Fundamenta Informaticae XX (2021) 1–35
DOI 10.3233/FI-2016-0000
IOS Press

Complexity assessments for decidable fragments of Set Theory. I: A taxonomy for the Boolean case.*

Domenico Cantone
Dept. of Mathematics and Computer Science
University of Catania, Italy
domenico.cantone@unict.it

Andrea De Domenico
Scuola Superiore di Catania,
University of Catania, Italy
andrea.dedomenico@studium.unict.it

Pietro Maugeri
Dept. of Mathematics and Computer Science
University of Catania, Italy
pietro.maugeri@unict.it

Eugenio G. Omodeo
Dept. of Mathematics and Geosciences
University of Trieste, Italy
eomodeo@units.it

Abstract. We report on an investigation aimed at identifying small fragments of set theory (typically, sublanguages of Multi-Level Syllogistic) endowed with polynomial-time satisfiability decision tests, potentially useful for automated proof verification. Leaving out of consideration the membership relator $\in$ for the time being, in this paper we provide a complete taxonomy of the polynomial and the NP-complete fragments involving, besides variables intended to range over the von Neumann set-universe, the Boolean operators $\cup, \cap, \setminus$, the Boolean relators $\subseteq, \not\subseteq, =, \neq$, and the predicates '$\cdot = \varnothing$' and '$\mathsf{Disj}(\cdot, \cdot)$', meaning 'the argument set is empty' and 'the arguments are disjoint sets', along with their opposites '$\cdot \neq \varnothing$' and '$\neg\mathsf{Disj}(\cdot, \cdot)$'. We also examine in detail how to test for satisfiability the formulae of six sample fragments: three sample problems are shown to be NP-complete, two to admit quadratic-time decision algorithms, and one to be solvable in linear time.

Keywords: Satisfiability problem, Computable set theory, Boolean set theory, Expressibility, NP-completeness, Proof verification.

Address for correspondence: D. Cantone, DMI, Università degli Studi di Catania, viale Andrea Doria, 6 — 95125 — Catania (I)

*We gratefully acknowledge partial support from project "STORAGE—Università degli Studi di Catania, Piano della Ricerca 2020/2022, Linea di intervento 2", and from INdAM-GNCS 2019 and 2020 research funds.

Introduction

The decision problem for fragments of set theory, namely the problem of establishing algorithmically for any formula φ in a given fragment whether or not φ is valid in the von Neumann universe of sets, has been thoroughly investigated over the last four decades within the field named *Computable Set Theory*. Research has mainly focused on the equivalent *satisfiability problem*, namely the problem of establishing in an effective manner, for any formula in a given fragment, whether an assignment of sets to free variables exists that makes the formula true.

The initial goal (back in 1978, cf. [1]) envisaged an automated proof checker based on set theory, within which it would become possible to carry out an extensive formalization of classical mathematics, as well as program-correctness verifications. The inferential kernel of such a proof assistant should have embodied decision procedures intended to capture the 'obvious' (deduction steps). Very soon, and long before the proof checker came into existence, the initial goal sparked a foundational quest aimed at drawing the precise frontier between the decidable and the undecidable in set theory (and also in other important mathematical theories). This inspired much of the subsequent work. Several extensions of the progenitor fragments MLS and MLSS of set theory were proved to have a solvable satisfiability problem, which led to a substantial body of results, partly comprised in the monographs [2, 3, 4, 5, 6].

We recall that MLS—an acronym for *Multi-Level Syllogistic*, see [7, 8, 9]—copes with propositional combinations of literals of the forms

$$x = y \cup z, \quad x = y \cap z, \quad x = y \setminus z, \quad x \neq y, \quad x \in y, \quad x \notin y, \tag{1}$$

(where x, y, z stand for set-variables); MLSS adds the singleton operator $\{\cdot\}$ to these constructs.[1] Unfortunately, as shown in [12], the satisfiability problem for either MLS or MLSS is NP-complete, even if restricted to conjunctions of literals of type (1). All extensions of MLS will hence have, in their turn, an NP-hard satisfiability problem (in fact, even hyperexponential in some cases; see [13, 14, 15]). Notwithstanding, the decision algorithm for an enriched variant of MLSS, implemented along the guidelines of [16], has come to play a key role among the inference mechanisms available in the proof-checker ÆtnaNova, aka Ref [4]. In view of the pervasiveness of that mechanism in actual uses of ÆtnaNova (as discussed, e.g., in [17, Sect. 3], in [5, Sect. 5.3.1], and in the sections on 'blobbing' of [4]), it will pay off to circumvent whenever possible the poor performances occasionally originating from the full-strength decision algorithm.

This is why we recently undertook an investigation aimed at identifying useful 'small' fragments of set theory (which in most cases are subfragments of MLS) endowed with polynomial-time decision tests.

In this note we report on results mainly focused, for the time being, on fragments that exclude the membership relator $\in$.[2] We provide a complete taxonomy of the polynomial and the NP-complete fragments involving, besides set variables intended to range over the

[1] Note that by adding Cartesian square literals $x = y \times y$ and cardinality literals $|x| = |y|$, $|x| \neq |y|$ to MLS, one makes the satisfiability problem undecidable (see [10] and the recent [11]).

[2] Anyhow, some results involving the membership relator $\in$, and treated elsewhere, will be reviewed in Sect. 3.1.

von Neumann universe of all sets (see below), the Boolean operators $\cup, \cap, \setminus$ and relators $\subseteq, \not\subseteq, =, \neq$, and the predicates (both affirmed and negated) '$\cdot = \varnothing$' and '$\mathsf{Disj}(\cdot, \cdot)$', expressing respectively that a specified set is empty and that two specified sets are disjoint.

The paper is organized as follows: Sect. 1 introduces the syntax and semantics of a language in which several hundreds of decidable fragments will be framed in this note; a subsection of it defines 'expressibility', a notion which eases the systematic assessment of the complexities of the satisfiability decision tests. Sect. 2 highlights a complexity-based classification of the fragments under consideration; two subsections of it examine in detail, respectively: three minimal NP-complete fragments; three emblematic, polynomial-complexity decision algorithms. Next, in Sections 3.1 and 3.2, the authors survey results published elsewhere [18, 19], regarding a fragment of set theory that involves the membership relator: a complexity taxonomy for that fragment is recalled, and the possibility to perform a quadratic-time satisfiability-preserving translation of it into a purely Boolean language is pointed out. This translation preserves satisfiability and requires quadratic time, hence it bridges the matter discussed herein and the material treated in [18]. Sect. 3.4 mentions complexity taxonomies developed by other authors for fragments of various logical systems. We conclude the paper with a few hints at future research. Then an appendix offers the proofs of two lemmas on expressibility matters.

1. Boolean set theory

We now introduce an interpreted language regarding sets, whose acronym $\mathbb{BST}$ stands for 'Boolean set theory'. The constructs of $\mathbb{BST}$ are borrowed from the algebraic theory of Boolean rings (see [20, Ch.VII]), but its variables are meant to range over a universe of nested (as opposed to 'flat') sets. We dub $\mathbb{BST}$ a 'theory' just to emphasize that its satisfiability problem is decidable. In the ongoing we will browse a wide range of satisfiability subproblems of the one referring to the entire $\mathbb{BST}$, and will assess their algorithmic complexities.

We postpone to other reports (see, for the time being, [18]) the treatment of $\in$, the membership relation. Adding $\in$ to $\mathbb{BST}$ does not disrupt its decidability (see Sect. 3.1) and truly calls for nested sets.

1.1. Syntax

The fragments of set theory investigated within the project we are reporting about are parts, delimited syntactically, of a specific quantifier-free language

$$\mathbb{BST} \coloneqq \mathsf{BST}(\cup, \cap, \setminus, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq).$$

This is the collection of all conjunctions of literals of the types

$$\begin{array}{cccc} s = \varnothing, & s \neq \varnothing, & \mathsf{Disj}(s,t)\,, & \neg\mathsf{Disj}(s,t)\,, \\ s \subseteq t, & s \not\subseteq t, & s = t, & s \neq t, \end{array}$$

where s and t stand for terms assembled from a denumerably infinite supply of set variables $x, y, z, \ldots$ by means of the Boolean operators: union $\cup$, intersection $\cap$, and set difference $\setminus$.

More generally, we shall denote by $\mathsf{BST}(\mathsf{op}_1, \ldots, \mathsf{pred}_1, \ldots)$ the subtheory of $\mathbb{BST}$ involving only the set operators $\mathsf{op}_1, \ldots$ (drawn from the collection $\{\cup, \cap, \setminus\}$) and the predicate symbols $\mathsf{pred}_1, \ldots$ (drawn from $\{=\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq\}$).

1.2. Semantics

For any $\mathbb{BST}$-conjunction φ, we shall denote by $\mathit{Vars}(\varphi)$ the collection of set variables occurring in φ; $\mathit{Vars}(\tau)$ is defined likewise, for any $\mathbb{BST}$-term τ.

A *set assignment* M is any function sending a collection of set variables V (called the *domain* of M and denoted $\mathsf{dom}(M)$) into the von Neumann universe $\mathcal{V}$ of well-founded sets. We recall that the *von Neumann universe* (see [21, pp. 95–102]), aka *von Neumann cumulative hierarchy*, is built up in stages as the union $\mathcal{V} \coloneqq \bigcup_{\alpha \in On} \mathcal{V}_\alpha$ of the levels $\mathcal{V}_\alpha \coloneqq \bigcup_{\beta<\alpha} \mathscr{P}(\mathcal{V}_\beta)$, with α ranging over the class On of all ordinal numbers, where $\mathscr{P}(\cdot)$ is the powerset operator.

Natural designation rules attach recursively a value to every term τ of $\mathbb{BST}$ such that $\mathit{Vars}(\tau) \subseteq \mathsf{dom}(M)$, for any set assignment M; here is how:

$$M(s \cup t) \coloneqq Ms \cup Mt, \quad M(s \cap t) \coloneqq Ms \cap Mt, \quad \text{and} \quad M(s \setminus t) \coloneqq Ms \setminus Mt.$$

We also put $ML \coloneqq \{\, Mv \mid v \in L\}$ for every $L \subseteq \mathsf{dom}(M)$, and

$$M(s = \varnothing) \coloneqq \begin{cases} \mathsf{true} & \text{if } Ms = \emptyset \\ \mathsf{false} & \text{otherwise,} \end{cases} \qquad M(\mathsf{Disj}(s,t)) \coloneqq \begin{cases} \mathsf{true} & \text{if } Ms \cap Mt = \emptyset \\ \mathsf{false} & \text{otherwise,} \end{cases}$$

$$M(s \neq \varnothing) \coloneqq \neg M(s = \varnothing), \qquad M\big(\neg\mathsf{Disj}(s,t)\big) \coloneqq \neg M\big(\mathsf{Disj}(s,t)\big),$$

for all literals $s = \varnothing$, $s \neq \varnothing$, $\mathsf{Disj}(s,t)$, and $\neg\mathsf{Disj}(s,t)$ of $\mathbb{BST}$; and proceed similarly with the literals of $\mathbb{BST}$ of the remaining types $s \subseteq t$, $s \not\subseteq t$, $s = t$, and $s \neq t$. Then we put, recursively,

$$M(\varphi_1 \wedge \cdots \varphi_k) \coloneqq M\varphi_1 \wedge \cdots M\varphi_k$$

when $\varphi_1, \ldots, \varphi_k$ are $\mathbb{BST}$-conjunctions and $\mathit{Vars}(\varphi_i) \subseteq \mathsf{dom}(M)$, for $i = 1, \ldots, k$.

Given a conjunction φ and a set assignment M such that $\mathit{Vars}(\varphi) \subseteq \mathsf{dom}(M)$, we say that M *satisfies* φ, and write $M \models \varphi$, if $M\varphi = \mathsf{true}$. When M satisfies φ, we also say that M is a *model* of φ. If all assignments M such that $\mathit{Vars}(\varphi) \subseteq \mathsf{dom}(M)$ satisfy φ, then we write $\models \varphi$.

A conjunction φ is said to be *satisfiable* if it has some model, else *unsatisfiable*.

Example 1.1. Here are three unsatisfiable $\mathbb{BST}$ conjunctions (see [19]):

$$\mathsf{Disj}(x \cup y\,,\, x' \cup y') \;\wedge\; x \cup x' = y \cup y' \;\wedge\; x \neq y\,,$$
$$\mathsf{Disj}(x \cup y \cup z\,,\, x' \cup y' \cup z') \;\wedge\; (x \cup x') \setminus (y \cup y') = z \cup z' \;\wedge\; x \setminus y \neq z\,,$$
$$\mathsf{Disj}(x \cup y \cup z\,,\, x' \cup y' \cup z') \;\wedge\; x \setminus y = z \;\wedge\; x' \setminus y' = z' \;\wedge\; (x \cup x') \setminus (y \cup y') \neq z \cup z'\,. \qquad \dashv$$

$\mathbb{BST}$ is a sublanguage of MLS, which has an NP-complete satisfiability problem [12]; since, in their turn, the fragments of set theory which we shall examine are included in $\mathbb{BST}$, their satisfiability problems belong to NP.

1.3. Expressibility

The reduction technique to be highlighted next has been our main tool in the construction of the complexity taxonomy of $\mathbb{BST}$-fragments which will be treated at length in Sect. 2.

Most of our reductions will be based on the standard notion of 'context-free' expressibility:

Definition 1.2. (Expressibility)
A formula $\psi(\vec{x})$ is said to be *expressible* in a fragment $\mathcal{T}$ of $\mathbb{BST}$, if there exists a $\mathcal{T}$-conjunction $\Psi(\vec{x},\vec{y})$ such that

$$\models \psi(\vec{x}) \longleftrightarrow (\exists\vec{y})\,\Psi(\vec{x},\vec{y}),$$

where $\vec{x}$ and $\vec{y}$ stand for tuples of set variables.

We also devised a more general notion of 'context-sensitive' expressibility, also characterized by its complexity. We named it $\mathcal{O}(f)$-*expressibility*, where $f\colon \mathbb{N}\to\mathbb{N}$ is any mapping intended to bound the complexity of the underlying rewriting procedure.[3]

Definition 1.3. ($\mathcal{O}(f)$-expressibility)
Let $\mathcal{T}$ and f be a fragment of $\mathbb{BST}$ and a specific mapping $f\colon \mathbb{N}\to\mathbb{N}$, respectively. A formula $\psi(\vec{x})$—typically involving a construct which one intends to eliminate—is said to be $\mathcal{O}(f)$-*expressible* in $\mathcal{T}$ if there exists a mapping

$$\varphi(\vec{y}) \mapsto \Psi_\varphi(\vec{x},\vec{y},\vec{z}) \tag{2}$$

from $\mathcal{T}$ into $\mathcal{T}$ (viz., each formula φ of $\mathcal{T}$ is sent into a formula of $\mathcal{T}$) such that the following conditions are satisfied by every φ:

(a) the mapping (2) can be computed in $\mathcal{O}(f(|\varphi|))$-time,

(b) if $\varphi(\vec{y})\wedge(\exists\vec{z})\Psi_\varphi(\vec{x},\vec{y},\vec{z})$ is satisfiable, so is $\varphi(\vec{y})\wedge\psi(\vec{x})$,

(c) $\models \big(\varphi(\vec{y}) \wedge \psi(\vec{x})\big) \longrightarrow (\exists\vec{z})\Psi_\varphi(\vec{x},\vec{y},\vec{z})$.

Remark 1.4. In Def. 1.3, conditions (b) and (c) have a joint bearing lesser than the condition

$$\models \big(\varphi(\vec{y})\wedge\psi(\vec{x})\big) \longleftrightarrow \big(\varphi(\vec{y})\wedge(\exists\vec{z})\,\Psi_\varphi(\vec{x},\vec{y},\vec{z})\big),$$

akin to the one characterizing simple expressibility in Def. 1.2. Indeed, while condition (c) requires that each model of $\varphi(\vec{y})\wedge\psi(\vec{x})$ is also a model of $\varphi(\vec{y})\wedge(\exists\vec{z})\,\Psi_\varphi(\vec{x},\vec{y},\vec{z})$, condition (b) just requests that when $\varphi(\vec{y})\wedge(\exists\vec{z})\,\Psi_\varphi(\vec{x},\vec{y},\vec{z})$ is satisfied by some set assignment M, then $\varphi(\vec{y})\wedge\psi(\vec{x})$ is satisfiable, possibly by a set assignment other than M.

It turns out that standard expressibility is a special case of $\mathcal{O}(1)$-expressibility. This is stated in the following lemma.

[3] A more general notion of 'cross'-expressibility recently appeared in [19, p. 218].

Lemma 1.5. If a formula $\psi(\vec{x})$ is expressible in a fragment $\mathcal{T}$ of $\mathbb{BST}$, then it is also $\mathcal{O}(1)$-expressible in $\mathcal{T}$.

Proof:
Let $\psi(\vec{x})$ be any formula expressible in $\mathcal{T}$, and let $\Psi(\vec{x},\vec{z})$ be a $\mathcal{T}$-conjunction such that

$$\models \psi(\vec{x}) \longleftrightarrow (\exists\vec{z})\Psi(\vec{x},\vec{z})\,. \tag{3}$$

Consider the mapping

$$\varphi(\vec{y}) \mapsto \Psi(\vec{x},\vec{z}) \tag{4}$$

from $\mathcal{T}$ into $\mathcal{T}$, where $\vec{z}$ is any tuple of distinct set variables. Plainly, the mapping (4) can be computed in $\mathcal{O}(1)$ time. In addition, by (3), we have

$$\models \big(\varphi(\vec{y}) \wedge \psi(\vec{x})\big) \longleftrightarrow \big(\varphi(\vec{y}) \wedge (\exists\vec{z})\Psi(\vec{x},\vec{z})\big).$$

Hence, in particular, the formulae $\varphi(\vec{y}) \wedge \psi(\vec{x})$ and $\varphi(\vec{y}) \wedge (\exists\vec{z})\Psi(\vec{x},\vec{z})$ are equisatisfiable, and we also have

$$\models \big(\varphi(\vec{y}) \wedge \psi(\vec{x})\big) \longrightarrow (\exists\vec{z})\Psi(\vec{x},\vec{z}).$$

Thus, conditions (b) and (c) of Def. 1.3 are also satisfied, proving that the formula $\psi(\vec{x})$ is $\mathcal{O}(1)$-expressible in $\mathcal{T}$. □

Various expressibility and inexpressibility results are collected in the following two lemmas, whose proofs are provided in Appendix A.

Lemma 1.6. (a) $x = y \setminus z$ is expressible in $\mathsf{BST}(\cup, \mathsf{Disj}, =)$;

(b) $x = y \cap z$ and $x = y \cup z$ are expressible in $\mathsf{BST}(\setminus, =)$;

(c) $x = y$ is expressible in $\mathsf{BST}(\subseteq)$;

(d) $x \subseteq y$ is expressible both in $\mathsf{BST}(\cup, =)$ and in $\mathsf{BST}(\cap, =)$;

(e) $x \not\subseteq y$ is expressible both in $\mathsf{BST}(\cup, \neq)$ and in $\mathsf{BST}(\cap, \neq)$;

(f) $x \neq \varnothing$ is expressible in $\mathsf{BST}(\subseteq, \neq)$, and therefore in $\mathsf{BST}(\cup, =, \neq)$; moreover, $x \neq \varnothing$ is expressible in $\mathsf{BST}(\not\subseteq)$, in $\mathsf{BST}(\neq, \mathsf{Disj})$, in $\mathsf{BST}(=\varnothing, \neq)$, and in $\mathsf{BST}(\neg\mathsf{Disj})$;

(g) $x = \varnothing$ is expressible in $\mathsf{BST}(\mathsf{Disj})$;

(h) $\mathsf{Disj}(x, y)$ is expressible both in $\mathsf{BST}(\cap, =\varnothing)$ and in $\mathsf{BST}(\setminus, =)$, and $\neg\mathsf{Disj}(x, y)$ is expressible both in $\mathsf{BST}(\cap, \neq\varnothing)$ and in $\mathsf{BST}(\subseteq, \neq\varnothing)$;

(i) $\neg\mathsf{Disj}(x, y)$ (i.e., $x \cap y \neq \varnothing$) is expressible in $\mathsf{BST}(\subseteq, \neq)$, and therefore expressible in $\mathsf{BST}(\cup, =, \neq)$;

(j) $x = \varnothing$ is not expressible in $\mathsf{BST}(\cup, \cap, =, \neq)$;

(k) $x = y \setminus z$ is not expressible in $\mathsf{BST}(\cup, \cap, =, \neq)$.

Lemma 1.7. The mapping $\varphi(\vec{y}) \mapsto \Psi_\varphi(x, \vec{y})$ from $\mathsf{BST}(\cup, =, \neq)$ into itself, where x is any set variable (possibly in φ) and

$$\Psi_\varphi(x, \vec{y}) \coloneqq \bigwedge_{y \in Vars(\varphi)} y \cup x = y,$$

enjoys the properties

- if $\varphi(\vec{y}) \wedge \Psi_\varphi(x, \vec{y})$ is satisfiable, so is $\varphi(\vec{y}) \wedge x = \varnothing$, and
- $\models \big(\varphi(\vec{y}) \wedge x = \varnothing\big) \longrightarrow \Psi_\varphi(x, \vec{y})$.

Hence, the literal $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cup, =, \neq)$ via $\Psi_\varphi(x, \vec{y})$.

2. Complexity taxonomy of the fragments of $\mathbb{BST}$

Of a fragment of $\mathbb{BST}$, we say that *it is* NP*-complete* if it has an NP-complete satisfiability problem (see [22]). Likewise, we say that *it is polynomial* if its satisfiability problem has polynomial complexity.

The overall number of fragments of $\mathbb{BST}$ is $2^3 \cdot (2^8 - 1) = 2040$; of these, 1278 are NP-complete and the remaining 762 are polynomial. The complexity of any fragment of $\mathbb{BST}$ can be efficiently identified once the *minimal* NP-complete fragments (namely the NP-complete fragments of $\mathbb{BST}$ that do not strictly contain any NP-complete fragment of $\mathbb{BST}$) and the *maximal* polynomial fragments (namely the polynomial fragments of $\mathbb{BST}$ that are not strictly contained in any polynomial fragment of $\mathbb{BST}$) have been singled out. Indeed, any $\mathbb{BST}$-fragment either is contained in some maximal polynomial $\mathbb{BST}$-fragment or contains some minimal NP-complete fragment.

Table 1 reports the 18 minimal NP-complete fragments and the 5 maximal polynomial fragments of $\mathbb{BST}$. Each row represents the fragment involving the operators and the relators that are marked with a '$\star$' symbol.

2.1. Minimal NP-complete fragments of $\mathbb{BST}$

Concerning the NP-complete fragments, initially we proved that the fragments

$$\mathsf{BST}(\setminus, \neq), \quad \mathsf{BST}(\cup, \cap, \neq), \quad \mathsf{BST}(\cup, \cap, =\varnothing, \neq\varnothing), \text{ and } \quad \mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$$

are NP-complete, by reducing the famous NP-complete problem 3SAT [22] to each of them.

Then, referring to the the initial blocks of Table 1, it can be observed that:

first block: the NP-completeness of the fragments $\mathsf{BST}(\setminus, \not\subseteq)$, $\mathsf{BST}(\setminus, \neg\mathsf{Disj})$, and $\mathsf{BST}(\setminus, \neq\varnothing)$ can be obtained by much the same technique used to reduce 3SAT to $\mathsf{BST}(\setminus, \neq)$;[4]

[4]The proof of NP-completeness of $\mathsf{BST}(\setminus, \neq)$ is so closely analogous to the one regarding $\mathsf{MST}(\setminus, \in)$ as provided in [18, Sec. 3.2] that it would be pointless to replicate it here.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg$Disj	$\subseteq$	$\not\subseteq$	$=$	$\neq$	Complexity	Section
		⋆								⋆	NP-complete	2.1
		⋆						⋆			NP-complete	2.1
		⋆				⋆					NP-complete	2.1
		⋆		⋆							NP-complete	2.1
⋆	⋆									⋆	NP-complete	2.1.1
⋆	⋆							⋆			NP-complete	2.1.1
⋆	⋆		⋆	⋆							NP-complete	2.1.3
⋆	⋆				⋆	⋆					NP-complete	2.1.3
⋆	⋆		⋆			⋆					NP-complete	2.1.3
⋆	⋆			⋆	⋆						NP-complete	2.1.3
⋆					⋆	⋆			⋆		NP-complete	2.1.2
⋆				⋆	⋆				⋆		NP-complete	2.1.2
⋆					⋆	⋆	⋆				NP-complete	2.1.2
⋆					⋆				⋆	⋆	NP-complete	2.1.2
⋆					⋆			⋆	⋆		NP-complete	2.1.2
⋆				⋆	⋆		⋆				NP-complete	2.1.2
⋆					⋆		⋆			⋆	NP-complete	2.1.2
⋆					⋆		⋆	⋆			NP-complete	2.1.2
⋆	⋆	⋆	⋆		⋆		⋆		⋆		$\mathcal{O}(1)$	2.2
⋆	⋆			⋆		⋆	⋆		⋆		$\mathcal{O}(1)$	2.2
⋆			⋆	⋆	⋆	⋆		⋆		⋆	$\mathcal{O}(n^3)$	2.2
⋆			⋆	⋆		⋆	⋆	⋆	⋆	⋆	$\mathcal{O}(n^3)$	2.2
	⋆		⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	$\mathcal{O}(n^4)$	2.2

Table 1. Complete taxonomy of minimal NP-complete and maximal polynomial fragments of $\mathbb{BST}$

second block: the proof of NP-completeness of the fragment $\mathsf{BST}(\cup, \cap, \not\subseteq)$ can be achieved by much the same technique used to reduce 3SAT to $\mathsf{BST}(\cup, \cap, \neq)$;

third block: the NP-completeness of $\mathsf{BST}(\cup, \cap, \mathsf{Disj}, \neg\mathsf{Disj})$, $\mathsf{BST}(\cup, \cap, =\varnothing, \neg\mathsf{Disj})$, and $\mathsf{BST}(\cup, \cap, \neq\varnothing, \mathsf{Disj})$ can be obtained by much the same technique used to reduce 3SAT to $\mathsf{BST}(\cup, \cap, =\varnothing, \neq\varnothing)$; and

fourth block: the NP-completeness of the fragment $\mathsf{BST}(\cup, =, \neq\varnothing, \mathsf{Disj})$ can be shown by much the same reduction technique used for $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$.

Finally, by resorting to some of the expressibility results listed in Lemma 1.6, it can readily be proved that:

- $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$ can be reduced in linear time
  to $\mathsf{BST}(\cup, \subseteq, \mathsf{Disj}, \neg\mathsf{Disj})$, by Lemma 1.6(c),

- $\mathsf{BST}(\cup, =, \neq\varnothing, \mathsf{Disj})$ can be reduced in linear time
  to $\mathsf{BST}(\cup, =, \neq, \mathsf{Disj})$, by Lemma 1.6(f),
  to $\mathsf{BST}(\cup, =, \not\subseteq, \mathsf{Disj})$, by Lemma 1.6(f),
  to $\mathsf{BST}(\cup, \subseteq, \neq\varnothing, \mathsf{Disj})$, by Lemma 1.6(c),
  to $\mathsf{BST}(\cup, \subseteq, \neq, \mathsf{Disj})$, by Lemma 1.6(c)(f),
  to $\mathsf{BST}(\cup, \subseteq, \not\subseteq, \mathsf{Disj})$, by Lemma 1.6(c)(f).

2.1.1. NP-completeness of $\mathsf{BST}(\cup, \cap, \neq)$

The fragment $\mathsf{BST}(\cup, \cap, \neq)$ consists of all conjunctions of literals of type $t_1 \neq t_2$, where t_1, t_2 are any terms involving the set operators $\cup$ and $\cap$. We will now reduce the problem 3SAT to the satisfiability problem of $\mathsf{BST}(\cup, \cap, \neq)$.

Let F be an instance of 3SAT of the form

$$F := C_1 \wedge \cdots \wedge C_m,$$

where $C_i = L_{i1} \vee L_{i2} \vee L_{i3}$, and the L_{ij}'s are propositional literals. Let $P_1, \ldots, P_n$ be the distinct propositional variables in F, and let $X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n$ be $2n$ pairwise distinct set variables. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, set

$$u_{ij} := \begin{cases} X_k & \text{if } L_{ij} = P_k \text{ for some } k \\ \overline{X}_k & \text{if } L_{ij} = \neg P_k \text{ for some } k. \end{cases}$$

Then, for each $i = 1, \ldots, m$, define

$$\begin{aligned} t_i &:= u_{i1} \cup u_{i2} \cup u_{i3}, \\ X &:= (X_1 \cup \overline{X}_1) \cap \cdots \cap (X_n \cup \overline{X}_n), \\ Y &:= (X_1 \cap \overline{X}_1) \cup \cdots \cup (X_n \cap \overline{X}_n), \\ T &:= t_1 \cap \cdots \cap t_m. \end{aligned}$$

Finally put

$$\Phi_F \coloneqq T \cap X \neq T \cap X \cap Y.$$

It can easily be checked that $|\Phi_F| = \mathcal{O}(|F|)$.

Lemma 2.1. Any instance F of the 3SAT problem is propositionally satisfiable if and only if the $\mathsf{BST}(\cup, \cap, \neq)$-formula Φ_F resulting from the above linear-time construction is satisfied by a set assignment.

Proof:
(Necessity.) First, we show that if the 3SAT instance F is propositionally satisfiable, then Φ_F is satisfied by a set assignment. To this purpose, let $\mathfrak{v}$ a Boolean valuation satisfying F, and let M^* be the set assignment such that

$$\begin{aligned} M^*X_k &:= b \quad \text{and} \quad M^*\overline{X}_k := \emptyset \quad \text{if } \mathfrak{v}(P_k) = \mathbf{t} \\ M^*X_k &:= \emptyset \quad \text{and} \quad M^*\overline{X}_k := b \quad \text{if } \mathfrak{v}(P_k) = \mathbf{f}, \end{aligned}$$

where b is any fixed non-empty set. Regardless of the value of $\mathfrak{v}(P_i)$, we have $(M^*X_i \cup M^*\overline{X}_i) = b$ and $(M^*X_i \cap M^*\overline{X}_i) = \emptyset$, so that $M^*X = b$ and $M^*Y = \emptyset$. Hence to prove that $M^* \models \Phi_F$, we just need to show that $M^*T = b$. Preliminarily, we notice that $M^*x \in \{b, \emptyset\}$, for each set variable x. Since $\mathfrak{v}$ satisfies F, we have $\mathfrak{v}(C_i) = \mathbf{t}$, for each $i = 1, \ldots, m$. Hence, for each $i = 1, \ldots, m$, there exists a $j = 1, 2, 3$ such that $\mathfrak{v}(L_{ij}) = \mathbf{t}$. So, if $L_{ij} = P_k$ for some k, then $M^*u_{ij} = M^*X_k = b$, while if $L_{ij} = \neg P_k$ then $M^*u_{ij} = M^*\overline{X}_k = b$. Hence, $M^*t_i = b$, for $i = 1, \ldots, m$, and also $M^*T = b$.

(Sufficiency.) Concerning the converse, assume that Φ_F is satisfiable. Since there is only one negative literal in Φ_F, there exists a set assignment M that satisfies Φ_F and is such that, for some non-empty set σ, $Mv \in \{\sigma, \emptyset\}$ for every set variable $v \in \mathit{Vars}(\Phi_F)$.[5] Since M satisfies Φ_F, plainly $MX \neq \emptyset$ and $MY = \emptyset$, so that $(MX_1 \cup M\overline{X}_1) \cap \ldots \cap (MX_n \cup M\overline{X}_n) \neq \emptyset$ and also $(MX_1 \cap M\overline{X}_1) \cup \ldots \cup (MX_n \cap M\overline{X}_n) = \emptyset$. Hence, for every $i = 1, \ldots, n$, we have $(MX_1 \cup M\overline{X}_1) \neq \emptyset$ and $(MX_i \cap M\overline{X}_i) = \emptyset$, from which the implication

$$MX_i \neq \emptyset \quad \longrightarrow \quad M\overline{X}_i = \emptyset \tag{5}$$

follows, and conversely. Moreover, since $MT \neq \emptyset$, then $Mt_i \neq \emptyset$ for each $i = 1, \ldots, m$. Hence, for each $i = 1, \ldots, m$, there exists a $j \in \{1, 2, 3\}$ such that $Mu_{ij} \neq \emptyset$. Consider the following Boolean valuation:

$$\mathfrak{v}(P_k) := \begin{cases} \mathbf{t} & \text{if } MX_k \neq \emptyset \\ \mathbf{f} & \text{otherwise.} \end{cases}$$

We only need to show that, for each $i = 1, \ldots, m$, there exists a $j \in \{1, 2, 3\}$ such that $\mathfrak{v}(L_{ij}) = \mathbf{t}$. We know that, for $i = 1, \ldots, m$, there exists a $\overline{j} = 1, 2, 3$ such that $Mu_{i\overline{j}} \neq \emptyset$, if $L_{i\overline{j}} = P_k$ for some k. Then $u_{i\overline{j}} = X_k$, so that $MX_k \neq \emptyset$. Hence $\mathfrak{v}(L_{i\overline{j}}) = \mathbf{t}$. On the other

[5] This is proved in [6, Sec.2.3].

hand, if $L_{i\bar{j}} = \neg P_k$ for some k, then $Mu_{i\bar{j}} = M\overline{X}_k \neq \emptyset$. so that, by (5), $MX_k = \emptyset$, $\mathfrak{v}(P_k) = \mathbf{f}$, and $\mathfrak{v}(L_{i\bar{j}}) = \mathbf{t}$. □

We can then conclude that:

Corollary 2.2. The satisfiability problem of $\mathsf{BST}(\cup, \cap, \neq)$ is NP-hard, and therefore it is NP-complete.

2.1.2. NP-completeness of $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$

The fragment $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$ of $\mathbb{BST}$ consists of all conjunctions of literals of the forms

$$\begin{aligned} l_1 \cup \cdots \cup l_n &= r_1 \cup \cdots \cup r_m, \\ \mathsf{Disj}(u_1 \cup \cdots \cup u_h\,,\ t_1 \cup \cdots \cup t_k), \\ \neg\mathsf{Disj}(s_1 \cup \cdots \cup s_p\,,\ z_1 \cup \cdots \cup z_q), \end{aligned}$$

where the l_i's, r_i's, u_i's, t_i's, s_i's, and z_i's are set variables and $n, m, h, k, p, q \geqslant 1$.

We will show that the satisfiability problem for this fragment belongs to the NP-complete problem class, by showing that any instance of the 3SAT problem can be reduced to an instance of the satisfiability problem for $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$.

Let F be a 3SAT instance:

$$F \coloneqq C_1 \wedge \cdots \wedge C_m\,,$$

where $C_i = L_{i1} \vee L_{i2} \vee L_{i3}$, and the L_{ij}'s are propositional literals. Let $P_1, \ldots, P_n$ be the distinct propositional letters in F; associate with them $2\,n + 1$ pairwise distinct set variables $X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n, \mathcal{X}$. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, define

$$T_{ij} \coloneqq \begin{cases} X_k & \text{if } L_{ij} = P_k \text{ for some } k \\ \overline{X}_k & \text{if } L_{ij} = \neg P_k \text{ for some } k, \end{cases}$$

then define

$$\mathcal{C}_i \coloneqq T_{i1} \cup T_{i2} \cup T_{i3} = \mathcal{X},$$

and finally put

$$\Phi_F \coloneqq \bigwedge_{i=1}^{m} \mathcal{C}_i \ \wedge \ \bigwedge_{k=1}^{n} \Big(\mathsf{Disj}\big(X_k, \overline{X}_k\big) \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X}\Big) \ \wedge \ \neg\mathsf{Disj}(\mathcal{X}, \mathcal{X})\ . \tag{6}$$

Lemma 2.3. Any instance F of the 3SAT problem is propositionally satisfiable if and only if the $\mathsf{BST}(\cup, =\,, \mathsf{Disj}, \neg\mathsf{Disj})$-formula Φ_F resulting from the above construction is satisfied by some set assignment.

Proof:
(Sufficiency.) To prove sufficiency, suppose that Φ_F is satisfiable. Since $\neg\mathsf{Disj}(\mathcal{X},\mathcal{X})$ is the only negative constraint in Φ_F, there exists a model M of Φ_F such that[6]

$$MX_k, M\overline{X}_k \in \{\emptyset, b\} \tag{7}$$

holds for each $k = 1,\ldots,n$, where b is a fixed non-empty set.

Since $M \models \Phi_F$, we have

$$\begin{array}{ll} M \models \neg\mathsf{Disj}(\mathcal{X},\mathcal{X}) & \therefore \\ M \models \mathcal{X} \cap \mathcal{X} \neq \varnothing & \therefore \\ M \models \mathcal{X} \neq \varnothing & \therefore \\ M\mathcal{X} \neq \emptyset, & \end{array} \tag{8}$$

and also, for each $k = 1,\ldots,n$,

$$\begin{array}{ll} M \models \mathsf{Disj}\big(X_k, \overline{X}_k\big) \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X} & \therefore \\ M \models X_k \cap \overline{X}_k = \varnothing \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X} & \therefore \\ MX_k \cap M\overline{X}_k = \emptyset \ \wedge \ MX_k \cup M\overline{X}_k \neq \emptyset & \text{(by (8))}. \end{array} \tag{9}$$

By combining (7) and (9), we obtain:

$$\big(\, MX_k = b \ \wedge \ M\overline{X}_k = \emptyset \,\big) \quad \vee \quad \big(\, MX_k = \emptyset \ \wedge \ M\overline{X}_k = b \,\big). \tag{10}$$

Now consider the following truth-value assignment:

$$\mathfrak{v}(P_k) = \begin{cases} \mathsf{true} & \text{if } MX_k \neq \emptyset \\ \mathsf{false} & \text{otherwise.} \end{cases}$$

We have assumed that $M \models \Phi_F$; therefore $M \models \mathcal{C}_i$ holds, for each $i = 1,\ldots,m$, and hence:

$$\begin{array}{ll} M\,(T_{i1} \cup T_{i2} \cup T_{i3}) = M\mathcal{X} & \therefore \\ MT_{i1} \cup MT_{i2} \cup MT_{i3} \neq \emptyset & \text{(by (8))}. \end{array}$$

Thus, for each $k = 1,\ldots,n$, there exists a $j = 1,2,3$, such that $MT_{ij} \neq \emptyset$. There are only two cases to be examined: $T_{ij} = X_k$ and $T_{ij} = \overline{X}_k$. In the former case, we have $L_{ij} = P_k$ for some k, and also $MX_k = MT_{ij} \neq \emptyset$; hence $\mathfrak{v}(L_{ij}) = \mathfrak{v}(P_k) = \mathsf{true}$, and thus $\mathfrak{v}(C_i) = \mathsf{true}$. In the latter case, we have $L_{ij} = \neg P_k$ for some k, and also $M\overline{X}_k = MT_{ij} \neq \emptyset$; hence, by (10), $MX_k = \emptyset$, and $\mathfrak{v}(\neg P_k) = \mathfrak{v}(L_{ij}) = \mathsf{true}$, and thus $\mathfrak{v}(C_i) = \mathsf{true}$. We conclude that the truth-value assignment $\mathfrak{v}$ satisfies the instance F of 3SAT; hence F is propositionally satisfiable, in consequence of Φ_F being satisfied by a set assignment.

[6] This is proved in [6, Sec.2.3].

(Necessity.) For the necessity part of this lemma, suppose that $\mathfrak{v}$ is a truth-value assignment satisfying the instance F of 3SAT, and define the following set assignment:

$$\begin{aligned} &M\mathcal{X} = b, \\ &MX_k = \begin{cases} b & \text{if } \mathfrak{v}(P_k) = \mathsf{true} \\ \emptyset & \text{otherwise,} \end{cases} \\ &M\overline{X}_k = b \setminus MX_k, \end{aligned}$$

where b is a non-empty set.

Plainly the set assignment M satisfies $\neg\mathsf{Disj}(\mathcal{X}, \mathcal{X})$. Since, for each $k = 1, \ldots, n$, when $MX_k = b$ holds then $M\overline{X}_k = b \setminus MX_k = \emptyset$, and when $MX_k = \emptyset$ then $M\overline{X}_k = b \setminus MX_k = b$, we have:

$$MX_k \cap M\overline{X}_k = \emptyset \quad \therefore \quad M \models \mathsf{Disj}\big(X_k, \overline{X}_k\big)$$

and

$$MX_k \cup M\overline{X}_k = b = M\mathcal{X}\,,$$

so that

$$M \models \bigwedge_{k=1}^{n} \Big(\mathsf{Disj}\big(X_k, \overline{X}_k\big) \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X}\Big).$$

It remains to be proved that $M \models \mathcal{C}_i$ for each $i = 1, \ldots, m$. Since $\mathfrak{v}$ propositionally satisfies F, we have that $\mathfrak{v}(C_i) = \mathsf{true}$ holds for each $i = 1, \ldots, m$; hence for each C_i there must be at least one L_{ij}, $j \in \{1, 2, 3\}$, whose truth value is true. This means that for each $\mathcal{C}_i$ there is a T_{ij} such that $MT_{ij} = b$, hence $M \models \mathcal{C}_i$. In fact, if $L_{ij} = P_k$ for some k, then $T_{ij} = X_k$ and $\mathfrak{v}(P_k) = \mathsf{true}$, hence $MT_{ij} = b$; if $L_{ij} = \neg P_k$ for some k, then $T_{ij} = \overline{X}_k$ and $\mathfrak{v}(P_k) = \mathsf{false}$, hence $M\overline{X}_k = b \setminus MX_k = b \setminus \emptyset = b$. We conclude that $M \models \Phi_F$; thus, there exists a set assignment satisfying Φ_F in consequence of F being propositionally satisfiable. □

The lemma just seen readily yields that:

Corollary 2.4. The satisfiability problem for $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$ belongs to the class of NP-complete problems.

2.1.3. NP-completeness of $\mathsf{BST}(\cup, \cap, {=}\varnothing, {\neq}\varnothing)$

The fragment $\mathsf{BST}(\cup, \cap, {=}\varnothing, {\neq}\varnothing)$ consists of all conjunctions of literals of the forms

$$t_1 = \varnothing, \quad t_2 \neq \varnothing,$$

where t_1, t_2 stand for any terms involving the set operators $\cup$ and $\cap$. We will reduce the 3SAT problem to the satisfiability problem of $\mathsf{BST}(\cup, \cap, {=}\varnothing, {\neq}\varnothing)$.

Let F be an instance of 3SAT defined by

$$F \coloneqq C_1 \wedge \cdots \wedge C_m,$$

where $C_i = L_{i1} \vee L_{i2} \vee L_{i3}$, and the L_{ij}'s are propositional literals. Let $P_1, \ldots, P_n$ be the distinct propositional variables in F, and let $X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n$ be $2n$ pairwise distinct set variables. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, set

$$u_{ij} := \begin{cases} X_k & \text{if } L_{ij} = P_k \text{ for some } k, \\ \overline{X}_k & \text{if } L_{ij} = \neg P_k \text{ for some } k. \end{cases}$$

Then, for each $i = 1, \ldots, m$, define

$$\begin{aligned} t_i &:= u_{i1} \cup u_{i2} \cup u_{i3}, \\ y_i &:= (X_i \cap \overline{X}_i) = \varnothing, \\ Y &:= y_1 \wedge \cdots \wedge y_n \\ T &:= t_1 \cap \cdots \cap t_m, \end{aligned}$$

and finally put

$$\Phi_F := Y \wedge (X_1 \cup \overline{X}_1) \cap \cdots \cap (X_n \cup \overline{X}_n) \cap T \neq \varnothing. \tag{11}$$

Plainly, the formula Φ_F can be constructed in $\mathcal{O}(|F|)$ time.

Lemma 2.5. Any instance F of the 3SAT problem is propositionally satisfiable if and only if the $\mathsf{BST}(\cup, \cap, =\varnothing, \neq\varnothing)$-formula Φ_F resulting from the above construction is satisfied by some set assignment.

Proof:
(Necessity.) We will show that if the 3SAT instance F is propositionally satisfiable, then Φ_F is satisfied by a set assignment. Let $\mathfrak{v}$ be a Boolean valuation satisfying F, and M^* the set assignment such that

$$\begin{aligned} M^*X_k := b \quad \text{and} \quad M^*\overline{X}_k := \emptyset \quad \text{if } \mathfrak{v}(P_k) = \mathbf{t} \\ M^*X_k := \emptyset \quad \text{and} \quad M^*\overline{X}_k := b \quad \text{if } \mathfrak{v}(P_k) = \mathbf{f}, \end{aligned}$$

where b is any non-empty set. Regardless of the value of $\mathfrak{v}(P_i)$, we have $(M^*X_i \cup M^*\overline{X}_i) = b$ and $(M^*X_i \cap M^*\overline{X}_i) = \emptyset$, so that for each $i = 1, \ldots, n$ $M^* \models y_i$ and therefore $M^* \models Y$. Hence to prove that $M^* \models \Phi_F$, we just need to prove that $M^*T = b$. First, we notice that $M^*x \in \{b, \emptyset\}$, for each set variable x. Since $\mathfrak{v}$ satisfies F, we have $\mathfrak{v}(C_i) = \mathbf{t}$, for each $i = 1, \ldots, m$. Hence, for each $i = 1, \ldots, m$, there exists a $j \in \{1, 2, 3\}$ such that $\mathfrak{v}(L_{ij}) = \mathbf{t}$. So, if $L_{ij} = P_k$ for some k, then $M^*u_{ij} = M^*X_k = b$, while if $L_{ij} = \neg P_k$ then $M^*u_{ij} = M^*\overline{X}_k = b$. Hence $M^*t_i = b$, for $i = 1, \ldots, m$, and also $M^*T = b$.

(Sufficiency.) Concerning the converse, assume that Φ_F is satisfiable. Since there is only one negative literal in Φ_F, there exists a set assignment M that satisfies Φ_F and is such that $Mv \in \{\sigma, \emptyset\}$, for every set variables $v \in \mathit{Vars}(\Phi_F)$ and for some fixed non-empty set σ. Since M satisfies Φ_F, plainly $MT \neq \emptyset$ and $(MX_i \cup \overline{X}_i) \neq \emptyset$, for every $i = 1, \ldots, n$. Furthermore,

$M \models Y$ holds. Thus $(X_i \cap \overline{X}_i) = \emptyset$, for each $i = 1, \ldots, n$. We infer that the following implication holds

$$MX_i \neq \emptyset \quad \longrightarrow \quad M\overline{X}_i = \emptyset, \tag{12}$$

and conversely. Moreover, since $MT \neq \emptyset$, then $Mt_i \neq \emptyset$ for each $i = 1, \ldots, m$. Hence, for each $i = 1, \ldots, m$, there exists a $j \in \{1, 2, 3\}$ such that $Mu_{ij} \neq \emptyset$. Consider the following Boolean valuation

$$\mathfrak{v}(P_k) := \begin{cases} \mathbf{t} & \text{if } MX_k \neq \emptyset \\ \mathbf{f} & \text{otherwise.} \end{cases}$$

We only need to show that, for each $i = 1, \ldots, m$, there exists a $j \in \{1, 2, 3\}$ such that $\mathfrak{v}(L_{ij}) = \mathbf{t}$. We know that, for $i = 1, \ldots, m$, there exists a $\overline{j} \in \{1, 2, 3\}$ such that $Mu_{i\overline{j}} \neq \emptyset$, provided that $L_{i\overline{j}} = P_k$ for some k. Then $u_{i\overline{j}} = X_k$, so that $MX_k \neq \emptyset$. Hence $\mathfrak{v}(L_{i\overline{j}}) = \mathbf{t}$. On the other hand, if $L_{i\overline{j}} = \neg P_k$ for some k, then $Mu_{i\overline{j}} = M\overline{X}_k \neq \emptyset$, so that, by (12), $MX_k = \emptyset$, $\mathfrak{v}(P_k) = \mathbf{f}$, and therefore $\mathfrak{v}(L_{i\overline{j}}) = \mathbf{t}$. □

We can then conclude that:

Corollary 2.6. The satisfiability problem of $\mathsf{BST}(\cup, \cap, = \varnothing, \neq \varnothing)$ is NP-hard, and therefore is NP-complete.

2.2. Maximal polynomial fragments of $\mathbb{BST}$

The maximal polynomial fragments of $\mathbb{BST}$ are

$\mathsf{BST}(\cup, \cap, \setminus, = \varnothing, \mathsf{Disj}, \subseteq, =)$, $\mathsf{BST}(\cup, \cap, =, \neq \varnothing, \neg\mathsf{Disj}, \subseteq)$,
$\mathsf{BST}(\cup, = \varnothing, \neq \varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$, $\mathsf{BST}(\cap, = \varnothing, \neq \varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$,
$\mathsf{BST}(\cup, = \varnothing, \neq \varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$.

The first two of the above fragments are trivial: in fact, since they contain only satisfiable conjunctions, they admit a $\mathcal{O}(1)$ satisfiability test.

Notice that the first fragment comprises all the positive relators and the complete suite of Boolean operators. It is immediate to check that each of its conjunctions φ is satisfied by the *null* set-assignment $M_\emptyset$ over $Vars(\varphi)$ such that $M_\emptyset x = \emptyset$ for each $x \in Vars(\varphi)$.

Concerning the second fragment, it can easily be verified that each of its conjunctions ψ is satisfied by any *constant nonnull* set assignment M_a over $Vars(\psi)$, where a is a nonempty set and $M_a x = a$ for every $x \in Vars(\psi)$.

Next, we provided $\mathcal{O}(n^3)$ satisfiability tests for the fragments $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj}, \neq)$ and $\mathsf{BST}(\cup, =, \neq)$, and a $\mathcal{O}(n^4)$ satisfiability test for the fragment $\mathsf{BST}(\cap, = \varnothing, =, \neq)$.[7]

Since

- $\models x \not\subseteq y \iff x \cup y \neq y$ (cf. Lemma 1.6(e)) and

[7]Due to space concerns, these complexity results are not proved in this paper: their publication is delayed.

- $\models x = \varnothing \longleftrightarrow \mathsf{Disj}(x, x)$ (cf. Lemma 1.6(f),(g)),

the $\mathcal{O}(n^3)$ satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj}, \neq)$ yields a $\mathcal{O}(n^3)$ satisfiability test for $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$.

In addition, since

- $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cup, =, \neq)$ (cf. Lemma 1.7),
- $x \neq \varnothing$ is expressible in $\mathsf{BST}(=\varnothing, \neq)$ (cf. Lemma 1.6(f)),
- $x \subseteq y$ is expressible in $\mathsf{BST}(\cup, =)$,
- $x \not\subseteq y$ is expressible in $\mathsf{BST}(\cup, \neq)$, and
- $\neg\mathsf{Disj}(x, y)$ is expressible in $\mathsf{BST}(\neq\varnothing, \subseteq)$,

the $\mathcal{O}(n^3)$ satisfiability test for $\mathsf{BST}(\cup, =, \neq)$ yields a $\mathcal{O}(n^3)$ satisfiability test for the fragment $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$.

Finally, since

- $x \neq \varnothing$ is expressible in $\mathsf{BST}(=\varnothing, \neq)$ (cf. Lemma 1.6(f)),
- $\mathsf{Disj}(x, y)$ and $\neg\mathsf{Disj}(x, y)$ are expressible in $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing)$ (cf. Lemma 1.6(h)), and
- $x \subseteq y$ and $x \not\subseteq y$ are expressible in $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing)$ (cf. Lemma 1.6(d),(e)),

it follows that the $\mathcal{O}(n^4)$ satisfiability test for $\mathsf{BST}(\cap, =\varnothing, =, \neq)$ yields a $\mathcal{O}(n^4)$ satisfiability test for the fragment $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$.

It can be checked, through comparison of the symbols appearing in the rows, that:

(A) none of the fragments listed in Table 1 is strictly contained in another fragment in the same table, namely that no fragment in Table 1 comprises all of the symbols of another fragment in the same table; moreover

(B) for every fragment $\mathcal{T}$ of $\mathbb{BST}$, there is a fragment in Table 1 that either contains $\mathcal{T}$ or is contained in $\mathcal{T}$. Thus, any fragment of $\mathbb{BST}$ not appearing in Table 1 is such that either all of its symbols are contained in a polynomial fragment in the table, or it comprises all symbols of an NP-complete fragment in the table.

Properties (A) and (B) imply that the 18 NP-complete fragments in Table 1 are indeed minimally NP-complete and, symmetrically, the 5 polynomial fragments in Table 1 are maximally polynomial.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg\mathsf{Disj}$	$\subseteq$	$\not\subseteq$	$=$	$\neq$	Complexity
$\star$			$\star$	$\star$	$\star$			$\star$		$\star$	$\mathcal{O}(n)$
	$\star$		$\star$	$\star$	$\star$					$\star$	$\mathcal{O}(n^2)$
$\star$					$\star$	$\star$					$\mathcal{O}(n^2)$

Table 2. Three non-maximal polynomial fragments of $\mathbb{BST}$

2.3. Emblematic, non-maximal, polynomial-complexity decision algorithms

While there is a limited interest in further investigating the non-minimal NP-complete fragments of $\mathbb{BST}$, things are not so with the non-maximal polynomial fragments: the latter may, in fact, admit decision tests outperforming any of the maximal polynomial fragments extending them.

We briefly report, below, some preliminary results obtained so far in this direction. Among others, we devised (see Table 2):

- a linear-time decision test for the fragment $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$, which readily generalizes, by Lemma 1.6(e),(f),(g), to a linear-time satisfiability test for the extended fragment $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \not\subseteq, \neq)$;

- a quadratic-time algorithm for the fragment $\mathsf{BST}(\cap, =\varnothing, \neq)$, whence one gets a quadratic-time satisfiability test for the extended fragment $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neq)$, thanks to Lemma 1.6(f),(h);

- a quadratic-time algorithm for the fragment $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$.

2.3.1. A linear-time satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$

Here we provide an account of a linear-time satisfiability test for the fragment $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$.

For convenience, we shall represent terms of the form $x_1 \cup \cdots \cup x_h$ as $\bigcup\{x_1, \ldots, x_h\}$. Thus, for a set assignment M and a finite nonempty collection of set variables $L \subseteq \mathit{Vars}(\varphi)$, we shall have $M(\bigcup L) = \bigcup ML = \bigcup_{x \in L} Mx$. We shall also assume that every formula of $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$ is represented in the format

$$\bigwedge_{i=1}^{p} \bigcup L_i \neq \bigcup R_i \;\wedge\; \bigwedge_{j=p+1}^{q} \mathsf{Disj}(\bigcup L_j, \bigcup R_j)\,, \tag{13}$$

where the L_h's and the R_h's are nonempty finite collections of set variables.[8]

Here all inequality literals of an arbitrary $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-conjunction ψ have been grouped together and isolated from the remaining literals, and it should be clear that this regrouping can be performed in linear time, along a single scan of ψ.

[8]The conjunction will comprise no inequality when $p = 0$, and will comprise no literal of type $\mathsf{Disj}(s, t)$ if $q = p$.

Towards a linear satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$, let φ be a satisfiable conjunction of the form (13), and let M be a set assignment over $Vars(\varphi)$ satisfying φ.

As a preliminary remark note that, for each $x \in \bigcup_{j=p+1}^{q}(L_j \cap R_j)$, the inclusion $Mx \subseteq (\bigcup ML_j) \cap (\bigcup MR_j)$ holds for some $j \in \{p+1, \ldots, q\}$. Hence, since $\mathsf{Disj}(\bigcup L_j, \bigcup R_j)$ is one of the conjuncts in φ, we have $(\bigcup ML_j) \cap (\bigcup MR_j) = \emptyset$, which in turn yields $Mx = \emptyset$.

Next, for each conjunct of type $\bigcup L_i \neq \bigcup R_i$ in φ, if any, we have $\bigcup ML_i \neq \bigcup MR_i$, i.e., $\big((\bigcup ML_i) \cup (\bigcup MR_i)\big) \setminus \big((\bigcup ML_i) \cap (\bigcup MR_i)\big) \neq \emptyset$. Thus, there is an $x \in (L_i \cup R_i) \setminus (L_i \cap R_i)$ such that $Mx \neq \emptyset$, whence $(L_i \cup R_i) \setminus (L_i \cap R_i) \setminus \bigcup_{j=p+1}^{q}(L_j \cap R_j) \neq \emptyset$ follows, by the preceding remark. Summing up, by assuming the satisfiability of φ we have established the following condition:

(C1) $(L_i \cup R_i) \setminus (L_i \cap R_i) \setminus \bigcup_{j=p+1}^{q}(L_j \cap R_j) \neq \emptyset$, for every $i = 1, \ldots, p$.

Conversely, let φ be a $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-conjunction of the form (13) for which the condition (C1) is true, and let $x_1, \ldots, x_k$ be the distinct variables in $Vars(\varphi) \setminus \bigcup_{j=p+1}^{q}(L_j \cap R_j)$. Consider any assignment M^* over $Vars(\varphi)$ such that

(i) $M^*x = \emptyset$, for each $x \in \bigcup_{j=p+1}^{q}(L_j \cap R_j)$, and

(ii) $M^*x_1, \ldots, M^*x_k$ are nonempty pairwise disjoint sets.

Then, it is not hard to check that M^* satisfies φ. Indeed, let $\mathsf{Disj}(\bigcup L_j, \bigcup R_j)$ be a conjunct of φ, and let $s \in \bigcup M^*L_j$. Hence, letting $x \in L_j$ be the set variable such that $s \in M^*x$, by (i) we have $x \notin L_j \cap R_j$, so that $x \notin R_j$. Thus, by (ii), $s \notin \bigcup M^*R_j$. The arbitrariness of s yields $\bigcup M^*L_j \cap \bigcup M^*R_j = \emptyset$, namely $M^* \models \mathsf{Disj}(\bigcup L_j, \bigcup R_j)$. Next, let $\bigcup L_i \neq \bigcup R_i$ be a conjunct of φ, with $i \in \{1, \ldots, p\}$. In view of condition (C1), there exists $x \in (L_i \cup R_i) \setminus (L_i \cap R_i) \setminus \bigcup_{j=p+1}^{q}(L_j \cap R_j) \neq \emptyset$, and w.l.o.g. we may assume that $x \in L_i \setminus R_i$, so that $M^*x \subseteq \bigcup M^*L_i$. From (i) and (ii), we get $M^*x \neq \emptyset$ and $M^*x \cap \bigcup M^*R_i = \emptyset$. Hence, $\bigcup M^*L_i \neq \bigcup M^*R_i$, namely $M^* \models \bigcup L_i \neq \bigcup R_i$.

We have just proved the lemma below, which yields a satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$:

Lemma 2.7. Let φ be a $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-conjunction of the form (13). Then φ is satisfiable if and only if condition (C1) holds.

Concerning the complexity of the satisfiability test implicit in Lemma 2.7, we observe that condition (C1) can be tested in $\mathcal{O}(|\varphi|)$ time, where $|\varphi|$ is the length of the whole conjunction φ since

- the set $\bigcup_{j=p+1}^{q}(L_j \cap R_j)$ can be computed in $\mathcal{O}\big(\sum_{j=p+1}^{q}(|L_j| + |R_j|)\big) = \mathcal{O}(|\varphi|)$ time;[9]
- the set $(L_i \cup R_i) \setminus \bigcup_{j=p+1}^{q}(L_j \cap R_j)$ can be computed in $\mathcal{O}(|L_i| + |R_i|)$ time and tested for emptiness in constant time, for each $i = 1, \ldots, p$; the resulting overall time hence is $\mathcal{O}\big(\sum_{i=1}^{p}(|L_i| + |R_i| + 1)\big) = \mathcal{O}(|\varphi|)$.

Therefore,

[9] $|L_j|$ is the cardinality of the collection of set variables L_j, namely the number of distinct set variables in L_j.

Lemma 2.8. The satisfiability problem for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-conjunctions can be solved in linear time.

2.3.2. A quadratic-time satisfiability test for $\mathsf{BST}(\cap, =\varnothing, \neq)$

The fragment $\mathsf{BST}(\cap, =\varnothing, \neq)$ is the collection of all conjunctions of the general form

$$\bigwedge_{i=1}^{m} \bigcap L_i \neq \bigcap R_i \;\wedge\; \bigwedge_{j=1}^{p} \bigcap D_j = \varnothing, \tag{14}$$

where, as in (13), the L_i's, R_i's and D_j's are *nonempty* finite collections of set variables.

Theorem 2.9. A conjunction φ of the form (14) is satisfiable if and only if the following holds for $i = 1, \ldots, m$:

$$L_i \neq R_i\,; \tag{15}$$

$$\text{there exist no } j, j' \in \{1, \ldots, p\} \text{ such that } D_j \subseteq L_i \;\wedge\; D_{j'} \subseteq R_i\,. \tag{16}$$

Proof:
(Necessity.) Let φ be a satisfiable conjunction of the form (14) and let M be a model of φ. Plainly no $i \in \{1, \ldots, m\}$ can violate condition (15), else $\bigcap L_i = \bigcap R_i$. Arguing by contradiction, suppose that an i violating condition (16) exists, so that there exist indices j, j' such that $D_j \subseteq L_i$ and $D_{j'} \subseteq R_i$. Since $M \models \varphi$, we have $\bigcap MD_j = \bigcap MD_{j'} = \emptyset$ and, accordingly,

$$\bigcap ML_i \subseteq \bigcap MD_j \quad \therefore \quad \bigcap ML_i = \emptyset\,;$$

likewise, $\bigcap MR_i = \emptyset$. However, this contradicts the consequence $\bigcap ML_i \neq \bigcap MR_i$ of $M \models \varphi$.

(Sufficiency.) Take $2^{|\mathit{Vars}(\varphi)|}$ disjoint sets v_S with $S \subseteq \mathit{Vars}(\varphi)$, where $|\mathit{Vars}(\varphi)|$ is the number of distinct set variables appearing in φ. Then put

$$Mx \coloneqq \{\, v_S \mid x \in S \wedge D_1 \nsubseteq S \wedge \cdots \wedge D_p \nsubseteq S \,\} \tag{17}$$

for each set variable x, so that

$$\bigcap MV = \{\, v_S \mid V \subseteq S \wedge D_1 \nsubseteq S \wedge \cdots \wedge D_p \nsubseteq S \,\} \tag{18}$$

plainly holds for every collection $V \subseteq \mathit{Vars}(\varphi)$. We will now show that if φ satisfies both (15) and (16) then M is a model for φ.

As regards conjuncts of type $\bigcap D_j = \varnothing$ in φ, we get

$$\bigcap MD_j = \{\, v_S \mid D_j \subseteq S \wedge D_1 \nsubseteq S \wedge \cdots \wedge D_p \nsubseteq S \,\}$$

from (18). Since $j \in \{1, \ldots, p\}$, this set is empty; therefore $\bigcap MD_j = \emptyset$.

Concerning conjuncts of type $\bigcap L_i \neq \bigcap R_i$ in φ, we get from (15) that either $L_i \nsubseteq R_i$ or $R_i \nsubseteq L_i$ holds. For definiteness assume that $L_i \nsubseteq R_i$ (the treatment of the other case being symmetrical), so that $v_{R_i} \notin \bigcap ML_i$. If $D_1 \nsubseteq R_i \wedge \cdots \wedge D_p \nsubseteq R_i$, then, by (18), $v_{R_i} \in \bigcap MR_i$, which proves $\bigcap ML_i \neq \bigcap MR_i$. Otherwise, $D_{j'} \subseteq R_i$ holds for some j' such that $1 \leqslant j' \leqslant p$, and hence $\bigcap MR_i = \emptyset$ (for, $R_{j'} \subseteq S$ implies $D_{j'} \subseteq S$) and, by condition (16), $D_j \nsubseteq L_i$ holds for $j = 1, \ldots, p$; thus, plainly, $v_{L_i} \in \bigcap ML_i$ holds, again proving $\bigcap ML_i \neq \bigcap MR_i$. □

Satisfiability Test

Theorem 2.9 states that, in order to establish whether or not a $\mathsf{BST}(\cap, =\varnothing, \neq)$-conjunction is satisfiable, it suffices to check whether the conjunction satisfies both conditions (15) and (16). Through the analysis of Algorithm 1, we will now show that the task of performing that check can be implemented to run in quadratic time.

Algorithm 1 Satisfiability test for $\mathsf{BST}(\cap, =\varnothing, \neq)$-conjunctions

Require: A conjunction φ of the form (14).
Ensure: Is φ satisfiable?

```
1:  for i ∈ {1,...,m} do
2:      if L_i = R_i then
3:          return "unsatisfiable"
4:  for i ∈ {1,...,m} do
5:      B_L := B_R := false
6:      for j ∈ {1,...,p} do
7:          if D_j ⊆ L_i then
8:              B_L := true
9:          if D_j ⊆ R_i then
10:             B_R := true
11:     if B_L ∧ B_R then
12:         return "unsatisfiable"
13: return "satisfiable"
```

To analyze the complexity of Algorithm 1, it is convenient to first analyze Algorithm 2.

It is easy to show that Algorithm 2 returns ***true*** if and only if $S \subseteq T$, by noticing that if $S \nsubseteq T$ then there is an element $s \in S$ such that $\mathcal{A}[s] = 0$ holds at line 6, so that from that point on c will always equal 0 and the algorithm will output ***false***.

As far as we are concerned, S, T will be collections of set variables, any two of which, $x, y \in S \cup T$, are treated as distinct regardless of their values. Moreover, Algorithm 2 trivially runs in $\mathcal{O}(|S| + |T|)$ time, and since its execution leaves the array $\mathcal{A}$ unchanged at the end, we can call this algorithm multiple times using the same array repeatedly: in particular, we can call the algorithm twice to check whether or not $S = T$ holds in $\mathcal{O}(|S| + |T|)$ time.

Concerning the complexity of Algorithm 1, indicate by n the length $\sum_{i=1}^{m}(|L_i| + |R_i|) + \sum_{j=1}^{p}|D_j|$ of the conjunction. By using Algorithm 2 where the array $\mathcal{A}$ has size $|Vars(\varphi)|$ and is indexed by the set variables in the conjunction, the **for**-loop at lines 1–3 runs in

Algorithm 2 Subset check

Require: Two sets, S and T.
Ensure: Is $S \subseteq T$?

```
1: Let 𝒜 be an array of size |S ∪ T|, indexed by the elements of S ∪ T, containing only 0's
2: c ≔ 1
3: for t ∈ T do
4:     𝒜[t] ≔ 1
5: for s ∈ S do
6:     c ≔ 𝒜[s] · c
7: for t ∈ T do
8:     𝒜[t] ≔ 0
9: if c = 1 then
10:    return true
11: return false
```

$\mathcal{O}\big(\sum_{i=1}^{m}(|L_i|+|R_i|)\big)$ time and the **for**-loop at lines 4–12 runs in $\mathcal{O}\big(\sum_{i=1}^{m}\sum_{j=1}^{p}(|D_j|+|L_i|+|R_i|)\big) = \mathcal{O}\big(p\sum_{i=1}^{m}(|L_i|+|R_i|)+m\sum_{j=1}^{p}D_j\big)$ time. Finally, by noticing that the array can be built and initialized in $\mathcal{O}(n)$ time and that $m < n$ and $p < n$, we conclude that Algorithm 1 runs in $\mathcal{O}(n^2)$ time, and hence:

Lemma 2.10. The satisfiability problem for $\mathsf{BST}(\cap, =\varnothing, \neq)$ can be solved in quadratic time.

2.3.3. A quadratic-time satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$

The fragment $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$ consists of all conjunctions of literals of the two forms:

$$\mathsf{Disj}(\bigcup L, \bigcup R)\ , \quad \neg\mathsf{Disj}(\bigcup L, \bigcup R)\,,$$

where L, R stand for finite nonempty collections of set variables. The form of any φ in this fragment hence is

$$\bigwedge_{i=1}^{p} \mathsf{Disj}(\bigcup L_i, \bigcup R_i) \ \wedge \bigwedge_{j=p+1}^{m} \neg\mathsf{Disj}(\bigcup L_j, \bigcup R_j)\,. \tag{19}$$

We denote by $\otimes$ the following operation akin to Cartesian product:

$$S \otimes T \coloneqq \big\{\, \{s\,,\ t\} \,|\, s \in S\,,\ t \in T \,\big\}\,. \tag{20}$$

Let us associate with such a φ the collection $\mathcal{C} \subseteq \mathit{Vars}(\varphi) \otimes \mathit{Vars}(\varphi)$ of set-variable pairs to which a model can assign intersecting sets, namely

$$\mathcal{C} \coloneqq \big(\mathit{Vars}(\varphi) \otimes \mathit{Vars}(\varphi)\big) \setminus \bigcup_{i=1}^{p}(L_i \otimes R_i)\,. \tag{21}$$

Theorem 2.11. Let φ be a conjunction of the form (19). Then φ is satisfiable if and only if

$$(\forall j \in \{p+1, \ldots, m\})\ (L_j \otimes R_j) \cap \mathcal{C} \neq \emptyset. \tag{22}$$

Proof:
(Necessity.) Let φ be a satisfiable $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$ conjunction, and let M be a model of φ. By way of contradiction, assume that (22) does not hold, so that

$$(L_j \otimes R_j) \cap \mathcal{C} = \emptyset. \tag{23}$$

holds for a suitable $j \in \{p+1, \ldots, m\}$. Since $M \models \varphi$, for such a j we have

$$(\exists u \in L_j\,,\ t \in R_j)\ Mu \cap Mt \neq \emptyset.$$

The pair $\{u\,,\ t\}$ belongs to $L_j \otimes R_j$ and so, by (23), $\{u\,,\ t\} \notin \mathcal{C}$; therefore, by (21):

$$(\exists i \in \{1, \ldots, p\})\ \{u, t\} \in L_i \otimes R_i.$$

So $\bigcup ML_i \cap \bigcup MR_i \neq \emptyset$, contradicting $M \models \mathsf{Disj}(\bigcup L_i, \bigcup R_i)$. From this contradiction we conclude that if φ is satisfiable then condition (22) must hold.

(Sufficiency.) For the sufficiency part of the theorem, we associate a set b_c with each $c \in \mathcal{C}$ so that the b_c's are pairwise disjoint; then we define the following set assignment:

$$Mx \coloneqq \big\{b_c \mid x \in c \in \mathcal{C}\big\}.$$

We will show that if φ satisfies condition (22), then $M \models \varphi$.

First we show that M satisfies literals of type $\neg\mathsf{Disj}(L_j, R_j)$. By condition (22):

$$\begin{array}{ll}
(\exists u \in L_j\,,\ t \in R_j)\ \{u\,,\ t\} \in \mathcal{C} & \therefore \\
b_{\{u,t\}} \in Mu \subseteq \bigcup ML_j \ \wedge\ b_{\{u,t\}} \in Mt \subseteq \bigcup MR_j & \therefore \\
b_{\{u,t\}} \in \bigcup ML_j \cap \bigcup MR_j, &
\end{array}$$

hence $M \models \neg\mathsf{Disj}(\bigcup L_j, \bigcup R_j)$.

Next, concerning literals of type $\mathsf{Disj}(\bigcup L_i, \bigcup R_i)$, arguing by contradiction suppose

$$\begin{array}{ll}
M \not\models \mathsf{Disj}(\bigcup L_i, \bigcup R_i)\,, & \text{so that} \\
(\exists\, b_c\,,\ c \in \mathcal{C})\ b_c \in \bigcup ML_i \cap \bigcup MR_i & \therefore \\
(\exists\, l \in L_i\,,\ r \in R_i)\ b_c \in Ml \cap Mr & \therefore \\
c = \{l, r\} \in L_i \otimes R_i & \therefore \\
c \notin \mathcal{C}, &
\end{array}$$

a contradiction. Therefore we conclude that $M \models \mathsf{Disj}(\bigcup L_i, \bigcup R_i)$, ending our proof. □

Satisfiability test

We now prove that Algorithm 3 is a valid satisfiability tester for $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$ conjunctions.

First, $\mathcal{M}$ is initialized as being the matrix of size $|\mathit{Vars}(\varphi)|^2$ each of whose entries has value 1; then the **for**-loop at lines 2–5 sets each entry of $\mathcal{M}$ addressed by a pair in $\bigcup_{i=1}^{p}(L_i \otimes R_i)$ to zero; at this point, the value of each entry of $\mathcal{M}$ turns out to be 1 if it is addressed by a pair in $\mathcal{C}$, and to be 0 otherwise.

Finally, at line 11 the value of c will be found to be 0 for some $j \in \{p+1, \ldots, m\}$ if and only if no pairs in $L_j \otimes R_j$ also belong to $\mathcal{C}$. By Theorem 2.11, this situation eventually arises if and only if the conjunction φ is unsatisfiable; hence the algorithm is correct.

Algorithm 3 Satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$

Require: A $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$-formula φ.
Ensure: Is φ satisfiable?

```
 1: Initialize a matrix M of size |Vars(φ)|^2 each of whose entries has value 1
 2: for i ∈ {1, ..., p} do
 3:     for l ∈ L_i do
 4:         for r ∈ R_i do
 5:             M[l][r] := 0
 6: for j ∈ {p + 1, ..., m} do
 7:     c := 0
 8:     for l ∈ L_j do
 9:         for r ∈ R_j do
10:             c := c + M[l][r]
11:     if c = 0 then
12:         return false
13: return true
```

Concerning the complexity of Algorithm 3, let φ be a conjunction of the form (19), and let $n = \sum_{i=1}^{m} \big(|L_i| + |R_i|\big)$. We prove that Algorithm 3 has complexity $\mathcal{O}(n^2)$.

Plainly, initializing matrix $\mathcal{M}$ requires $\mathcal{O}\big(|\mathit{Vars}(\varphi)|^2\big)$ time. Then, each iteration of the **for**-loop at lines 2–5 sets to zero exactly $|L_i|\,|R_i|$ entries of $\mathcal{M}$; the **for**-loop is iterated for each $i \in \{1, \ldots, p\}$ and is accomplished in $\mathcal{O}\big(\sum_{i=1}^{p} |L_i|\,|R_i|\big)$ time. Analogously, the **for**-loop at lines 6–12 is accomplished in $\mathcal{O}\big(\sum_{j=p+1}^{m} |L_j|\,|R_j|\big)$ time.

The overall complexity of Algorithm 3 hence is $\mathcal{O}\big(|\mathit{Vars}(\varphi)|^2 + \sum_{i=1}^{m} |L_i|\,|R_i|\big)$ and, since it is easily proved that $\sum_{i=1}^{m} |L_i|\,|R_i| \leqslant \big(\sum_{i=1}^{m}(|L_i| + |R_i|)\big)^2 = n^2$ and, moreover, the length of φ cannot be exceeded by the number of distinct variables in it, viz. $|\mathit{Vars}(\varphi)| \leqslant n$, we get an overall complexity $\mathcal{O}(n^2)$. It readily follows that:

Lemma 2.12. The satisfiability problem for $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$ is solvable in quadratic time.

2.4. Strength-reduction examples

The results summarized in Table 1 and in Lemma 1.6 could drive a proof-assistant in systematically choosing from a portfolio of decision algorithms one that ensures a fast route to a needed syllogism. A few examples of this follow.

Example 2.13. An automated prover, confronted with the task of checking that the implication

$$x = y \setminus x \longrightarrow y = \varnothing$$

is valid, must prove the unsatisfiability of the conjunction

$$x = y \setminus x \ \wedge \ y \neq \varnothing.$$

By indicating that the satisfiability problems for $\mathsf{BST}(\setminus, =, \neq)$ and for $\mathsf{BST}(\cap, \mathsf{Disj}, =, \neq)$ are, respectively, NP-complete and polynomially solvable, Table 1 suggests reducing instances of the former problem to the latter, when possible.

This is feasible in our case: in fact, by Lemma 1.6(a), $x = y \setminus x$ reduces to $x \cap x = \varnothing \ \wedge \ y \subseteq x \cup x$ and thus, by Lemma 1.6(d), the goal becomes the one of showing the unsatisfiability of

$$\mathsf{Disj}(x, x) \ \wedge \ x \cap y = y \ \wedge \ y \neq x \cap x.$$

Example 2.14. In order to verify that

$$\mathscr{P}(A) \cup \mathscr{P}(B) \subseteq \mathscr{P}(A \cup B),$$

an interactive formal prover could resort to the satisfiability problem for MLSP, which is known to be decidable [13, 14], but which is intractable in practice by means of the today known decision algorithms. To ease the task of establishing that the formula

$$\mathscr{P}(A) \cup \mathscr{P}(B) \nsubseteq \mathscr{P}(A \cup B)$$

is unsatisfiable, a human intervention could suggest the equivalence

$$\mathscr{P}(A) \cup \mathscr{P}(B) \nsubseteq \mathscr{P}(A \cup B) \longleftrightarrow (\exists x)\big((x \subseteq A \vee x \subseteq B) \ \wedge \ x \nsubseteq A \cup B\big)$$

to the prover, so that it could reach the goal by simply analyzing the conjunctions:

$$x \subseteq A \ \wedge \ x \nsubseteq A \cup B,$$
$$x \subseteq B \ \wedge \ x \nsubseteq A \cup B;$$

both of these easily turn out to be unsatisfiable when tested by the satisfiability algorithm for $\mathsf{BST}(\cup, \subseteq, \nsubseteq)$, which is polynomial.

Example 2.15. Distributivity of union over intersection amounts to the unsatisfiability of

$$(A \cap B) \cup C \neq (A \cup C) \cap (B \cup C).$$

At first view, this is an instance of the satisfiability problem for $\mathsf{BST}(\cup, \cap, \neq)$, which is NP-complete. Alternatively, the issue can be split into two. On the one hand, to prove that

$$(A \cup C) \cap (B \cup C) \nsubseteq (A \cap B) \cup C,$$

is unsatisfiable, we reduce this issue to the unsatisfiability of the conjunction

$$S \neq \varnothing \ \wedge \ S \subseteq A \cup C \ \wedge \ S \subseteq B \cup C \ \wedge \ \mathsf{Disj}((A \cap B) \cup C, S)\,,$$

which implies

$$\neg\mathsf{Disj}(S, A \cup C) \ \wedge \neg\mathsf{Disj}(S, B \cup C) \ \wedge \ \mathsf{Disj}(S, C) \wedge \ (\mathsf{Disj}(A, S) \ \vee \ \mathsf{Disj}(B, S));$$

this can be proved unsatisfiable through two applications of the satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj})$, which is polynomial. On the other hand, the unsatisfiability of

$$(A \cap B) \cup C \nsubseteq (A \cup C) \cap (B \cup C),$$

is equivalent to the unsatisfiability of the conjunction

$$S \neq \varnothing \ \wedge \ S \subseteq (A \cap B) \cup C \ \wedge \ \big(\mathsf{Disj}(S, A \cup C) \vee \mathsf{Disj}(S, B \cup C)\big),$$

which implies

$$\big(\neg\mathsf{Disj}(S, A \cap B) \vee \neg\mathsf{Disj}(S, C)\big) \wedge \mathsf{Disj}(S, C) \wedge \mathsf{Disj}(S, A \cap B)\,;$$

this can be proved unsatisfiable through two applications of the satisfiability test for $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing)$, which is polynomial.

3. Related work and analogous taxonomies

3.1. Boolean set theory versus membership set theory

This section offers a quick summary of the complexity taxonomy for a fragment,

$$\mathbb{MST} := \mathsf{MST}(\cup, \cap, \setminus, \in, \notin),$$

of set theory constructed in close analogy with $\mathbb{BST}$.[10] It also reports on an algorithm performing a satisfiability-preserving translation of $\mathbb{MST}$ into a purely Boolean language.

[10] As recalled in the Introduction, by adding the equality relator $=$ alone (or, respectively, the singleton operator $\{\cdot\}$ along with $=$) to the constructs of $\mathbb{MST}$—whose acronym stands for 'membership set theory'—, one moves into the theory MLS (or, resp., into MLSS). Note that, thanks to the equivalences

$$\begin{array}{rcl} x = y \cap z & \longleftrightarrow & x = y \setminus (y \setminus z)\,, \\ x \neq y & \longleftrightarrow & \exists\, w\,(\, x \in w \ \wedge \ y \notin w\,)\,, \end{array} \qquad \begin{array}{rcl} x = y \cup z & \longleftrightarrow & (x \setminus y = z \setminus y \ \wedge \ y \setminus x = x \setminus x)\,, \\ y \notin w & \longleftrightarrow & \exists\, v\,(\, y \in v \ \wedge \ w \cap v = v \setminus v\,)\,, \\ x \in w & \longleftrightarrow & \{x\} \cap w = \{x\}\,, \end{array}$$

$\cap$, $\cup$ are eliminable from MLS; so are $\in, \notin$ from MLSS; and $\neq, \notin$ are expressible in terms of $=, \in$, and $\setminus$.

As discussed in [23, 18], the number of distinct fragments of $\mathbb{MST}$ is 24: of those, 10 are NP-complete and 14 are polynomial. Much as for $\mathbb{BST}$, the complexity of any fragment of $\mathbb{MST}$ can be readily determined once the *minimal* NP-complete fragments and the *maximal* polynomial fragments have been singled out. Table 3 shows the minimal NP-complete and the maximal polynomial fragments of $\mathbb{MST}$.

$\cup$	$\cap$	$\setminus$	$\in$	$\notin$	Complexity
		$\star$	$\star$		NP-complete
$\star$	$\star$		$\star$		NP-complete
	$\star$		$\star$	$\star$	$\mathcal{O}(n^2)$
$\star$			$\star$	$\star$	$\mathcal{O}(n)$
$\star$	$\star$	$\star$		$\star$	$\mathcal{O}(1)$

Table 3. Maximal polynomial and minimal NP-complete fragments of $\mathbb{MST}$

The maximal polynomial fragment $\mathsf{MST}(\cup, \cap, \setminus, \notin)$ trivially admits a constant time complexity, because all conjunctions φ composing it are satisfiable: indeed, any of them is satisfied by the *null* set assignment sending every $x \in \mathit{Vars}(\varphi)$ to $\emptyset$.

In order to get a $\mathcal{O}(n)$ satisfiability test for $\mathsf{MST}(\cup, \in, \notin)$, we first provided a $\mathcal{O}(n)$ satisfiability test for the fragment $\mathsf{MST}(\cup, \in)$ and then managed to translate every $\mathsf{MST}(\cup, \in, \notin)$-conjunction into an equisatisfiable $\mathsf{MST}(\cup, \in)$-conjunction in $\mathcal{O}(n)$ time.

We also provided a $\mathcal{O}(n^2)$-time satisfiability test for the fragment $\mathsf{MST}(\cap, \in)$, which was extended into a satisfiability test for $\mathsf{MST}(\cap, \in, \notin)$ that retains the same complexity.

Both of the fragments $\mathsf{MST}(\cup, \cap, \in)$ and $\mathsf{MST}(\setminus, \in)$ were shown to be NP-complete; resorting to the axiom of regularity was necessary to prove the NP-completeness of the former, while regularity was not needed to prove the NP-completeness of the latter.

3.2. Bridging Boolean set theory and membership set theory

In [19], the authors have proposed a quadratic-time translation of conjunctions of literals of the forms $x = y \setminus z$, $x \neq y$, and $x \in y$, where x, y, z stand for variables ranging over the von Neumann universe, into unquantified Boolean formulae of a streamlined conjunctive normal form. The formulae in the target language involve variables ranging over a Boolean field of sets, along with a difference operator and relators designating equality, non-disjointness and inclusion; the result of each translation is a conjunction of literals of the forms $x = y \setminus z$, $x \neq y$ and of implications whose antecedents are isolated literals and whose consequents are either inclusions $x \subseteq y$ and $x \subsetneq y$ between variables, or equalities between variables.

3.3. Novelties with respect to a prior conference paper

This paper is an extended version of [24], by the same authors;[11] hence we have the duty to indicate what are the novelties here.

1. The proof of Proposition A.3, preparatory to Lemma 1.7, is now shown—see below.
2. The most challenging NP-completeness proofs regarding fragments of $\mathbb{BST}$ are provided in Sections 2.1.1–2.1.3.
3. A new non-maximal, polynomial-complexity decision algorithm has been pointed out (third row of Table 2), and the complexity of the decision algorithm for $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neq)$ (second row of Table 2) has been lowered from cubic to quadratic.
4. The decision algorithms of two quadratic-time satisfiability tests are presented in Sections 2.3.2 and 2.3.3.
5. Various examples have been introduced—see in particular Sect. 2.4.

3.4. Other complexity taxonomies for decidable logics

Several authors have developed complexity taxonomies of decidable fragments, delimited syntactically, of various logical systems. Among precursors of this approach, we must cite the landmark paper [25], where Harry R. Lewis analyzed the computational complexity of determining satisfiability for prenex sentences belonging to certain fragments of the classical predicate calculus. A complexity classification, concerning the point-based temporal logics LTL, CTL, and CTL*, appears in [26], which singles out very low complexity fragments. Quite recently, regarding model-checking in the framework of Halpern and Shoham's modal logic of time intervals, we have [27, 28].

4. Conclusion and future work

We highlighted some preliminary results of an investigation aimed at identifying small fragments of set theory endowed with polynomial-time satisfiability decision tests, potentially useful for automated proof verification and, more generally, in the symbolic manipulation of declarative specifications (cf., e.g., [29, 30, 31]). In this initial phase, we mainly focused on 'Boolean Set Theory', namely the fragment of quantifier-free formulae of set theory involving variables, the Boolean set operators $\cup, \cap, \setminus$, the Boolean relators $\subseteq, \not\subseteq, =, \neq$, and the predicates '$\cdot =\varnothing$' and '$\mathsf{Disj}(\cdot, \cdot)$', along with their opposites.

Future work will concentrate on the analysis of the sub-maximal polynomial fragments of $\mathbb{BST}$: as Sect. 2.3 illustrates, a finer complexity taxonomy of the collection of all $\mathbb{BST}$ fragments will thus emerge. We also intend to enrich the endowment of set operators and relators of

[11] It is also closely related to [23, 18], as discussed in Sect. 3.1.

$\mathbb{BST}$, e.g. with the *symmetric difference* operator $\triangle$ such that $\models x \triangle y = (x \setminus y) \cup (y \setminus x)$. We also plan to deepen the study of membership fragments akin to those surveyed in Sect. 3.1, and ultimately to analyze, from the complexity standpoint, situations arising from the interplay among constructs related to membership and Boolean constructs.

We envisage a confluence of the line of research centered on satisfiability testers, to which this paper and its companion [18] contribute, with another active line of research centered on set-unification algorithms (for an up-to-date survey on those, see [32]). In various customary situations, in fact, a quick satisfiability tester is less helpful than a solver able to produce a symbolic solution of maximum possible generality, or a bunch of solution templates which cover, collectively, all possible models of a given formula. E.g., the Büttner–Simonis unification algorithm for Boolean algebras [33] produces, when a model exists, a most general unifier; its application realm does border the fragments addressed in this paper (cf. [3, pp. 64–66 and pp. 168–179]). An example in which a most general unifier does not exist—but nevertheless all models are covered—is the set-unification algorithm presented in [34] (see also [3, pp. 256–259]), wherein membership can be expressed in terms of the *adjunction* operator $\{\cdot \mid \cdot\}$ such that $\models \{x \mid y\} = y \cup \{x\}$ and hence $\models x \in y \longleftrightarrow \{x \mid y\} = y$; the application realm of this unification algorithm is contiguous to the fragments addressed in [23, 18].

Acknowledgements

We are grateful to the anonymous referees for their valuable comments and suggestions.

References

[1] Schwartz JT. Instantiation and decision procedures for certain classes of quantified set-theoretic formulae. Technical Report 78-10, NASA Langley Research Center, Hampton, Virginia, 1978.

[2] Cantone D, Ferro A, Omodeo EG. Computable set theory. Number 6 in International Series of Monographs on Computer Science, Oxford Science Publications. Clarendon Press, Oxford, UK, 1989.

[3] Cantone D, Omodeo EG, Policriti A. Set theory for computing - From decision procedures to declarative programming with sets. Monographs in Computer Science. Springer-Verlag, New York, 2001.

[4] Schwartz JT, Cantone D, Omodeo EG. Computational logic and set theory: Applying formalized logic to analysis. Springer-Verlag, 2011. ISBN 978-0-85729-807-2. doi:10.1007/978-0-85729-808-9. Foreword by M. Davis.

[5] Omodeo EG, Policriti A, Tomescu AI. On Sets and Graphs: Perspectives on Logic and Combinatorics. Springer, 2017. ISBN 978-3-319-54981-1. URL `https://link.springer.com/book/10.1007%2F978-3-319-54981-1`.

[6] Cantone D, Ursino P. An Introduction to the Technique of Formative Processes in Set Theory. Springer International Publishing, 2018. ISBN 978-3-319-74778-1. doi:10.1007/978-3-319-74778-1.

[7] Ferro A, Omodeo EG, Schwartz JT. Decision procedures for some fragments of Set Theory. In: Proceedings of CADE-5, Les Arcs, France, July 8–11, 1980, volume 87 of *LNCS*. Springer-Verlag, 1980 pp. 88–96.

[8] Ferro A, Omodeo EG, Schwartz JT. Decision Procedures for Elementary Sublanguages of Set Theory. I: Multilevel Syllogistic and Some Extensions. *Commun. Pur. Appl. Math.*, 1980. **33**:599–608.

[9] Breban M, Ferro A, Omodeo E, Schwartz J. Decision Procedures for Elementary Sublanguages of Set Theory. II: Formulas involving Restricted Quantifiers, together with Ordinal, Integer, Map, and Domain Notions. *Commun. Pur. Appl. Math.*, 1981. **34**:177–195.

[10] Cantone D, Cutello V, Policriti A. Set-theoretic reductions of Hilbert's Tenth Problem. In: Börger E, Büning HK, Richter MM (eds.), CSL '89, 3rd Workshop on Computer Science Logic, Kaiserslautern, Germany, October 2-6, 1989, Proceedings, volume 440 of *LNCS*. Springer, 1990 pp. 65–75.

[11] Cantone D, Omodeo EG, Panettiere M. From Hilbert's 10^{th} problem to slim, undecidable fragments of set theory. In: Cordasco G, Gargano L, Rescigno A (eds.), Proceedings of the 21st Italian Conference on Theoretical Computer Science, ICTCS 2020, Ischia, Italy, September 14-16, 2020, CEUR Workshop Proceedings. CEUR-WS.org, To appear.

[12] Cantone D, Omodeo EG, Policriti A. The Automation of Syllogistic. II: Optimization and Complexity Issues. *J. Automat. Reasoning*, 1990. **6**(2):173–187.

[13] Cantone D. Decision procedures for elementary sublanguages of Set Theory. X. Multilevel syllogistic extended by the singleton and powerset operators. *J. Automat. Reasoning*, 1991. **7**(2):193–230.

[14] Cantone D, Omodeo EG, Ursino P. Formative processes with applications to the decision problem in set theory. I: Powerset and singleton operators. *Inform. Comput.*, 2002. **172**(2):165–201. doi: 10.1006/inco.2001.3096.

[15] Cantone D, Ursino P. Formative processes with applications to the decision problem in set theory. II: Powerset and singleton operators, finiteness predicate. *Inform. Comput.*, 2014. **237**:215–242.

[16] Cantone D, Zarba CG. A new fast tableau-based decision procedure for an unquantified fragment of set theory. In: Caferra R, Salzer G (eds.), Automated Deduction in Classical and Non-Classical Logics, volume 1761 of *LNAI*, pp. 127–137. Springer-Verlag, 2000.

[17] Omodeo EG, Tomescu AI. Set graphs. III. Proof Pearl: Claw-free graphs mirrored into transitive hereditarily finite sets. *J. Automat. Reasoning*, 2014. **52**(1):1–29.

[18] Cantone D, Maugeri P, Omodeo EG. Complexity assessments for decidable fragments of set theory. II: A taxonomy for 'small' languages involving membership. *Theoretical Computer Science*, 2020. doi:https://doi.org/10.1016/j.tcs.2020.08.023. URL `http://www.sciencedirect.com/science/article/pii/S0304397520304825`.

[19] Cantone D, De Domenico A, Maugeri P, Omodeo EG. A quadratic reduction of constraints over nested sets to purely Boolean formulae in CNF. In: Calimeri F, Perri S, Zumpano E (eds.), Proceedings of the 35th Italian Conference on Computational Logic, Rende, Italy, October 13-15, 2020, volume 2710 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020 pp. 214–230. ISSN 1613-0073.

[20] Jacobson N. Lectures in Abstract Algebra, Vol.1 – Basic Concepts. D. Van Nostrand, New York, 1951.

[21] Manin YI. A course in mathematical logic. Graduate texts in Mathematics. Springer-Verlag, 1977.

[22] Garey MR, Johnson DS. Computers and Intractability: A Guide to the Theory of NP-Completeness. Series of Books in the Mathematical Sciences. W. H. Freeman, 1979. ISBN 0716710455.

[23] Cantone D, Maugeri P. Polynomial-Time Satisfiability Tests for 'Small' Membership Theories. In: Cherubini A, Sabadini N, Tini S (eds.), Proceedings of the 20th Italian Conference on Theoretical Computer Science, ICTCS 2019, Como, Italy, September 9-11, 2019, volume 2504 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019 pp. 261–273. URL `http://ceur-ws.org/Vol-2504/paper29.pdf`.

[24] Cantone D, Domenico AD, Maugeri P, Omodeo EG. Polynomial-time satisfiability tests for Boolean fragments of Set Theory. In: Casagrande A, Omodeo EG (eds.), Proceedings of the 34th Italian Conference on Computational Logic, Trieste, Italy, June 19-21, 2019, volume 2396 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019 pp. 123–137. URL `http://ceur-ws.org/Vol-2396/paper22.pdf`.

[25] Lewis HR. Complexity results for classes of quantificational formulas. *J. Comput. Syst. Sci.*, 1980. **21**(3):317–353. URL `https://doi.org/10.1016/0022-0000(80)90027-6`.

[26] Meier A, Thomas M, Vollmer H, Mundhenk M. The complexity of satisfiability for fragments of CTL and CTL*. *Int. J. Found. Comput. Sci.*, 2009. **20**(5):901–918. URL `https://doi.org/10.1142/S0129054109006954`.

[27] Bozzelli L, Molinari A, Montanari A, Peron A, Sala P. Which fragments of the interval temporal logic HS are tractable in model checking? *Theor. Comput. Sci.*, 2019. **764**:125–144. URL `https://doi.org/10.1016/j.tcs.2018.04.011`.

[28] Bozzelli L, Molinari A, Montanari A, Peron A, Sala P. Model checking for fragments of the interval temporal logic HS at the low levels of the polynomial time hierarchy. *Inf. Comput.*, 2018. **262**(Part):241–264. URL `https://doi.org/10.1016/j.ic.2018.09.006`.

[29] Stolzenburg F. Membership-Constraints and Complexity in Logic Programming with Sets. In: Baader F, Schulz KU (eds.), Frontiers of Combining Systems, First International Workshop FroCoS 1996, Munich, Germany, March 26-29, 1996, Proceedings, volume 3 of *Applied Logic Series*. Kluwer Academic Publishers, 1996 pp. 285–302.

[30] Dovier A, Piazza C, Rossi G. A uniform approach to constraint-solving for lists, multisets, compact lists, and sets. *ACM Trans. Comput. Log.*, 2008. **9**(3):15:1–15:30. URL `https://doi.org/10.1145/1352582.1352583`.

[31] Cristiá M, Rossi G. Solving Quantifier-Free First-Order Constraints Over Finite Sets and Binary Relations. *J. Autom. Reason.*, 2020. **64**(2):295–330. URL `https://doi.org/10.1007/s10817-019-09520-4`.

[32] Dovier A, Pontelli E, Rossi G. Set unification. *Theor. Pract. Log. Prog.*, 2006. **6**(6):645–701. URL `https://doi.org/10.1017/S1471068406002730`.

[33] Büttner W, Simonis H. Embedding Boolean Expressions into Logic Programming. *J. Symb. Comput.*, 1987. **4**(2):191–205. URL `https://doi.org/10.1016/S0747-7171(87)80065-2`.

[34] Dovier A, Omodeo EG, Pontelli E, Rossi GF. A Language for Programming in Logic with Finite Sets. *J. Logic Program.*, 1996. **28**(1):1–44.

A. Proofs of three major lemmas

This appendix provides proofs of the Lemmas 1.6 and 1.7.

Proof of Lemma 1.6:

(a) $\models x = y \setminus z \longleftrightarrow (x \cap z = \varnothing \ \wedge \ x \cup z = y \cup z)$.

(b) $\models x = y \cup z \longleftrightarrow \big(x \setminus y = z \setminus y \ \wedge \ y \setminus x = x \setminus x\big)$ and $\models x = y \cap z \longleftrightarrow x = y \setminus (y \setminus z)$.

(c) $\models x = y \longleftrightarrow x \subseteq y \ \wedge y \subseteq x$.

(d) $\models x \subseteq y \longleftrightarrow x \cup y = y$ and $\models x \subseteq y \longleftrightarrow x \cap y = x$

(e) $\models x \not\subseteq y \longleftrightarrow x \cup y \neq y$ and $\models x \not\subseteq y \longleftrightarrow x \cap y \neq x$

(f) We have that:

- $\models x \neq \varnothing \longleftrightarrow (\exists y, z)(y \subseteq x \ \wedge \ z \subseteq x \ \wedge \ z \neq y)$,
- from the latter and (c), it follows that $x \neq \varnothing$ is also expressible in $\mathsf{BST}(\cup, =, \neq)$,
- $\models x \neq \varnothing \longleftrightarrow (\exists y)(x \not\subseteq y)$,
- $\models x \neq \varnothing \longleftrightarrow (\exists y)(x \neq y \ \wedge \ \mathsf{Disj}(y, y))$,
- $\models x \neq \varnothing \longleftrightarrow (\exists y)(x \neq y \ \wedge \ y = \varnothing)$,
- $\models x \neq \varnothing \longleftrightarrow \neg\mathsf{Disj}(x, x)$,

(g) $\models x = \varnothing \longleftrightarrow \mathsf{Disj}(x, x)$.

(h) We have that:

- $\models \mathsf{Disj}(x, y) \longleftrightarrow x \cap y = \varnothing$ and $\models \mathsf{Disj}(x, y) \longleftrightarrow x \setminus (x \setminus y) = x \setminus x$,
- $\models \neg\mathsf{Disj}(x, y) \longleftrightarrow x \cap y \neq \varnothing$ and $\models \neg\mathsf{Disj}(x, y) \longleftrightarrow (\exists z)(z \subseteq x \ \wedge \ z \subseteq y \ \wedge \ z \neq \varnothing)$.

(i) $\models x \cap y \neq \varnothing \longleftrightarrow (\exists w, w')\big(w \subseteq x \ \wedge \ w \subseteq y \ \wedge \ w' \subseteq w \ \wedge \ w' \neq w\big)$.
From the latter and (d), it follows that $\neg\mathsf{Disj}(x, y)$ is also expressible in $\mathsf{BST}(\cup, =, \neq)$.

(j) By way of contradiction, assume that there exists a $\mathsf{BST}(\cup, \cap, =, \neq)$-conjunction $\Phi_\varnothing(x, \vec{y})$ such that

$$\models x = \varnothing \longleftrightarrow (\exists \vec{y}\,)\ \Phi_\varnothing(x, \vec{y}\,) \tag{24}$$

(so that $(\exists \vec{y})\Phi_\varnothing(x, \vec{y})$, and therefore $\Phi_\varnothing(x, \vec{y})$, is satisfiable).
Let M be any set assignment such that $M \models \Phi_\varnothing(x, \vec{y})$, and set $M'z := Mz \cup C$, for every $z \in \mathit{Vars}(\Phi_\varnothing)$, where C is any nonempty set that is disjoint from Mz, for every

$z \in \mathit{Vars}(\Phi_\varnothing)$ (namely, such that $C \cap \bigcup M(\mathit{Vars}(\Phi_\varnothing)) = \emptyset$). Then, for $y_1, \ldots, y_n \in \mathit{Vars}(\Phi_\varnothing)$, we have:

$$\begin{aligned} M'(y_1 \cup \ldots \cup y_n) &= M'y_1 \cup \ldots \cup M'y_n \\ &= (My_1 \cup C) \cup \ldots \cup (My_n \cup C) \\ &= (My_1 \cup \ldots \cup My_n) \cup C \\ &= M(y_1 \cup \ldots \cup y_n) \cup C \end{aligned}$$

and

$$\begin{aligned} M'(y_1 \cap \ldots \cap y_n) &= M'y_1 \cap \ldots \cap M'y_n \\ &= (My_1 \cup C) \cap \ldots \cap (My_n \cup C) \\ &= (My_1 \cap \ldots \cap My_n) \cup C \\ &= M(y_1 \cap \ldots \cap y_n) \cup C. \end{aligned}$$

Therefore:

(j_1) if the literal $y_1 \cup \ldots \cup y_n = z_1 \cup \ldots \cup z_m$ is in $\Phi_\varnothing$, then

$$\begin{aligned} M'(y_1 \cup \ldots \cup y_n) &= M(y_1 \cup \ldots \cup y_n) \cup C \\ &= M(z_1 \cup \ldots \cup z_m) \cup C = M'(z_1 \cup \ldots \cup z_m); \end{aligned}$$

(j_2) if the literal $y_1 \cap \ldots \cap y_n = z_1 \cap \ldots \cap z_m$ is in $\Phi_\varnothing$, then

$$\begin{aligned} M'(y_1 \cap \ldots \cap y_n) &= M(y_1 \cap \ldots \cap y_n) \cup C \\ &= M(z_1 \cap \ldots \cap z_m) \cup C = M'(z_1 \cap \ldots \cap z_m); \end{aligned}$$

(j_3) if the literal $y \neq z$ is in $\Phi_\varnothing$, then we have

$$M'y = My \cup C, \quad M'z = Mz \cup C, \quad \text{and} \quad My \neq Mz,$$

implying $M'y \neq M'z$ because of the disjointness between C and My, Mz.

From (j_1), (j_2), and (j_3), it follows that $M' \models \Phi_\varnothing$, so that we have also $M' \models (\exists \vec{y})\Phi_\varnothing$. In addition we have $M'x = Mx \cup C \neq \emptyset$. Thus, $M' \not\models (\exists \vec{y})\Phi_\varnothing \longrightarrow x = \varnothing$, contradicting (24) and hence showing that $x = \varnothing$ is not expressible in $\mathsf{BST}(\cup, \cap, =, \neq)$.

(k) If $x = y \setminus z$ were expressible in $\mathsf{BST}(\cup, \cap, =, \neq)$, then there would exist a conjunction $\Psi(x, y, z, \vec{w})$ in $\mathsf{BST}(\cup, \cap, =, \neq)$ such that

$$\models x = y \setminus z \longleftrightarrow (\exists \vec{w})\Psi(x, y, z, \vec{w}). \tag{25}$$

From (25) we get

$$\models x = y \setminus y \longleftrightarrow (\exists \vec{w})\Psi(x, y, y, \vec{w}),$$

$$
\begin{aligned}
\therefore\quad & \models x = \varnothing \;\longleftrightarrow\; (\exists \vec{w})\Psi(x,y,y,\vec{w}) \\
\therefore\quad & \models (\forall y)\big[x = \varnothing \;\longleftrightarrow\; (\exists \vec{w})\Psi(x,y,y,\vec{w})\big] \\
\therefore\quad & \models (\forall y)\big[\big((\exists \vec{w})\Psi(x,y,y,\vec{w}) \;\longrightarrow\; x = \varnothing\big) \\
& \qquad\qquad \wedge\; \big(x = \varnothing \;\longrightarrow\; (\exists \vec{w})\Psi(x,y,y,\vec{w})\big)\big] \\
\therefore\quad & \models \big[(\forall y)((\exists \vec{w})\Psi(x,y,y,\vec{w}) \;\longrightarrow\; x = \varnothing) \\
& \qquad\qquad \wedge\; (\forall y)\big(x = \varnothing \;\longrightarrow\; (\exists \vec{w})\Psi(x,y,y,\vec{w})\big)\big] \\
\therefore\quad & \models \big[\big((\forall y)\neg(\exists \vec{w})\Psi(x,y,y,\vec{w}) \;\vee\; x = \varnothing\big) \\
& \qquad\qquad \wedge\; \big(x = \varnothing \;\longrightarrow\; (\forall y)(\exists \vec{w})\Psi(x,y,y,\vec{w})\big)\big] \\
\therefore\quad & \models \big[\big(\neg(\exists \vec{w})(\exists y)\Psi(x,y,y,\vec{w}) \;\vee\; x = \varnothing\big) \\
& \qquad\qquad \wedge\; \big(x = \varnothing \;\longrightarrow\; (\exists \vec{w})(\exists y)\Psi(x,y,y,\vec{w})\big)\big] \\
\therefore\quad & \models \big[\big((\exists \vec{w})(\exists y)\Psi(x,y,y,\vec{w}) \;\longrightarrow\; x = \varnothing\big) \\
& \qquad\qquad \wedge\; \big(x = \varnothing \;\longrightarrow\; (\exists \vec{w})(\exists y)\Psi(x,y,y,\vec{w})\big)\big]
\end{aligned}
$$

and therefore may conclude

$$\models x = \varnothing \;\longleftrightarrow\; (\exists \vec{w})(\exists y)\Psi(x,y,y,\vec{w});$$

i.e., $x = \varnothing$ would be expressible in $\mathsf{BST}(\cup,\cap,=,\neq)$, contradicting (j). Thus, $x = y \setminus z$ is not expressible in $\mathsf{BST}(\cup,\cap,=,\neq)$. □

Remark A.1. (Concerning the expressibility of union in terms of set-difference) Above, in proving Lemma 1.6(b), we have shown how to formulate union literals as conjunctions of equalities between $\setminus$*-terms* (viz., terms built from set-variables by means of the set-difference operator). Concretely, we have pointed out that $\models x = y \cup z \longleftrightarrow \big(t_1 = t_2 \wedge t_3 = t_4\big)$ holds for suitable $\setminus$-terms t_1, t_2, t_3, t_4 . To show that this result cannot be ameliorated, we will now prove that *no single equality $s_1 = s_2$ between $\setminus$-terms suffices to express $x = y \cup z$.*

By way of contradiction, suppose that $\models s_1 = s_2 \;\longrightarrow\; x = y \cup z$ holds, where s_1 and s_2 are $\setminus$-terms and x, y, z are distinct variables. To enforce that $\mathsf{left}(s)$ denotes the leftmost variable in s, we define recursively:

$$\mathsf{left}(s) \;:=\; \begin{cases} s & \text{when } s \text{ is a variable} \\ \mathsf{left}(s') & \text{when } s = s' \setminus s'', \end{cases}$$

for any $\setminus$-term s; thus, by straight induction, $\models s \subseteq \mathsf{left}(s)$ will hold for every s. There exists at least one $w \in \{x,y,z\} \setminus \{\, \mathsf{left}(s_1), \mathsf{left}(s_2) \,\}$; thus, after putting $L_w := \{x,y,z\} \setminus \{w\}$, we get

$$\models s_1 = s_2 \;\longrightarrow\; w \setminus \bigcup L_w = \varnothing$$

from our initial hypothesis; for, trivially, $\models x = y \cup z \longrightarrow \big(\, x \setminus (y \cup z) = \varnothing \wedge y \setminus x = \varnothing \wedge z \setminus x = \varnothing \,\big)$.

However, this is untenable. To see why, consider a set-assignment M such that $Mw \neq \emptyset$ holds along with $Mu = \emptyset$, for $u \in L_w \cup \{\, \mathsf{left}(s_1), \mathsf{left}(s_2) \,\}$. Then

$$M \models s_1 = \varnothing \ \wedge \ s_2 = \varnothing \ \wedge \ w \setminus \bigcup L_w \neq \varnothing$$

holds, which leads us to the sought contradiction. □

In preparation for the proof of Lemma 1.7, we state two propositions, the former of which is obvious and will easily yield the other one.

Proposition A.2. Consider two lists $S_0, \ldots, S_m$ and $T_0, \ldots, T_n$ of sets, along with a set C included in each S_i and in each T_j, so that $C \subseteq S_0 \cap \cdots \cap S_m \cap T_0 \cap \cdots \cap T_n$. Then we have:

$$\bigcup_{i=0}^{m} S_i = \bigcup_{j=0}^{n} T_j \quad \longleftrightarrow \quad \bigcup_{i=0}^{m} (S_i \setminus C) = \bigcup_{j=0}^{n} (T_j \setminus C). \tag{26}$$

Proposition A.3. Let φ be a conjunction in $\mathsf{BST}(\cup, =, \neq)$, and let x be any variable occurring in φ such that the conjunction

$$\varphi \ \wedge \bigwedge_{y \in \mathit{Vars}(\varphi)} y \cup x = y \tag{27}$$

is satisfiable. Then φ has a model M^* such that $M^*x = \emptyset$.

Proof:
Le M be any model for (27), so that $Mx \subseteq My$ holds for all $y \in \mathit{Vars}(\varphi)$. By putting

$$M^*y \coloneqq My \setminus Mx, \quad \text{for } y \in \mathit{Vars}(\varphi),$$

define the set assignment M^* with $\mathsf{dom}(M^*) = \mathit{Vars}(\varphi)$. Thus, for all $L, R \subseteq \mathit{Vars}(\varphi)$,

$$\begin{aligned} \bigcup_{y \in L} My = \bigcup_{y \in R} My \quad &\longleftrightarrow \quad \bigcup_{y \in L} (My \setminus Mx) = \bigcup_{y \in R} (My \setminus Mx) \\ &\longleftrightarrow \quad \bigcup_{y \in L} M^*y = \bigcup_{y \in R} M^*y \,, \end{aligned}$$

will hold (in particular, the former of these bi-implications follows directly from Proposition A.2). Consequently, taking into account that $M(s_0 \cup \cdots \cup s_\kappa) = M\, s_0 \cup \cdots \cup M\, s_\kappa$ holds for any nonempty set $\{s_0, \ldots, s_\kappa\} \subseteq \mathit{Vars}(\varphi)$, we get $M^* \models \varphi$ from $M \models \varphi$. Since we also have $M^*x = Mx \setminus Mx$, we readily reach the desired conclusion. □

We are now ready to prove Lemma 1.7.

Proof of Lemma 1.7: Let $\varphi \mapsto \Psi_\varphi(\vec{y}, x)$ be the mapping from $\mathsf{BST}(\cup, =, \neq)$ into itself where

$$\Psi_\varphi(x, \vec{y}) \coloneqq \bigwedge_{y \in Vars(\varphi)} y \cup x = y\,.$$

Plainly, $\models x = \varnothing \longrightarrow \bigwedge_{y \in Vars(\varphi)} y \cup x = y$. Hence, *a fortiori*,

$$\models \big(\varphi(\vec{y}) \ \wedge \ x = \varnothing\big) \longrightarrow \bigwedge_{y \in Vars(\varphi)} y \cup x = y\,,$$

proving condition (c) of Def. 1.3.

As for condition (b) of Def. 1.3, we have to show that if $\varphi(\,\vec{y}\,) \ \wedge \ \bigwedge_{y \in Vars(\varphi)} y \cup x = y$ is satisfiable, so is $\varphi(\,\vec{y}\,) \ \wedge \ x = \varnothing$. But this follows at once from Proposition A.3.

Finally, by observing that the mapping $\varphi \mapsto \bigwedge_{y \in Vars(\varphi)} y \cup x = y$ clearly can be computed in $\mathcal{O}(n)$-time, we conclude that the literal $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cup, =, \neq)$. $\square$