Theoretical Computer Science 954 (2023) 113786

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Complexity assessments for decidable fragments of Set Theory. III: Testers for crucial, polynomial-maximal decidable Boolean languages ☆,☆☆

Domenico Cantone [a,*], Pietro Maugeri [a], Eugenio G. Omodeo [b]

[a] Dept. of Mathematics and Computer Science, University of Catania, Italy
[b] Dept. of Mathematics and Earth Sciences, University of Trieste, Italy

ARTICLE INFO

Article history:
Received 6 July 2022
Received in revised form 1 December 2022
Accepted 22 February 2023
Available online 7 March 2023
Communicated by O. Arieli

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

ABSTRACT

We continue our investigation aimed at spotting small fragments of Set Theory (in this paper, sublanguages of Boolean Set Theory) that might be of use in automated proof-checkers based on the set-theoretic formalism. Here we propose a method that leads to a cubic-time satisfiability decision test for the language involving, besides variables intended to range over the von Neumann set-universe, the Boolean operator $\cup$ and the logical relators $=$ and $\neq$. It can be seen that the dual language involving the Boolean operator $\cap$ and, again, the relators $=$ and $\neq$, also admits a cubic-time satisfiability decision test; noticeably, the same algorithm can be used for both languages. Suitable pre-processing can reduce richer Boolean languages to the said two fragments, so that the same cubic satisfiability test can be used to treat the relators $\subseteq$ and $\not\subseteq$, and the predicates '$\blacksquare = \varnothing$' and '$\mathrm{DISJ}(\blacksquare, \blacksquare)$', meaning 'the argument is empty' and 'the arguments are disjoint sets', along with their opposites '$\blacksquare \neq \varnothing$' and '$\neg\mathrm{DISJ}(\blacksquare, \blacksquare)$'. Those richer languages are 'polynomial maximal', in the sense that each language strictly containing either of them and whose formulae are conjunctions of literals has an NP-hard satisfiability problem.
A generalized version of the two said satisfiability tests can treat the relator $\not\subset$, though at the price of a worsening of the algorithmic complexity (from cubic to quintic time).

0. Introduction

The field named *Computable Set Theory* [5] pursued, with long-standing efforts, languages reconciling ease of symbolic manipulation with high expressive power, so that an armory of reasoning tools—foremost, satisfiability testers for unquantified fragments of theories concerning sets and classes—could shape a friendly proof-development environment.

Long before the envisaged proof-verifier ÆtnaNova [17] came into being, a foundational quest aimed at carefully drawing the frontier between the decidable and the undecidable in set theory sparked interest in the field [15]. The unquantified set-theoretic language called Multi-Level Syllogistic (MLS for short) and some extensions of it [11], whose satisfiability

☆ This article belongs to Section B: Logic, semantics and theory of programming, Edited by Don Sannella.

☆☆ The authors 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.

* Corresponding author.
E-mail addresses: domenico.cantone@unict.it (D. Cantone), pietro.maugeri@unict.it (P. Maugeri), eomodeo@units.it (E.G. Omodeo).

https://doi.org/10.1016/j.tcs.2023.113786
0304-3975/

Table 1
Complexity of the analyzed fragments: in each row, markers occur in correspondence of the operators/relators of the language. The polynomial-maximal fragments of $\mathbb{BST}$ bear the marker $\circledast$ instead of $\star$ (cf. [4] and Remark 3); the symbol $\star$ is circled in the two fragments which lie at the core of this paper.

$\cup$	$\cap$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg$Disj	$\subseteq$	$\not\subseteq$	$=$	$\neq$	$\subset$	$\not\subset$	Complexity	Section
⊛								⊛	⊛			$O(n^3)$	2
★		★	★		★	★	★	★	★			$O(n^3)$	3
	⊛							⊛	⊛			$O(n^3)$	2
	★	★	★	★	★	★	★	★	★			$O(n^3)$	3
★									★		★	$O(n^5)$	4.2
✻		✻	✻		✻	✻	✻	✻	✻	✻	✻	$O(n^5)$	4.4
	★								★		★	$O(n^5)$	4.3
	✻	✻	✻	✻	✻	✻	✻	✻	✻	✻	✻	$O(n^5)$	4.4

problems were shown to be NP-complete in [7], foreran that quest. In recent years we undertook a related investigation, aimed at spotting fragments of MLS endowed with low-degree polynomial-time satisfiability tests, which hence promise to be useful in set-based proof technology. In [4], we reported about the complexity taxonomy of all sublanguages of the so-called Boolean Set Theory ($\mathbb{BST}$ for short); here we treat in detail two core fragments, or 'theories', in that taxonomy and produce cubic-time tests for their satisfiability problems. Specifically, we will study two languages, consisting of all nonempty conjunctions of literals of the two forms

$$(=)\quad x_1 \star \cdots \star x_h = y_1 \star \cdots \star y_k$$

$$(\neq)\quad u_1 \star \cdots \star u_m \neq v_1 \star \cdots \star v_p,$$

where $h, k, m, p \geqslant 1$ hold, $\star$ stands for a fixed dyadic set operation, and x's, y's, u's, v's stand for set variables.

By instantiating $\star$ as $\cup$, we obtain the fragment $\mathsf{BST}(\cup, =, \neq)$; by instantiating it as $\cap$, we obtain the fragment $\mathsf{BST}(\cap, =, \neq)$.

The paper is organized as follows. In Section 1, we introduce an equivalence relation $\sim_\varphi$ between nonempty subsets of the set of variables occurring in a formula φ of either fragment and elicit some of its key properties. Then, in Section 2, we present our main decidability results together with a cubic-time satisfiability test, stressing the commonalities between the two decidable theories. In Section 3, we discuss how to adapt the proposed tests to richer sublanguages of $\mathbb{BST}$, while preserving the cubic complexity. Section 4 offers two variants of the satisfiability tests for $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$: those can handle the negated strict inclusion relator $\not\subset$ and have much in common with their counterpart tests just mentioned. As regards expressive power, it is shown that one of the new fragments, $\mathsf{BST}(\cup, \not\subset, \neq)$, extends $\mathsf{BST}(\cup, =, \neq)$, while the other, $\mathsf{BST}(\cap, \not\subset, \neq)$, extends $\mathsf{BST}(\cap, =, \neq)$; as regards the algorithmic complexity of the satisfiability tests, they perform worse than their counterparts, due to the disjunction implicit in $\not\subset$ (to wit, $x \not\subset y \iff (x \not\subseteq y \vee x = y)$) which pushes the time complexity up from $O(n^3)$ to $O(n^5)$. Then we draw conclusions.

An orthogonal view of the content of this paper is provided in Table 1, summarizing the complexity results to be discussed. In that table:

- The four fragments shown at the top, and the complexity of their satisfiability tester, were briefly discussed in [3], which however did not explain the estimated complexity (as will be done here) — actually, the complexity of the tester for the third fragment was then overestimated.
- The four fragments shown at the bottom are new, because they do not belong to the $\mathbb{BST}$ family: they involve, in fact, the strict inclusion relator which, in earlier papers of this series, could at best be emulated via a disjunction. Here, instead, they are regarded as primitive constructs and are treated by an entirely *ad hoc* approach.

A detailed comparison with the results concerning $\mathbb{MST}$ (see [6]) would make little sense, because membership is a construct available in all fragments of the $\mathbb{MST}$ family, but is not expressible in the purely Boolean view underlying this work as well as [3].

Although the views underlying this paper and [6] differ from one another, we made an effort to exploit the same semantics for the respective languages. This uniformity goal forced us to refer the semantics to a universe of nested sets; in the light of Lemma 2 of [3], though, we could have insisted without loss of generality that the elements of the sets assigned as values to the variables of a formula must all have the same rank. But then it would be an easy matter to replace the elements of such sets by memberless individuals (sometimes called 'atoms' or 'urelements') — concerning this, also see footnote 1 below.

1. The equivalence relation $\sim_\varphi$ and the $\sim_\varphi$-closure operator

Any given formula φ of $\mathsf{BST}(\star, =, \neq)$ can be represented by two sets:

$$\Phi_\varphi^{=} := \Big\{\big\{\{x_1, \cdots, x_h\}, \{y_1, \cdots, y_k\}\big\} \mid x_1 \star \cdots \star x_h = y_1 \star \cdots \star y_k \text{ is in } \varphi\Big\},$$
$$\Phi_\varphi^{\neq} := \Big\{\big\{\{u_1, \cdots, u_m\}, \{v_1, \cdots, v_p\}\big\} \mid u_1 \star \cdots \star u_m \neq v_1 \star \cdots \star v_p \text{ is in } \varphi\Big\}.$$

A central role will be played by the following relation $\sim_\varphi$ on $\mathscr{P}^+(Vars(\varphi))$, where $Vars(\varphi)$ is defined to be the collection of all variables occurring in φ, and $\mathscr{P}^+(S) := \mathscr{P}(S) \setminus \{\emptyset\}$ for any set S. Specifically, $\sim_\varphi$ is the intersection—hence the inclusion-minimal—of all *equivalence relations* $\sim$ on $\mathscr{P}^+(Vars(\varphi))$ that satisfy the following closure conditions:

(Cl1) $\Phi_\varphi^{=} \subseteq \sim$,
(Cl2) if $A \sim B$, then $A \cup C \sim B \cup C$, for all $A, B, C \in \mathscr{P}^+(Vars(\varphi))$.

We will in fact prove (see Theorems 1 and 2 below), for any φ in $\mathsf{BST}(\star, =, \neq)$, that φ is satisfiable if and only if $\{u_1, \ldots, u_m\} \not\sim_\varphi \{v_1, \ldots, v_p\}$ holds for every literal of the form $(\neq)$ in φ. We will exploit a useful *closure operator* to test conditions of the form $Z_1 \not\sim_\varphi Z_2$. Specifically, we will prove that the set

$$\widetilde{Z} := \bigcup\{W \mid W \sim_\varphi Z\},$$

called the $\sim_\varphi$-*closure of* Z (or, more simply, the *closure of* Z, when the relation $\sim_\varphi$ is understood), is the largest set $\sim_\varphi$-equivalent to Z and then will provide a quadratic algorithm for computing it. Accordingly, a set Z such that $\widetilde{Z} = Z$ will be said to be $\sim_\varphi$-closed or just *closed*.

Lemma 1. *Let $Z \in \mathscr{P}^+(Vars(\varphi))$. Then the closure $\widetilde{Z}$ of Z, namely the set $\bigcup\{W \mid W \sim_\varphi Z\}$, is the largest subset of $Vars(\varphi)$ that is $\sim_\varphi$-equivalent to Z.*

Proof. Let W_1, W_2 be such that $W_1 \sim_\varphi Z$ and $W_2 \sim_\varphi Z$. By applying (Cl2) twice, we have: $W_1 \cup W_2 \sim_\varphi Z \cup W_2 \sim_\varphi Z$. In view of the finiteness of $\{W \mid W \sim_\varphi Z\}$, by induction it follows that

$$\Big(\bigcup\{W \mid W \sim_\varphi Z\}\Big) \sim_\varphi Z\,.$$

Since, for every W' such that $W' \sim_\varphi Z$, we plainly have $W' \subseteq \bigcup\{W \mid W \sim_\varphi Z\}$, we may conclude that $\bigcup\{W \mid W \sim_\varphi Z\}$ is the largest set in $\mathscr{P}^+(Vars(\varphi))$ that is $\sim_\varphi$-equivalent to Z. □

The equivalence relation $\sim_\varphi$ captures which equalities among terms must be true in every model, if any, of a given formula φ: this will be shown in Lemma 2.

A *set assignment* M is any function sending a collection $\mathsf{dom}(M)$ of set variables into the von Neumann universe[1] $\mathcal{V} := \bigcup_{\alpha \in On} \mathcal{V}_\alpha$ of all sets, resulting from the union of the levels $\mathcal{V}_\alpha := \bigcup_{\beta<\alpha} \mathscr{P}(\mathcal{V}_\beta)$, with α ranging over the class On of all ordinal numbers, where $\mathscr{P}(\blacksquare)$ is the powerset operator.

Recursive designation rules attach a value to each compound term and to each literal in $\mathsf{BST}(\star, =, \neq)$ that only involve variables belonging to $\mathsf{dom}(M)$:

$$M(\sigma \star \tau) := M\sigma \star M\tau\,;$$
$$M(\sigma = \tau) := \begin{cases} \mathsf{true} & \text{if } M\sigma = M\tau\,, \\ \mathsf{false} & \text{otherwise;} \end{cases}$$
$$M(\sigma \neq \tau) := \neg M(\sigma = \tau)\,.$$

Then we put, for conjunctions of literals ℓ_i:

$$M(\ell_1 \wedge \cdots \wedge \ell_k) := M\ell_1 \wedge \cdots \wedge M\ell_k$$

when $Vars(\ell_i) \subseteq \mathsf{dom}(M)$, for $i = 1, \ldots, k$.

[1] Notice that a universe formed by subsets of an infinite collection of *individuals* (namely, entities devoid of elements that differ from the empty set) would also be adequate for the needs of this paper; here $\mathcal{V}$ is adopted as the standard interpretation domain in order to have a uniform semantics for the languages $\mathbb{BST}$ and $\mathbb{MST}$, both treated in this series of papers.

Given a conjunction φ and a set assignment M such that $Vars(\varphi) \subseteq \mathsf{dom}(M)$, 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.

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

For convenience, in what follows we will use the shortening notations

$$\begin{aligned} M\{x_1, \ldots, x_k\} &\coloneqq \{M x_1, \ldots, M x_k\}, \\ \bigstar\{x_1, \ldots, x_k\} &\coloneqq x_1 \star \cdots \star x_k, \\ \bigstar\{s_1, \ldots, s_k\} &\coloneqq s_1 \star \cdots \star s_k, \end{aligned}$$

where M represents a set assignment, $x_1, \ldots, x_k$ and $s_1, \ldots, s_k$ denote variables and sets, respectively, and the $\star$'s stand for alike symbols/operations '$\cup$' or '$\cap$'.

Lemma 2. *Let φ be any formula in $\mathsf{BST}(\star, =, \neq)$, and let M be any set assignment over $Vars(\varphi)$ satisfying φ. Then*

$$Z_1 \sim_\varphi Z_2 \implies \bigstar M Z_1 = \bigstar M Z_2,$$

for all $Z_1, Z_2 \in \mathscr{P}^+(Vars(\varphi))$.

Proof. In view of the inclusion-minimality of $\sim_\varphi$, it suffices to prove that the equivalence relation $\sim_M$ over $\mathscr{P}^+(Vars(\varphi))$ defined by

$$Z_1 \sim_M Z_2 \overset{\text{Def.}}{\iff} \bigstar M Z_1 = \bigstar M Z_2$$

satisfies the closure conditions (Cl1) and (Cl2).

Concerning (Cl1), if $\{L, R\} \in \Phi_\varphi^=$ then $\bigstar ML = \bigstar MR$; so $L \sim_M R$ holds, proving $\Phi_\varphi^= \subseteq \sim_M$.

As for (Cl2), let $A \sim_M B$ and $C \subseteq Vars(\varphi)$. Then $\bigstar MA = \bigstar MB$, and therefore

$$\bigstar M(A \cup C) = (\bigstar MA) \star (\bigstar MC) = (\bigstar MB) \star (\bigstar MC) = \bigstar M(B \cup C),$$

from which $A \cup C \sim_M B \cup C$ follows. □

We next prove the following key property, which will be used in the correctness proof of our fast algorithm presented in Section 2.1 for computing $\sim_\varphi$-closures.

Lemma 3. *Given a $\mathsf{BST}(\star, =, \neq)$-formula φ, let $Z \in \mathscr{P}^+(Vars(\varphi))$ be such that*

$$L \subseteq Z \iff R \subseteq Z, \qquad \text{for every } \{L, R\} \in \Phi_\varphi^=. \tag{1}$$

Then, for all $W_1, W_2 \in \mathscr{P}^+(Vars(\varphi))$ such that $W_1 \sim_\varphi W_2$, we have

$$W_1 \subseteq Z \iff W_2 \subseteq Z. \tag{2}$$

Proof. Let φ and Z be as in the hypothesis. In view of the $\subseteq$-minimality of $\sim_\varphi$, it suffices to prove that the equivalence relation over $\mathscr{P}^+(Vars(\varphi))$ defined by

$$W_1 \sim_Z W_2 \quad \overset{\text{Def.}}{\iff} \quad \big(W_1 \subseteq Z \iff W_2 \subseteq Z\big) \tag{3}$$

satisfies the closure conditions (Cl1) and (Cl2).

As for (Cl1), just from the hypothesis it follows that $L \sim_Z R$, for every $\{L, R\} \in \Phi_\varphi^=$. Concerning (Cl2), let $A \sim_Z B$ and $C \subseteq Vars(\varphi)$, and assume that $A \cup C \subseteq Z$. Then, $A \subseteq Z$ and $C \subseteq Z$, so that by (3) we have $B \cup C \subseteq Z$. Symmetrically, it can be shown that $B \cup C \subseteq Z$ implies $A \cup C \subseteq Z$. Hence,

$$A \cup C \subseteq Z \iff B \cup C \subseteq Z$$

holds and, by (3), we readily have $A \cup C \sim_Z B \cup C$. The arbitrariness of A, B, and C yields that even the closure condition (Cl2) holds for $\sim_Z$.

Finally, from the $\subseteq$-minimality of $\sim_\varphi$, we get $\sim_\varphi \subseteq \sim_Z$. Therefore, if $W_1 \sim_\varphi W_2$, then $W_1 \sim_Z W_2$, which by (3) implies $W_1 \subseteq Z \iff W_2 \subseteq Z$. □

Further useful properties of the $\sim_\varphi$-closure operator and of the equivalence relation $\sim_\varphi$ are reported in the following lemma.

Lemma 4. *Let $Z, Z_1, Z_2 \in \mathscr{P}^+(Vars(\varphi))$. Then*

(a) $Z \subseteq \widetilde{Z}$ *and* $Z \sim_\varphi \widetilde{Z}$; [Extensivity]
(b) $\widetilde{Z} = \widetilde{\widetilde{Z}}$; [Idempotency]
(c) *if* $Z_1 \sim_\varphi \widetilde{Z}_2$, *then* $Z_1 \subseteq \widetilde{Z}_2$;
(d) $Z_1 \sim_\varphi Z_2$ *if and only if* $\widetilde{Z}_1 = \widetilde{Z}_2$; [Representativity]
(e) *if* $Z_1 \subseteq Z_2$, *then* $\widetilde{Z}_1 \subseteq \widetilde{Z}_2$; [Monotonicity]
(f) $Z_1 \subseteq \widetilde{Z}_2$ *if and only if* $\widetilde{Z}_1 \subseteq \widetilde{Z}_2$;
(g) *if* $Z_1 \subseteq Z$ *or* $Z_2 \subseteq Z$ *holds and* $Z_1 \sim_\varphi Z_2$, *then* $Z \sim_\varphi Z \cup Z_1 \cup Z_2$.

Proof. Property (a) follows directly from Lemma 1.

Concerning (b): the transitivity of $\sim_\varphi$ yields $\widetilde{\widetilde{Z}} \sim_\varphi Z$ so that, by the definition of $\widetilde{Z}$, $\widetilde{\widetilde{Z}} \subseteq \widetilde{Z}$ holds. By (a) we have $\widetilde{Z} \subseteq \widetilde{\widetilde{Z}}$, therefore $\widetilde{Z} = \widetilde{\widetilde{Z}}$.

As for (c), if $Z_1 \sim_\varphi \widetilde{Z}_2$, then $Z_1 \subseteq \widetilde{\widetilde{Z}}_2 = \widetilde{Z}_2$.

Concerning (d), if $Z_1 \sim_\varphi Z_2$, then the transitivity of $\sim_\varphi$ yields $\widetilde{Z}_1 \sim_\varphi \widetilde{Z}_2$. Thus, by (c), we get $\widetilde{Z}_1 = \widetilde{Z}_2$. Conversely if $\widetilde{Z}_1 = \widetilde{Z}_2$, then $\widetilde{Z}_1 \sim_\varphi \widetilde{Z}_2$, thus by transitivity $Z_1 \sim_\varphi Z_2$ holds.

Regarding (e), let $Z_1 \subseteq Z_2$. Since by (a) $Z_2 \subseteq \widetilde{Z}_2$ and $Z_1 \sim_\varphi \widetilde{Z}_1$ hold, by (Cl2) we have

$$\widetilde{Z}_2 = Z_1 \cup (\widetilde{Z}_2 \setminus Z_1) \sim_\varphi \widetilde{Z}_1 \cup (\widetilde{Z}_2 \setminus Z_1) = \widetilde{Z}_1 \cup \widetilde{Z}_2 .$$

Thus, by (c), we get the inclusion $\widetilde{Z}_1 \cup \widetilde{Z}_2 \subseteq \widetilde{Z}_2$, and therefore $\widetilde{Z}_1 \subseteq \widetilde{Z}_2$.

Concerning (f), if $Z_1 \subseteq \widetilde{Z}_2$, then by (e) and (b) we have $\widetilde{Z}_1 \subseteq \widetilde{\widetilde{Z}}_2 = \widetilde{Z}_2$. Conversely, if $\widetilde{Z}_1 \subseteq \widetilde{Z}_2$, then by (a) we have $Z_1 \subseteq \widetilde{Z}_1 \subseteq \widetilde{Z}_2$.

Finally, as for (g), suppose that we have $Z_1 \sim_\varphi Z_2$ and either $Z_1 \subseteq Z$ or $Z_2 \subseteq Z$ holds. By (Cl2), we have $Z \cup Z_1 \sim_\varphi Z \cup Z_2$. But $\{Z \cup Z_1, Z \cup Z_2\} = \{Z, Z \cup Z_1 \cup Z_2\}$, hence $Z \sim_\varphi Z \cup Z_1 \cup Z_2$ follows. □

Notice that the properties of the closure operator $\widetilde{\blacksquare}$ in Lemmas 1 and 4 depend solely on the closure condition (Cl2).

2. Satisfiability in BST$(\cup, =, \neq)$ and in BST$(\cap, =, \neq)$

Below we will present a necessary condition that also suffices to ensure that a given formula in either BST$(\cup, =, \neq)$ or BST$(\cap, =, \neq)$ is satisfiable. Noticeably, this condition is essentially the same for both languages, so that the same algorithm can be used to test formulae of either language for satisfiability.

Theorem 1. *Let* φ *be a* BST$(\cup, =, \neq)$*-formula. Then* φ *is satisfiable if and only if* $L \nsim_\varphi R$ *(namely* $\widetilde{L} \neq \widetilde{R}$*) holds for every literal of the form* $\bigcup L \neq \bigcup R$ *in* φ.

Proof. *(Necessity.)* Let M be a model for φ. By way of contradiction, assume that there exists a literal $\bigcup L \neq \bigcup R$ such that $L \sim_\varphi R$. Then by Lemma 2 we would have $\bigcup ML = \bigcup MR$, a contradiction. Therefore for all literals $\bigcup L \neq \bigcup R$ we must have $L \nsim_\varphi R$, completing the necessity part of the proof.

(Sufficiency.) Next, let us assume that, for each $\{L, R\} \in \Phi_\varphi^{\neq}$, we have $L \nsim_\varphi R$. We will construct a set assignment M that satisfies φ.

Let us assign a nonempty set $b_{\widetilde{V}}$ to each $\widetilde{V}$ such that $V \in \bigcup \Phi_\varphi^{\neq}$ in such a way that the $b_{\widetilde{V}}$'s are pairwise distinct.[2] Then we define the set assignment M over $Vars(\varphi)$ by putting, for each $x \in Vars(\varphi)$,

$$Mx := \{b_{\widetilde{V}} \mid x \notin \widetilde{V} \text{ and } V \in \bigcup \Phi_\varphi^{\neq}\},$$

so that we have

$$\bigcup MU := \{b_{\widetilde{V}} \mid U \not\subseteq \widetilde{V} \text{ and } V \in \bigcup \Phi_\varphi^{\neq}\}, \tag{4}$$

for every $U \subseteq Vars(\varphi)$.

We first show that $\bigcup ML = \bigcup MR$ holds whenever $\{L, R\} \in \Phi_\varphi^{=}$. Thus, let $\{L, R\} \in \Phi_\varphi^{=}$ and let $b_{\widetilde{V}} \in \bigcup ML$, for some $V \in \Phi_\varphi^{\neq}$ such that $L \not\subseteq \widetilde{V}$. Then $\widetilde{L} \not\subseteq \widetilde{V}$, by Lemma 4(f). Since $\{L, R\} \in \Phi_\varphi^{=}$, then by (Cl1) we have $L \sim_\varphi R$, so that, by (d) and (f) of Lemma 4, $R \not\subseteq \widetilde{V}$ follows. Hence $b_{\widetilde{V}} \in \bigcup MR$, and by the arbitrariness of $b_{\widetilde{V}}$ we have $\bigcup ML \subseteq \bigcup MR$.

[2] For definiteness, such $b_{\widetilde{V}}$'s can be drawn from the collection $\{\{\emptyset\}, \{\{\emptyset\}\}, \{\{\{\emptyset\}\}\}, \dots\}$ of Zermelo's non-zero numerals. (Incidentally, this shows that each variable occurring in a satisfiable formula could be interpreted as a set of non-zero numerals). Alternatively—in a semantics devoid of nested sets (see footnote 1)—, the entities $b_{\widetilde{V}}$ could be drawn from a collection of individuals.

Analogously one can prove $\bigcup MR \subseteq \bigcup ML$, so $\bigcup ML = \bigcup MR$ holds, as we intended to show.

Next we prove that $\bigcup ML \neq \bigcup MR$ holds, whenever $\{L, R\} \in \Phi_\varphi^{\neq}$. Thus, let $\{L, R\} \in \Phi_\varphi^{\neq}$, so that by our assumption we have $L \nsim_\varphi R$. Hence, by Lemma 4(d), $\widetilde{L} \neq \widetilde{R}$. W.l.o.g., let us assume that $\widetilde{L} \not\subseteq \widetilde{R}$. Since $R \subseteq \widetilde{R}$ (by Lemma 4(a)) and plainly $R \in \bigcup \Phi_\varphi^{\neq}$, then $b_{\widetilde{R}} \notin \bigcup MR$, by (4). On the other hand, by Lemma 4(f), $L \not\subseteq \widetilde{R}$, hence $b_{\widetilde{R}} \in \bigcup ML$, again by (4), and therefore $\bigcup ML \neq \bigcup MR$, concluding the proof of the theorem. □

Example 1. As will turn out, the BST$(\cup, =, \neq)$-formula

$$x = y \cup z \ \wedge \ z \cup w = k \cup h \ \wedge \ x \cup w \neq y \cup z, \tag{†}$$

is satisfiable and a satisfying model can be constructed by our technique.

Let $L := \{x, w\}$ and $R := \{y, z\}$. By Theorem 1, the above formula is satisfiable if and only if $\widetilde{L} \neq \widetilde{R}$. Since $\widetilde{L} = \{x, y, z, w, k, h\}$ while $\widetilde{R} = \{x, y, z\}$, they indeed are different, which shows that the above formula is satisfiable.

In order to find a set assignment satisfying (†), we can follow the proof of Theorem 1. Let a, b be any two different sets, for example $a = \{\{\emptyset\}\}$ and $b = \{\emptyset, \{\emptyset\}\}$. Assign a and b to $\widetilde{L}$ and $\widetilde{R}$, respectively. By the procedure discussed within the proof of Theorem 1, we obtain the following set assignment M:

$$Mx = My = Mz := \emptyset, \quad Mw = Mh = Mk = \{b\}.$$

To see that M duly models (†), observe that:

$$Mx = \emptyset = My \cup Mz,$$
$$Mz \cup Mw = \{b\} = Mh \cup Mk,$$
$$Mx \cup Mw = \{b\} \neq \emptyset = My \cup Mz. \qquad \dashv$$

The following result, analogous to Theorem 1, is proved by means of an argument dual to the one used above:

Theorem 2. *Let φ be a* BST$(\cap, =, \neq)$*-formula. Then φ is satisfiable if and only if $L \nsim_\varphi R$ (namely $\widetilde{L} \neq \widetilde{R}$) holds for every literal of the form $\bigcap L \neq \bigcap R$ in φ.*

2.1. A cubic-time satisfiability test for BST$(\cup, =, \neq)$ and for BST$(\cap, =, \neq)$

Theorems 1 and 2 imply that any BST$(\star, =, \neq)$-formula φ, where $\star \in \{\cup, \cap\}$, is satisfiable if and only if $\widetilde{L} \neq \widetilde{R}$ holds for every literal $\bigstar L \neq \bigstar R$ in φ. Hence, they yield straight decision procedures for both of the theories BST$(\star, =, \neq)$.

The next step will then be to provide a quadratic algorithm for computing the closure $\widetilde{Z}$ of any input $Z \in \mathscr{P}^+(Vars(\varphi))$, namely, the largest set in $\mathscr{P}^+(Vars(\varphi))$ which is $\sim_\varphi$-equivalent to Z.

In Algorithm 1 below, we provide a high-level specification of the function Closure, intended to compute the closure $\widetilde{Z}$ of any given $Z \in \mathscr{P}^+(Vars(\varphi))$, for a BST$(\star, =, \neq)$-formula φ. After proving its correctness, we will illustrate a lower-level implementation whose time complexity is *quadratic* (as opposed to the cubic-time complexity which would ensue from a naive implementation).

Algorithm 1 Satisfiability tester.

Require: A BST$(\star, =, \neq)$-formula φ represented by the sets of pairs $\Phi_\varphi^{=}$ and $\Phi_\varphi^{\neq}$.
Ensure: Is φ satisfiable?

```
for each {L, R} in Φ_φ^≠ do
    if Closure(L, Φ_φ^=) = Closure(R, Φ_φ^=) then
        return false;
return true;
```

```
function Closure(Z, Φ_φ^=)
    Input: a set Z and the set Φ_φ^=
    Output: the ∼_φ-closure 𝒵 of Z
    𝒵 ← Z;
    while there exists {L, R} ∈ Φ_φ^= such that L ⊆ 𝒵 ⟺ R ⊈ 𝒵 do
        𝒵 ← 𝒵 ∪ L ∪ R;
    return 𝒵;
```

We can now prove quite easily the correctness of the function Closure.

Lemma 5. *The function* Closure *computes closures correctly.*

Proof. Given a $\mathsf{BST}(\star, =, \neq)$-formula φ, with input a set $Z \subseteq \mathit{Vars}(\varphi)$ and the collection $\Phi_\varphi^=$, the while-loop of the function Closure plainly terminates within a number $k \leqslant |\Phi_\varphi^=|$ of iterations. Let $\mathcal{Z}_i$ be the value of the variable $\mathcal{Z}$ after i iterations, so that $\mathcal{Z}_0 = Z$. Preliminarily, we prove by induction on $i = 0, 1, \ldots, k$ that $\mathcal{Z}_i \sim_\varphi Z$ and $Z \subseteq \mathcal{Z}_i$.

The base case $i = 0$ is trivial.

Next, let $\{L, R\} \in \Phi_\varphi^=$ be the pair selected by the while-loop during its i-th iteration, with $i \geqslant 1$. Hence, we have:

$$L \sim_\varphi R, \quad L \subseteq \mathcal{Z}_{i-1} \iff R \nsubseteq \mathcal{Z}_{i-1}, \quad \text{and } \mathcal{Z}_i = \mathcal{Z}_{i-1} \cup L \cup R.$$

Thus, by inductive hypothesis and Lemma 4(g), we have

$$Z \sim_\varphi \mathcal{Z}_{i-1} \sim_\varphi \mathcal{Z}_{i-1} \cup L \cup R = \mathcal{Z}_i \quad \text{and} \quad \mathcal{Z}_i = \mathcal{Z}_{i-1} \cup L \cup R,$$

from which $Z \sim_\varphi \mathcal{Z}_i$ and $Z \subseteq \mathcal{Z}_i$ follow. Hence, by induction, we have $Z \sim_\varphi \mathcal{Z}^{\mathrm{f}}$ and $Z \subseteq \mathcal{Z}^{\mathrm{f}}$, where $\mathcal{Z}^{\mathrm{f}} \coloneqq \mathcal{Z}_k$ is the final value of the variable $\mathcal{Z}$ returned by the execution of Closure$(Z, \Phi_\varphi^=)$.

In addition, the terminating condition for the while-loop yields that $L \subseteq \mathcal{Z}^{\mathrm{f}} \iff R \subseteq \mathcal{Z}^{\mathrm{f}}$, for all $\{L, R\} \in \Phi_\varphi^=$. Thus, from Lemma 3 it follows that

$$W_1 \subseteq \mathcal{Z}^{\mathrm{f}} \iff W_2 \subseteq \mathcal{Z}^{\mathrm{f}}, \tag{5}$$

for all $W_1, W_2 \in \mathscr{P}^+(\mathit{Vars}(\varphi))$ such that $W_1 \sim_\varphi W_2$.

In order to prove that $\mathcal{Z}^{\mathrm{f}} = \widetilde{Z}$, we observe that, since $Z \sim_\varphi \widetilde{Z}$ and $Z \subseteq \mathcal{Z}^{\mathrm{f}}$, by (5) we have $\widetilde{Z} \subseteq \mathcal{Z}^{\mathrm{f}}$. Moreover, by Lemma 4(a),(d) and since $Z \sim_\varphi \mathcal{Z}^{\mathrm{f}}$, we readily get $\mathcal{Z}^{\mathrm{f}} \subseteq \widetilde{\mathcal{Z}^{\mathrm{f}}} = \widetilde{Z}$. The latter inclusion, together with the previously established one $\widetilde{Z} \subseteq \mathcal{Z}^{\mathrm{f}}$, implies $\mathcal{Z}^{\mathrm{f}} = \widetilde{Z}$, i.e., $\mathcal{Z}^{\mathrm{f}}$ is the closure of Z, proving that the call to Closure$(Z, \Phi_\varphi^=)$ computes the closure $\widetilde{Z}$ of Z correctly. □

Theorems 1 and 2, together with Lemma 5 and Lemma 4(d), readily yield that Algorithm 1 is a valid satisfiability test for formulae in the languages $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$.

2.1.1. A quadratic implementation of the function Closure

Next, we provide a quadratic implementation of the function Closure, which, for a given $\mathsf{BST}(\star, =, \neq)$-formula φ, takes as input the collection $\Phi_\varphi^=$ and a set $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$ of which one wants to compute the closure $\widetilde{Z}$. As internal data structures, the function Closure uses: a doubly linked list Ripe of sets of the form $(L \cup R) \setminus \mathcal{Z}$, where $\{L, R\} \in \Phi_\varphi^=$ and $\mathcal{Z}$ is the internal variable whose value will converge to $\widetilde{Z}$ at termination; a doubly linked list Unripe of pairs of form $\langle (L \setminus \mathcal{Z}), (R \setminus \mathcal{Z}) \rangle$, with $\{L, R\} \in \Phi_\varphi^=$; and an array Aux of m lists of pointers to nodes either in Ripe or in Unripe, where m is the number of the distinct variables $x_1, \ldots, x_m$ in φ, intended to support fast retrieval of nodes in the lists Ripe and Unripe.

We will express the complexity of the main procedure in Algorithm 1 and of our efficient implementation of the function Closure in terms of the four quantities m, n, p, q, where $m \coloneqq |\mathit{Vars}(\varphi)|$ is the number of distinct set variables in φ, $n \coloneqq |\varphi|$ is the size of φ (intended as the total number of symbol occurrences in φ), $p \coloneqq |\Phi_\varphi^=|$ is the number of literals of the form $(=)$ in φ, and $q \coloneqq |\Phi_\varphi^\neq|$ is the number of literals of the form $(\neq)$ in φ. Plainly, we have $m, p, q \leqslant n$.

Much as in [6], we can index the variables in $\mathit{Vars}(\varphi)$ from 1 to $m = |\mathit{Vars}(\varphi)|$, so that every subset A of $\mathit{Vars}(\varphi)$ can be represented as a Boolean array of size m such that any set variable x_i belongs to A if and only if $A[i] = 1$. In fact, it is possible to build such an index and initialize accordingly all the arrays corresponding to the collections of set variables present in $\bigcup \Phi_\varphi^= \cup \bigcup \Phi_\varphi^\neq$ in $O(m(p+q)+n)$ time, even starting from a formula φ in plain text, yet to be parsed. Specifically, for each literal ℓ in φ of the form $\bigstar L = \bigstar R$ or $\bigstar L \neq \bigstar R$, we let π_L and π_R be pointers to the sets L and R, respectively. Then, while parsing the formula φ, we construct the collection of pairs

$$\Pi \coloneqq \left\{ \langle x, \pi_A \rangle \mid x \in A \in \bigcup \Phi_\varphi^= \cup \bigcup \Phi_\varphi^\neq \right\}$$

and sort it in $O(n)$ time, according to the first components, using the lexicographic sorting algorithm of strings of varying length as described in [1, Algorithm 3.2 (pp. 80–84)].

Having sorted Π, we can easily collect the $m \leqslant n$ distinct variables of φ and index them using the integers $1, \ldots, m$. By means of such an indexing, in $O(m(p+q))$ time (where $p = |\Phi_\varphi^=|$ and $q = |\Phi_\varphi^\neq|$) we can represent as an m-bit array each set of variables in $\bigcup \Phi_\varphi^= \cup \bigcup \Phi_\varphi^\neq$, and accordingly represent $\Phi_\varphi^=$ and $\Phi_\varphi^\neq$ as lists of pairs of m-bit arrays. Such preliminary encoding phase can be carried out in $O(m(p+q)+n)$ time.

We make use of the auxiliary functions Add(List,S) and Remove(List,Ptr) to add S to List (S can be either a set or a pair of sets) and to remove the node pointed to by Ptr from List, respectively. Since the two lists we use, namely Ripe and

```
1:  function CLOSURE(Z, Φ=_φ)
2:      Z ← Z;
3:      for each {L, R} ∈ Φ=_φ do
4:          if L ∪ R ⊈ Z then
5:              if L ⊆ Z or R ⊆ Z then PTR ← ADD(RIPE, (L ∪ R) \ Z);
6:                  ▷ PTR is a pointer to the node just added to the list RIPE ◁
7:              else PTR ← ADD(UNRIPE, ⟨(L \ Z), (R \ Z)⟩);
8:              for each index i such that x_i ∈ L ∪ R do ADD(AUX[i], PTR);
9:      while RIPE is not empty do
10:         S ← EXTRACT(RIPE);                          ▷ Extracts the first set in RIPE
11:         for each index i such that x_i ∈ S do
12:             Z ← Z ∪ {x_i};
13:             for each pointer PTR in AUX[i] do
14:                 if PTR is in RIPE then              ▷ PTR points to the set PTR.DATA
15:                     PTR.DATA ← PTR.DATA \ {x_i};
16:                     if PTR.DATA = ∅ then REMOVE(RIPE, PTR);
17:                 else                                ▷ PTR points to the pair PTR.DATA in the list UNRIPE
18:                     L_PTR ← PTR.DATA.LEFT ← PTR.DATA.LEFT \ {x_i};
19:                     R_PTR ← PTR.DATA.RIGHT ← PTR.DATA.RIGHT \ {x_i};
20:                     if L_PTR = ∅ or R_PTR = ∅ then
21:                         REMOVE(UNRIPE, PTR);
22:                         if L_PTR ≠ ∅ then ADD(RIPE, L_PTR)
23:                         else if R_PTR ≠ ∅ then ADD(RIPE, R_PTR)
24:                 REMOVE(AUX[i], PTR);
25:     return Z;
```

UNRIPE, are maintained as doubly linked lists, both operations can be performed in $O(1)$ time. The function ADD returns also a pointer to the newly inserted node. Finally, we use the function EXTRACT(LIST) to access in $O(1)$ time the pointer to the first node of LIST while removing it.

The function CLOSURE comprises two phases: an initialization phase, lines 2–8, and a computation phase, lines 9–25.

For each m-bit array, we maintain a counter of its bits set to 1, so that emptiness tests can be performed in $O(1)$ time. Plainly, unions, set differences, and inclusion tests of sets represented as m-bit arrays can easily be performed in $O(m)$ time. Also, membership tests and the operations of singleton addition and singleton removal can be performed in $O(1)$ time.

Thus, the initialization phase of the function CLOSURE (lines 2–8) can be performed in $O(mp)$ time.

At the end of the initialization phase and at each subsequent step, the lists RIPE and UNRIPE contain only sets disjoint from $\mathcal{Z}$, and each of them has length at most $p = |\Phi^{=}_{\varphi}|$. Specifically, the list RIPE contains the set $(L \cup R) \setminus \mathcal{Z}$, for all $\{L, R\} \in \Phi^{=}_{\varphi}$ such that $L \cup R \nsubseteq \mathcal{Z}$ but either $L \subseteq \mathcal{Z}$ or $R \subseteq \mathcal{Z}$ holds. Instead, the list UNRIPE contains the pair $\langle L \setminus \mathcal{Z}, R \setminus \mathcal{Z}\rangle$, for all $\{L, R\} \in \Phi^{=}_{\varphi}$ such that $L \nsubseteq \mathcal{Z}$ and $R \nsubseteq \mathcal{Z}$ both hold.

In view of the assignments at lines 15, 18, and 19, the disjointness property from $\mathcal{Z}$ of the sets $(L \cup R) \setminus \mathcal{Z}$ in RIPE and the sets $L \setminus \mathcal{Z}$ and $R \setminus \mathcal{Z}$ such that $\{L \setminus \mathcal{Z}, R \setminus \mathcal{Z}\}$ is in UNRIPE is maintained at each iteration of the **while**-loop at lines 9–24. Hence, at each extraction of a set S from the list RIPE at line 10, none of the set variables in S has already been selected and processed by the **for**-loop 11–24. Therefore, each set variable in $Vars(\varphi)$ is processed by the **for**-loop at lines 11–24 at most once, yielding that, in the overall, the **for**-loop 11–24 is executed at most m times, which amounts to a total $O(mp)$ time, since each execution of the **for**-loop 11–24, say relative to a set variable x_i, is dominated by the time taken by the internal **for**-loop 13–24, which is $O(p)$. Indeed, (i) at the end of the initialization phase, the list AUX[i] contains at most p pointers to nodes in the lists RIPE and UNRIPE; (ii) once a pointer in the list AUX[i] is processed, it is then removed (line 24), so that it will never be processed again; and (iii) each line of the **for**-loop 13–24 can be executed in constant time.

Since each extraction at line 10 takes $O(1)$ time and, as observed, the list RIPE has length at most p at the end of the initialization phase, it follows that the **while**-loop at lines 9–24 takes $O(mp)$ time.

Thus, the overall complexity of the function CLOSURE is $O(mp)$ time. Since $m, p \leqslant n$, we have also that the function CLOSURE takes quadratic time $O(n^2)$ in the size n of the formula φ being tested for satisfiability.

Finally, our satisfiability tester (Algorithm 1) checks at most q pairs $\{L, R\} \in \Phi^{\neq}_{\varphi}$ in $O(mpq)$ time, by computing the closures $\widetilde{L}$ and $\widetilde{R}$ by means of calls to the function CLOSURE and comparing them. By taking into account also the preliminary encoding phase, which has a $O(m(p+q)+n)$-time complexity, the overall complexity of Algorithm 1 is $O(mpq+n)$, which is $O(n^3)$ since $m, p, q \leqslant n$.

Concerning the space complexity of our satisfiability tester, it is immediate to check that all data structures used in Algorithm 1 and in the function CLOSURE require $O(mp)$ space, namely $O(n^2)$ space since $m, p \leqslant n$.

Summarizing, we have proved the following result:

Theorem 3. *The satisfiability decision problem for the language* $\mathsf{BST}(\cup, =, \neq)$*, as well as for* $\mathsf{BST}(\cap, =, \neq)$*, can be solved in cubic time and quadratic space.*

Remark 1. It is not hard to see that our satisfiability tester (Algorithm 1) and its auxiliary function CLOSURE can be refined in order that an explicit set assignment modeling the input conjunction φ is returned when φ is satisfiable.

3. Extensions of BST$(\cup, =, \neq)$ and BST$(\cap, =, \neq)$

Let us begin by recalling here two notions, called *existential expressibility* and $O(f)$*-expressibility* (stemming from [4] and from [3]), through which one can reduce, in constant or $O(f)$ time, a set-theoretic formula of a richer language to an equisatisfiable formula belonging to a smaller language. Those notions rely on techniques for replacing formulae that involve operators or relators not belonging to the smaller language with convenient surrogates, only comprising operators of the smaller language.

If the target (smaller) language has a decidable satisfiability problem, also the source (larger) language does, because a satisfiability-preserving preprocessing transforms every instance of the latter problem into an instance of the former. We must also insist on fast reduction techniques—this is why $O(f)$ enters into play—, because the algorithmic complexity of the preprocessing phase should not overwhelm the cost of the satisfiability test.

Most of our reductions are based on the following notion of 'context-free' expressibility:

Definition 1 *(Existential expressibility).* A formula $\psi(\vec{x})$ is said to be *existentially expressible* in a theory $\mathcal{T}$ if there exists a $\mathcal{T}$-formula $\Psi(\vec{x}, \vec{z})$ such that

$$\models \ \psi \iff (\exists \vec{z})\, \Psi,$$

where $\vec{x}$ and $\vec{z}$ stand for disjoint tuples of set variables. ⊣

We also devised a more general notion of 'context-sensitive' expressibility, embodying complexity-related information.

Definition 2 *($O(f)$-expressibility).* Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be any theories and $f : \mathbb{N} \to \mathbb{N}$ be a given map. A collection C of formulae is $O(f)$*-expressible from* $\mathcal{T}_1$ *into* $\mathcal{T}_2$ (or $O(f)$*-expressible in* $\mathcal{T}_1$, when $\mathcal{T}_1 = \mathcal{T}_2$) if a map computable in $O\big(f(|\varphi \wedge \psi|)\big)$ time

$$\big\langle \varphi(\vec{y}), \psi(\vec{x}) \big\rangle \mapsto \ \Xi_\varphi^\psi(\vec{x}, \vec{y}, \vec{z}) \tag{6}$$

from $\mathcal{T}_1 \times C$ into $\mathcal{T}_2$ exists, where no variable in $\vec{z}$ occurs in either $\vec{x}$ or $\vec{y}$, such that the following two conditions are satisfied for all φ in $\mathcal{T}_1$ and ψ in C:

(a) if $\varphi \wedge \Xi_\varphi^\psi$ is satisfiable, so is $\varphi \wedge \psi$,
(b) $\models \big(\varphi \wedge \psi\big) \longrightarrow (\exists \vec{z})\, \Xi_\varphi^\psi$. ⊣

Remark 2. An immediate consequence of conditions (a) and (b) in Definition 2 is that the conjunctions $\varphi \wedge \psi$ and $\varphi \wedge \Xi_\varphi^\psi$ are equisatisfiable, for all φ in $\mathcal{T}_1$ and ψ in C. ⊣

Regarding the languages BST$(\cup, =, \neq)$ and BST$(\cap, =, \neq)$, we have:

(a) the literal $x = \varnothing$ is $O(n)$-expressible in BST$(\cup, =, \neq)$;
(b) the literal $x = \varnothing$ is $O(n)$-expressible in BST$(\cap, =)$[3];
(c) the literal $x \subseteq y$ is existentially expressible in BST$(\cup, =)$ and in BST$(\cap, =)$;
(d) the literal $x \not\subseteq y$ is existentially expressible in BST$(\cup, \neq)$ and in BST$(\cap, \neq)$;
(e) the literal DISJ(x, y) is existentially expressible in BST$(\cap, =\varnothing)$ and therefore, by (b), it is $O(n)$-expressible in BST$(\cap, =)$;
(f) the literal ¬DISJ(x, y) is existentially expressible in BST$(\subseteq, \neq)$; therefore, by (c), it is existentially expressible in both of BST$(\cup, =, \neq)$ and BST$(\cap, =, \neq)$.

Wrapping up, we have that (a), (c), (d), and (f) ensure that any BST$(\cup, =\varnothing, \neq\varnothing, \neg\text{DISJ}, \subseteq, \not\subseteq, =, \neq)$-formula can be reduced in linear time to an equisatisfiable BST$(\cup, =, \neq)$-formula. Similarly (b), (c), (d), and (e), (f) ensure that any BST$(\cap, =\varnothing, \neq\varnothing, \text{DISJ}, \neg\text{DISJ}, \subseteq, \not\subseteq, =, \neq)$-formula can be reduced in linear time to an equisatisfiable BST$(\cap, =, \neq)$-formula. Therefore:

Lemma 6. *The satisfiability decision problem for either one of the languages* BST$(\cup, =\varnothing, \neq\varnothing, \neg\text{DISJ}, \subseteq, \not\subseteq, =, \neq)$, BST$(\cap, =\varnothing, \neq\varnothing, \text{DISJ}, \neg\text{DISJ}, \subseteq, \not\subseteq, =, \neq)$ *can be solved in cubic time.*

[3] The proofs of (a) and of (c)–(f) appear in [4, Appendix A]; the proof of (b), which is new, is provided further on in this section (see Lemma 8).

Remark 3. In [4] it is also proved that

$$\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\textsc{Disj}, \subseteq, \not\subseteq, =, \neq) \quad \text{and} \quad \mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \textsc{Disj}, \neg\textsc{Disj}, \subseteq, \not\subseteq, =, \neq)$$

are polynomial-maximal; that is, every language that consists of all conjunctions of literals involving a selection from the set

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

of operators/relators, and strictly contains either one of the said two languages, has an NP-hard satisfiability decision problem. ⊣

We close this section with a highlight of the reduction technique based on $O(f)$-expressibility, by proving that the literal $x=\varnothing$ is $O(n)$-expressible in $\mathsf{BST}(\cap,=)$, albeit not existentially expressible in $\mathsf{BST}(\cap,=,\neq)$.

Lemma 7. *The literal $x=\varnothing$ is not existentially expressible in* $\mathsf{BST}(\cap,=,\neq)$.

Proof. Assume by way of contradiction that $x=\varnothing$ is existentially expressible in $\mathsf{BST}(\cap,=,\neq)$, so that there exists a $\mathsf{BST}(\cap,=,\neq)$-formula $\Psi(x,\vec{z})$, where the variable x does not occur in $\vec{z}$, such that

$$\models x=\varnothing \iff (\exists\vec{z})\,\Psi\,. \tag{7}$$

Let M be the set assignment over $\{x\}$ such that $Mx=\emptyset$, so that $M\models x=\varnothing$. Then, by (7), we also have $\models(\exists\vec{z})\,\Psi$, and so M can be extended over the variables $\vec{z}$ in such a way that $M\models\Psi$. Let c be any set not belonging to $\bigcup_{v\in Vars(\Psi)} Mv$, and consider the set assignment M' over $Vars(\Psi)$ such that

$$M'v = Mv\cup\{c\}, \quad \text{for every } v\in Vars(\Psi).$$

We claim that $M'\models\Psi$. Indeed, for each $V\in\mathscr{P}^+(Vars(\Psi))$, we have $M'\bigcap V=\bigcap M'V=(\bigcap MV)\cup\{c\}$ and so

$$M'\models\bigcap L=\bigcap R \iff M\models\bigcap L=\bigcap R\,,$$

for all $L,R\in\mathscr{P}^+(Vars(\Psi))$.

Thus, since $M\models\Psi$ and Ψ is a conjunction of literals of the forms

$$\bigcap L=\bigcap R \quad\text{and}\quad \bigcap L\neq\bigcap R\,,$$

with $L,R\in\mathscr{P}^+(Vars(\Psi))$, it readily follows that $M'\models\Psi$ as well, and so $M'\models(\exists\vec{z})\,\Psi$.

On the other hand, since $M'x=\{c\}$, we have $M'\not\models x=\varnothing$, and therefore, by (7), we must also have $M'\not\models(\exists\vec{z})\,\Psi$, which is a contradiction. Hence, our claim follows. □

Lemma 8. *The literal $x=\varnothing$ is $O(n)$-expressible from* $\mathsf{BST}(\cap,=)$ *into* $\mathsf{BST}(\cap,=)$.

Proof. Consider the mapping

$$\langle\varphi, x=\varnothing\rangle \ \mapsto\ \big(x\cap\bigcap Vars(\varphi)\big)=x, \tag{8}$$

defined for all $\varphi\in\mathsf{BST}(\cap,=)$, which can be plainly computed in time $O(|\varphi|)$. We show that conditions (a) and (b) of Definition 2 are satisfied. Thus, let φ be any conjunction in $\mathsf{BST}(\cap,=)$.

Concerning condition (a), let M be any model for $\varphi\wedge\big(x\cap\bigcap Vars(\varphi)\big)=x$. Then we have $Mx=Mx\cap My$, namely $Mx\subseteq My$, for all $y\in Vars(\varphi)$. Put

$$M'v := Mv\setminus Mx,$$

for every $v\in Vars(\varphi)\cup\{x\}$. Plainly $M'x=Mx\setminus Mx=\emptyset$, so that $M'\models x=\varnothing$. Having shown that M' satisfies the literal $x=\varnothing$, next we prove that it also satisfies all the literals of φ. Consider any literal $\bigcap L=\bigcap R$ in φ. Since $\bigcap_{v\in L}Mv=\bigcap_{v\in R}Mv$, we readily have:

$$\bigcap M'L=\bigcap_{v\in L}M'v=\bigcap_{v\in L}(Mv\setminus Mx)=\Big(\bigcap_{v\in L}Mv\Big)\setminus Mx=\Big(\bigcap_{v\in R}Mv\Big)\setminus Mx=\bigcap_{v\in R}(Mv\setminus Mx)=\bigcap_{v\in R}M'v=\bigcap M'R,$$

proving that M' models correctly every literal of φ. Hence, $M'\models\varphi\wedge\psi$, and so $\varphi\wedge\psi$ is satisfiable.

Next, as for (b), it is enough to prove that any model M for $\varphi\wedge x=\varnothing$ is also a model for the literal $\big(x\cap\bigcap Vars(\varphi)\big)=x$. But this follows at once, since $Mx=\emptyset$ and so

$$M\big(x\cap\bigcap Vars(\varphi)\big)=Mx\cap M\bigcap Vars(\varphi)=\emptyset=Mx.$$

Whence the claim follows. □

4. Satisfiability in BST(∪, ⊄, ≠) and in its dual BST(∩, ⊄, ≠)

In this section we address the satisfiability problem for the theories $\mathsf{BST}(\cup, \not\subset, \neq)$ and $\mathsf{BST}(\cap, \not\subset, \neq)$ that consist of all the conjunctions of literals of the two forms

$$(\not\subset) \quad x_1 \star \cdots \star x_h \not\subset y_1 \star \cdots \star y_k$$
$$(\neq) \quad u_1 \star \cdots \star u_m \neq v_1 \star \cdots \star v_p,$$

where $h, k, m, p \geqslant 1$, $\star$ is a fixed dyadic set operation, and the x's, y's, u's, and v's are set variables. Specifically, by instantiating $\star$ as $\cup$ we obtain $\mathsf{BST}(\cup, \not\subset, \neq)$, while by instantiating $\star$ as $\cap$ we obtain $\mathsf{BST}(\cap, \not\subset, \neq)$.

On the one hand, in view of the equivalences

$$x = y \iff x \not\subset x \cup y \ \wedge \ y \not\subset x$$
$$x = y \iff x \cap y \not\subset x \ \wedge \ y \not\subset x,$$

the theories $\mathsf{BST}(\cup, \not\subset, \neq)$ and $\mathsf{BST}(\cap, \not\subset, \neq)$ can be regarded as extensions of $\mathsf{BST}(\cup, =, \neq)$ and of $\mathsf{BST}(\cap, =, \neq)$, respectively. On the other hand, the relation $x \not\subset y$, which amounts to $x \not\subseteq y \ \vee \ x = y$, embodies a disjunction; we will prove that, notwithstanding, both of the theories $\mathsf{BST}(\cup, \not\subset, \neq)$ and $\mathsf{BST}(\cap, \not\subset, \neq)$ admit a deterministic polynomial-time decision procedure, albeit of a degree greater than cubic, as was the degree for $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$.

4.1. The equivalence relation $\backsim_\varphi$ and the $\backsim_\varphi$-closure operator for BST(∪, ⊄, ≠)-conjunctions

We suitably tailor to $\mathsf{BST}(\cup, \not\subset, \neq)$ the equivalence relation $\sim_\varphi$ and related closure operator introduced in Section 1 in connection with the satisfiability problem for $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$, thus obtaining a new equivalence relation $\backsim_\varphi$ and its related closure operator $\widehat{\ \cdot\ }$.

Any conjunction φ of $\mathsf{BST}(\cup, \not\subset, \neq)$ can be conveniently represented by the following two sets of ordered and unordered pairs, respectively,

$$\Phi_\varphi^{\not\subset} \coloneqq \Big\{ \big\langle \{x_1, \cdots, x_h\}, \{y_1, \cdots, y_k\} \big\rangle \mid x_1 \cup \cdots \cup x_h \not\subset y_1 \cup \cdots \cup y_k \text{ is in } \varphi \Big\},$$
$$\Phi_\varphi^{\neq} \coloneqq \Big\{ \big\{ \{u_1, \cdots, u_m\}, \{v_1, \cdots, v_p\} \big\} \mid u_1 \cup \cdots \cup u_m \neq v_1 \cup \cdots \cup v_p \text{ is in } \varphi \Big\},$$

where $\langle X, Y \rangle$ stands for the ordered pair of the sets X and Y and the doubleton $\{X, Y\}$ represents the unordered pair of X and Y.

The set $\Phi_\varphi^{\not\subset}$ must consist of ordered pairs, since the relation $x_1 \cup \cdots \cup x_h \not\subset y_1 \cup \cdots \cup y_k$ is asymmetrical. On the other hand, the symmetry of the relation $u_1 \cup \cdots \cup u_m \neq v_1 \cup \cdots \cup v_p$ allows us to use unordered pairs in $\Phi_\varphi^{\neq}$.

We denote by $\mathsf{dom}(\Phi_\varphi^{\not\subset})$ the collection of all the first components of the ordered pairs in $\Phi_\varphi^{\not\subset}$. We will also be interested in the *field* of $\Phi_\varphi^{\not\subset}$, namely the collection of all the sets of variables that appear as first or second component in any ordered pair in $\Phi_\varphi^{\not\subset}$. By representing ordered pairs $\langle X, Y \rangle$ *à la* Kuratowski, namely by putting $\langle X, Y \rangle \coloneqq \{\{X\}, \{X, Y\}\}$, it is easy to check that the field of $\Phi_\varphi^{\not\subset}$ is equal to the set $\bigcup \bigcup \Phi_\varphi^{\not\subset}$, which to ease the notation will be written as $\bigcup\!\!\bigcup \Phi_\varphi^{\not\subset}$ in the rest of the paper. Similarly, the field of $\Phi_\varphi^{\neq}$ is just equal to the set $\bigcup \Phi_\varphi^{\neq}$.

For any conjunction φ in $\mathsf{BST}(\cup, \not\subset, \neq)$, we define an equivalence relation $\backsim_\varphi$ intended to capture all the equalities $\bigcup L = \bigcup R$, with $L, R \in \mathscr{P}^+(\mathit{Vars}(\varphi))$, entailed by the subconjunction of all the literals of type $(\not\subset)$ in φ (in fact, it will turn out that φ is satisfiable if and only if $L \not\backsim_\varphi R$ for every literal $\bigcup L \neq \bigcup R$ of type $(\neq)$ in φ).

Specifically, $\backsim_\varphi$ is the smallest equivalence relation $\backsim$ on $\mathscr{P}^+(\mathit{Vars}(\varphi))$ satisfying the following two closure conditions:

(Cl1*) if $\langle L, R \rangle \in \Phi_\varphi^{\not\subset}$ and $L \subseteq V \backsim R$, for some $V \in \mathscr{P}^+(\mathit{Vars}(\varphi))$, then $L \backsim R$;

(Cl2) if $A \backsim B$, then $A \cup C \backsim B \cup C$ for all $A, B, C \in \mathscr{P}^+(\mathit{Vars}(\varphi))$.

In analogy with Section 1, we also define a closure operator $\widehat{\ \cdot\ }$ on $\mathscr{P}^+(\mathit{Vars}(\varphi))$ associated with the relation $\backsim_\varphi$ by putting

$$\widehat{Z} \coloneqq \bigcup \{W \mid W \backsim_\varphi Z\}$$

for every $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$.

As already remarked at the end of Section 1, the properties of the closure operator $\widetilde{\ \cdot\ }$ in Lemmas 1 and 4 depend solely on condition (Cl2). Since the equivalence relation $\backsim_\varphi$ shares the same second closure condition (Cl2) with the relation $\sim_\varphi$ defined in Section 1, the closure operator $\widehat{\ \cdot\ }$ enjoys the same properties listed in Lemmas 1 and 4. For convenience, we collect them in the following lemma, adapted to $\backsim_\varphi$ and $\widehat{\ \cdot\ }$:

Lemma 9. *Let $Z, Z_1, Z_2 \in \mathscr{P}^+(\mathit{Vars}(\varphi))$. Then*

(a) $Z \subseteq \overset{\frown}{Z}$ *and* $Z \backsim_\varphi \overset{\frown}{Z}$;
(b) $\overset{\frown}{Z} = \overset{\frown}{\overset{\frown}{Z}}$;
(c) *if* $Z_1 \backsim_\varphi \overset{\frown}{Z}_2$, *then* $Z_1 \subseteq \overset{\frown}{Z}_2$;
(d) $Z_1 \backsim_\varphi Z_2$ *if and only if* $\overset{\frown}{Z}_1 = \overset{\frown}{Z}_2$;
(e) *if* $Z_1 \subseteq Z_2$, *then* $\overset{\frown}{Z}_1 \subseteq \overset{\frown}{Z}_2$;
(f) $Z_1 \subseteq \overset{\frown}{Z}_2$ *if and only if* $\overset{\frown}{Z}_1 \subseteq \overset{\frown}{Z}_2$;
(g) *if* $Z_1 \subseteq Z$ *or* $Z_2 \subseteq Z$ *holds and* $Z_1 \backsim_\varphi Z_2$, *then* $Z \backsim_\varphi Z \cup Z_1 \cup Z_2$;
(h) *the* $\backsim_\varphi$*-closure* $\overset{\frown}{Z}$ *of* Z *is the largest subset of* $Vars(\varphi)$ *that is* $\backsim_\varphi$*-equivalent to* Z.

Towards the discovery of a necessary and sufficient condition for the satisfiability of $\mathsf{BST}(\cup, \not\subset, \neq)$-formulae φ, we begin by showing that $A \backsim_\varphi B$ is a sufficient condition for the equality $\bigcup A = \bigcup B$ to be entailed by φ.

Lemma 10. *Let* φ *be any formula of* $\mathsf{BST}(\cup, \not\subset, \neq)$, *and let* $A, B \in \mathscr{P}^+(Vars(\varphi))$ *be any pair of* $\backsim_\varphi$*-equivalent sets. Then the equality* $\bigcup A = \bigcup B$ *is satisfied by every model of* φ.

Proof. Let M be any model for φ. In view of the inclusion-minimality of $\backsim_\varphi$, it is sufficient to prove that the equivalence relation $\backsim_{\mathsf{M}}$ over $\mathscr{P}^+(Vars(\varphi))$ defined by

$$X \backsim_{\mathsf{M}} Y \quad \overset{\text{Def.}}{\Longleftrightarrow} \quad \mathsf{M} \bigcup X = \mathsf{M} \bigcup Y$$

satisfies both closure conditions (Cl1*) and (Cl2).

Concerning (Cl1*), let $\langle L, R\rangle \in \Phi^{\not\subset}_\varphi$ and let $C \in \mathscr{P}^+(Vars(\varphi))$ be any superset of L such that $C \backsim_{\mathsf{M}} R$. We need to prove that $L \backsim_{\mathsf{M}} R$ holds. From the very definition of $\backsim_{\mathsf{M}}$ and since $L \subseteq C$, we have $\mathsf{M} \bigcup L \subseteq \mathsf{M} \bigcup C = \mathsf{M} \bigcup R$. In addition, since $\bigcup L \not\subset \bigcup R$ belongs to φ, then $\mathsf{M} \bigcup L \not\subset \mathsf{M} \bigcup R$. Together with $\mathsf{M} \bigcup L \subseteq \mathsf{M} \bigcup R$, the latter relation readily yields $\mathsf{M} \bigcup L = \mathsf{M} \bigcup R$, and therefore $L \backsim_{\mathsf{M}} R$ holds.

Concerning (Cl2), we can reason exactly as in the proof of Lemma 2.

Thus, if $A \backsim_\varphi B$, then $A \backsim_{\mathsf{M}} B$ and therefore $\mathsf{M} \bigcup A = \mathsf{M} \bigcup B$, proving that M satisfies the equality $\bigcup A = \bigcup B$. □

The following theorem suggests a satisfiability test for $\mathsf{BST}(\cup, \not\subset, \neq)$-formulae, analogous to the one in Theorem 1 for $\mathsf{BST}(\cup, =, \neq)$-formulae.

Theorem 4. *Let* φ *be a* $\mathsf{BST}(\cup, \not\subset, \neq)$*-formula. Then* φ *is satisfiable if and only if* $L \not\backsim_\varphi R$ *(namely* $\overset{\frown}{L} \neq \overset{\frown}{R}$*) holds for every literal of type* $\bigcup L \neq \bigcup R$ *in* φ.

Proof. The necessity part of the proof is an immediate consequence of the preceding lemma.

As for the sufficiency part of the proof, let us assume that for each literal $\bigcup L \neq \bigcup R$ in φ we have $L \not\backsim_\varphi R$. We will use such an assumption to construct a set assignment $\overset{\frown}{\mathsf{M}}$ that satisfies φ.

Let us assign a nonempty set $b_{\overset{\frown}{V}}$ to each $\overset{\frown}{V}$ such that $V \in \mathscr{P}^+(Vars(\varphi))$, where the $b_{\overset{\frown}{V}}$'s are assumed to be pairwise distinct. Then we define the set assignment $\overset{\frown}{\mathsf{M}}$ over $Vars(\varphi)$ by putting, for each $x \in Vars(\varphi)$,

$$\overset{\frown}{\mathsf{M}} x := \{b_{\overset{\frown}{V}} \mid x \notin \overset{\frown}{V} \text{ and } V \in \mathscr{P}^+(Vars(\varphi))\}, \tag{9}$$

so that we have

$$\overset{\frown}{\mathsf{M}} \bigcup U := \{b_{\overset{\frown}{V}} \mid U \not\subseteq \overset{\frown}{V} \text{ and } V \in \mathscr{P}^+(Vars(\varphi))\}$$

for every $U \in \mathscr{P}^+(Vars(\varphi))$.

We claim that the assignment $\overset{\frown}{\mathsf{M}}$ satisfies φ.

We begin by proving that $\overset{\frown}{\mathsf{M}}$ satisfies all the literals in φ of type $(\not\subset)$. Thus, let $\bigcup L \not\subset \bigcup R$ be in φ. Either $L \backsim_\varphi R$ or $L \not\backsim_\varphi R$ holds. We prove that if $L \backsim_\varphi R$ then $\overset{\frown}{\mathsf{M}} \bigcup L = \overset{\frown}{\mathsf{M}} \bigcup R$, whereas if $L \not\backsim_\varphi R$ then $\overset{\frown}{\mathsf{M}} \bigcup L \not\subseteq \overset{\frown}{\mathsf{M}} \bigcup R$, so that in any case $\overset{\frown}{\mathsf{M}} \bigcup L \not\subset \overset{\frown}{\mathsf{M}} \bigcup R$ holds.

Assume first that $L \backsim_\varphi R$ holds, and let $b_{\overset{\frown}{V}} \in \overset{\frown}{\mathsf{M}} \bigcup L$, for some $V \in \mathscr{P}^+(Vars(\varphi))$ such that $L \not\subseteq \overset{\frown}{V}$. Then $R \not\subseteq \overset{\frown}{V}$, since otherwise by Lemma 9(f) we would have $L \subseteq \overset{\frown}{L} = \overset{\frown}{R} \subseteq \overset{\frown}{V}$, a contradiction. Hence, $b_{\overset{\frown}{V}} \in \overset{\frown}{\mathsf{M}} \bigcup R$, so that $\overset{\frown}{\mathsf{M}} \bigcup L \subseteq \overset{\frown}{\mathsf{M}} \bigcup R$. Analogously, it can be proved that the reverse inclusion $\overset{\frown}{\mathsf{M}} \bigcup R \subseteq \overset{\frown}{\mathsf{M}} \bigcup L$ must hold, and therefore we have $\overset{\frown}{\mathsf{M}} \bigcup L = \overset{\frown}{\mathsf{M}} \bigcup R$, which yields $\overset{\frown}{\mathsf{M}} \bigcup L \not\subset \overset{\frown}{\mathsf{M}} \bigcup R$.

On the other hand, if $L \not\backsim_\varphi R$, then $L \not\subseteq \overset{\frown}{R}$, otherwise (Cl1*) would yield $L \backsim_\varphi R$. Hence $b_{\overset{\frown}{R}} \in \overset{\frown}{\mathsf{M}} \bigcup L$. In addition, since $R \subseteq \overset{\frown}{R}$ (by Lemma 9(a)) then $b_{\overset{\frown}{R}} \notin \overset{\frown}{\mathsf{M}} \bigcup R$. Thus, $\overset{\frown}{\mathsf{M}} \bigcup L \not\subset \overset{\frown}{\mathsf{M}} \bigcup R$ holds, namely $\overset{\frown}{\mathsf{M}}$ satisfies the literal $\bigcup L \not\subset \bigcup R$.

Next, we prove that the assignment $\overset{\frown}{\mathsf{M}}$ satisfies also the remaining literals of type $(\neq)$ in φ. Thus, let $\bigcup L \neq \bigcup R$ be in φ. Then, just by hypothesis, it holds that $L \not\backsim_\varphi R$, and so $\overset{\frown}{L} \neq \overset{\frown}{R}$. Hence, either $\overset{\frown}{L} \not\subseteq \overset{\frown}{R}$ or $\overset{\frown}{R} \not\subseteq \overset{\frown}{L}$, so that by Lemma 9(f)

either $L \not\subseteq \breve{R}$ or $R \not\subseteq \breve{L}$ holds. If $L \not\subseteq \breve{R}$, then $b_{\widehat{R}} \in \breve{\mathrm{M}} \bigcup L \setminus \breve{\mathrm{M}} \bigcup R$, whereas if $R \not\subseteq \breve{L}$ then $b_{\widehat{L}} \in \breve{\mathrm{M}} \bigcup R \setminus \breve{\mathrm{M}} \bigcup L$. In any case, $\breve{\mathrm{M}} \bigcup L \neq \breve{\mathrm{M}} \bigcup R$, proving that $\breve{\mathrm{M}}$ satisfies the literal $\bigcup L \neq \bigcup R$.

In conclusion, the assignment $\breve{\mathrm{M}}$ satisfies all the literals of φ, proving that φ is satisfiable, and in turn completing the proof of the lemma. □

Remark 4. In the light of (9), a by-product of the proof of the sufficiency part of Theorem 4 is that any satisfiable $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ admits a model $\breve{\mathrm{M}}$ such that $|\breve{\mathrm{M}}\, x| < 2^{|Vars(\varphi)|}$, for every $x \in Vars(\varphi)$.

However, notice that the proof of Theorem 4 requires the existence of $b_{\widehat{V}}$ only for those sets of variables $V \subseteq Vars(\varphi)$ that occur in literals of type $(\neq)$ or as right-hand sides of literals of type $(\not\subset)$. Thus, by letting $\mathcal{P}_\varphi$ be the collection of all such sets V, we could safely replace the definition (9) by the following one:

$$\breve{\mathrm{M}}\, x \coloneqq \{b_{\widehat{V}} \mid x \notin \breve{V} \text{ and } V \in \mathcal{P}_\varphi\},$$

for $x \in Vars(\varphi)$.

As a result, we obtain that any satisfiable $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ admits a model $\breve{\mathrm{M}}$ such that the size of each set $\breve{\mathrm{M}}\, x$ is linearly bounded in the size of φ, rather than merely exponentially bounded in the size of $Vars(\varphi)$. ⊣

Theorem 4 states that in order to check the satisfiability of any $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ we need to verify that each unordered pair $\{L, R\}$ in $\Phi_\varphi^{\neq}$ is formed by sets of variables L and R that are not $\backsim_\varphi$-equivalent. An immediate implementation of such a check would require one to compute the whole relation $\backsim_\varphi$. However, since the field of $\backsim_\varphi$ is the collection $\mathscr{P}^+(Vars(\varphi))$, whose size is exponential, such an approach would take exponential time.

The next section presents a polynomial satisfiability test for $\mathsf{BST}(\cup, \not\subset, \neq)$-formulae.

4.2. A satisfiability test for $\mathsf{BST}(\cup, \not\subset, \neq)$

Rather than computing the whole equivalence relation $\backsim_\varphi$, a more efficient strategy will take advantage of property (d) in Lemma 9, which states that

$$Z_1 \backsim_\varphi Z_2 \iff \breve{Z}_1 = \breve{Z}_2$$

for any two sets $Z_1, Z_2 \in \mathscr{P}^+(Vars(\varphi))$. However, to avoid an exponential cost, we will show that the computation of $\backsim_\varphi$-closures can be confined to the members of $\bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq}$ only. As we will prove, this is just what the function ClosureMap below does, when executed with its second argument set to $\bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq}$. However, for convenience, we first show that the function ClosureMap with inputs a $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ and the whole collection $\mathscr{P}^+(Vars(\varphi))$ calculates the correct $\backsim_\varphi$-closure operator over $\mathscr{P}^+(Vars(\varphi))$. Subsequently, we will argue that the call ClosureMap$(\varphi,\ \mathcal{V})$ (and in particular the call ClosureMap$(\varphi,\ \bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq})$) computes correctly the $\backsim_\varphi$-closure map over the collection in its second argument $\mathcal{V}$, provided that it contains $\bigcup \Phi_\varphi^{\not\subset}$ as a subset. Finally, we will show that the execution of ClosureMap$(\varphi,\ \bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq})$ takes polynomial time in the size of φ.

To be more specific, when called with inputs a $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ and a collection $\mathcal{V} \subseteq \mathscr{P}^+(Vars(\varphi))$ such that $\bigcup \Phi_\varphi^{\not\subset} \subseteq \mathcal{V}$, the function ClosureMap, after initializing each variable $\widehat{A}$ to A (for every $A \in \mathcal{V}$), executes repeatedly expansion assignments of the form

$$\widehat{L} \leftarrow \widehat{L} \cup \widehat{R},$$

where $\langle L, R\rangle$ is any *unsteady pair* in $\mathcal{V} \times \bigcup \Phi_\varphi^{\not\subset}$ (in the sense of Definition 3 below), until no unsteady pairs remain.

Definition 3. A pair $\langle L, R\rangle \in \mathcal{V} \times \bigcup \Phi_\varphi^{\not\subset}$ is *unsteady* if the following two conditions hold:

(a) $\widehat{R} \not\subseteq \widehat{L}$,
(b) $R \not\subseteq \widehat{L} \longrightarrow \big(\langle L, R\rangle \in \Phi_\varphi^{\not\subset} \wedge L \subseteq \widehat{R}\big)$.

4.2.1. Termination

When an assignment $\widehat{L} \leftarrow \widehat{L} \cup \widehat{R}$ at line 5 of ClosureMap is executed, the set $\widehat{L}$ gets new elements, as the pair $\langle L, R\rangle$ is unsteady beforehand and so $\widehat{R} \not\subseteq \widehat{L}$ must hold. It therefore follows that during the execution of ClosureMap$(\varphi, \mathcal{V})$, for any collection $\mathcal{V}$ such that $\bigcup \Phi_\varphi^{\not\subset} \subseteq \mathcal{V} \subseteq \mathscr{P}^+(Vars(\varphi))$, the **while**-loop 4–5 is executed at most the following number of times

$$\sum_{L \in \mathcal{V}} \big(|Vars(\varphi)| - |L|\big) = |\mathcal{V}| \cdot |Vars(\varphi)| - \sum_{L \in \mathcal{V}} |L|, \tag{10}$$

and so, in particular, ClosureMap$(\varphi, \mathcal{V})$ terminates.

Algorithm 2

1: **function** ClosureMap(φ, $\mathcal{V}$)
Input: a BST($\cup, \not\subset, \neq$)-conjunction φ and a collection of sets $\mathcal{V}$ such that $\bigcup \Phi_\varphi^{\not\subset} \subseteq \mathcal{V} \subseteq \mathscr{P}^+(Vars(\varphi))$
Output: the $\backsim_\varphi$-closure operator over $\mathcal{V}$
2: **for** all $A \in \mathcal{V}$ **do**
3: $\widehat{A} \leftarrow A$;
4: **while** there are unsteady pairs $\langle L, R\rangle$ in $\mathcal{V} \times \bigcup \Phi_\varphi^{\not\subset}$ **do**
5: $\widehat{L} \leftarrow \widehat{L} \cup \widehat{R}$; ▷ $\langle L, R\rangle$ *is any such unsteady pair*
6: **return** the map $A \mapsto \widehat{A}$ on $\mathcal{V}$;

We also remark that, since at termination of ClosureMap($\varphi, \mathcal{V}$) no pair $\langle L, R\rangle \in \mathcal{V} \times \bigcup \Phi_\varphi^{\not\subset}$ is unsteady, the following disjunction holds, for every $\langle L, R\rangle \in \mathcal{V} \times \bigcup \Phi_\varphi^{\not\subset}$:

$$\neg(\widehat{R} \not\subseteq \widehat{L}) \ \vee \ \neg\big(R \not\subseteq \widehat{L} \longrightarrow (\langle L, R\rangle \in \Phi_\varphi^{\not\subset} \wedge L \subseteq \widehat{R})\big).$$

The latter disjunction is equivalent to

$$\widehat{R} \subseteq \widehat{L} \ \vee \ \big(R \not\subseteq \widehat{L} \wedge (\langle L, R\rangle \notin \Phi_\varphi^{\not\subset} \vee L \not\subseteq \widehat{R})\big),$$

which in its turn is equivalent to the conjunction

$$(R \subseteq \widehat{L} \longrightarrow \widehat{R} \subseteq \widehat{L}) \ \wedge \ \big(\langle L, R\rangle \in \Phi_\varphi^{\not\subset} \longrightarrow (L \subseteq \widehat{R} \longrightarrow \widehat{R} \subseteq \widehat{L})\big).$$

Thus, at termination of the execution of ClosureMap($\varphi, \mathcal{V}$), the following two terminating conditions hold:

(T_1) $R \subseteq \widehat{L} \longrightarrow \widehat{R} \subseteq \widehat{L}$, for every $\langle L, R\rangle \in \mathcal{V} \times \bigcup \Phi_\varphi^{\not\subset}$;
(T_2) $L \subseteq \widehat{R} \longrightarrow \widehat{R} \subseteq \widehat{L}$, for every $\langle L, R\rangle \in \Phi_\varphi^{\not\subset}$.

4.2.2. Correctness

For convenience, we first prove the correctness of the algorithm ClosureMap when, for a given BST($\cup, \not\subset, \neq$)-formula φ, ClosureMap is called with its second argument set to $\mathscr{P}^+(Vars(\varphi))$. Subsequently, we will argue that, when called with any subcollection $\mathcal{V}$ of $\mathscr{P}^+(Vars(\varphi))$ containing $\bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq}$ as a subset, ClosureMap computes correctly the $\backsim_\varphi$-closure map over $\mathcal{V}$.

To start with, we prove that the map $A \mapsto \widehat{A}$ over $\mathscr{P}^+(Vars(\varphi))$ computed by the execution of ClosureMap(φ, $\mathscr{P}^+(Vars(\varphi))$) is an *approximate* $\backsim_\varphi$*-closure operator*, according to the following definition.

Definition 4. Given a BST($\cup, \not\subset, \neq$)-conjunction φ, an *approximate* $\backsim_\varphi$*-closure operator* is any map $V \mapsto \widehat{V}$ over $\mathscr{P}^+(Vars(\varphi))$ such that the inclusions

$$V \subseteq \widehat{V} \subseteq \widehat{\widehat{V}}$$

hold for every $V \in \mathscr{P}^+(Vars(\varphi))$.

Lemma 11. *The map $A \mapsto \widehat{A}$ over $\mathscr{P}^+(Vars(\varphi))$ computed by the execution of* ClosureMap(φ, $\mathscr{P}^+(Vars(\varphi))$)*, for a given* BST($\cup, \not\subset, \neq$)*-formula φ, is an approximate $\backsim_\varphi$-closure operator.*

Proof. Plainly, we have $A \subseteq \widehat{A}$ for all $A \in \mathscr{P}^+(Vars(\varphi))$, as the sets $\widehat{A}$ grow monotonically during the execution of ClosureMap and initially each $\widehat{A}$ is set to A in the **for**-loop 2–3.

Concerning the inclusions $\widehat{A} \subseteq \widehat{\widehat{A}}$, these hold trivially just after the execution of the initializing **for**-loop 2–3. Thus, it suffices to prove by induction that the inclusions $\widehat{A} \subseteq \widehat{\widehat{A}}$ are preserved after each execution of the assignment at line 5.

Let us first consider the case when $R \not\subseteq \widehat{L}$ holds just before the execution of the assignment under consideration at line 5. Hence, we have

- $\widehat{R} \not\subseteq \widehat{L}$ (by condition (a) of Definition 3)
- $\langle L, R\rangle \in \Phi_\varphi^{\not\subset}$ and $L \subseteq \widehat{R}$ (by condition (b) of Definition 3).

Also, let us assume inductively that $\widehat{L} \subseteq \widehat{\widehat{L}}$ and $\widehat{R} \subseteq \widehat{\widehat{R}}$ hold at that time, so that $L \subseteq \widehat{\widehat{R}}$ holds as well. Then, by (Cl1*), we have $L \backsim_\varphi R$, namely $\widehat{\widehat{L}} = \widehat{\widehat{R}}$, and so the set $\widehat{L} \cup \widehat{R}$ assigned to $\widehat{L}$ is contained in $\widehat{\widehat{L}}$, proving that $\widehat{L} \subseteq \widehat{\widehat{L}}$ continues to hold after the execution of the assignment under consideration.

Next, we consider the case when $R \subseteq \widehat{L}$ holds just before the execution of the assignment under examination at line 5.

Let us inductively assume that the inclusions $\widehat{L} \subseteq \widehat{\widehat{L}}$ and $\widehat{R} \subseteq \widehat{\widehat{R}}$ hold at that time, so that $\widehat{L} \cup \widehat{R} \subseteq \widehat{\widehat{L}} \cup \widehat{\widehat{R}}$ holds as well. From $R \subseteq \widehat{L} \subseteq \widehat{\widehat{L}}$, Lemma 9(f) yields $\widehat{R} \subseteq \widehat{\widehat{R}} \subseteq \widehat{\widehat{L}}$, and so the set $\widehat{L} \cup \widehat{R}$ assigned to $\widehat{L}$ at line 5 is contained in $\widehat{\widehat{L}}$, proving that the inclusion $\widehat{L} \subseteq \widehat{\widehat{L}}$ continues to hold after the execution of the assignment under consideration. □

Next we show that when an approximate closure operator enjoys certain properties, it is indeed a closure operator. Subsequently, we will use such a result to prove that the execution of ClosureMap(φ, $\mathscr{P}^+(Vars(\varphi))$) computes correctly the $\backsim_\varphi$-closure operator.

Lemma 12. *Let φ be a* BST$(\cup, \not\subset, \neq)$*-conjunction, and let $V \mapsto \widehat{V}$ be an approximate $\backsim_\varphi$-closure over $\mathscr{P}^+(Vars(\varphi))$ such that the following conditions hold:*

(i) if $L \subseteq \widehat{R}$ then $\widehat{L} = \widehat{R}$, for $\langle L, R\rangle \in \Phi_\varphi^{\not\subset}$;
(ii) if $U \subseteq V$ then $\widehat{U} \subseteq \widehat{V}$, for $U, V \in \mathscr{P}^+(Vars(\varphi))$;
(iii) $\widehat{V} = \widehat{\widehat{V}}$, for $V \in \mathscr{P}^+(Vars(\varphi))$.

Then the operator $\widehat{\blacksquare}$ coincides with $\breve{\blacksquare}$, namely $\widehat{V} = \breve{V}$ holds for all $V \in \mathscr{P}^+(Vars(\varphi))$.

Proof. Let $\hat{\frown}$ be the equivalence relation defined on $\mathscr{P}^+(Vars(\varphi))$ by setting

$$A \mathbin{\hat{\frown}} B \quad \overset{\text{Def.}}{\Longleftrightarrow} \quad \widehat{A} = \widehat{B}.$$

To begin with, we show that the relation $\hat{\frown}$ satisfies both conditions (Cl1*) and (Cl2), thereby proving by the minimality of $\backsim_\varphi$ the implication

$$A \backsim_\varphi B \quad \longrightarrow \quad A \mathbin{\hat{\frown}} B, \tag{11}$$

for all $A, B \in \mathscr{P}^+(Vars(\varphi))$.

Concerning (Cl1*), let $\langle L, R\rangle \in \Phi_\varphi^{\not\subset}$ and let $V \in \mathscr{P}^+(Vars(\varphi))$ be such that $L \subseteq V$ and $R \mathbin{\hat{\frown}} V$ hold, so that $\widehat{R} = \widehat{V}$. Then we have $L \subseteq V \subseteq \widehat{V} = \widehat{R}$ and therefore, by (i), $L \mathbin{\hat{\frown}} R$ readily follows.

As for the closure condition (Cl2), preliminarily we observe that the implication

$$U \subseteq \widehat{V} \quad \longrightarrow \quad \widehat{U} \subseteq \widehat{V} \tag{12}$$

holds for all $U, V \in \mathscr{P}^+(Vars(\varphi))$. In fact, by (ii) and (iii), $U \subseteq \widehat{V}$ implies $\widehat{U} \subseteq \widehat{\widehat{V}} = \widehat{V}$.

Now, let $A \mathbin{\hat{\frown}} B$ for some $A, B \in \mathscr{P}^+(Vars(\varphi))$, and let $C \in \mathscr{P}^+(Vars(\varphi))$. Plainly, $C \subseteq A \cup C \subseteq \widehat{A \cup C}$ and, analogously, $A \subseteq \widehat{A \cup C}$. Thus, by (12), $\widehat{A} \subseteq \widehat{A \cup C}$ holds and so $B \subseteq \widehat{B} = \widehat{A} \subseteq \widehat{A \cup C}$. Hence, $B \cup C \subseteq \widehat{A \cup C}$, which again by (12) implies $\widehat{B \cup C} \subseteq \widehat{A \cup C}$. Analogously, one can show that the reverse inclusion $\widehat{A \cup C} \subseteq \widehat{B \cup C}$ is valid as well, and so $\widehat{A \cup C} = \widehat{B \cup C}$ holds. Hence, we have $A \cup C \mathbin{\hat{\frown}} B \cup C$, proving that $\hat{\frown}$ satisfies the second closure condition too.

As anticipated, the minimality of $\backsim_\varphi$ readily yields the implication (11). In particular, since $V \backsim_\varphi \breve{V}$ by Lemma 9(a), from (11) we get $V \mathbin{\hat{\frown}} \breve{V}$, and so we have

$$\breve{V} \subseteq \widehat{\breve{V}} = \widehat{V}, \tag{13}$$

for all $V \in \mathscr{P}^+(Vars(\varphi))$.

Finally, recalling that $\widehat{\blacksquare}$ is an approximate $\backsim_\varphi$-closure operator (and so $\widehat{V} \subseteq \breve{V}$ trivially holds), from (13) we get $\widehat{V} = \breve{V}$ for all $V \in \mathscr{P}^+(Vars(\varphi))$, proving the lemma. □

By a direct application of Lemma 12, next we prove that the approximate $\backsim_\varphi$-closure $A \mapsto \widehat{A}$ computed by ClosureMap(φ, $\mathscr{P}^+(Vars(\varphi))$) is indeed the $\backsim_\varphi$-closure operator.

Theorem 5. *For a given* BST$(\cup, \not\subset, \neq)$*-conjunction φ, the map $A \mapsto \widehat{A}$ computed by* ClosureMap(φ, $\mathscr{P}^+(Vars(\varphi))$) *coincides with the $\backsim_\varphi$-closure operator.*

Proof. On the basis of Lemma 12, it is enough to prove that the approximate $\backsim_\varphi$-closure operator $A \mapsto \widehat{A}$ computed by ClosureMap(φ, $\mathscr{P}^+(Vars(\varphi))$) satisfies its conditions (i)–(iii).

Concerning condition (i), let $\langle L, R\rangle \in \Phi_\varphi^{\not\subset}$ and assume that $L \subseteq \widehat{R}$ holds. From the terminating condition (T_2), we readily get $R \subseteq \widehat{R} \subseteq \widehat{L}$, whereas from condition (T_1), but applied to the pair $\langle R, L\rangle$, we have $\widehat{L} \subseteq \widehat{R}$. Hence, $\widehat{L} = \widehat{R}$ follows, proving (i).

As for condition (ii), let U, V be any two members of $\mathscr{P}^+(Vars(\varphi))$ such that $U \subseteq V$. We prove that $\widehat{U} \subseteq \widehat{V}$ holds.

Since $U \subseteq \widehat{V}$, we can immediately rule out the case in which $U \in \mathrm{dom}(\Phi_\varphi^{\not\subset})$, as in this case $\widehat{U} \subseteq \widehat{V}$ follows readily from the terminating condition (T_1).

Thus, let us assume that $U \notin \mathrm{dom}(\Phi_\varphi^{\not\subset})$. In this case, the set $\widehat{U}$ is the result of the execution of a sequence of assignments of the form '$\widehat{U} \leftarrow \widehat{U} \cup \widehat{R}$' at line 5 in ClosureMap (to be called *$\widehat{U}$-assignments*), where $\langle U, R\rangle \in \left(\mathscr{P}^+(Vars(\varphi)) \times \bigcup \Phi_\varphi^{\not\subset}\right) \setminus \Phi_\varphi^{\not\subset}$.

Accordingly, let $\left\{\widehat{U} \leftarrow \widehat{U} \cup \widehat{R}_i\right\}_{i=1,\ldots,k}$ be the sequence of the $\widehat{U}$-assignments in the order in which they are executed by the algorithm ClosureMap during the computation of the map $\widehat{\blacksquare}$. For each $i = 1, \ldots, k$, we denote by $\overline{R}_i^{(i)}$ and $\overline{U}^{(i)}$ the

values of the algorithm's variables '$\widehat{R}_i$' and '$\widehat{U}$', respectively, at the time the i-th $\widehat{U}$-assignment '$\widehat{U} \leftarrow \widehat{U} \cup \widehat{R}_i$' is about to be executed. Then, we plainly have:

$$\overline{U}^{(i)} = U \cup \bigcup_{j=1}^{i-1} \overline{R}_j^{(j)} \subseteq U \cup \bigcup_{j=1}^{i-1} \widehat{R}_j, \tag{14}$$

for each $i = 1, \ldots, k$, and[4]

$$\widehat{U} \subseteq U \cup \bigcup_{j=1}^{k} \widehat{R}_j. \tag{15}$$

Since each pair $\langle U, R_i \rangle$ is unsteady just before the execution of the i-th $\widehat{U}$-assignment '$\widehat{U} \leftarrow \widehat{U} \cup \widehat{R}_i$' and $\langle U, R_i \rangle \notin \Phi_\varphi^{\not\subset}$, condition (b) of Definition 3 yields $R_i \subseteq \overline{U}^{(i)}$. Hence, by (14), we have

$$R_i \subseteq U \cup \bigcup_{j=1}^{i-1} \widehat{R}_j, \quad \text{for } i = 1, \ldots, k. \tag{16}$$

In order to prove the inclusion $\widehat{U} \subseteq \widehat{V}$ (where—recall—we are assuming $U \subseteq V$), it is enough to show that

$$R_i \subseteq \widehat{V}, \quad \text{for } i = 1, \ldots, k. \tag{17}$$

Indeed, in the light of condition (T_1), from (17) it would follow

$$\widehat{R}_i \subseteq \widehat{V}, \quad \text{for } i = 1, \ldots, k,$$

and so from (15) we would get

$$\widehat{U} \subseteq U \cup \bigcup_{i=1}^{k} \widehat{R}_i \subseteq \widehat{V}.$$

We prove (17) by induction on $i = 1, \ldots, k$.
For $i = 1$, by (16) we have

$$R_1 \subseteq U \subseteq V \subseteq \widehat{V}.$$

Next, let us inductively assume that $\bigcup_{j=1}^{i-1} R_j \subseteq \widehat{V}$ holds for $2 \leqslant i \leqslant k$, and show that $R_i \subseteq \widehat{V}$. Since, by condition ($T_1$), $\bigcup_{j=1}^{i-1} \widehat{R}_j \subseteq \widehat{V}$ holds, using (16) we have

$$R_i \subseteq U \cup \bigcup_{j=1}^{i-1} \widehat{R}_j \subseteq \widehat{V},$$

and so, again by condition (T_1),

$$\widehat{R}_i \subseteq \widehat{V}.$$

Thus, by induction and by (15), we have

$$\widehat{U} \subseteq U \cup \bigcup_{i=1}^{k} \widehat{R}_i \subseteq \widehat{V}.$$

Finally, concerning condition (iii) of Lemma 12, let $V \in \mathscr{P}^+(Vars(\varphi))$ and put $W \coloneqq \widehat{V}$. We prove that $\widehat{W} = W$, namely $\widehat{\widehat{V}} = \widehat{V}$.

[4] The inclusion that we are about to prove can be strengthened into

$$\widehat{U} = U \cup \bigcup_{i=1}^{k} \widehat{R}_i. \tag{$\star$}$$

Indeed, since $R_i \subseteq \overline{R}_i^{(i)}$, then by (15) we have $R_i \subseteq \widehat{U}$ and so the terminating condition (T_1) yields $\widehat{R}_i \subseteq \widehat{U}$ for all $i = 1, \ldots, k$, thus proving

$$U \cup \bigcup_{i=1}^{k} \widehat{R}_i \subseteq \widehat{U}.$$

Together with (15), the latter inclusion yields ($\star$).

Arguing by contradiction, assume that $\widehat{W} \neq W$ holds. If $W \in \bigcup \Phi_\varphi^{\not\subset}$, then $\langle V, W\rangle \in \mathscr{P}^+(Vars(\varphi)) \times \bigcup \Phi_\varphi^{\not\subset}$, and so, by condition ($T_1$), at termination we would have

$$\widehat{V} = W \not\subseteq \widehat{W} \subseteq \widehat{V},$$

which is untenable. On the other hand, if $W \notin \bigcup \Phi_\varphi^{\not\subset}$, let '$\widehat{W} \leftarrow \widehat{W} \cup \widehat{R}$' be the first $\widehat{W}$-assignment during the execution of ClosureMap and, much as above, let $\overline{W}$ and $\overline{R}$ be the values of the algorithm's variables '$\widehat{W}$' and '$\widehat{R}$' just before the execution of such $\widehat{W}$-assignment, so that $\overline{W} = W$ plainly holds. As the pair $\langle W, R\rangle$ must be unsteady just before the execution of the $\widehat{W}$-assignment at hand and in this case $\langle W, R\rangle \notin \Phi_\varphi^{\not\subset}$, by conditions (a) and (b) of Definition 3 we must have:

(A) $\overline{R} \not\subseteq \overline{W}$
(B) $R \subseteq \overline{W} = W = \overline{V}$

respectively. Thus, from the terminating condition (T_1), the inclusion (B) yields

$$\overline{R} \subseteq \widehat{R} \subseteq \widehat{V} = W = \overline{W},$$

which contradicts (A).

Since we have reached a contradiction in either case, we must have $\widehat{W} = W$, namely $\widehat{\widehat{V}} = \widehat{V}$, proving (iii) by the arbitrariness of $V \in \mathscr{P}^+(Vars(\varphi))$.

Having shown that the approximate $\backsim_\varphi$-closure operator $A \mapsto \widehat{A}$ satisfies conditions (i)–(iii) of Lemma 12, we may conclude that it coincides with the $\backsim_\varphi$-closure operator. □

The correctness of ClosureMap$(\varphi, \mathscr{P}^+(Vars(\varphi)))$, proved in Theorem 5, readily implies the correctness of ClosureMap$(\varphi, \mathcal{V})$, for every subcollection $\mathcal{V} \subseteq \mathscr{P}^+(Vars(\varphi))$ containing $\bigcup \Phi_\varphi^{\not\subset}$ as a subset. It is enough to observe that the assignment at line 5 of ClosureMap has the form '$\widehat{L} \leftarrow \widehat{L} \cup \widehat{R}$', with $L \in \mathcal{V}$ and $R \in \bigcup \Phi_\varphi^{\not\subset}$. Therefore, only the values $\widehat{R}$ for $R \in \bigcup \Phi_\varphi^{\not\subset}$ are needed for the proper functioning of the computation of ClosureMap$(\varphi, \mathcal{V})$, and these values are available since in our case $\bigcup \Phi_\varphi^{\not\subset} \subseteq \mathcal{V}$ holds.

On the grounds of the above results, the theory $\mathsf{BST}(\cup, \not\subset, \neq)$ admits the following satisfiability test, whose complexity is assessed next.

Algorithm 3 Satisfiability test for $\mathsf{BST}(\cup, \not\subset, \neq)$.

Require: a $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ;
Ensure: is φ satisfiable?
1: let $A \mapsto \widehat{A}$ be the map computed by ClosureMap$(\varphi, \bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq})$;
2: **for** each $\{L, R\} \in \Phi_\varphi^{\neq}$ **do**
3: **if** $\widehat{L} = \widehat{R}$ **then**
4: **return** false;
5: **return** true;

4.2.3. A polynomial implementation of the satisfiability test for $\mathsf{BST}(\cup, \not\subset, \neq)$

For an input $\mathsf{BST}(\cup, \not\subset, \neq)$-formula φ, the time complexity of ClosureMap$(\varphi, \bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq})$ and of Algorithm 3 will be expressed in terms of the following quantities:

- $m \coloneqq |Vars(\varphi)|$ (the number of distinct variables in φ),
- $n \coloneqq |\varphi|$ (the size of φ, namely the total number of symbols in φ),
- $\ell \coloneqq |\Phi_\varphi^{\not\subset}| + |\Phi_\varphi^{\neq}|$ (the total number of literals in φ).

Plainly, we have $m, \ell \leqslant n$.

As in Section 2.1, we assume that the distinct set variables occurring in φ have been indexed from 1 to m, so that each of the relevant subsets V of $Vars(\varphi)$ can be regarded as a bit-array of size m, where $V[i] = 1$ if and only if $x_i \in V$. As remarked in Section 2.1, such an indexing and the subsequent initialization of all the bit-arrays relative to the conjunction φ require $O(m\ell + n)$ time. Notice that once all the relevant sets of variables have been represented as bit-arrays, each operation involving them in ClosureMap and each set-inclusion test can be executed in $O(m)$ time.

The function ClosureMap presents two parts: an *initialization phase* (lines 2–3) and a *construction phase* (lines 4–5). The initialization phase creates just a bit-array of size m for each of the $O(\ell)$ sets of variables in $\bigcup \Phi_\varphi^{\not\subset} \cup \bigcup \Phi_\varphi^{\neq}$, thus taking

$O(m\ell)$ time. Concerning the construction phase, from (10) it follows that the while-loop at lines 4–5 is executed $O(m\ell)$ times. Each iteration takes $O(m)$ time for the assignment at line 5 and $O(m\ell^2)$ time for the evaluation of condition at line 4. Hence, the overall complexity of the construction phase of ClosureMap is $O(m^2\ell^3)$, and therefore the execution cost of line 1 in Algorithm 3 is $O(m^2\ell^3)$ too. The **for**-loop at lines 2–4 of Algorithm 3 is executed $O(\ell)$ times and each execution takes $O(m)$ time, for a total of $O(m\ell)$ time. Hence, the overall time complexity of Algorithm 3 is $O(m^2\ell^3)$.

Summing up, we can conclude that

Theorem 6. *The satisfiability problem for* $\mathsf{BST}(\cup, \not\subset, \neq)$ *can be solved in quintic time.*

4.3. A satisfiability test for $\mathsf{BST}(\cap, \not\subset, \neq)$

The satisfiability problem for the theory $\mathsf{BST}(\cap, \not\subset, \neq)$ can be reduced by duality to that of $\mathsf{BST}(\cup, \not\subset, \neq)$ in linear time. Specifically, for any $\mathsf{BST}(\cap, \not\subset, \neq)$-conjunction ψ, we define its *dual* $\psi^\star$ as the $\mathsf{BST}(\cup, \not\subset, \neq)$-conjunction obtained from ψ by

- replacing each occurrence of the operator '$\cap$' with its dual '$\cup$', and then
- exchanging the left-hand side with the right-hand side in any of the literals of type $(\not\subset)$.

Since dual conjunctions can be constructed in linear time, we obtain that the satisfiability problem for $\mathsf{BST}(\cap, \not\subset, \neq)$ can be solved in quintic time once we prove that each $\mathsf{BST}(\cap, \not\subset, \neq)$-conjunction and its dual are equisatisfiable.

Lemma 13. *A* $\mathsf{BST}(\cap, \not\subset, \neq)$*-conjunction is satisfiable if and only if so is its dual.*

Proof. Let ψ be a $\mathsf{BST}(\cap, \not\subset, \neq)$-conjunction and let M be any set assignment over $Vars(\psi)$. Also, let $D_M \coloneqq \bigcup_{x\in Vars(\psi)} Mx$ be the *domain* of M.

We define the *dual assignment* $M^\star$ of M by setting

$$M^\star x \coloneqq D_M \setminus Mx, \qquad \text{for all } x \in Vars(\psi),$$

and prove that the following equivalence holds:

$$M \models \psi \iff M^\star \models \psi^\star. \tag{18}$$

It is enough to show that (18) holds for every literal of ψ.

Literals of type $\bigcap X \neq \bigcap Y$. We have:

$$\begin{aligned}
M \models \bigcap X \neq \bigcap Y &\longleftrightarrow M \bigcap X \neq M \bigcap Y \\
&\longleftrightarrow \bigcap_{x\in X} Mx \neq \bigcap_{y\in Y} My \\
&\longleftrightarrow D_M \setminus \bigcap_{x\in X} Mx \neq D_M \setminus \bigcap_{y\in Y} My \\
&\qquad (\text{since } \bigcap_{x\in X} Mx, \bigcap_{y\in Y} My \subseteq D_M) \\
&\longleftrightarrow \bigcup_{x\in X} (D_M \setminus Mx) \neq \bigcup_{y\in Y} (D_M \setminus My) \\
&\qquad (\text{by De Morgan's Law of intersection}) \\
&\longleftrightarrow \bigcup_{x\in X} M^\star x \neq \bigcup_{y\in Y} M^\star y \\
&\longleftrightarrow M^\star \bigcup X \neq M^\star \bigcup Y \\
&\longleftrightarrow M^\star \models \bigcup X \neq \bigcup Y \\
&\longleftrightarrow M^\star \models \left(\bigcap X \neq \bigcap Y\right)^\star.
\end{aligned}$$

Literals of type $\bigcap X \not\subset \bigcap Y$. We have:

$$
\begin{aligned}
M \models \bigcap X \not\subset \bigcap Y \;&\longleftrightarrow\; M \bigcap X \not\subset M \bigcap Y \\
&\longleftrightarrow\; \bigcap_{x\in X} Mx \not\subset \bigcap_{y\in Y} My \\
&\longleftrightarrow\; D_M \setminus \bigcap_{y\in Y} My \not\subset D_M \setminus \bigcap_{x\in X} Mx \\
&\qquad (\text{since } \textstyle\bigcap_{x\in X} Mx,\ \bigcap_{y\in Y} My \subseteq D_M) \\
&\longleftrightarrow\; \bigcup_{y\in Y} (D_M \setminus My) \not\subset \bigcup_{x\in X} (D_M \setminus Mx) \\
&\qquad (\text{by De Morgan's Law of intersection}) \\
&\longleftrightarrow\; \bigcup_{y\in Y} M^\star y \neq \bigcup_{x\in X} M^\star x \\
&\longleftrightarrow\; M^\star \bigcup Y \not\subset M^\star \bigcup X \\
&\longleftrightarrow\; M^\star \models \bigcup Y \not\subset \bigcup X \\
&\longleftrightarrow\; M^\star \models \left(\bigcap X \not\subset \bigcap Y\right)^\star. \quad \square
\end{aligned}
$$

In view of Lemma 13, Algorithm 4 is a satisfiability test for $\mathsf{BST}(\cap, \not\subset, \neq)$-conjunctions.

Algorithm 4 Satisfiability test for $\mathsf{BST}(\cap, \not\subset, \neq)$.

Require: a $\mathsf{BST}(\cap, \not\subset, \neq)$-formula ψ;
Ensure: is ψ satisfiable?
1: let $A \mapsto \widehat{A}$ be the map computed by ClosureMap($\psi^\star$, $\bigcup \Phi^{\not\subset}_{\psi^\star} \cup \bigcup \Phi^{\neq}_{\psi^\star}$);
2: **for** each $\{L, R\} \in \Phi^{\neq}_{\psi^\star}$ **do**
3: **if** $\widehat{L} = \widehat{R}$ **then**
4: **return** false;
5: **return** true;

Thus, we have:

Theorem 7. *The satisfiability problem for* $\mathsf{BST}(\cap, \not\subset, \neq)$ *can be solved in quintic time.*

Remark 5. Again by duality, it can be shown that the satisfiability problem for $\mathsf{BST}(\cup, \not\subset, \neq)$ can be reduced in linear time to the satisfiability problem for $\mathsf{BST}(\cap, \not\subset, \neq)$. ⊣

4.4. *Extending* $\mathsf{BST}(\cup, \not\subset, \neq)$ *and* $\mathsf{BST}(\cap, \not\subset, \neq)$

As stated at the beginning of Section 4, $\mathsf{BST}(\cup, \not\subset, \neq)$ and $\mathsf{BST}(\cap, \not\subset, \neq)$ generalize, respectively, $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$. This can be shown by using the tools of *existential expressibility* and of $O(f)$*-expressibility*.

Lemma 14.

(i) Literals of the forms

$$x \subseteq y \quad \text{and} \quad x = y$$

are expressible both in $\mathsf{BST}(\cup, \not\subset)$ *and in* $\mathsf{BST}(\cap, \not\subset)$*;*
(ii) literals of the form $x \subset y$ *are expressible both in* $\mathsf{BST}(\cup, \not\subset, \neq)$ *and in* $\mathsf{BST}(\cap, \not\subset, \neq)$.

Proof. As a consequence of the bi-implication

$$\models x \subseteq y \;\longleftrightarrow\; y \not\subset y \cup x, \tag{19}$$

literals of the form $x \subseteq y$ are expressible in $\mathsf{BST}(\cup, \not\subset)$, since $y \not\subset y \cup x$ is a $\mathsf{BST}(\cup, \not\subset)$-literal.

To prove (19), let M be any set assignment that satisfies $x \subseteq y$. Then $M \models y = x \cup y$, so that $M \models y \not\subset y \cup x$ holds. On the other hand, if a set assignment M models $y \not\subset y \cup x$, then $M \models y = y \cup x$, namely $M \models x \subseteq y$, as $\models y \subseteq y \cup x$.

Similarly, one can show that literals of the form $x \subseteq y$ are expressible in $\mathsf{BST}(\cap, \not\subset)$, by relying on the bi-implication:

$$\models x \subseteq y \;\longleftrightarrow\; y \cap x \not\subset x.$$

In the light of the preceding results, the conjunction

$$x \subseteq y \ \wedge \ y \subseteq x,$$

and therefore the literal

$$x = y,$$

is plainly expressible both in $\mathsf{BST}(\cup, \not\subset)$ and in $\mathsf{BST}(\cap, \not\subset)$, completing the proof of (i).

Concerning (ii), from the bi-implication

$$\models x \subset y \longleftrightarrow x \subseteq y \wedge x \neq y,$$

and from (i) it readily follows that $x \subset y$ is expressible both in $\mathsf{BST}(\cup, \not\subset, \neq)$ and in $\mathsf{BST}(\cap, \not\subset, \neq)$. □

The above lemma yields that $\mathsf{BST}(\cup, =, \neq)$- and $\mathsf{BST}(\cap, =, \neq)$-conjunctions can be expressed by $\mathsf{BST}(\cup, \not\subset, \neq)$- and $\mathsf{BST}(\cap, \not\subset, \neq)$-conjunctions, respectively. Thus, from Lemma 6 and Theorems 6 and 7, we have:

Lemma 15. *The satisfiability problem for the theories*

- $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\textsc{Disj}, \subseteq, \not\subseteq, =, \neq, \subset, \not\subset)$
- $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \textsc{Disj}, \neg\textsc{Disj}, \subseteq, \not\subseteq, =, \neq, \subset, \not\subset)$

can be solved in quintic time.

Related work and conclusions

In [4,6], we highlighted initial results on 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., [18,9,8,2]). At the outset, we focused on 'Boolean Set Theory', $\mathbb{BST}$, namely the language of quantifier-free formulae that involves set-variables, the Boolean set operators $\cup, \cap, \setminus$, the Boolean relators $\subseteq, \not\subseteq, =, \neq$, and the predicates '$\blacksquare = \varnothing$' and '$\textsc{Disj}(\blacksquare, \blacksquare)$' along with their opposites. That language, whose expressive power is greater than it may appear at first glance (cf. [3]), has an NP-complete satisfiability problem. In [4] we arranged the fragments of $\mathbb{BST}$ in a full complexity taxonomy which spots the 18 minimal NP-complete fragments, and the 5 maximal polynomial fragments. We then announced a study on sub-maximal polynomial fragments of $\mathbb{BST}$, which this paper has undertaken, but which is left largely undone.

Two of the four decidable fragments of set theory studied in this paper do not perfectly fit into the taxonomy introduced in [4], according to which a conjunct of the form $\blacksquare \not\subset \blacksquare$ should not be regarded as a literal but, rather, as the disjunction of two literals. This does not imply that we should redesign our $\mathbb{BST}$ taxonomy; anyhow, it slightly broadens the inventory of addressed satisfiability decision algorithms—a treatable formula is not always a conjunction of literals, but it can be a conjunction of literals and 'small' disjunctions. It should be noted that a move towards a similar broadening already took place in the translation [3] of MLS into $\mathbb{BST}$, where the formula resulting from a translation could be a conjunctive normal form involving only small disjunctions.

As said at the end of [4], we envisage a confluence of the line of research centered on satisfiability testers, to which this paper, along with [4,6,3], contributes, with another active line of research centered on set-unification algorithms (see, e.g., [10]).

Another foreseeable confluence has to do with a long-standing line of research initiated by [12,13], concerning the cooperation among decision algorithms (see also, among many, [16]). In connection with the problem of combining decision algorithms, it is worth noticing that not just $\mathbb{BST}$ but even the much richer decidable theory MLS turns out to be 'convex' in the sense explained in [14].

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Domenico Cantone reports financial support was provided by University of Catania. Eugenio G. Omodeo reports financial support was provided by Francesco Severi National Institute of Higher Mathematics, National Group of Scientific Calculations.

Data availability

No data was used for the research described in the article.

Acknowledgements

They wish to thank Andrea De Domenico for very fruitful discussions and the anonymous referees for carefully reading the manuscript and providing very helpful suggestions.

References

[1] A.V. Aho, J.E. Hopcroft, J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley Publishing Company, 1975.
[2] W. Büttner, H. Simonis, Embedding Boolean expressions into logic programming, J. Symb. Comput. 4 (2) (1987) 191–205.
[3] D. Cantone, A. De Domenico, P. Maugeri, E.G. Omodeo, A quadratic reduction of constraints over nested sets to purely Boolean formulae in CNF, in: F. Calimeri, S. Perri, E. Zumpano (Eds.), Proceedings of the 35th Italian Conference on Computational Logic - CILC 2020, Rende, Italy, October 13–15, 2020, in: CEUR Workshop Proceedings, vol. 2710, CEUR-WS.org, 2020, pp. 214–230.
[4] D. Cantone, A. De Domenico, P. Maugeri, E.G. Omodeo, Complexity assessments for decidable fragments of set theory. I: a taxonomy for the Boolean case, Fundam. Inform. 181 (1) (2021) 37–69.
[5] D. Cantone, A. Ferro, E.G. Omodeo, Computable Set Theory, International Series of Monographs on Computer Science, vol. 1, Clarendon Press, Oxford, UK, 1989.
[6] D. Cantone, P. Maugeri, E.G. Omodeo, Complexity assessments for decidable fragments of set theory. II: a taxonomy for 'small' languages involving membership, Theor. Comput. Sci. 848 (2020) 28–46.
[7] D. Cantone, E.G. Omodeo, A. Policriti, The automation of syllogistic. II: optimization and complexity issues, J. Autom. Reason. 6 (2) (June 1990) 173–188.
[8] M. Cristiá, G. Rossi, Solving quantifier-free first-order constraints over finite sets and binary relations, J. Autom. Reason. 64 (2) (2020) 295–330.
[9] A. Dovier, C. Piazza, G. Rossi, A uniform approach to constraint-solving for lists, multisets, compact lists, and sets, ACM Trans. Comput. Log. 9 (3) (2008) 15.
[10] A. Dovier, E. Pontelli, G. Rossi, Set unification, Theory Pract. Log. Program. 6 (6) (2006) 645–701.
[11] A. Ferro, E.G. Omodeo, J.T. Schwartz, Decision procedures for elementary sublanguages of set theory. I: multi-level syllogistic and some extensions, Commun. Pure Appl. Math. 33 (5) (1980) 599–608.
[12] G. Nelson, D.C. Oppen, Simplification by cooperating decision procedures, ACM Trans. Program. Lang. Syst. 1 (2) (1979) 245–257.
[13] G. Nelson, D.C. Oppen, Fast decision procedures based on congruence closure, J. ACM 27 (2) (1980) 356–364.
[14] D.C. Oppen, Complexity, convexity and combinations of theories, Theor. Comput. Sci. 12 (1980) 291–302.
[15] F. Parlamento, A. Policriti, Decision procedures for elementary sublanguages of set theory. IX: unsolvability of the decision problem for a restricted subclass of the Δ_0-formulas in set theory, Commun. Pure Appl. Math. 41 (2) (1988) 221–251.
[16] R.E. Shostak, Deciding combinations of theories, J. ACM 31 (1) (1984) 1–12.
[17] J.T. Schwartz, D. Cantone, E.G. Omodeo, Computational Logic and Set Theory – Applying Formalized Logic to Analysis, Springer, 2011, Foreword by Martin Davis.
[18] F. Stolzenburg, Membership-constraints and complexity in logic programming with sets, in: F. Baader, K.U. Schulz (Eds.), Frontiers of Combining Systems, First International Workshop, Proceedings, FroCoS 1996, Munich, Germany, March 26-29, 1996, in: Applied Logic Series, vol. 3, Kluwer Academic Publishers, 1996, pp. 285–302.