Università degli studi di Catania
XXXIV Ph.D in computer science

Doctoral Thesis

A complete complexity taxonomy of 'small' fragments of theory Multi-Level Syllogistic

---

Author:
Pietro Maugeri

Supervisor:
Prof. Domenico Cantone

Examiners:
Prof. Alberto Policriti – Università degli studi di Udine
Prof. Alexandru I. Tomescu – University of Helsinki

Academic Year 2020/2021

Contents

List of Tables

Chapter 1

Introduction

The research field of *Computable Set Theory* was born in the 1970's when Jacob T. Schwartz launched a long-term project to combine computer science and set theory. Since the end of the nineteenth century, set theory profoundly changed the language of mathematics due to the naturalness, conciseness and short of builtin semantics of any of its axiomatic version, leading to the mathematics we know today. The amenability of set theory does not end with those listed above, despite its compact axiomatic it permits a fruitful basis from speculations of deep issues, giving easy access to notions such as counting, finiteness, infinitness, etc.

In the 70's computer science was in its infancy so it was natural to combine it with such a useful theory. At an early stage this project was centered on the issue of software prototyping, and the first achievement was a programming language, called SETL [SDDS86], which proved ideal for software prototyping, but also for developing sophisticated algorithms.

The goals inside Computable Set theory shifted, after this initial stage, towards more theoretic goals rooted in the fields of logic and computability. Set-theoretic languages consist in unquantified formulae comprising variables, that will be interpreted as sets, and operations and predicates between sets. Every set-theoretic language is identified by the predicates and operators it comprises, and can be seen as sub-languages, or fragments, of a more generalized set theory comprising all possible operators and predicates. A foundational quest spawned naturally inside the field of Computable Set theory, namely the quest of studying the decidability of all the fragments of Set theory in order to find the boundaries between the decidable and undecidable.

This study generated a large plethora of works, [CFO89, COP01, SCO11, CU18] to cite a few, all of which revolved around one particular sub-language of set theory called *Multi-Level Syllogistic*, MLS for short, which comprises the operations and predicates that are considered a common core of Set theory, operations like *set union* and predicates as *x belongs to y*. MLS will also have a central role inside this dissertation. Such quest revealed to be of much interest not only for the decidability results themselves but also for new useful tools, such as *formative process*, that where developed to solve the decidability problem for some fragments of Set theory.

The study of decidability inside set theory allowed the creation of an automated proof verifier, written in the SETL programming language, called ÆtnaNova [SCO11], that certifies the correctness of theorems by finding appropriate satisfiable set-theoretic formulae, thus heavily relying on the decidability of the language comprising those formulae. The decidability problem of MLS, the simpler fragment studied at this point, was proven to be decidable in [FOS80] and NP-complete in [COP90], meaning that establishing the satisfiability of a MLS-formulae it is not an easy task and can require a huge amount of time, slowing down the performance of any tool that needs to solve such problem, like ÆtnaNova.

Recently a new quest similar to that of finding the boundaries between decidable and undecidable fragments arose in the field of Computable Set theory. The new goal is to study all the sub-fragments of MLS in order to find small sub-languages endowed with a deterministic polynomial time decision procedure. In order to make the study of the over two thousand fragments of MLS feasible, two new tools, of *existential expressibility* and $\mathcal{O}(f)$-*expressibility*, were developed. Also, we splitted this study between the two main sub-languages of MLS, namely BST and MST. These results can be found in recent works, [CDMO21, CMO20, CDDMO20, CMO21, CDDM21], and are the main focus of this dissertation.

This dissertation is organized as follows. In this introduction we will present the language MLS and its syntax and semantics. Then we show that the decidability problem for MLS belongs to the class of *Nondeterministic Polynomial* problems. In Chapter 2 we present the two tools of *existential expressibility* and $\mathcal{O}(f)$-*expressibility* alongside some main expressibility results. Later, in Chapters 3, 4, and 5 we present the complete complexity taxonomy of MLS, showing the boundaries between NP-complete and polynomial fragments. Specifically, first we present the taxonomy for BST and MST then we combine such results into the taxonomy of MLS. In Chapter 6 we show that MLS is a convex theory. Hence, we can to combine its polynomial fragments outside set theory without disrupting polynomiality. Finally, in the conclusions, we sum up all the result presented and highlight some possible future research paths.

1.1 Multi-Level Syllogistic

Multi-Level Syllogistic, MLS for short, is the focus of most of the results in the field of Computable Set theory. In this section we will present the syntax and semantics of it, alongside with the definition of the decision problem for MLS and its fragments.

1.1.1 Syntax of MLS

Here we present the symbols and the structure comprising the syntax of MLS.

The symbols of MLS are:

- infinitely many set variables $x, y, z, \ldots$;
- the constant symbol '$\varnothing$';
- the set operator symbols $\cdot' \cup '\cdot$, $\cdot' \cap '\cdot$, and $\cdot' \setminus '\cdot$;
- the set predicate symbols $\cdot' \subseteq '\cdot$, $\cdot' = '\cdot$, and $\cdot' \in '\cdot$;
- the logical connective symbols '$\neg$'$\cdot$, $\cdot' \wedge '\cdot$, $\cdot' \vee '\cdot$, $\cdot' \longrightarrow '\cdot$, and $\cdot' \longleftrightarrow '\cdot$.

The collection of MLS-terms is the smallest set of expressions such that:

- all the set variables and the constant $\varnothing$ are MLS-terms;
- if s and t are MLS-terms, so are $s \cup t$, $s \cap t$, and $s \setminus t$.

MLS-atoms have the forms

$$s \subseteq t, \quad s = t, \quad s \in t,$$

where s and t are MLS-terms. Finally MLS-formulae are propositional combination of MLS-atoms by means of the logical connective present in the theory. MLS-literals are MLS-atoms and their negations.

Example 1.1. *Some examples of* MLS*-formulae are:*

$$x \in (y \cup z) \setminus w \ \vee x = y \cup w \longrightarrow z = y;$$
$$x \cup y = z \cap y \wedge y \in w \setminus x \vee w \setminus x \in z \cap y;$$
$$x \cap y \cup w \subseteq z \longleftrightarrow x = y \wedge z \setminus w \in y.$$

Given any MLS-formula φ, we denote by $\mathit{Vars}(\varphi)$ the collection of all the set variables occurring inside φ. For instance,

$$\mathit{Vars}(x \in (y \cup z) \setminus w \ \vee x = y \cup w \longrightarrow z = y) = \{x, y, z, w\}.$$

Definition 1.1 (Fragments of MLS)**.** *A fragment of* MLS *is any language whose collection of symbols is a subset of the collection of symbols of* MLS*, and whose formulae follow the same structure of* MLS*-formulae.*

Notice that any formula of a fragment of MLS is also a formula of MLS.

To denote fragments, we use the name of their super-theory, MLS for instance, alongside the collection of symbols used in the fragment, for example $\mathsf{MLS}(\cup, =, \in)$ is the fragment of MLS comprising only the set operator symbol '$\cup$' and the set predicate $\cdot = \cdot$ and $\cdot \in \cdot$.

1.1.2 Semantics of MLS

The semantics of MLS are defined from *set assignments* in the most natural way.

Definition 1.2. *A set assignment is any map M from a collection V of set variables (called the variables domain of M) into the universe $\mathcal{V}$ of all sets.*

The set domain of M is the set $\bigcup M[V] = \bigcup_{v\in V} Mv$, and the rank of M is the rank of its set domain, namely,

$$\mathsf{rk}\,(M) \coloneqq \mathsf{rk}\,(\bigcup M[V])\,.$$

In the above definition we used the universe of all sets $\mathcal{V}$ and the notion of rank, we now take a slight detour to give the definition of those notions.

The von Neumann hierarchy of all sets

The von Neumann standard cumulative hierarchy is defined starting from the empty set $\emptyset$ following transfinite recursion over the class On of all the ordinal numbers in the following way:

$$\begin{aligned}\mathcal{V}_0 &\coloneqq \emptyset\\ \mathcal{V}_{\alpha+1} &\coloneqq \mathscr{P}(\mathcal{V}_\alpha)\\ \mathcal{V}_\lambda &\coloneqq \bigcup_{\mu<\lambda} \mathcal{V}_\mu\\ \mathcal{V} &\coloneqq \bigcup_{\mu\in On} \mathcal{V}_\mu;\end{aligned}$$

where α and μ are ordinals, $\alpha+1$ is the successor of α and λ is a limit ordinal. We recall that $\mathscr{P}(S) \coloneqq \{T \mid T \subseteq S\}$.

The notion of rank is strictly bounded to the von Neumann hierarchy:

Definition 1.3. *The rank of a set $S \in \mathcal{V}$, denoted by $\mathsf{rk}\,(S)$, is the least ordinal α such that $S \in \mathcal{V}_{\alpha+1}$.*

We now resume to the semantics of MLS

Definition 1.4 (Interpretation of set operators)**.** *Let M be a set assignment and V its variable domain. Let s, t be* MLS*-terms whose variables occurs in V. We put recursively,*

$$\begin{aligned}&M\varnothing \coloneqq \emptyset;\\ &M(s \cup t) \coloneqq Ms \cup Mt;\\ &M(s \cap t) \coloneqq Ms \cap Mt;\\ &M(s \setminus t) \coloneqq Ms \setminus Mt.\end{aligned}$$

Notice that for each MLS-term s, Ms is a set in the von Neumann universe $\mathcal{V}$.

Definition 1.5 (Interpretation of set predicate)**.** *For all* MLS*-terms s and t and all set assignments M, such that $Vars(s), Vars(t) \subseteq V$, where V is the variable*

domain of M, we put:

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

$$M(s \in t) \coloneqq \begin{cases} \mathsf{true} & \text{if } Ms \in Mt \\ \mathsf{false} & \text{otherwise.} \end{cases}$$

Notice that applying a set assignment to any MLS-atom we obtain a truth value, either true or false.

Definition 1.6 (Interpretation of propositional connective)**.** *For all set assignment M whose variable domain is V, and all MLS-formulae Φ and Ψ such that $Vars(\Phi)$, $Vars(\Psi) \subseteq V$, we put recursively*

$$M(\Phi \wedge \Psi) \coloneqq M\Phi \wedge M\Psi, \qquad M(\Phi \vee \Psi) \coloneqq M\Phi \vee M\Psi,$$
$$M(\Phi \longrightarrow \Psi) \coloneqq M\Phi \longrightarrow M\Psi, \qquad M(\Phi \longleftrightarrow \Psi) \coloneqq M\Phi \longleftrightarrow M\Psi,$$
$$M(\neg\Phi) \coloneqq \neg(M\Phi).$$

Notice that likewise MLS-atoms, by applying a set assignment to a MLS-formula we obtain a truth value.

Definition 1.7. *A set assignment M is said to* SATISFY *an MLS-formula Φ if $M\Phi = \mathsf{true}$ holds, in which case we also say that M is a* MODEL *for Φ and write $M \models \Phi$. If ϕ has a model, we say that Φ is* SATISFIABLE*; otherwise we say it is* UNSATISFIABLE*. If $M' \models \Phi$ for every set assignment M' defined over $Vars(\Phi)$, we say that Φ is* TRUE *and write $\models \Phi$. Finally, two MLS-formulae Φ and Ψ are said* EQUISATISFIABLE *if ϕ is satisfiable if and only if so is Ψ.*

Example 1.2. *The MLS-formulae of Example 1.1 are all satisfiable. For instance, the formula*

$$x \cup y = z \cap y \wedge y \in w \setminus x \vee w \setminus x \in z \cap y;$$

is satisfied by the set assignment

$$Mx = My = \{\emptyset\}, \quad Mz = \{\emptyset\}, \quad My = \{\emptyset, \{\emptyset\}\}.$$

1.2 Decidability of MLS

Having defined the syntax and semantics of MLS, we can prove that the decision problem for MLS, and all its fragment, is solvable, (i.e., MLS is decidable).

First we need to define the decision problem for MLS.

Definition 1.8. *Given any sub-language $\mathfrak{L}$ of MLS, a satisfiability test for $\mathfrak{L}$ is any algorithm that given as input any $\mathfrak{L}$-formula φ returns true if φ is satisfiable, otherwise returns false.*

Definition 1.9. *Given any sub-language* $\mathfrak{L}$ *of* MLS*, the decision problem (or satisfiability problem) for* $\mathfrak{L}$ *is the problem of establishing whether or not* $\mathfrak{L}$ *admits a satisfiability test.*

Definition 1.10. *If the decision problem for a fragment* $\mathfrak{L}$ *of* MLS *belongs to the class of* NP*-complete problems, we say that* $\mathfrak{L}$ *is* NP*-complete. If a fragment* $\mathfrak{L}$ *of* MLS *admits a satisfiability test whose complexity is deterministic polynomial with respect to the length of the input formula we say that* $\mathfrak{L}$ *is polynomial.*

Having defined the decision problem for MLS, we will now show that MLS is decidable, more so NP-complete. Recall that to prove that a problem is NP-complete, we need to prove that it belongs to both the classes of NP and NP-hard problems.

The following is a simplified version of the proof that the decision problem for theory MLSS belongs to the class NP adapted to MLS. The original proof can be found in [Can13].

The first step we take is that to simplify our theory MLS. As we will see in the next chapter with Corollary 2.2, any MLS formula can be reduced to an equisatisfiable conjunction of literals of the following two types:

$$x = y \setminus z, \quad x \in y, \tag{1.1}$$

where x, y, and z are set variables.

For the rest of this section, by MLS-formula we will always mean conjunctions of literals of types (1.1).

Now we derive some necessary (computable) conditions for the satisfiability of any MLS-formula φ, and later we show that those conditions are also sufficient, thereby proving the decidability of MLS.

Assume φ is a satisfiable MLS-formula and let M be one of its models. Let $\sim_M$ be the equivalence relation over the set variables of φ defined as

$$x \sim_M y \longleftrightarrow Mx = My.$$

Naturally, such equivalence relation partitions the collection $Vars(\varphi)$ into equivalence classes. For each of such equivalence classes, let us pick a $\sim_M$-representative, and let φ' be the formula obtained by replacing each set variable of φ with its $\sim_M$-representative. Plainly $M \models \varphi'$ and also $Mx \neq My$ for any two distinct variables x and y of φ'. Thus M is injective over $V = Vars(\varphi')$.

Let $U = \bigcup_{v \in V} Mv$, and Σ be the Venn partition of U:

$$\Sigma \coloneqq \{\bigcap \mathcal{E} \setminus \bigcup (U \setminus \mathcal{E}) \mid \emptyset \neq \mathcal{E} \subseteq U\} \setminus \{\emptyset\}.$$

It can easily be checked that this is the partition induced by the equivalence relation on U:

$$s \sim_\Sigma t \longleftrightarrow s \in Mx \iff t \in Mx, \quad \text{for all set variables } x \in V.$$

Observe that for any $x \in V$, each region $\sigma \in \Sigma$ is either fully contained in Mx or disjoint from it. Therefore we can associate to each region σ a Boolean-valued map π_σ over V by putting

$$\pi_\sigma(x) \coloneqq \begin{cases} \mathsf{true} & \text{if } \sigma \subseteq Mx \\ \mathsf{false} & \text{if } \sigma \cap Mx = \emptyset. \end{cases} \tag{1.2}$$

Let Π be the collection of all such maps. The following property holds for all the maps $\pi \in \Pi$:

$$\pi(x) \iff \pi(y) \wedge \neg\pi(z) \text{ whenever } x = y \setminus z \text{ appears in } \varphi'. \tag{1.3}$$

Moreover we say that any Boolean-valued map that satisfies the above condition is a *place* for φ'.

Towards finding enough properties to guarantee the existence of a model for φ', we say that a collection of places Π' for φ' is *ample* if for any two variables $x, y \in \mathit{Vars}(\varphi')$ there exists a place $\pi' \in \Pi'$ such that $\pi'(x) \neq \pi'(y)$. Notice that the collection of places Π defined above is ample. Indeed let x, y be two distinct set variables in φ'. Then, as observed above, $Mx \neq My$, so that there exists a region $\sigma \in \Sigma$ such that $\sigma \subseteq Mx$ if and only if $\sigma \cap My = \emptyset$, thus $\pi_\sigma(x) \neq \pi_\sigma(y)$.

It could be shown that the existence of an ample collection of places is sufficient to prove the satisfiability for all the MLS-formulae involving no literals of type $x \in y$. To take care of membership literals, we define the collection L of all the set variables x that appear in a literal of type $x \in y$ on the left-hand side of $\in$. We call such variables *left-hand* variables.

For each left-hand variable x, there exists at least one variable y such that $Mx \in My$. Thus, there exists a unique region σ^x of Σ such that $Mx \in \sigma^x$. It is immediate to check that the place π_{σ^x} (or π^x for short) satisfies the following condition,

$$\pi^x(y) = \mathsf{true} \text{ if the literal } x \in y \text{ belongs to } \varphi'. \tag{1.4}$$

We call any place π fulfilling the above property a *place at* x. Trivially some of the places of Π are places at x for some left-hand variable x, while others are not.

Finally, since well-founded set theory forbids membership cycles of the form,

$$s \in \ldots \in s,$$

it is possible to sort the set variables inside $\mathit{Vars}(\varphi')$ to obtain a total order $\prec$ such that $x \prec y$ whenever $Mx \in My$. Then, for all left-hand x variables we

have

$$\text{if } \pi^x(y) = \mathsf{true} \text{ then } x \prec y.$$

We call any order of variables satisfying the the latter property an *acceptable ordering*.

The necessary properties just enumerated are also sufficient to guarantee the existence of a model for φ, as proven in the next theorem.

Theorem 1.1. *Let φ be an* MLS*-formula. Then φ is satisfiable if and only if there exists an equivalence relation $\sim$ over the set variables of φ, such that by letting φ' be the* MLS*-formula obtained by replacing each set variable in φ with its $\sim$-representative, the following conditions hold:*

(i) φ' admits an ample set π of places.

(ii) For each left-hand variable x of φ' there exists a place π^x at x in Π. Moreover, there exists an order of $Vars(\varphi')$ such that $\pi^x(y)$ is false unless x precedes y in that order, where x is any left-hand variable.

Proof. (Necessity). We have already proven that both conditions are necessary.

(Sufficiency). Suppose both condition holds. For each place $\pi \in \Pi$, we choose a distinct singleton $\overline{\pi}$ such that the cardinality of its only member is larger than $m + n$, where m is the number of left-hand variables and n is the cardinality of Π. We choose such singletons so that they are all pairwise disjoint. Define the set assignment M for all $v \in Vars(\varphi')$ as follows,

$$Mv \coloneqq \{Mx \mid x \in L \wedge \pi^x(v)\} \cup \bigcup\{\overline{\pi} \mid \pi \in \Pi \wedge \pi(v)\}. \tag{1.5}$$

First notice that each set Mv is well-founded thanks to condition (ii). In fact to build each of such set, it is enough to follow the order defined in (ii).

By definition of M, we have that the cardinality of each Mv can not exceed $m + n$, while the cardinality of each of the sets $\overline{\pi}$ is greater then $m + n$. Thus, we have

$$\{Mz \mid z \in L\} \cap \bigcup\{\overline{\pi} \mid \pi \in \Pi\} = \emptyset,$$

and therefore

$$\{Mz \mid z \in L \wedge \pi^z(x)\} \cap \bigcup\{\overline{\pi} \mid \pi \in \Pi \wedge \pi(y)\} = \emptyset, \tag{1.6}$$

for every pair x, y in $Vars(\varphi')$.

Concerning literals of type $x \in y$, we have that x is a left-hand variable. Thus, by (ii), there exists a place π^x at x so that $\pi^x(y) = \mathsf{true}$, since $x \in y$ appears inside φ'. Therefore, by (1.5), $Mx \in My$. Thus by the generality of $x \in y$, M models correctly all the literals of type $x \in y$ of φ'.

Concerning literals of type $x = y \setminus z$, we first show that M is injective. Let x and y be any two distinct set variables.

Since Π is ample, by condition (i) there must exists a place π' such that one between $\pi'(x)$ and $\pi'(y)$ is true while the other is false. Suppose for definiteness

that $\pi'(x)$ is true. Then $\pi'(y)$ is false. Since the $\overline{\pi}$'s are pairwise disjoint, we must have:

$$\overline{\pi'} \subseteq \bigcup\{\overline{\pi} \mid \pi \in \Pi \wedge \pi(x)\} \setminus \bigcup\{\overline{\pi} \mid \pi \in \Pi \wedge \pi(y)\},$$

and since the only element of $\overline{\pi'}$ has cardinality greater than $m+n$, we have

$$\overline{\pi'} \subseteq Mx \setminus My.$$

Thus M is injective.

Finally let $x = y \setminus z$ be a literal of φ'. By the injectivity of M, the disjointedness of the sets $\overline{\pi}$, and (1.6), we have

$$\begin{aligned}
Mx &= \{Mv \mid v \in L \wedge \pi^v(x)\} \cup \bigcup\{\overline{\pi} \mid \pi \in \Pi \wedge \pi(x)\} \\
&= \{Mv \mid v \in L \wedge \pi^v(y) \wedge \neq \pi^v(z)\} \cup \bigcup\{\overline{\varphi} \mid \pi \in \Pi \wedge \pi(y) \wedge \neq \pi(z)\} \\
&= (\{Mv|\ v \in L \wedge \pi^v(y)\} \setminus \{Mv|\ v \in l \wedge \pi^v(z)\}) \cup \bigcup(\{\overline{\pi}|\ \pi \in \Pi \wedge \pi(y)\} \setminus \{\overline{\pi}|\ \pi \in \Pi \wedge \pi(z)\}) \\
&= (\{Mv|\ v \in L \wedge \pi^v(y)\} \cup \bigcup\{\overline{\pi}|\ \pi \in \Pi \wedge \pi(y)\}) \setminus (\{Mv|\ v \in L \wedge \pi^v(z)\} \cup \bigcup\{\overline{\pi}|\ \pi \in \Pi \wedge \pi(z)\}) \\
&= My \setminus Mz.
\end{aligned}$$

By the generality of the literal $x = y \setminus z$, M models correctly all such literals of φ'.

Summing up $M \models \varphi'$. Then by extending M to all the set variables $x \in \mathit{Vars}(\varphi) \setminus \mathit{Vars}(\varphi')$ in such a way that Mx is equal to the set assigned to its $\sim$-representative, we plainly have that $M \models \varphi$. □

Theorem 1.1 readily implies the decidability of MLS. However, as for now, we can only infer that solving the satisfiability problem for MLS takes *nondeterministic exponential* time rather then *nondeterministic polynomial* time. This is due to the size of the collection Π having no polynomial bound, in fact Π can have size up to 2^p, where p is the number of variables occurring in φ'. Luckily, we can prove that we can constrain the size of Π to be linear in p.

Let $\varphi', V, L, M, \Sigma$, and Π as before, so that, in particular, M is injective and Π is the collection of all the places π_σ for $\sigma \in \Sigma$.

Given $\Sigma' \subseteq \Sigma$ and $V' \subseteq V$, we say that Σ' distinguishes V' if for each pair of distinct variables x, y of V' there exists a region $\sigma \in \Sigma'$ that is contained in one of the sets Mx, My and is disjoint from the other.

Lemma 1.2. *Let $\Sigma' \subseteq \Sigma$ and $V' \subseteq V$. If Σ' distinguishes V', then for each $x \in V$ there exists a $\sigma \in \Sigma$ such that $\Sigma' \cup \{\sigma\}$ distinguishes $V' \cup \{x\}$.*

Proof. Assume that V' does not distinguish $V' \cup \{x\}$. Then there exists a variable $y \in V'$ such that for all $\sigma' \in \Sigma'$ it is either $\sigma' \subseteq Mx \cap My$ or $\sigma' \cap (Mx \cup My) = \emptyset$. There must exist exactly one such y inside V', otherwise V' would not distinguish V'. Notice that since M is injective, $Mx \neq My$. Hence there must be a region $\sigma \in \Sigma$ that is fully contained in exactly one between Mx and My and is disjoint from the other. Therefore, it is trivial to show that $\Sigma \cup \{\sigma\}$ distinguishes $V' \cup \{x\}$. □

Notice that the singleton $\{x\}$, where x is any variable of V, is vacuously distinguished by the empty partition, then by applying iteratively the latter lemma we obtain a partition Σ' of size $p-1$ that distinguishes V. It can be easily checked that any subset of Π that contains all the places π_σ such that $\sigma \in \Sigma'$ is ample, in particular

$$\Pi' \coloneqq \{\pi_\sigma \mid \sigma \in \Sigma'\} \cup \{\pi^x \mid x \in L\},$$

whose cardinality is $p + m - 1$ (where $m = |L|$) is ample.

We can observe that Π can be used for Theorem 1.1. Thus condition condition (i) can be checked in *nondeterministic linear* time. Since this is also the case for checking condition (ii) end to find the equivalence $\sim$, we can conclude that:

Lemma 1.3. *The decidability problem for* MLS *belongs to the class of* NP *problems.*

In Chapters 3, 4, and 5, we provide several NP-hard fragments of MLS. NP-hardness of a fragment is inherited by all of its extensions, likewise to prove that the decision problem for a theory is in NP, it is enough to say the same for all of its fragments. Therefore we can conclude

Lemma 1.4. *The satisfiability problem for* MLS *is* NP*-complete.*

1.2.1 Extensions of MLS

The decision problem of several extensions of MLS has been solved throughout the years, whereas the same problem remains open for another extensions. Here we list the decidable extensions of MLS, identifying each language by the symbols it comprises.

MLSS :	$\cup, \cap, \setminus, \subseteq, =, \in, \{\cdot\}$	([FOS80])
MLS$\bigcup$:	$\cup, \cap, \setminus, \subseteq, =, \in, \bigcup(\cdot)$	([CFS87])
MLSS$\bigcap$:	$\cup, \cap, \setminus, \subseteq, =, \in, \{\cdot\}, \bigcap(\cdot)$	([CC89])
MLSP :	$\cup, \cap, \setminus, \subseteq, =, \in, \mathscr{P}(\cdot)$	([CSF85])
MLSSP :	$\cup, \cap, \setminus, \subseteq, =, \in, \{\cdot\}, \mathscr{P}(\cdot)$	([Can91, CUO02])
MLSSPF :	$\cup, \cap, \setminus, \subseteq, =, \in, \{\cdot\}, \mathscr{P}(\cdot), Finite(\cdot)$	([CU14])

Not all the extensions of MLS are decidable however, for instance the theory MLSuC$^+$, comprising the symbols, $\cup, \cap, \setminus, \in, \subseteq, \in, \otimes, \uplus$, that is obtained from MLS by adding the two atoms:

$$x \otimes y, \quad x = \uplus(y),$$

where x and y are set variables, has an undecidable satisfiability problem. We recall that $s \otimes t \coloneqq \{\{a, b\} \mid a \in s \wedge b \in t\}$ is the collection of the unordered pairs of the elements of s and t, while $\uplus(T) \coloneqq \{s \mid (\exists! t)(t \in T \wedge s \in t)\}$ is the disjoint union of the elements of T.

In this dissertation we will focus on MLS and its fragments, so remaining amply inside the realm of the decidable. In fact, since all the fragments we will study are sub-languages of MLS the same proof of decidability we presented early in this chapter will also apply to them. Therefore, all the fragments we will study not only are decidable, but their satisfiability problem also belongs to the class of nondeterministic polynomial problems.

Chapter 2

Existential expressibility and $\mathcal{O}(f)$-expressibility

In this chapter we present two reduction techniques that have been used as powerful tools in the construction of the complete taxonomy of MLS fragments, the main topic of this dissertations.

Most of the reduction we present are based on the notion of **existential expressibility**, which can be seen as a *context-free* expressibility in a sense that will be declared soon.

Definition 2.1 (Existential expressibility)**.** *A formula $\psi(\vec{x})$ is* existentially expressible *in a theory $\mathcal{T}$ if there exists a $\mathcal{T}$-formula $\Psi(\vec{x},\vec{z})$ such that*

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

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

We also provide a more general notion of *context-sensitive* expressibility called $\mathcal{O}(f)$**-expressibility**. This second form of expressibility is strictly bound to its complexity, hence the "$\mathcal{O}(f)$" in the name.

Definition 2.2 ($\mathcal{O}(f)$-expressibility)**.** *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be any theories and $f\colon \mathbb{N}\to\mathbb{N}$ be a given map. A collection $\mathcal{C}$ of formulae is said to be* $\mathcal{O}(f)$-expressible from $\mathcal{T}_1$ into $\mathcal{T}_2$ *if there exists a map*

$$\langle\,\varphi(\vec{y})\,,\,\psi(\vec{x})\,\rangle \;\mapsto\; \Xi^{\psi}_{\varphi}(\vec{x},\vec{y},\vec{z}) \tag{2.1}$$

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

(a) the mapping (2.1) can be computed in $\mathcal{O}\big(f(|\varphi\wedge\psi|)\big)$-time,

(b) if $\varphi(\vec{y})\wedge\Xi^{\psi}_{\varphi}(\vec{x},\vec{y},\vec{z})$ is satisfiable, so is $\varphi(\vec{y})\wedge\psi(\vec{x})$,

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

It is quite evident that existential expressibility is a particular form of $\mathcal{O}(1)$-expressibility. In fact, if $\Psi(\vec{x},\vec{z})$ is the formula that existentially expresses $\psi(\vec{x})$, then the map $\psi(\vec{x})\mapsto\Psi(\vec{x},\vec{z})$ satisfies conditions (a)–(c).

Example 2.1. *As we will show below the atom $x = y \cap y$ is expressible in the language $\mathsf{MLS}(\setminus, =, \neq)$ (which is the fragment of MLS that comprises only the operator $\setminus$ and the relations $=$ and $\neq$) due to the well-known equivalence*

$$\models x = y \cap z \longleftrightarrow x = y \setminus (y \setminus z). \tag{2.2}$$

Consider the following formula in the theory $\mathsf{MLS}(\setminus, \cap, =, \neq)$:

$$\varphi \coloneqq x = y \setminus w \ \wedge \ w \neq y \cap z \ \wedge \ x = z \setminus w.$$

Using existential expressibility, we can translate the formula φ into an equisatisfiable formula φ' inside $\mathsf{MLS}(\setminus, =, \neq)$, where

$$\varphi' \coloneqq x = y \setminus w \ \wedge \ w \neq x' \ \wedge \ x' = y \setminus x'' \ \wedge \ x'' = y \setminus z \ \wedge x = z \setminus w.$$

In this example, the collection of variables $\vec{x}$ is $\{x, y, z, w\}$ and the collection $\vec{z}$ is $\{x', x''\}$.

2.1 Expressibility results

Existential expressibility can be easily used to reduce the satisfiability problem of a theory to that of one of its fragments. Let $\mathcal{T}_1$ be a fragment of MLS that admits a sub-language $\mathcal{T}_2$ such that every literal that is in $\mathcal{T}_1$ but not in $\mathcal{T}_2$ is existentially expressible in $\mathcal{T}_2$. Then, following Definition 2.1, given any formula φ_1 of $\mathcal{T}_1$, we can replace any literal of that formula that is not in $\mathcal{T}_2$ with a $\mathcal{T}_2$-formula Ψ to obtain a formula φ_2 of $\mathcal{T}_2$ that is equisatisfiable with φ_1.

Existential expressibility enjoys a transitive property in the following sense. Once we found that a certain formula ψ is existentially expressible inside a theory $\mathcal{T}_1$, namely we found a $\mathcal{T}$-formula Ψ_1 such that

$$\models \psi(\vec{x}) \longleftrightarrow (\exists \vec{z}) \Psi_1(\vec{x}, \vec{z}),$$

then ψ will be also existential expressible in any theory $\mathcal{T}_2$ where Ψ is existentially expressible. Indeed, in this case there exists a $\mathcal{T}_2$-formula Ψ_2 such that

$$\models \Psi_1(\vec{x}) \longleftrightarrow (\exists \vec{z}) \Psi_2(\vec{x}, \vec{z}).$$

Hence we obtain

$$\models \psi(\vec{x}) \longleftrightarrow (\exists \vec{z}) \Psi_1(\vec{x}, \vec{z}) \longleftrightarrow (\exists \vec{z}, \vec{w}) \Psi_2(\vec{x}, \vec{z}, \vec{w}).$$

Thus, by setting $\vec{y} = \vec{z} \cup \vec{w}$ we have

$$\models \psi(\vec{x}) \longleftrightarrow (\exists \vec{y}) \Psi_2(\vec{x}, \vec{y}).$$

$\mathcal{O}(f)$-expressibility can be used to reduce the satisfiability problem of a theory to that of one of its sub-theories in the same fashion of existential expressibility. The main difference in the use between existential and $\mathcal{O}(f)$-expressibility is

that while any existential expressibility result for a theory carries over to every of its super-theory, the same is not true for $\mathcal{O}(f)$-expressibility. However, if a formula is $\mathcal{O}(f)$-expressible in a theory $\mathcal{T}$, then it is also $\mathcal{O}(f)$-expressible inside all the sub-theories of $\mathcal{T}$ obtained by reducing the theory by means of existential expressibility.

In what follows, we will often refer to a theory called *Boolean Set Theory*, BST, and its fragments. The theory BST will be studied in Chapter 3, where a more precise definition of its syntax will be given. For what concerns us in this chapter, it is enough to define BST as the theory obtained by removing the membership relation $\in$ from MLS.

Lemma 2.1. *(a)* $x = y \setminus z$ *is expressible in* $\mathsf{BST}(\cup, \mathsf{Disj}, =)$*;*

(b) $x = y \cup z$ *is expressible in* $\mathsf{BST}(\setminus, =)$*;*

(c) $x = y \cap z$ *is expressible in* $\mathsf{BST}(\setminus, =)$*;*

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

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

(f) $x \nsubseteq y$ *is expressible both in* $\mathsf{BST}(\cup, \neq)$ *and in* $\mathsf{BST}(\cap, \neq)$*;*

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

(h) $x \neq \varnothing$ *is expressible in* $\mathsf{BST}(\subseteq, \neq)$, $\mathsf{BST}(=\varnothing, \neq)$, $\mathsf{BST}(\neg\mathsf{Disj})$, *and in* $\mathsf{MLS}(\in)$*;*

(i) $\mathsf{Disj}\,(x, y)$[1] *is expressible in* $\mathsf{BST}(\cap, =\varnothing)$*;*

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

(k) $x \notin y$ *is expressible in* $\mathsf{MLS}(\setminus, =, \in)$*;*

(l) $x \neq y$ *is expressible in* $\mathsf{MLS}(\cup, \cap, \in, \notin)$.

Proof. Most of the following proofs are trivial once the equivalence $\bigcup\psi \longleftrightarrow \Psi$ is given, so that we will only linger on some of those proofs.

(a) $\models x = y \setminus z \longleftrightarrow (\mathsf{Disj}\,(x, z) \ \wedge x \cup z = y \cup z)$.

First, let $M \models x = y \setminus z$. Then plainly $\mathsf{Disj}\,(Mx, Mz)$ and $Mx \cup Mz = My \cup Mz$ hold. If conversely $M \models \mathsf{Disj}\,(x, z) \wedge x \cup z = y \cup z$, we have that if $s \in My \setminus Mz$ then, since $My \subseteq My \cup Mz = Mx \cup Mz$, $s \in Mx$, so that $My \setminus Mz \subseteq Mx$. Moreover, if $s \in Mx$ then likewise $s \in My \cup Mz$, and since $\mathsf{Disj}\,(Mx, Mz)$, then $s \in My \setminus Mz$, so that $Mx = My \setminus Mz$.

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

First, let $M \models x = y \cup z$. Then $My \setminus Mx = \emptyset = MX \setminus Mx$, and also $Mx \setminus My = Mz \setminus My$. If $M \models x \setminus y = z \setminus y \wedge y \setminus x = x \setminus x$ we have that $My \setminus Mx = \emptyset$, and thus $My \subseteq Mx$. Let $s \in Mx$. Then, either $s \in My$, so that $s \in My \cup Mz$, or $s \notin My$ so that $s \in Mx \setminus My = Mz \setminus My \subseteq$

[1] Given two sets S, T the proposition "S is disjoint from T", $\mathsf{Disj}\,(S, T)$, means that no element of S belongs to T and vice-versa. Equivalently we can write $S \cap T = \emptyset$

$My \cup Mz$. Therefore $Mx \subseteq My \setminus Mz$. If $s \in My \cup Mz$, we reason as follows. If $s \in My$, then we have $s \in Mx$, otherwise if $s \in Mz \setminus My$ then $s \in Mx \setminus My \subseteq Mx$. Hence, $Mx = My \cup Mz$.

(c) $\models x = y \cap z \longleftrightarrow x = y \setminus (y \setminus z)$.

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

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

(f) $\models x \nsubseteq y \longleftrightarrow y \neq x \cup y$ and $\models x \subseteq y \longleftrightarrow x \neq x \cap y$.

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

(h) We have that

- $\models x \neq \varnothing \longleftrightarrow (\exists y, z)(y \subseteq x \ \wedge \ y \neq x)$.

  Let $M \models x \neq \varnothing$. Then there exists an $s \in Mx$. Let $My = \emptyset$. Then plainly $My \subseteq Mx$, and $Mx \neq My$. If $M \models y \subseteq x \wedge y \neq x$, then there exists an $s \in Mx \setminus My$, so that $Mx \neq \emptyset$.

- $\models \neq \varnothing \longleftrightarrow (\exists y) y = \varnothing \ \wedge \ x \neq y$.

- $\models x \neq \varnothing \longleftrightarrow \neg\mathsf{Disj}\,(x, x)$.

- $\models x \neq \varnothing \longleftrightarrow (\exists y) y \in x$.

(i) $\mathsf{Disj}\,(x, y) \longleftrightarrow x \cap y = \varnothing$.

(j) We have that:

- $\neg\mathsf{Disj}\,(x, y) \longleftrightarrow x \cap y \neq \varnothing$;

- $\models \neg\mathsf{Disj}\,(x, y) \longleftrightarrow (\exists z)(z \subseteq x \ \wedge \ z \subseteq y \wedge \ z \neq \varnothing)$, (recall that by (g) $z \neq \varnothing$ is expressible in $\mathsf{BST}(\subseteq, \neq)$).

(k) $\models x \notin y \longleftrightarrow (\exists z)(x \in z \ \wedge z = z \setminus y$.

If $M \models x \notin y$, then putting $Mz = \{Mx\}$ we have both $Mx \in Mz$ and $Mz = Mz \setminus My$. Otherwise, if $M \models z \in x \wedge z = z \setminus y$, then $Mx \in Mz$ and $\mathsf{Disj}\,(Mz, My)$. Thus $Mx \notin My$.

(l) $\models x \neq y \longleftrightarrow (\exists z)(z \in x \cup y \wedge z \notin x \cap y)$.

□

We are ready to use the above results to reduce MLS to a smaller fragment.

Corollary 2.2. *Every* MLS*-formula can be reduced to an equisatisfiable conjunction of atoms of types*

$$x = y \setminus z, \quad x \in y.$$

Proof. Without loss of generality we can consider just MLS-formulae in conjunctive normal form, thus, we may assume that any formula will be a conjunction

of atoms of types:

$$\begin{array}{l} x = y \cup z,\ x = y \cap z,\ x = y \setminus z, \\ x = \varnothing,\ x \subseteq y,\ x = y,\ \mathsf{Disj}\,(x, y)\,, x \in y \\ x \neq \varnothing,\ x \nsubseteq y,\ x \neq y,\ \mathsf{Disj}\,(x, y)\,, x \notin y. \end{array}$$

By applying Lemma 2.1, we obtain a conjunction of atoms of types

$$x = y \setminus x,\ x = y,\ x \in y.$$

We can also get rid of the atoms of type $x = y$ since

$$\models x = y \longleftrightarrow (\exists z)(x = y \setminus z\ \wedge z = z \setminus z).$$

□

Similarly we can also prove that

Corollary 2.3. *Every* BST*-formula can be reduced to an equisatisfiable conjunction of atoms of types*

$$x = y \setminus z, \quad x \neq y.$$

When a literal is not expressible in a given theory, we may check if it is $\mathcal{O}(f)$-expressible, for some function f.

Lemma 2.4. *Any satisfiable* MLS$(\cup, \cap, =, \neq, \in, \notin)$*-formula* φ *admits a model* M *such that*

$$Mx \neq \emptyset,$$

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

Proof. Let M' be a model for φ and let α be a rank such that

$$\mathsf{rk}\,(M'x) < \alpha,$$

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

Let σ be any set of rank α, S be the collection of all the sets of rank strictly smaller than α, and $\mathcal{V}$ be the von Neumann universe of all sets. Define the function $f_\sigma : S \mapsto \mathcal{V}$ as

$$f_\sigma(s) \coloneqq \{f_\sigma(t) \mid t \in s\} \cup \{\sigma\}.$$

First we prove that f_σ is injective. We proceed by proving

$$f_\sigma(s') = f_\sigma(s'') \longrightarrow s' = s''$$

by induction over the rank of s'.

First let $\mathsf{rk}(s') = 0$, then $s' = \emptyset$. By definition $f_\sigma(\emptyset) = \{\sigma\}$. Notice that for every nonempty set $t \in S$, $\mathsf{rk}(f_\sigma(t)) > \alpha + 1$. In fact, let $s \in t$. Then $\sigma \in f_\sigma(s) \in f_\sigma(t)$, so that if $f_\sigma(s'') = f_\sigma(s') = \{\sigma\}$ we must have $s'' = \emptyset = s'$.

Let $\mathsf{rk}(s') > 0$ and let $t' \in s'$. We have that $f_\sigma(t') \in f_\sigma(s') = f_\sigma(s'')$. Hence there exists a $t'' \in s''$ such that $f_\sigma(t') = f_\sigma(t'')$. Plainly, $\mathsf{rk}(t') < \mathsf{rk}(s')$. Thus, by inductive hypothesis $t' = t''$, we have $s' \subseteq s''$. Analogously we can prove $s'' \subseteq s'$, so that $s' = s''$ holds.

Now define the set assignment M as

$$Mx \coloneqq f_\sigma(M'x)$$

for all $x \in \mathit{Vars}(\varphi)$.

By proving that for each term T of φ we have $MT = f_\sigma(M'T)$, the injectiveness of f_σ directly yields that $M \models \varphi$.

We proceed by structural induction, first let $T = x$ for some variable x of φ. Then $MT = f_\sigma(M'T)$, by definition of M.

Now let $T = T_1 \cup T_2$. Then, by inductive hypothesis, $MT = MT' \cup MT'' = f_\sigma(M'T') \cup f_\sigma(M'T'')$, and plainly $M'T = M'T' \cup M'T''$. Let $s \in f_\sigma(M'T)$. If $s = \sigma$, then $s \in f_\sigma(M'T') \subseteq MT$. Otherwise there must exist a $t \in M'T$ such that $s = f_\sigma(t) \in f_\sigma(M'T)$. Then plainly $t \in M'T' \cup M'T''$. W.l.o.g., let $t \in M'T'$. Thus, $s \in f_\sigma(M'T') \subseteq MT$ and therefore $f_\sigma(M'T) \subseteq MT$. Now let $s \in MT = f_\sigma(M'T') \cup f_\sigma(M'T'')$. W.l.o.g. let $s \in f_\sigma(M'T')$. Then either $s = \sigma$ or there exists a $t \in M'T'$ such that $s = f_\sigma(t)$. If $s = \sigma$, then plainly $s \in f_\sigma(M'T)$, otherwise $t \in M'T$ and $s \in f_\sigma(M'T)$. Hence, $MT \subseteq f_\sigma(M'T)$. Summing up, $Mt = f_\sigma(M'T)$.

Finally, let $T = T' \cap T''$. Then, by inductive hypothesis $MT = MT' \cap MT'' = f_\sigma(M'T') \cap f_\sigma(M'T'')$, and plainly $M'T = M'T' \cap M'T''$. Let $s \in MT$. Then $s \in f_\sigma(M'T') \cap s_\sigma(M'T'')$. If $s = \sigma$, then plainly $s \in f_\sigma(M'T)$, otherwise since f_σ is injective there exists a $t \in M'T' \cap M'T''$ such that $s = f_\sigma(t)$. Hence $s \in f_\sigma(M'T)$. Let now $s \in f_\sigma(M'T)$. Then if $s = \sigma$ plainly $s \in f_\sigma(M'T') \cap f_\sigma(M'T'') = MT$. Otherwise there exists a $t \in M'T$ such that $s = f_\sigma(t)$. Hence $t \in M'T' \cap M'T''$, so that $s \in f_\sigma(M'T') \cap f_\sigma(M'T'') = Mt$. Therefore $MT = f_\sigma(M'T)$.

Plainly $\sigma \in Mx$, for all $x \in \mathit{Vars}(\varphi)$, hence our thesis follows. □

Lemma 2.5. *The literal $x = \varnothing$ is not existentially expressible in $\mathsf{MLS}(\cup, \cap, =, \neq, \in, \notin)$.*

Proof. By way contradiction, assume there exists a $\mathsf{MLS}(\cup, \cap, =, \neq, \in, \notin)$-formula Ψ involving x and such that

$$\models x = \varnothing \longleftrightarrow \Psi.$$

Plainly $x = \varnothing$ is satisfiable, thus also Ψ is satisfiable. Then, by Lemma 2.4, there exists a model M for Ψ such that $Mv \neq \emptyset$, for all $v \in \mathit{Vars}(\Psi)$. In

particular, $Mx \neq \emptyset$. Hence,

$$M \not\models x = \varnothing \longleftrightarrow \Psi,$$

a contradiction. □

Corollary 2.6. *The literal $x = y \backslash z$ is not existential expressible in $\mathsf{MLS}(\cup, \cap, =, \neq, \in, \notin)$.*

Proof. This is a direct consequence of Lemmas 2.5 and 2.1(g). □

As a consequence of Lemma 2.5, the literal $x = \varnothing$ is not expressible inside all the fragments of $\mathsf{MLS}(\cup, \cap, =, \neq, \in, \notin)$. However, we can prove that $x = \varnothing$ is $\mathcal{O}(n)$-expressible inside both $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$.

Lemma 2.7. *The literal $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cup, =, \neq)$.*

Proof. Let φ be any $\mathsf{BST}(\cup, =, \neq)$-formula, and consider the map

$$\langle\, x = \varnothing\,,\ \varphi \,\rangle \mapsto \bigwedge_{v \in Vars(\varphi)} x \subseteq v. \tag{2.3}$$

We prove that the map (2.3) satisfies (a), (b), and (c).

Concerning (a), to compute the map we just need to scan the formula φ and add the literal $x \subseteq v$ to the conjunction $\bigwedge x \subseteq v$ for each variable in φ. Letting n be the length of the formula φ, the map (2.3) can be computed in $\mathcal{O}(n)$-time.

Concerning (b), let $M \models \varphi \wedge \bigwedge_{v \in Vars(\varphi)} x \subseteq v$. Let us put

$$M'v \coloneqq Mv \setminus Mx,$$

for all $v \in Vars(\varphi) \cup \{x\}$.

Plainly $M'x = \emptyset$. Let ℓ be any literal of φ. Then ℓ is either of the form $\bigcup L = \bigcup R$ or $\bigcup L \neq \bigcup R$.

If $\ell \coloneqq \bigcup L = \bigcup R$, we have that

$$\begin{aligned} \bigcup M'L &= (\bigcup ML) \setminus Mx \\ &= (\bigcup MR) \setminus Mx \\ &= \bigcup M'R. \end{aligned}$$

If $\ell \coloneqq \bigcup L \neq \bigcup R$, we have that $Mx \subseteq \bigcup ML \cap \bigcap MR$, and $\bigcup ML \neq \bigcup MR$. W.l.o.g., let $s \in \bigcup ML \setminus \bigcup MR$. Then $s \notin Mx$ and therefore $s \in (\bigcup ML \setminus Mx) \setminus (\bigcup MR \setminus Mx)$, that is $s \in \bigcup M'L \setminus \bigcup M'R$.

Therefore we have $M' \models \varphi \wedge x = \varnothing$.

Finally, concerning (c), let $M \models \varphi \wedge x = \varnothing$. Then plainly $Mx = \emptyset \subseteq My$, for each variable $v \in Vars(\varphi)$,. Thus $M \models \varphi \wedge \bigwedge_{v \in Vars(\varphi)} x \subseteq v$. □

Analogously, we can also prove that:

Lemma 2.8. *The literal $x = \varnothing$ is $\mathcal{O}(n)$-expressible in* $\mathsf{BST}(\cap, =, \neq)$.

In the next section we will provide a more complex example of $\mathcal{O}(f)$-expressibility. Specifically, we will show that all the formulae in the language MLSS are $\mathcal{O}(n^2)$-expressible in the an extension of BST called BST+, where the theory MLSS is a super-theory of MLS obtained by also allowing atoms of type $x = \{y\}$.

2.2 An example of $\mathcal{O}(f)$-expressibility: MLSS is $\mathcal{O}(n^2)$-expressible in BST+

By applying Corollary 2.2, we obtain that MLSS is the collection of all the conjunctions of atoms of types

$$x = y \setminus z,\ x \in y,\ x = \{y\}.$$

Notice, however, that the equivalence

$$\models x \in y \longleftrightarrow (\exists z) z = \{x\} \ \wedge \ z \subseteq y$$

holds.

Thus, using existential expressibility, we can further simplify MLSS to the collection of all the conjunctions of atoms of types

$$x = y \setminus z,\ x = \{y\}.$$

We shall also be interested in the extension BST+ of the theory BST.

Definition 2.3. *The theory* BST+ *consists of all the propositional combinations (resulting from unrestrained use of the logical connectives '$\wedge$', '$\vee$', '$\longrightarrow$', '$\longleftrightarrow$', and '$\neg$') of atomic formulae of type $x = y \setminus z$.*

It is not hard to check that the satisfiability problem for BST+ can be reduced to the satisfiability problem for BST in nondeterministic polynomial time, and therefore it is NP-complete.

Here we show that atoms of type $z = \{x\}$ are not existentially expressible in BST+, whereas any conjunction of such atoms turns out to be $\mathcal{O}(n^2)$-expressible from BST into BST+. The following proof is an extension of the proof that literals of type $x \in y$ are $\mathcal{O}(n^2)$-expressible inside BST+ presented in [CDDMO20].

2.2.1 Existential inexpressibility of $z = \{x\}$ in BST+

If singletons atoms of the form $z = \{x\}$ were existentially expressible in BST+, membership atoms $x \in y$ would also be expressible; in fact, as stated before, such atoms are existential expressible in presence of atoms of singleton atoms inside BST.

Lemma 2.9. $\models x \in y \longleftrightarrow (\exists z)(z = \{x\} \wedge z \subseteq y)$, *if x, y, z are distinct variables.*

Proof. If $M \models z = \{x\} \wedge z \subseteq y$ holds, then $M \models x \in y$, since $Mx \in Mz \subseteq My$.

Conversely, if $M \models x \in y$ holds, let us extend M by putting $Mz \coloneqq \{Mx\}$; then $Mx \in My$ yields $M \models z = \{x\} \wedge z \subseteq y$, so that $M \models (\exists z)(z = \{x\} \wedge z \subseteq y)$. □

In this section, we show that membership atoms $x \in y$ are not existentially expressible in BST+; therefore, by the lemma just seen, atoms $z = \{x\}$ are not existentially expressible in BST+ as well. Specifically, we prove that every satisfiable BST+-formula Ψ admits a "flat" model M, namely a model that guarantees

$$s \in Mv \longrightarrow \mathsf{rk}\,(s) = \rho,$$

for some positive rank ρ and for each variables v of Ψ. Consequently, $Mx \notin My$ for any $x, y \in \mathit{Vars}(\Psi)$, and hence $M \not\models x \in y$.

Definition 2.4. *For every ordinal $\rho \geqslant 1$, a set assignment M over a collection V of variables is said to be* ρ-flat *if all sets belonging to $\bigcup\{\, Mv : v \in V \,\}$ have rank ρ.*

No membership atom $x \in y$ is satisfied by any ρ-flat set assignment:

Lemma 2.10. *Let M be a ρ-flat set assignment over a collection V of variables. Then $Mx \notin My$, for any $x, y \in V$.*

Proof. Because of the ρ-flatness of M, for every $x \in V$ either $\mathsf{rk}\,(Mx) = 0$ (when $Mx = \emptyset$) or $\mathsf{rk}\,(Mx) = \rho + 1$ (when $Mx \neq \emptyset$). Hence, in any case $\mathsf{rk}\,(Mx) \neq \rho$ (since ρ-flatness presupposes $\rho \geqslant 1$), and therefore $Mx \notin \bigcup\{My \mid y \in V\}$. □

As proved in the next lemma, any satisfiable BST+-formula always admits a ρ-flat model, for sufficiently large ρ.

Lemma 2.11. *Every satisfiable formula φ of BST+ admits a ρ-flat set model, for any $\rho \geqslant |\mathit{Vars}(\Phi)| + 1$.*

Proof. Let φ be a satisfiable formula of BST+, and let M be a set model of φ. Let ϕ_M^+ be the conjunction of all distinct atoms $x = y \setminus z$ occurring in φ that are satisfied by M. Likewise, let ϕ_M^- be the conjunction of all distinct literals $x \neq y \setminus z$ satisfied by M and such that $x = y \setminus z$ occurs in φ. Finally, let

$$\phi_M \coloneqq \phi_M^+ \wedge \phi_M^-. \tag{2.4}$$

Plainly, M satisfies ϕ_M by construction. Additionally, by propositional reasoning, every set model of ϕ_M satisfies our initial formula φ. Thus, it is enough to show that the conjunction ϕ_M admits a ρ-flat set model for every $\rho \geqslant n + 1$, where

$$n \coloneqq |\mathit{Vars}(\phi_M)| = |\mathit{Vars}(\Phi)|.$$

We prove that ϕ_M admits a ρ-flat set model by contracting each nonempty region R_W of M of the form

$$R_W \coloneqq \Big(\bigcap\{Mx \mid x \in W\}\Big) \setminus \bigcup\{My \mid y \in \mathit{Vars}(\phi_M) \setminus W\},$$

for $\emptyset \neq W \subseteq \mathit{Vars}(\phi_M)$, into a distinct singleton of rank $\rho+1$ (hence containing a single member of rank ρ).

Since the map $\kappa \mapsto 2^\kappa - \kappa$ is strictly increasing for $\kappa \geqslant 1$, for every integer $\rho \geqslant n+1$ we have

$$\begin{aligned}\left|\mathcal{V}_\rho^{\#}\right| &= |\mathcal{V}_{\rho+1}| - |\mathcal{V}_\rho| \\ &= 2^{|\mathcal{V}_\rho|} - |\mathcal{V}_\rho| \\ &\geqslant 2^{|\mathcal{V}_{n+1}|} - |\mathcal{V}_{n+1}| \\ &= |\mathcal{V}_{n+2}| - |\mathcal{V}_{n+1}| \\ &= \left|\mathcal{V}_{n+1}^{\#}\right| \geqslant 2^n.\end{aligned}$$

Hence, there exists an injective map $\Im_\rho \colon \mathscr{P}(\mathit{Vars}(\phi_M)) \to \mathcal{V}_\rho^{\#}$ from the collection of the nonempty subsets of $\mathit{Vars}(\phi_M)$ into $\mathcal{V}_\rho^{\#}$ of the (hereditary finite) sets of rank ρ.

Next, we define a set assignment M_ρ^* over $\mathit{Vars}(\phi_M)$ by putting

$$M_\rho^* x \coloneqq \{\Im_\rho(W) : x \in W \subseteq \mathit{Vars}(\phi_M) \wedge R_W \neq \emptyset\}.$$

By construction, the assignment M_ρ^* is ρ-flat. In addition, it is not hard to check that, for every $\emptyset \neq W \subseteq \mathit{Vars}(\phi_M)$, the region R_W^* of M^* defined by

$$R_W^* \coloneqq \Big(\bigcap\{M_\rho^* x : x \in W\}\Big) \setminus \bigcup\{M_\rho^* y : y \in \mathit{Vars}(\phi_M) \setminus W\}$$

is nonempty if and only if so is its corresponding region R_W of M. Thus, M_ρ^* satisfies ϕ_M. □

We are now ready to prove that membership atoms $x \in y$—and hence, atoms of the form $z = \{x\}$—are not existentially expressible in BST+.

Theorem 2.1. *The atom $x \in y$ is not existentially expressible in* BST+.

Proof. By way of contradiction, assume the atom $x \in y$ is existentially expressible in BST+. Thus there exists a BST+-formula $\psi(\vec{x}, x, y)$ such that

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

Since $x \in y$ is plainly satisfiable, so it is $\psi(\vec{x}, x, y)$ for some $\vec{x}$. Thus, by Lemma 2.11, it admits a ρ-flat model M_ρ. Therefore we obtain

$$M_\rho \not\models x \in y \wedge \psi(\vec{x}, x, y),$$

a contradiction. □

2.2.2 $\mathcal{O}(n^2)$-expressibility in BST+ of singleton-atom conjunctions

Conforming with Definition 2.2, we shall prove that any conjunction $\psi(\vec{x})$ of atoms of the form $x = \{y\}$ is $\mathcal{O}(n^2)$-expressible from BST into BST+ by exhibiting a map

$$\langle\, \varphi(\vec{y}\,),\ \psi(\vec{x}\,)\,\rangle \ \mapsto\ \Xi^{\psi}_{\varphi}(\vec{x},\vec{y},\vec{z}\,)$$

that can be computed in quadratic time and such that conditions (b) and (c) of Definition 2.2 are satisfied, where $\varphi(\vec{y}\,)$ ranges over the collection of BST-conjunctions and the variables in $\vec{z}$ are distinct from those in $\vec{x}$ and in $\vec{y}$.

Thus, let $\varphi(\vec{y})$ be any BST-conjunction and $\psi(\vec{x})$ be of the said form. For each variable $x \in \mathit{Vars}(\varphi\wedge\psi)$, we introduce a new distinct auxiliary variable $\tilde{x}$ (these variables will enforce that $x \in y$ only if $\tilde{x} \subsetneq \tilde{y}$). Then we put:

$$\begin{aligned}\Xi^{\psi}_{\varphi} \coloneqq \bigwedge_{x=\{y\}\text{ in }\psi} & x \not\subseteq y \ \wedge \\ & \bigwedge_{\substack{x=\{y\}\text{ in }\psi\\ x'=\{y'\}\text{ in }\psi}} \big(y = y' \longleftrightarrow x = x'\big) \ \wedge \bigwedge_{\substack{x=\{y\}\text{ in }\psi\\ v\in \mathit{Vars}(\varphi\wedge\psi)}} \big(\neg\mathsf{Disj}\,(x,v,\longrightarrow)\, x \subseteq v\big) \ \wedge \\ & \bigwedge_{\substack{x=\{y\}\text{ in }\psi\\ v\in \mathit{Vars}(\varphi\wedge\psi)}} \big(\neg\mathsf{Disj}\,(x,v,\longrightarrow)\, \tilde{y} \subsetneq \tilde{v}\big) \ \wedge \bigwedge_{x,y\in \mathit{Vars}(\varphi\wedge\psi)} \big(x = y \longrightarrow \tilde{x} = \tilde{y}\big)\end{aligned}$$

(thus, the list $\vec{z}$ of variables in Definition 2.2 is the collection $\widetilde{\vec{x}}$ of all auxiliary set variables $\tilde{x}$).

Plainly, Ξ^{ψ}_{φ} is a BST+-formula—in fact, it is a conjunction of a rather simple form—, which satisfies the following proposition, implying condition (a) of Definition 2.2:

Lemma 2.2. $\Xi^{\psi}_{\varphi} = \Theta\left(|\,\mathit{Vars}(\varphi\wedge\psi)|^2\right)$. □

The proof of this lemma is postponed to Sec. 2.2.1. In the next two subsections, we shall prove that

- if $\varphi(\vec{y}\,)\wedge\Xi^{\psi}_{\varphi}(\vec{x},\vec{y},\widetilde{\vec{x}}\,)$ is satisfiable, then so is $\varphi(\vec{y}\,)\wedge\psi(\vec{x}\,)$, and
- every model of $\varphi(\vec{y}\,)\wedge\psi(\vec{x}\,)$ can be extended into a model of $\Xi^{\psi}_{\varphi}(\vec{x},\vec{y},\widetilde{\vec{x}}\,)$,

thus showing that also conditions (b) and (c) of Definition 2.2 are fulfilled, which proves that every singleton-atom conjunction is $\mathcal{O}(n^2)$-expressible from BST into BST+.

Translation examples

Here we digress to provide a few examples illustrating how the conjunction Ξ^{ψ}_{φ} renders the formula $\varphi\wedge\psi$.

Singletons are characterized by the fact they contain exactly one element, that is why both

$$\bigwedge_{x=\{y\} \text{ in } \psi} x \not\subseteq y \quad \text{and} \quad \bigwedge_{\substack{x=\{y\} \text{ in } \psi \\ v \in Vars(\varphi \wedge \psi)}} \big(\neg \mathsf{Disj}\,(x, v, \longrightarrow)\, x \subseteq v\big)$$

(the former of which entails $x \neq \varnothing$) occur in our translation. Moreover the singleton-atom $x = \{y\}$ plainly implies $y \in x$, therefore the ordering relation induced by $\in$ must be preserved.[2] This is the rationale for including, in our translation, the constraints

$$\bigwedge_{\substack{x=\{y\} \text{ in } \psi \\ v \in Vars(\varphi \wedge \psi)}} \big(\neg \mathsf{Disj}\,(x, v, \longrightarrow)\, \tilde{y} \subsetneq \tilde{v}\big) \quad \text{and} \quad \bigwedge_{x,y \in Vars(\varphi \wedge \psi)} \big(x = y \longrightarrow \tilde{x} = \tilde{y}\big)\,.$$

Example 2.2. *The conjunction $\varphi \wedge \psi$, where*

$$\varphi \coloneqq a = b \setminus c \qquad \text{and} \qquad \psi \coloneqq x = \{y\} \ \wedge \ y = \{z\} \ \wedge \ z = \{x\},$$

is plainly unsatisfiable because it implies the membership cycle $x \in z \in y \in x$. To reflect this, Ξ^{ψ}_{φ} encompasses the literals and implications

$$\begin{aligned} x \neq \varnothing\,, &\quad \big(\neg \mathsf{Disj}\,(x, x, \longrightarrow)\, \tilde{y} \subsetneq \tilde{x}\big)\,, \\ y \neq \varnothing\,, &\quad \big(\neg \mathsf{Disj}\,(y, y, \longrightarrow)\, \tilde{z} \subsetneq \tilde{y}\big)\,, \\ z \neq \varnothing\,, &\quad \big(\neg \mathsf{Disj}\,(z, z, \longrightarrow)\, \tilde{x} \subsetneq \tilde{z}\big)\,, \end{aligned}$$

showing that it is unsatisfiable due to the cycle $\tilde{x} \subsetneq \tilde{z} \subsetneq \tilde{y} \subsetneq \tilde{x}$.

Example 2.3. *The conjunction $\varphi \wedge \psi$, where $\varphi \coloneqq y = x \setminus z$ and $\psi \coloneqq x = \{y\} \wedge y = \{z\}$, is not satisfiable; in fact any set assignment M satisfying this formula is such that $My = Mx \setminus Mz$, $Mx = \{My\}$, and $My = \{Mz\}$; thus $My = Mz$, and hence $My = \{My\}$, a contradiction.*

Our translation Ξ^{ψ}_{φ} comprises:

$$y \neq \varnothing \ \wedge \ x \not\subseteq y \ \wedge \ \big(\neg \mathsf{Disj}\,(x, y, \longrightarrow)\, x \subseteq y\big).$$

Any model M for $\varphi \wedge \Xi^{\psi}_{\varphi}$ is such that, $My \subseteq Mx$ and $My \neq \emptyset$ hold, so that $\neg \mathsf{Disj}\,(Mx, My)$; but then $Mx \subseteq My$ must hold, which conflicts with $x \not\subseteq y$.

As we will prove in Sec. 2.2.1, satisfiability carries over from $\varphi \wedge \psi$ to $\varphi \wedge \Xi^{\psi}_{\varphi}$. Here is an interesting example of this:

Example 2.4. *The conjunction $\varphi \wedge \psi$, where*

$$\varphi \coloneqq x = y \setminus y' \ \wedge \ x = z \setminus z', \quad \text{and} \quad \psi \coloneqq z = \{y\},$$

[2] See, below, the beginning of Sec. 2.2.1.

is satisfiable. In fact, $\varphi \wedge \psi$ is plainly satisfied by every set assignment M over $Vars(\varphi \wedge \psi)$ of the form

$$Mx \coloneqq \emptyset, \quad My \coloneqq s, \quad Mz \coloneqq \{s\}, \quad My' \coloneqq s \cup s', \quad Mz' \coloneqq \{s\} \cup s'', \quad (2.5)$$

where s, s', and s'' are any well-founded sets.

It can easily be checked that, for every set assignment M of the form (2.5), the extension M^+ of M over the auxiliary variables $\tilde{v}$, for $v \in Vars(\varphi \wedge \psi)$ and where

$$M^+\tilde{v} \coloneqq \begin{cases} s \cup \{s\} & \text{if } s \in Mv \\ \emptyset & \text{otherwise,} \end{cases}$$

satisfies Ξ^ψ_φ, so that $M \models (\exists\, \tilde{x}, \tilde{y}, \tilde{z}, \tilde{y}', \tilde{z}')\Xi^\psi_\varphi$ holds. Thus, to show that condition (c) of Definition 2.2 is satisfied, namely that $\models \big(\varphi \wedge \psi\big) \longrightarrow (\exists\, \tilde{x}, \tilde{y}, \tilde{z}, \tilde{y}', \tilde{z}')\Xi^\psi_\varphi$ holds, it is enough to check that the conjunction $\varphi \wedge \psi$ is satisfied by set assignments of the form (2.5) only. Let then $\overline{M}$ be any model for $\varphi \wedge \psi$. Since $\overline{M}x \subseteq \overline{M}z$ and $\overline{M}z = \{\overline{M}y\}$, either $\overline{M}x = \emptyset$ or $\overline{M}x = \{\overline{M}y\}$ holds. The latter case can be readily ruled out, for in view of $\overline{M}x \subseteq \overline{M}y$ it would follow $\overline{M}y \in \overline{M}y$, which is untenable in the realm of well-founded sets. Thus, $\overline{M}x = \emptyset$ must hold. Letting $s \coloneqq \overline{M}y$—so that $\overline{M}z = \{s\}$—, since $\overline{M}y \subseteq \overline{M}y'$ and $\overline{M}z \subseteq \overline{M}z'$, we have $\overline{M}y' = s \cup s'$ and $\overline{M}z' = \{s\} \cup s''$, for some sets s' and s'', and therefore $\overline{M}$ has the form (2.5).

2.2.3 If $\varphi \wedge \Xi^\psi_\varphi$ is satisfiable, then so is $\varphi \wedge \psi$

In order to convert a model $M^{(0)}$ of $\varphi \wedge \Xi^\psi_\varphi$ into a model of $\varphi \wedge \psi$, we can assume that $M^{(0)}$ is ρ-flat over $Vars(\varphi \wedge \Xi^\psi_\varphi)$ for some integer $\rho > \big|Vars(\varphi \wedge \Xi^\psi_\varphi)\big|$: on the one hand, Lemma 2.11 enables us to do so; on the other hand, Lemma 2.10 tells us that such an $M^{(0)}$ does not model any of the atoms $x = \{y\}$ in ψ. For $i = 1, \ldots, m$ (where m is a suitable integer, $0 \leqslant m \leqslant |\{\text{atoms in } \psi\}|$), we will lift $M^{(i-1)}$ into a model $M^{(i)}$ of $\varphi \wedge \Xi^\psi_\varphi$ such that the atoms in ψ modeled by $M^{(i-1)}$ remain true in $M^{(i)}$, while at least one conjunct $x_i = \{y_i\}$ of ψ, false in $M^{(i-1)}$, becomes true in $M^{(i)}$. This iterative process will end as soon as all atoms in ψ are true; actually, if ψ_i is the conjunction of those atoms in ψ which $M^{(i)}$ makes true for the first time, $\psi_1 \wedge \cdots \wedge \psi_m$ will coincide with ψ—up to an atom rearrangement.

In our set up, each ψ_i must hence comprise at least a conjunct $x_i = \{y_i\}$ of ψ not appearing in any ψ_j with $j < i$. The selection of $x_i = \{y_i\}$ will shape the transformation of $M^{(i-1)}$ into $M^{(i)}$ and will be based on the criterion that the atom in question be minimal, in ψ (deprived of all atoms in $\psi_1 \wedge \cdots \wedge \psi_{i-1}$), in regard to the ordering $\prec^{M^{(0)}}$ induced on the conjuncts of ψ as specified below:

Definition 2.5. *Given a model M of $\varphi \wedge \Xi^\psi_\varphi$, we put $\ell \mathrel{\dot\prec^M} \ell'$ for all atoms $\ell \coloneqq x{=}\{y\}$ and $\ell' \coloneqq x'{=}\{y'\}$ in ψ such that $\neg\mathsf{Disj}\,(Mx, My')$ holds. Then, for all conjuncts ℓ and ℓ' in ψ, we put $\ell \prec^M \ell'$ provided that*

$$\ell = \ell_0 \mathrel{\dot\prec^M} \ell_1 \mathrel{\dot\prec^M} \cdots \mathrel{\dot\prec^M} \ell_n = \ell',$$

for some $\ell_0, \ell_1, \ldots, \ell_n$ *in* ψ *with* $n \geqslant 1$.

This definition enforces that

Lemma 2.3. *The relation* $\prec^M$ *is a strict partial order.*

Proof. Plainly, $\prec^M$ is a transitive relation; it will hence suffice to prove that there is no cycle $\ell_0 \prec^M \cdots \prec^M \ell_n = \ell_0$. Assuming, by way of contradiction, that such a cycle exists, there would be atoms $\ell'_0, \ell'_1, \ldots, \ell'_m$ in ψ such that

$$\ell'_0 \mathrel{\dot\prec^M} \ell'_1 \mathrel{\dot\prec^M} \cdots \mathrel{\dot\prec^M} \ell'_{m-1} \mathrel{\dot\prec^M} \ell'_m = \ell'_0,$$

where ℓ'_i has the form $x_i{=}\{y_i\}$ for $i = 0, 1, \ldots, m$. Then, we would have

$$\neg\mathsf{Disj}\,(Mx_0, My_1, ,) \ldots, \neg\mathsf{Disj}\,(Mx_{m-1}, My_0,)\,.$$

Since $M \models \Xi^\psi_\varphi$, then in particular

$$M \models \bigwedge_{\substack{x=\{y\} \text{ in } \psi \\ v \in Vars(\varphi\wedge\psi)}} \big(\neg\mathsf{Disj}\,(x, v, \longrightarrow)\,\tilde{y} \subsetneq \tilde{v}\big)\,,$$

and therefore

$$M\tilde{y}_0 \subsetneq M\tilde{y}_1 \subsetneq \cdots \subsetneq M\tilde{y}_{m-1} \subsetneq M\tilde{y}_0\,,$$

thus yielding the contradiction $M\tilde{y}_0 \subsetneq M\tilde{y}_0$.

The aciclicity of $\prec^M$ entails its reflexivity, whence the claim follows. □

Let M be a set assignment over the variables of $\varphi \wedge \Xi^\psi_\varphi$, and $x = \{y\}$ be any atom in ψ such that

$$\forall v \in Vars(\varphi \wedge \psi)\,\big(\,My \notin Mv\,\big)\,. \tag{2.6}$$

We define the set assignment $M_{x,y}$ by putting, for all $v \in Vars(\varphi \wedge \psi)$:

$$M_{x,y}v \coloneqq \begin{cases} Mv & \text{if } \mathsf{Disj}\,(Mx, Mv,)\,, \\ \big(\,Mv \setminus Mx\,\big) \uplus \{My\} & \text{otherwise,} \end{cases} \tag{2.7}$$

and

$$M_{x,y}\tilde{v} \coloneqq M\tilde{v}\,. \tag{2.8}$$

(As usual, $\uplus$ denotes disjoint set union.)

It turns out that the set assignments $M_{x,y}$ and M model the very same literals of type $u \subseteq v$ and $\mathsf{Disj}\,(u, v)$, with $u, v \in Vars(\varphi \wedge \psi)$. This is proved in the next two lemmas.

Lemma 2.4. *Let* $x = \{y\}$ *be an atom in* ψ *and* M *be a model of* $\varphi \wedge \Xi^\psi_\varphi$ *satisfying condition* (2.6)*; then* $Mu \subseteq Mv \longleftrightarrow M_{x,y}u \subseteq M_{x,y}v$ *, for all* $u, v \in Vars(\varphi\wedge\psi)$.

Proof. Plainly $Mu \subseteq Mv \longrightarrow M_{x,y}u \subseteq M_{x,y}v$. To get the converse, suppose that $M_{x,y}u \subseteq M_{x,y}v$ and $s \in Mu$: we must prove that $s \in Mv$. On the one hand, if $s \notin Mx$ then $s \in Mu \setminus Mx \subseteq M_{x,y}u \subseteq M_{x,y}v$ and $s \neq My$ hold, entailing $s \in Mv$. On the other hand, if $s \in Mx$ holds, then $\neg\mathsf{Disj}\,(Mx, Mu)$ follows from $s \in Mx \cap Mu$, and therefore $My \in M_{x,y}u \subseteq M_{x,y}v$ and $\neg\mathsf{Disj}\,(Mx, Mv)$ hold. Since $M \models \Xi^{\psi}_{\varphi}$, we then have $Mx \subseteq Mv$, and thus $s \in Mv$ again, as sought. □

Lemma 2.5. *Let $x = \{y\}$ be an atom in ψ and M be a model of $\varphi \wedge \Xi^{\psi}_{\varphi}$ satisfying* (2.6)*; then* $\mathsf{Disj}\,(Mu, Mv) \longleftrightarrow \mathsf{Disj}\,(M_{x,y}u, M_{x,y}v)$ *holds for all $u, v \in Vars(\varphi \wedge \psi)$.*

Proof. Plainly $\neg\mathsf{Disj}\,(Mu, Mv) \longrightarrow \neg\mathsf{Disj}\,(M_{x,y}u, M_{x,y}v)$. To get the converse, suppose that $s \in M_{x,y}u \cap M_{x,y}v$: we must prove that $Mu \cap Mv \neq \emptyset$. If $s \neq My$, then $s \in Mu \setminus Mx$ and $s \in Mv \setminus Mx$; so we are done. Otherwise, Mx intersects both of Mu and Mv; but then, since $M \models \Xi^{\psi}_{\varphi}$, we get $\emptyset \neq Mx \subseteq Mu \cap Mv$. □

Lemma 2.6. *Let $x = \{y\}$ be an atom in ψ and M be a model of $\varphi \wedge \Xi^{\psi}_{\varphi}$ satisfying* (2.6)*. Then $M_{x,y} \models \varphi \wedge \Xi^{\psi}_{\varphi} \wedge x = \{y\}$ and $M_{x,y}y = My$, so that $M_{x,y}x \neq Mx$.*

Proof. To prove that $M_{x,y} \models \varphi$, we will get $M_{x,y} \models \phi$ from $M \models \phi$ for each conjunct ϕ of φ.

Consider first a conjunct $u = v \setminus w$ of φ. By Lemmas 2.4 and 2.5, $M_{x,y}u \subseteq M_{x,y}v \setminus M_{x,y}w$ follows from $Mu = Mv \setminus Mw$. Supposing then that $s \in M_{x,y}v \setminus M_{x,y}w$, we must prove that $s \in M_{x,y}u$. In fact, either $s \in Mv \setminus Mx$ or $s = My$ and $Mx \cap Mv \neq \emptyset$ holds: in the former case, from $s \notin M_{x,y}w$ we get $s \notin Mw$, hence $s \in (Mv \setminus Mw) \setminus Mx = Mu \setminus Mx \subseteq M_{x,y}u$, so that $s \in M_{x,y}u$; in the latter case, we get $Mx \cap Mw = \emptyset$ and hence $Mx \cap Mu = Mx \cap (Mv \setminus Mw) = Mx \cap Mv \neq \emptyset$, and therefore again $s \in M_{x,y}u$. The inclusions $M_{x,y}u \subseteq M_{x,y}v \setminus M_{x,y}w$ and $M_{x,y}u \supseteq M_{x,y}v \setminus M_{x,y}w$ just shown sum up to $M_{x,y}u = M_{x,y}v \setminus M_{x,y}w$; that is, $M_{x,y} \models u = v \setminus w$.

Consider next a literal $u \neq v \setminus w$ in φ. If $Mu \not\subseteq Mv \setminus Mw$ then either $Mu \not\subseteq Mv$ or $Mu \subseteq Mv \wedge Mu \cap Mw \neq \emptyset$ holds; accordingly, by Lemmas 2.4 and 2.5, $M_{x,y}u \not\subseteq M_{x,y}v \setminus M_{x,y}w$ holds. On the other hand, if $Mv \setminus Mw \not\subseteq Mu$, then there exists an $s \in Mv$ such that $s \notin Mu \cup Mw$. If $s \notin Mx$, then $s \in M_{x,y}v$ and $s \notin M_{x,y}u \cup M_{x,y}w$, hence $s \in M_{x,y}v \setminus M_{x,y}w$ and $M_{x,y}u \not\supseteq M_{x,y}v \setminus M_{x,y}w$; otherwise, $My \in M_{x,y}v$ follows from $s \in Mx \cap Mv$, whereas $My \notin M_{x,y}u \cup M_{x,y}w$, else either $Mx \cap Mu \neq \emptyset$ or $Mx \cap Mw \neq \emptyset$ would hold, entailing (due to the fact that M models Ξ^{ψ}_{φ}) the contradiction $s \in Mx \subseteq Mu \cup Mw$; thus My witnesses that $M_{x,y}u \not\supseteq M_{x,y}v \setminus M_{x,y}w$ when $s \in Mx$. Summing up, we get $M_{x,y}u \neq M_{x,y}v \setminus M_{x,y}w$ in all cases; that is, $M_{x,y} \models u \neq v \setminus w$.

Concerning the atom $x = \{y\}$, it follows from $M \models \Xi^{\psi}_{\varphi}$ that $M \models x \not\subseteq y \wedge \big(x \not\subseteq y \longrightarrow \mathsf{Disj}\,(x, y)\big)$, and hence $M \models x \neq \varnothing \wedge \mathsf{Disj}\,(x, y)$; therefore, $M_{x,y}y = My$ and $M_{x,y}x = \{My\} = \{M_{x,y}y\}$ hold. That is, $M_{x,y} \models x = \{y\}$.

What precedes suffices to prove that $M_{x,y} \models \varphi \wedge x = \{y\}$; since moreover $M_{x,y}\tilde{v} \coloneqq M\tilde{v}$ holds according to (2.8), Lemmas 2.4 and 2.5 also yield $M_{x,y} \models \Xi^{\psi}_{\varphi}$. □

Corollary 2.12. *Let M be a model of $\varphi \wedge \Xi^{\psi}_{\varphi}$ satisfying* (2.6) *for some atom $x = \{y\}$ in ψ, and ψ' be the conjunction of all atoms $x' = \{y'\}$ in ψ satisfying $M_{x,y}x' \neq Mx'$. Then $M_{x,y} \models \psi'$. Also, $x = \{y\}$ belongs to ψ', $M_{x,y}y = My$, and $M_{x,y} \models x' = x \wedge y' = y$ for each $x' = \{y'\}$ in ψ.*

Proof. When $x' = \{y'\}$ in ψ, from $M_{x,y}x' \neq Mx'$ we get $\neg\mathsf{Disj}\,(Mx, Mx')$. Therefore, since

$$M \models \bigwedge_{\substack{x''=\{y''\} \text{ in } \psi \\ v \in Vars(\varphi\wedge\psi)}} \big(\neg\mathsf{Disj}\,(x'', v) \longrightarrow x'' \subseteq v\big),$$

we obtain $Mx = Mx'$; and then, by (2.7), $M_{x,y}x' = M_{x,y}x = \{My\}$. From $Mx = Mx'$ we also get $My = My'$, and thus $My = M_{x,y}y = M_{x,y}y'$, whence $M_{x,y}x' = \{M_{x,y}y'\}$. The genericness of $x' = \{y'\}$ in ψ' yields $M_{x,y} \models \psi'$. □

Lemma 2.7. *If $\varphi \wedge \Xi^{\psi}_{\varphi}$ is satisfiable, so is $\varphi \wedge \psi$.*

Proof. Suppose that $\varphi \wedge \Xi^{\psi}_{\varphi}$ is satisfiable; then it has a ρ-flat model $M^{(0)}$ with $\rho > \big|Vars(\varphi \wedge \Xi^{\psi}_{\varphi})\big| \geqslant |Vars(\varphi \wedge \psi)|$, by Lemma 2.11. Belaboring the idea sketched at the beginning of Sec. 2.2.1, we consider the ordering $\prec^{M^{(0)}}$ between atoms of ψ as defined in Definition 2.5 and repeatedly perform the following actions:

i. choose an atom $\ell_i \coloneqq x_i{=}\{y_i\}$ in ψ, minimal in regard to $\prec^{M^{(0)}}$, that does not appear in ψ_j for any $j < i$;

ii. define $M^{(i)} \coloneqq M^{(i-1)}_{x_i,y_i}$ and let ψ_i be the collection of all atoms $x' = \{y'\}$ in ψ such that $M^{(i)}x' \neq M^{(i-1)}x'$;

iii. finally, take m to be the value such that $\psi_1 \wedge \cdots \wedge \psi_m = \psi$.

We will prove by induction on $i = 1, \ldots, m$ that:

1) Condition (2.6) holds when $M = M^{(i-1)}$ and $y = y_i$. — Thus $M^{(i)} \models \varphi \wedge \Xi^{\psi}_{\varphi} \wedge \psi_i$ holds, where $x_i = \{y_i\}$ is a conjunct of ψ_i, by Lemma 2.6 and Corollary 2.12.

2) $M^{(i)}y = M^{(k-1)}y_k$ and $M^{(i)}x = \{M^{(k-1)}y_k\}$ hold for $1 \leqslant k \leqslant i$ and for every atom $x = \{y\}$ in ψ_k. — Consequently $M^{(i)} \models \psi_1 \wedge \cdots \wedge \psi_i$ holds.

3) $\mathsf{rk}\big(M^{(i)}v\big) \in \{0, \ldots, i\} \cup \{\rho{+}1, \ldots, \rho{+}1{+}i\}$ holds for every $v \in Vars(\varphi \wedge \psi)$; therefore $M^{(i)}v \notin \mathcal{V}^{\#}_{\rho}$, if we assume w.l.o.g. that $\rho > m$.

4) The inclusion $M^{(i)}v \subseteq \big\{M^{(i)}y : x = \{y\} \text{ in } \psi_1 \wedge \cdots \wedge \psi_i\big\} \uplus \mathcal{V}^{\#}_{\rho}$ holds for each $v \in Vars(\varphi \wedge \psi)$.

Thus, by 1)–2), $M^{(m)} \models \varphi \wedge \psi$ will finally hold, settling the claim of this lemma.

Notice, in passing and in view of Lemma 2.5, that $\ell \prec^{M^{(i)}} \ell' \longleftrightarrow \ell \prec^{M^{(i-1)}} \prec \ell'$ will also follow from our construction and induction hypotheses for all $\ell, \ell' \in \psi$; we will hence get (omitting the model's superscript) $\ell_{k'} \not\prec^{M} \ell_k$ for $1 \leqslant k < k' \leqslant m$.

Case $i = 1$. Concerning 1), since $M^{(0)}$ is ρ-flat, by Lemma 2.10 condition (2.6) holds when $M = M^{(0)}$ for each atom $x = \{y\}$ in ψ, and in particular when $y = y_1$. The claim 2) readily follows from Corollary 2.12 in this case, as it also does, for any i, when $i = k$.

To get 3), notice that $M^{(1)}v \subseteq M^{(0)}v \cup \{M^{(0)}y_1\}$ readily follows from the definition of $M^{(1)}$; then, since $M^{(0)}$ is ρ-flat, $\mathsf{rk}\left(M^{(0)}v\right) \in \{0, 1, \rho+1, \rho+2\}$. To get 4), rely on the inclusion $M^{(1)}v \subseteq \{M^{(0)}y_1\} \cup M^{(0)}v$ just noticed, where $M^{(0)}v \subseteq \mathcal{V}_\rho^{\#}$ (due to ρ-flatness), $M^{(0)}y_1 = M^{(1)}y_1 \notin \mathcal{V}_\rho^{\#}$ (by 2) and 3) already proved for $i = 1$), and $x_1 = \{y_1\}$ belongs to ψ_1 (as noted under 1)).

Case $i > 1$. As regards 1), arguing by contradiction suppose that $M^{(i-1)}y_i \in M^{(i-1)}v$ holds for some $v \in \mathit{Vars}(\varphi \wedge \psi)$. Since $M^{(i-1)}v \subseteq \left\{M^{(i-1)}y : x = \{y\} \text{ in } \psi_1 \wedge \cdots \wedge \psi_{i-1}\right\} \uplus \mathcal{V}_\rho^{\#}$ holds by the induction hypothesis 4) but $M^{(i-1)}y_i \in \mathcal{V}_\rho^{\#}$ is ruled out by 3), we get $M^{(i-1)}y_i \in \left\{M^{(i-1)}y : x = \{y\} \text{ in } \psi_1 \wedge \cdots \wedge \psi_{i-1}\right\}$; accordingly, $M^{(i-1)}y_i = M^{(i-1)}y_j$ holds for some $j < i$, thanks to Corollary 2.12. Through Lemma 2.4, our induction ensures that $M^{(j)} \models \Xi_\varphi^\psi$, $M^{(j)} \models y_i = y_j$, $M^{(j)} \models x_i = x_j \leftrightarrow y_i = y_j$, and thus $M^{(j)}x_j = M^{(j)}x_i$ and $M^{(j)}y_j = M^{(j-1)}y_j \in M^{(j)}x_i$. Since $M^{(j-1)}y_j \notin M^{(j-1)}x_i$ holds, by hypothesis 1), the literal $x_i = \{y_i\}$ must belong to ψ_j, which leads us to the sought contradiction.

As for 2), supposing the contrary, there should exist a k and a least h such that $k < h \leqslant i$ and that either $M^{(h)}y \neq M^{(h-1)}y$ or $M^{(h)}x \neq \{M^{(h-1)}y\}$ holds for some literal $x = \{y\}$ in ψ_k. In the former case, we should have $\neg\, \mathsf{Disj}\left(M^{(h-1)}x_h, M^{(h-1)}y_k\right)$, which leads to the contradiction $\ell_h \prec^{M^{(0)}} \ell_k$. Therefore, we must have $M^{(h)}y = M^{(h-1)}y = M^{(k-1)}y_k$ and $M^{(h)}x \neq \{M^{(k-1)}y_k\} = M^{(h-1)}x$. This is untenable, though: in fact, by Corollary 2.12, $M^{(h)}x \neq M^{(h-1)}x$ implies $M^{(h)}x = \{M^{(h)}y\}$, whence $M^{(h)}x = \{M^{(k-1)}y_k\}$ follows.

As regards 3), the construction of $M^{(i)}$ gives us $M^{(i)}v \subseteq M^{(i-1)}v \cup \{M^{(i-1)}y_i\}$ where, by the induction hypothesis 3), either $0 \leqslant \mathsf{rk}\left(M^{(i-1)}w\right) < i$ or $\rho + 1 \leqslant \mathsf{rk}\left(M^{(i-1)}w\right) \leqslant \rho + i$ holds for each w in $\mathit{Vars}(\varphi \wedge \psi)$, and in particular for w in $\{v, x_i\}$. We will have either $0 \leqslant \mathsf{rk}\left(M^{(i)}v\right) < i$ or $\rho + 1 \leqslant \mathsf{rk}\left(M^{(i)}v\right) \leqslant \rho + i$ when $M^{(i)}v \subseteq M^{(i-1)}v$; the upper bounds i and $\rho + i$ must be increased by one in case $M^{(i-1)}y_i \in M^{(i)}v$.

Finally, concerning 4), we have that:

$$\begin{aligned} M^{(i)}v &\subseteq M^{(i-1)}v \cup \{M^{(i-1)}y_1\} && \text{(by definition of } M^{(i)}) \\ &\subseteq \big\{M^{(i-1)}y : x = \{y\} \text{ in } \psi_1 \wedge \cdots \wedge \psi_{i-1}\big\} \uplus \mathcal{V}_\rho^{\#} \cup \{M^{(i-1)}y_i\} \\ &&& \text{(by the induction hypothesis 4))} \\ &= \big\{M^{(i)}y : x = \{y\} \text{ in } \psi_1 \wedge \cdots \wedge \psi_{i-1}\big\} \uplus \mathcal{V}_\rho^{\#} \cup \{M^{(i)}y_i\} \\ &&& \text{(by the induction hypothesis 2))} \\ &= \big\{M^{(i)}y : x = \{y\} \text{ in } \psi_1 \wedge \cdots \wedge \psi_i\big\} \uplus \mathcal{V}_\rho^{\#} \\ &&& \text{(since } M^{(i)} \models \Xi_\varphi^\psi \text{ and by the induction hypothesis 3)).} \end{aligned}$$

□

2.2.4 Each model of $\varphi \wedge \psi$ can be extended into a model of $\varphi \wedge \Xi_\varphi^\psi$

Let $\varphi \wedge \psi$ be satisfiable and let M be a model of it. Consider the relation $\prec$ between set variables induced by membership according to the rules

(i) $x \prec y$ if $Mx \in My$, and

(ii) if $x \prec y$ and $y \prec z$ then $x \prec z$,

so that, plainly, $\prec$ is a strict ordering.

Extend M to the set variables $\tilde{v}$ by putting, for each of them:

$$M\tilde{v} \coloneqq \{Mx : x \prec v\}.$$

To prove that $M \models \Xi_\varphi^\psi$, first notice that Mx is a singleton for all $x = \{y\}$ in ψ, so that we have

$$M \models \bigwedge_{\substack{x=\{y\} \text{ in } \psi \\ v \in Vars(\varphi \wedge \psi)}} \big(\neg \mathsf{Disj}\,(x, v, \longrightarrow)\, x \subseteq v\big).$$

Moreover M models all pair of atoms $x = \{y\}$ and $x' = \{y'\}$ in ψ so that

$$M \models \bigwedge_{x=\{y\} \text{ in } \psi} x \nsubseteq y$$

and

$$M \models \bigwedge_{\substack{x=\{y\} \text{ in } \psi \\ x'=\{y'\} \text{ in } \psi}} \big(y = y' \longleftrightarrow x = x'\big).$$

Concerning any of the implications

$$\neg \mathsf{Disj}\,(x, v, \longrightarrow)\, \tilde{y} \subsetneq \tilde{v}$$

with $x = \{y\}$ in ψ and $v \in Vars(\varphi \wedge \psi)$, assume that $\neg \mathsf{Disj}\,(Mx, Mv)$; then, since $Mx = \{My\}$, we have that $My \in Mv$ and hence $y \prec v$, so that transitivity and strictness of $\prec$ yield $\{My' \mid y' \prec y\} \subsetneq \{Mv' \mid v' \prec v\}$; that is, $M\tilde{y} \subsetneq M\tilde{v}$.

Finally, as for the implications

$$x = y \longrightarrow \tilde{x} = \tilde{y}$$

with $x, y \in \mathit{Vars}(\varphi \wedge \psi)$, assume $Mx = My$. When $v \prec x$, a tuple $v_1, \ldots, v_n$ of set variables exists such that $v = v_1$, $Mv_i \in Mv_{i+1}$ holds for each i, and $v_n = x$; thus $Mv = Mv_1 \in \cdots \in Mv_n = Mx = My$, and hence $v \prec y$. Analogously $v \prec x$ follows from $v \prec y$. Thus $\{v \mid v \prec x\} = \{v \mid v \prec y\}$; that is, $M\tilde{x} = M\tilde{y}$.

We have so extended a generic M such that $M \models \varphi \wedge \psi$ into a model of $\varphi \wedge \Xi^{\psi}_{\varphi}$; therefore we get

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

Putting together Lemmas 2.2 and 2.7, and the conclusion just reached, we get:

Theorem 2.8. *Membership conjunctions are $\mathcal{O}(n^2)$-expressible from* BST *into* BST+. □

2.2.5 Design and analysis of the translation algorithm

In order to prove Lemma 2.2, we provide a detailed specification of the algorithm that generates the formula Ξ^{ψ}_{φ} out of the conjunction $\varphi \wedge \psi$.

1: Initialize $\mathit{Vars}(\varphi \wedge \psi)$ as an empty list of set variables;
2: Initialize Ξ^{ψ}_{φ} as an empty list of conjuncts;
3: **for** each set variable x that appears in φ **do**
4: add x to $\mathit{Vars}(\varphi \wedge \psi)$;
5: **for** each conjunct $x = \{y\}$ in ψ **do**
6: add x and y to $\mathit{Vars}(\varphi \wedge \psi)$;
7: add $x \nsubseteq y$ to Ξ^{ψ}_{φ};
8: **for** each conjunct $x = \{y\}$ in ψ **do**
9: **for** each $v \in \mathit{Vars}(\varphi \wedge \psi)$ **do**
10: add $\big(\neg\mathsf{Disj}\,(x, v, \longrightarrow)\, x \subseteq v\big) \wedge \big(\neg\mathsf{Disj}\,(x, v, \longrightarrow)\, \tilde{y} \subsetneq \tilde{v}\big)$ to Ξ^{ψ}_{φ};
11: **for** each pair $x = \{y\}$, $x' = \{y'\}$ of distinct conjuncts in ψ **do**
12: add $\big(y = y' \longleftrightarrow x = x'\big)$ to Ξ^{ψ}_{φ};
13: **for** all $x, y \in \mathit{Vars}(\varphi \wedge \psi)$ **do**
14: add $(x = y \longrightarrow \tilde{x} = \tilde{y})$ to Ξ^{ψ}_{φ}.

Adding elements to $\mathit{Vars}(\varphi \wedge \psi)$ and to Ξ^{ψ}_{φ} will require constant time if these are implemented as lists of set variables and conjuncts.

The **for**-loop at lines 3 and 4 can be performed in $\Theta(|\varphi|)$-time, where $|\varphi|$ is the total length of the conjunction φ; similarly the **for**-loop from line 5 to line 7 can be performed in $\Theta(|\psi|)$-time. The **for**-loop from line 8 to line 10 is iterated $\Theta\left(|\psi \times \mathit{Vars}(\varphi \wedge \psi)|\right)$ times, the **for**-loop at lines 11 and 12 is iterated $\Theta\big(\,|\psi|^2\,\big)$ times, and the **for**-loop at lines 13 and 14 is iterated $\Theta\left(|\,\mathit{Vars}(\varphi \wedge \psi)|^2\right)$ times.

The overall time complexity then is $\Theta\big(\,|\varphi \wedge \psi| + |\psi \times \mathit{Vars}(\varphi \wedge \psi)| + |\psi|^2 + |\,\mathit{Vars}(\varphi \wedge \psi)|^2\,\big)$, and since $|\,\mathit{Vars}(\varphi \wedge \psi)| = \mathcal{O}\big(\,|\varphi \wedge \psi|\,\big)$, we can say that Ξ^{ψ}_{φ} can be generated in $\mathcal{O}\left(|\varphi \wedge \psi|^2\right)$-time.

Chapter 3

A complete complexity taxonomy for Boolean Set Theory

Boolean Set Theory (BST) is one of the two main sub-languages of MLS, defined as the fragment obtained by removing the membership relation symbol $\in$ from MLS. As for MLS the decision problem for BST also belongs to the class of NP-complete problems, as we will show in Section 3.1. Several of its fragments however admits polynomial-time decision procedure (see Sections 3.2,3.3 and 3.3.4). A comprehensive study of the fragments of BST is not only of foundational interest, finding the boundaries between NP-complete and deterministic polynomial-time problems inside BST, but it is also of a practical interest, as these results can be used to improve the performance of those systems that require to solve the decision problem inside BST to operate, as for example the automated proof verifier ÆtnaNova.

In this chapter we will present a complete complexity taxonomy of the fragments of BST. Such taxonomy was first presented in [CDMO21]. It comprises thousands of fragments, 2040 to be precise, 1278 whose satisfiability problem is NP-complete (for short we call those fragments **NP-complete fragments**), and the remaining 762 whose decision problem can be solved in deterministic polynomial time (for short we call such fragments **polynomial fragments**). A complete list of all the fragments would be overwhelming and difficult to read. To remedy this problem we focus on the symmetric families of:

- **minimal NP-complete fragments:** these are the NP-complete fragments whose sub-theories are polynomial, i.e., removing any symbol from them results in a polynomial fragment of BST; and
- **maximal polynomial fragments:** these are the polynomial fragments whose super-theories are NP-complete, i.e., by adding any symbol of BST to them results in a NP-complete fragment of BST.

Such two families contain far less fragments and can easily be presented in a single table (see Table 3.4), which plainly suffices as a tool to establish whether a fragment is NP-complete or polynomial. In fact any fragment will be either a sub-theory of a maximal polynomial fragment or a super-theory of a minimal NP-complete fragment.

Corollary 2.2 states that the language MLS can be reduced to the collection of all conjunctions of atoms of the following two types:

$$x = y \setminus z, \quad x \in y.$$

By applying the same reductions to BST, Corollary 2.3 allow us to reduce BST to the collection of all the conjunctions of atoms of the two types:

$$x = y \setminus z, \quad x \neq y,$$

where x, y and z are set variables. Although this reduced version of BST is quite useful to simplify otherwise long proofs (see Lemma 2.7 for instance), it is detrimental for our study of a complete BST complexity taxonomy. In fact we would have just 5 fragments of BST:

$$\mathsf{BST}(\setminus, =, \neq), \quad \mathsf{BST}(\setminus, =), \quad \mathsf{BST}(\setminus, \neq)$$
$$\mathsf{BST}(=), \quad \mathsf{BST}(\neq);$$

lacking most fragments, in particular most polynomial fragments such as $\mathsf{BST}(\cup, =, \neq)$.

We need to take a step backward and give, like we did for MLS, a broader definition of the syntax of BST.

Syntax of BST

The symbols of the language BST are:

- infinitely many set variables $x, y, z, \ldots$;
- the constant symbol '$\varnothing$';
- the set operator symbols ' $\setminus$ ', ' $\cup$ ', ' $\cap$ ';
- the set relation symbols ' $\subseteq$ ', ' $=$ ', ' $=\varnothing$', 'Disj';
- the logical connective symbols ' $\wedge$ ', '$\neg$'.

Define BST-terms as the smallest set of expression such that:

- all the set variables and the constant $\varnothing$ are BST-terms;
- if s and t are BST-terms, so are $s \setminus t$, $s \cup t$, and $s \cap t$;

hence if s and t are BST-terms the possible BST-atoms are:

$$s \subseteq t, \quad s = t, \quad s = \varnothing, \quad \text{and} \quad \mathsf{Disj}(s, t).$$

BST-literals are simply atoms, A, and their negations, $\neg A$. Finally BST-formulae are conjunctions of literals by means of the logical connective ' $\wedge$ '.

Usually we will forgo the symbol '$\neg$' and instead add the following relation symbols:

$$\text{' } \not\subseteq \text{ ', ' } \neq \text{ ', ' } \neq\varnothing\text{', '}\neg\mathsf{Disj}\text{'.}$$

For example, we write $s \neq t$ in place of $\neg(s = t)$.

Remark 3.1. *The name Boolean Set Theory is not incidental and its derived by the three operators it comprises,* '$\setminus$','$\cup$', *and* '$\cap$'. *Those operators are known as the Boolean set operators due to their likeness to the propositional connectives* '$\wedge \neg$','$\vee$', *and* '$\wedge$'.

Remark 3.2. *Defining the symbol admitted inside* BST*, we purposedly left out the logical connectives* '$\vee$', '$\rightarrow$', *and* '$\leftrightarrow$', *the rationale behind this decision lying on our goal of finding polynomial fragments. In fact, including any of the aforementioned logical connectives inside a fragment we can prove its* NP*-completeness regardless of the nature of the operator and propositional symbols of the fragment.*

The semantics for BST is defined exactly as for MLS therefore will not be discussed here.

In the above discussion we envisioned a study of a complete complexity taxonomy of the language BST and we proposed to store the result of such investigation into a table containing all the minimal NP-complete and maximal polynomial fragments. Table 3.1 is an example of how the table will look, each row represent a fragment and each column represent a symbol, if there is a '$\star$' inside a cell then the symbol represented by the column of the cell is present inside the fragment represented by the row of that cell. The last column contains the complexity of the fragment represented by each row. In Table 3.1 the only line represents the fragment $\mathsf{BST}(\setminus, \neq)$, which is indeed NP-complete, as we will show in Section 3.1.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg$Disj	$\subseteq$	$\not\subseteq$	$=$	$\neq$	Complexity
		$\star$								$\star$	NP-complete

TABLE 3.1: Early look at the table of containing the BST taxonomy

In the following sections we will study the fragments of BST in order to fill up the above table.

3.1 NP-Complete fragments of BST

In Chapter 1 we showed that the satisfiability problem for MLS belongs to the NP complexity class, as well the satisfiability problem for BST and for all the other fragments of MLS, in the same class. As anticipated, the satisfiability problem for BST, and then by extensions for MLS, belongs also to the class of NP-hard problems, pushing both problems in the class of the NP-complete problems. In the following section we will finally prove this fact.

Here we prove the NP-completeness of four BST fragments: $\mathsf{BST}(\setminus, \neq)$, $\mathsf{BST}(\cup, \cap, \neq)$, $\mathsf{BST}(\cup, \cap, =\varnothing, \neq\varnothing)$, and $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj}, =)$, by reducing the problem 3SAT

to their decision problem. Several other NP-complete fragments can be found inside BST, most of them are extensions of other NP-complete fragments, i.e., they are not minimal NP-complete fragments, while other can be proved to be NP-complete by slightly modifying the proof for one of the above fragments. In particular we have:

- the NP-completeness of the fragments $\mathsf{BST}(\setminus, \subseteq)$, $\mathsf{BST}(\setminus, \neg\mathsf{Disj})$, and $\mathsf{BST}(\setminus, \neq\varnothing)$ can be obtained much by the same technique used to reduce 3SAT to $\mathsf{BST}(\setminus, \neq)$;
- the proof of the NP-completeness of the fragment $\mathsf{BST}(\cup, \cap, \not\subseteq)$ can be achieved much by the same technique used to reduce 3SAT to $\mathsf{BST}(\cup, \cap, \neq)$;
- the NP-completeness of fragments $\mathsf{BST}(\cup, \cap, \mathsf{Disj}, \neg\mathsf{Disj})$, $\mathsf{BST}(\cup, \cap, =\varnothing, \neg\mathsf{Disj})$ and $\mathsf{BST}(\cup, \cap, \neq\varnothing, \mathsf{Disj})$ can be obtained by much the same technique used to reduce 3SAT to $\mathsf{BST}(\cup, \cap, =\varnothing, \neq\varnothing)$;
- finally the NP-completeness of $\mathsf{BST}(\cup, =, \neq\varnothing, \mathsf{Disj})$ can be shown much by the same approach used to reduce 3SAT to $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj}, =)$.

Moreover we can also apply some expressibility results shown in Chapter 2 to prove that:

- $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$ can be reduced in linear time
  - to $\mathsf{BST}(\cup, \subseteq, \mathsf{Disj}, \neg\mathsf{Disj})$, by Lemma 2.1(e),
- $\mathsf{BST}(\cup, =, \neq\varnothing, \mathsf{Disj})$ can be reduced in linear time
  - to $\mathsf{BST}(\cup, =, \neq, \mathsf{Disj})$, by Lemma 2.1(h),
  - to $\mathsf{BST}(\cup, =, \not\subseteq, \mathsf{Disj})$, by Lemma 2.1(h),
  - to $\mathsf{BST}(\cup, \subseteq, \neq\varnothing, \mathsf{Disj})$, by Lemma 2.1(e),
  - to $\mathsf{BST}(\cup, \subseteq, \neq, \mathsf{Disj})$, by Lemma 2.1(e)(h),
  - to $\mathsf{BST}(\cup, \subseteq, \not\subseteq, \mathsf{Disj})$, by Lemma 2.1(e)(h).

Before moving on to the actual NP-hardness proofs, we give the definition of the problem 3SAT, a well-known NP-complete problem.

Definition 3.1. *The problem 3SAT is the problem of establishing whether or not there exists a propositional evaluation $\mathfrak{v}$ over the variables of any given 3SAT-formula F, that makes the formula F true.*

A 3SAT-formula F is any formula of the form:

$$F := \bigwedge_{i=1}^{m} (L_{i1} \vee L_{i2} \vee L_{i3}), \tag{3.1}$$

where the L_{ij}'s are propositional literals, therefore L_{ij} is either a propositional variable P or its negation $\neg P$.

A propositional valuation $\mathfrak{v}$ is a map that assigns to any propositional variable P a truth value, true or false.

For simplicity of notation, we refer to each disjunction $(L_{i1} \vee L_{i2} \vee L_{i3})$ in (3.1) as C_i.

Before proceeding, we need one last technical lemma that is necessary for our NP-hardness proofs.

At the beginning of this section we showed that any BST-formula can be reduced to a conjunction of literals of types:

$$x = y \setminus z, \text{ or } x \neq y.$$

We will refer to literals of type $x = y \setminus z$ as *positive literals* and to literals of type $x \neq y$ as *negative literals*.

Lemma 3.3. *If φ is a satisfiable* BST*-formula that comprises exactly one negative literal then φ admits a model M such that $Mv \in \{\sigma, \emptyset\}$, for some set nonempty set σ and for all $v \in Vars(\varphi)$.*

Proof. Let M' be a model for φ, and let $x' \neq y'$ be the negative literal in φ, so that plainly $M'x' \neq M'y'$. W.l.o.g., we may assume that $M'x' \nsubseteq M'y'$, so that there exists a set s in $M'x' \setminus M'y'$. Let us define the set assignment M by putting

$$Mv \coloneqq \begin{cases} \{s\} & \text{if } s \in M'v \\ \emptyset & \text{otherwise,} \end{cases}$$

for all $v \in Vars(\varphi)$.

Trivially, we have $Mx' = \{s\}$ and $My' = \emptyset$, so that $M \models x' \neq y'$. Hence, we are left to prove that $M \models x = y \setminus z$, for each positive literal of φ.

By definition of M, for all $t \in Mx$, we have $t = s$ and $s \in M'x$. Since $M' \models \varphi$ then $s \in M'y \setminus M'z$ thus $My = \{s\}$ and $Mz = \emptyset$ thus $My \setminus Mz = \{s\}$, proving that $Mx \subseteq My \setminus Mz$.

On the other hand, for all $t \in My \setminus Mz$, by definition of M, $t = s$ and $My \setminus Mz = \{s\}$. We must have $My = \{s\}$ and $Mz = \emptyset$ thus $s \in M'y \setminus M'z = M'x$ then $Mx = \{s\}$, proving that $My \setminus Mz \subseteq Mx$. Combining the two last result we have $Mx = My \setminus Mz$ for all positive literals of φ, therefore $M \models \varphi$.

By putting $\sigma = \{s\}$ we obtain our thesis. □

All of the proofs of NP-hardness for fragments of BST will require the above lemma. As an intuition we can notice that using the set assignment generated by Lemma 3.3 we can easily associate the truth value true to the set σ and false to the empty set.

3.1.1 BST$(\setminus, \neq)$

The first fragment we analyze is BST$(\setminus, \neq)$, which is defined as:

Definition 3.2. *The fragment* $\mathsf{BST}(\setminus, \neq)$ *consists of all conjunctions of literals of the following type*

$$t_1 \neq t_2,$$

where t_1, t_2 *are terms involving only set variables and the set operator* ' $\setminus$ '.

In order to reduce 3SAT to $\mathsf{BST}(\setminus, \neq)$, let F be a generic 3SAT formula, and let $P_1, \ldots, P_n$ be its distinct propositional variables. Put $X_1, \ldots, X_n, \mathcal{X}$ as $n+1$ distinct set variables. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, we put:

$$T_{ij} := \begin{cases} X_k & \text{if } L_{ij} = P_k \text{ for some } k \\ \mathcal{X} \setminus X_k & \text{if } L_{ij} = \neg P_k \text{ for some } k. \end{cases}$$

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

$$\mathcal{C}_i := \big((\mathcal{X} \setminus T_{i1}) \setminus T_{i2}\big) \setminus T_{i3}.$$

Finally, let:

$$\Phi_F := (\cdots(((\mathcal{X} \setminus \mathcal{C}_1) \setminus \mathcal{C}_2) \setminus \mathcal{C}_3) \setminus \cdots \setminus \mathcal{C}_{m-1}) \setminus \mathcal{C}_m \neq \mathcal{X} \setminus \mathcal{X}. \tag{3.2}$$

We claim that:

Lemma 3.4. *Any 3-SAT instance* F *is propositionally satisfiable if and only if the* $\mathsf{BST}(\setminus, \neq)$*-formula* Φ_F *obtained from* F *using the above construction is satisfiable by a set assignment.*

Proof. (Necessity). To begin with, let us assume that F is propositionally satisfiable, and let $\mathfrak{v}$ be a Boolean valuation satisfying it. Consider the following set assignment:

$$M\mathcal{X} := b,$$
$$MX_k := \begin{cases} b & if\ \mathfrak{v}(P_k) = \mathsf{true} \\ \emptyset & if\ \mathfrak{v}(P_k) = \mathsf{false}, \end{cases}$$

for $k = 1, \ldots, n$, where b is any nonempty set. The set assignment M satisfies Φ_F. Indeed, since $\mathfrak{v}$ satisfies F, then $\mathfrak{v}(C_i) = \mathsf{true}$, for each $i = 1, \ldots, m$. Hence, for each $i = 1, \ldots, m$ there exists a $j = 1, 2, 3$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$. If $L_{ij} = P_k$, for some k, we have that $MT_{ij} = MX_k = b$, while if $L_{ij} = \neg P_k$, for some k, we have that $MT_{ij} = M\mathcal{X} \setminus MX_k = b \setminus \emptyset = b$. Hence, for each $i = 1, \ldots, m$, $M\mathcal{C}_i = \emptyset$, and therefore:

$$\begin{aligned} M\Phi_F &= \big((\cdots(((M\mathcal{X} \setminus M\mathcal{C}_1) \setminus M\mathcal{C}_2) \setminus M\mathcal{C}_3) \setminus \cdots \setminus M\mathcal{C}_{m-1}) \setminus M\mathcal{C}_m \neq M\mathcal{X} \setminus M\mathcal{X}\big) \\ &= \big((\cdots(((b \setminus \emptyset) \setminus \emptyset) \setminus \emptyset) \setminus \cdots \setminus \emptyset) \setminus \emptyset \neq b \setminus b\big) \\ &= \big(b \neq \emptyset\big) \\ &= \mathsf{true}. \end{aligned}$$

In conclusion, we have $M \models \Phi_F$, so if F is propositionally satisfiable, then Φ_F is satisfied by a set assignment.

(Sufficiency). Now assume Φ_F is satisfied by the set assignment M. By Lemma 3.3, since Φ_F has only one negative literal, we can pick M such that $Mv \in \{\emptyset, \sigma\}$, for every $v \in \mathit{Vars}(\Phi_F)$, and some nonempty set σ. We prove that the Boolean valuation $\mathfrak{v}$ defined by

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

for $k = 1, \ldots, n$, propositionally satisfies F. Since $M \models \Phi_F$, for each $i = 1, \ldots, m$ we have $M\mathcal{C}_i = \emptyset$. Hence there exists a $j = 1, 2, 3$ such that $MT_{ij} \neq \emptyset$, if $L_{ij} = P_k$ then $MT_{ij} = MX_k \neq \emptyset$ so that $\mathfrak{v}(L_{ij}) = \mathsf{true}$, while if $L_{ij} = \neg P_k$ then $MT_{ij} = M(\mathcal{X} \setminus X_k) \neq \emptyset$ so $\mathfrak{v}(L_{ij}) = \mathsf{true}$. Then for each $i = 1, \ldots, m$ there exists a $j = 1, 2, 3$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$ so that $\mathfrak{v}(C_i) = \mathsf{true}$, hence $\mathfrak{v}$ propositionally satisfies the 3SAT instance F. □

It is quite easy to show that the above construction can be used to transform any 3SAT instance F in the $\mathsf{BST}(\setminus, \neq)$-formula Φ_F in $\mathcal{O}(|F|)$-time. Hence the 3SAT problem can be reduced in linear-time to the satisfiability problem for $\mathsf{BST}(\setminus, \neq)$.

We have proved that the decision problem for $\mathsf{BST}(\setminus, \neq)$ belongs to the class NP-hard. In addition, since $\mathsf{BST}(\setminus, \neq)$ is a fragment of BST, it also belongs to the class NP. Therefore, we can conclude:

Lemma 3.5. *The satisfiability problem for* $\mathsf{BST}(\setminus, \neq)$ *belongs to the* NP*-complete complexity class.*

3.1.2 $\mathsf{BST}(\cup, \cap, \neq)$

Next is the fragment $\mathsf{BST}(\cup, \cap, \neq)$ defined as:

Definition 3.3. *The fragment* $\mathsf{BST}(\cup, \cap, \neq)$ *consists of all conjunctions of literals of type*

$$t_1 \neq t_2,$$

where t_1, t_2 *are terms involving the set operators* ' $\cup$ ' *and* ' $\cap$ ' *alongside set variables.*

Again, we reduce 3SAT to $\mathsf{BST}(\cup, \cap, \neq)$. So, let $P_1, \ldots, P_n$ be the distinct propositional variables in a generic 3SAT instance F and let $X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n$ be $2n$ pairwise distinct set variables. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, set

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

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

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

and finally put

$$\Phi_F := T \cap X \neq T \cap X \cap Y. \tag{3.3}$$

Lemma 3.6. *Any 3SAT instance F is propositionally satisfiable if and only if so is the $\mathsf{BST}(\cup, \cap, \neq)$-formula Φ_F obtained from F according to the above construction.*

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

$$\begin{aligned} MX_k := b \quad and \quad M\overline{X}_k := \emptyset \quad if\ \mathfrak{v}(P_k) = \mathsf{true} \\ MX_k := \emptyset \quad and \quad M\overline{X}_k := b \quad if\ \mathfrak{v}(P_k) = \mathsf{false}, \end{aligned}$$

where b is a nonempty set satisfies Φ_F. Notice that, regardless of the value of $\mathfrak{v}(P_i)$, we have $(MX_i \cup M\overline{X}_i) = b$ and $(MX_i \cap M\overline{X}_i) = \emptyset$, so that $MX = b$ and $MY = \emptyset$. Hence to prove that $M \models \Phi_F$, we just need to prove that $MT = b$. First, we notice that $Mx \in \{b, \emptyset\}$, for each set variable x. Since $\mathfrak{v}$ satisfies F, for each $i = 1, \ldots, m$, we have $\mathfrak{v}(C_i) = \mathsf{true}$. Hence, for each $i = 1, \ldots, m$, there exists a $j = 1, 2, 3$ such that . So, if $L_{ij} = P_k$, for some k, then $Mu_{ij} = MX_k = b$, while if $L_{ij} = \neg P_k$, for some k, then $Mu_{ij} = M\overline{X}_k = b$. Hence $Mt_i = b$, for $i = 1, \ldots, m$, and also $MT = b$ holds.

(Sufficiency). Concerning the converse, assume that Φ_F is satisfiable. Since there is only one negative literal in Φ_F, by Lemma 3.3 there exists a set assignment M that satisfies Φ_F and is such that $Mv \in \{\sigma, \emptyset\}$, for every set variables $v \in \mathit{Vars}(\Phi_F)$. Since M satisfies Φ_F, plainly $MX \neq \emptyset$ and $MY = \emptyset$, so that $(MX_1 \cup M\overline{X}_1) \cap \ldots \cap (MX_n \cup M\overline{X}_n) \neq \emptyset$ and also $(MX_1 \cap M\overline{X}_1) \cup \ldots \cup (MX_n \cap M\overline{X}_n) = \emptyset$. Hence, for every $i = 1, \ldots, n$, we have $(MX_i \cup M\overline{X}_i) \neq \emptyset$ and $(MX_i \cap M\overline{X}_i) = \emptyset$, from which it follows that

$$\text{if } MX_i \neq \emptyset \text{ then } M\overline{X}_i = \emptyset, \tag{3.4}$$

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

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

We only need to show that, for each $i = 1, \ldots, m$, there exists a $j = 1, 2, 3$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$. We know that, for $i = 1, \ldots, m$, there exists a $\overline{j} = 1, 2, 3$ such that $Mu_{i\overline{j}} \neq \emptyset$, if $L_{i\overline{j}} = P_k$ for some k. Then $u_{i\overline{j}} = X_k$, so that $MX_k \neq \emptyset$. Hence $\mathfrak{v}(L_{i\overline{j}}) = \mathsf{true}$, while if $L_{i\overline{j}} = \neg P_k$ for some k, we have $Mu_{i\overline{j}} = M\overline{X}_k \neq \emptyset$ so that, by (3.4), $MX_k = \emptyset$, $\mathfrak{v}(P_k) = \mathsf{false}$, and $\mathfrak{v}(L_{i\overline{j}}) = \mathsf{true}$. Thus $\mathfrak{v}$ satisfies F, concluding our proof. □

Since the above construction can be used to reduce in $\mathcal{O}(|F|)$-time any 3SAT instance F into the $\mathsf{BST}(\cup, \cap, \neq)$ formula Φ_F, we can then conclude that:

Lemma 3.7. *The satisfiability problem of* $\mathsf{BST}(\cup, \cap, \neq)$ *belongs to the* NP*-complete complexity class.*

3.1.3 $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$

Next in our analysis of the NP-complete fragments of BST we consider the fragment $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj}, =)$.

Definition 3.4. *The fragment* $\mathsf{BST}(\cup, =, \mathsf{Disj}, \neg\mathsf{Disj})$ *of* BST *consists of all conjunctions of literals of the forms*

$$t_1 = t_2, \ \mathsf{Disj}\,(t_3, t_4)\,, \ \neg\mathsf{Disj}\,(t_5, t_6)\,,$$

where t_1, t_2, t_3, t_4, t_5, *and* t_6 *are terms only comprising set variables and the set operator* $\cup$.

Given any 3SAT instance F let $P_1, \ldots, P_n$ be the distinct propositional letters found occurring in F ; associate with them $2n+1$ pairwise distinct set variables $X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n$, plus a set variable $\mathcal{X}$. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, define

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

and put

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

Finally, put

$$\Phi_F \coloneqq \bigwedge_{i=1}^{m} \mathcal{C}_i \ \wedge \ \bigwedge_{k=1}^{n} \Big(\mathsf{Disj}\,\big(X_k, \overline{X}_k\big) \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X} \Big) \ \wedge \ \neg\mathsf{Disj}\,(\mathcal{X}, \mathcal{X})\,. \quad (3.5)$$

Lemma 3.8. *Any 3SAT-instance* F *is propositionally satisfiable if and only if the* $\mathsf{BST}(\cup, =\,, \mathsf{Disj}, \neg\mathsf{Disj})$*-formula* Φ_F *defined as above is satisfied by some set assignment.*

Proof. (Sufficiency). To prove sufficiency, suppose that Φ_F is satisfiable. Since $\neg\mathsf{Disj}\,(\mathcal{X}, \mathcal{X})$ is the only negative constraint in Φ_F, according to Lemma 3.3 there exists a model M of Φ_F such that

$$MX_k, M\overline{X}_k \in \{\emptyset, \sigma\} \quad (3.6)$$

holds for each $k = 1, \ldots, n$, where σ is a fixed nonempty set.

Since $M \models \Phi_F$, we have

$$\begin{array}{ll} M \models \neg\mathsf{Disj}\,(\mathcal{X}, \mathcal{X}) & \therefore \\ M \models \mathcal{X} \cap \mathcal{X} \neq \varnothing & \therefore \\ M \models \mathcal{X} \neq \varnothing & \therefore \\ M\mathcal{X} \neq \emptyset, & \end{array} \tag{3.7}$$

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

$$\begin{array}{lll} M \models \mathsf{Disj}\left(X_k, \overline{X}_k\right) \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X} & & \therefore \\ M \models X_k \cap \overline{X}_k = \varnothing \ \wedge \ X_k \cup \overline{X}_k = \mathcal{X} & & \therefore \\ MX_k \cap M\overline{X}_k = \emptyset \ \wedge \ MX_k \cup M\overline{X}_k \neq \emptyset & \text{(by (3.7)).} & \end{array} \tag{3.8}$$

By combining (3.6) and (3.8), we obtain:

$$\left(\, MX_k = \sigma \ \wedge \ M\overline{X}_k = \emptyset \,\right) \quad \vee \quad \left(\, MX_k = \emptyset \ \wedge \ M\overline{X}_k = \sigma \,\right). \tag{3.9}$$

Now consider the following truth-value assignment:

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

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

$$\begin{array}{ll} M\left(T_{i1} \cup T_{i2} \cup T_{i3}\right) = M\mathcal{X} & \therefore \\ MT_{i1} \cup MT_{i2} \cup MT_{i3} \neq \emptyset & \text{(by (3.7)).} \end{array}$$

Thus, for each $k = 1, \ldots, n$, there exists a $j = 1, 2, 3$, such that $MT_{ij} \neq \emptyset$. There are only two cases to be examined: $T_{ij} = X_k$ and $T_{ij} = \overline{X}_k$. In the former case, we have $L_{ij} = P_k$ for some k, and also $MX_k = MT_{ij} \neq \emptyset$; hence $\mathfrak{v}(L_{ij}) = \mathfrak{v}(P_k) = \mathsf{true}$, and thus $\mathfrak{v}(C_i) = \mathsf{true}$. In the latter case, we have $L_{ij} = \neg P_k$ for some k, and also $M\overline{X}_k = MT_{ij} \neq \emptyset$; hence, by (3.9), $MX_k = \emptyset$, and $\mathfrak{v}(\neg P_k) = \mathfrak{v}(L_{ij}) = \mathsf{true}$, and thus $\mathfrak{v}(C_i) = \mathsf{true}$. We conclude that the truth-value assignment $\mathfrak{v}$ satisfies the instance F of 3SAT; hence F is propositionally satisfiable, in consequence of Φ_F being satisfied by a set assignment.

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

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

where b is a non-empty set.

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

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

and

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

so that

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

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

After noticing that any 3SAT instance F can be reduced to the conjunction Φ_F using the above construction in $\mathcal{O}(|F|)$-time, the lemma just seen readily yields that:

Lemma 3.9. *The satisfiability problem for* $\mathsf{BST}(\cup,=,\mathsf{Disj},\neg\mathsf{Disj})$ *belongs to the complexity class of* NP*-complete problems.*

3.1.4 $\mathsf{BST}(\cup,\cap,=\!\varnothing,\neq\!\varnothing)$

The last fragment of BST we prove to be NP-complete is $\mathsf{BST}(\cup,\cap,=\!\varnothing,\neq\!\varnothing)$.

Definition 3.5. *The fragment* $\mathsf{BST}(\cup,\cap,=\!\varnothing,\neq\!\varnothing)$ *consists of all conjunctions of literals of the forms*

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

where t_1, t_2 *stand for any terms involving the set operators* $\cup$ *and* $\cap$*, in addition to set variables.*

Reducing the 3SAT problem, associate to each propositional letter P_k inside an instance F of the 3SAT problem two set variables X_k and $\overline{X}_k$, then build the following $\mathsf{BST}(\cup,\cap,=\!\varnothing,\neq\!\varnothing)$-formula:

$$\Phi_F \coloneqq \bigwedge_{i=1}^{m}(T_{i1} \cap T_{i2} \cap T_{i3} = \varnothing) \ \wedge \ \bigwedge_{k=1}^{n}(X_k \cap \overline{X}_k = \varnothing) \ \wedge \ \mathcal{C}, \tag{3.10}$$

where

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

and

$$\mathcal{C} \coloneqq (X_1 \cup \overline{X}_1) \cap \cdots \cap (X_n \cup \overline{X}_n) \neq \varnothing.$$

Intuitively speaking, here we are representing true by the set constant $\varnothing$; thus, e.g., $T_{i1} \cap T_{i2} \cap T_{i3} = \varnothing$ means that at least one of T_{i1}, T_{i2}, T_{i3} must be 'true'.

Lemma 3.10. *Any instance F of the 3SAT problem is propositionally satisfiable if and only if the $\mathsf{BST}(\cup, \cap, =\varnothing, \neq\varnothing)$-formula Φ_F resulting from the above construction is satisfied by some set assignment.*

Proof. (Satisfiability). First suppose that Φ_F is satisfiable. Since $\mathcal{C}$ is the only negative constraint in Φ_F, following Lemma 3.3 there exists a model M of Φ_F sending each $v \in \mathit{Vars}(\Phi_F)$ to $Mv \in \{\emptyset, \sigma\}$, where σ is a fixed non-empty set. Since $M \models \mathcal{C}$ and $M \models X_k \cap \overline{X}_k = \emptyset$, *either*

$$MX_k \neq \emptyset \text{ and } M\overline{X}_k = \emptyset$$
$$\textit{or}$$
$$MX_k = \emptyset \text{ and } M\overline{X}_k \neq \emptyset$$

must hold for each $k = 1, \ldots, n$.

Now we prove that the truth-value assignment

$$\mathfrak{v}(P_k) \coloneqq \begin{cases} \mathsf{true} & \text{if } MX_k = \emptyset \\ \mathsf{false} & \text{if } M\overline{X}_k = \emptyset \end{cases}$$

satisfies F. Since $M \models \Phi_F$, for each $i = 1, \ldots, m$ there exists a $j \in \{1, 2, 3\}$ such that $MT_{ij} = \emptyset$. If T_{ij} coincides with X_k for some k, then $L_{ij} = P_k$ and $MX_k = \emptyset$; hence $\mathfrak{v}(L_{ij}) = \mathfrak{v}(P_k) = \mathsf{true}$. On the other hand, if $T_{ij} = \overline{X}_k$ for some k, then $L_{ij} = \neg P_k$ and $M\overline{X}_k = \emptyset$; hence $\mathfrak{v}(L_{ij}) = \neg\mathfrak{v}(P_k) = \mathsf{true}$. We therefore conclude that $\mathfrak{v}$ satisfies F.

(Necessity). For the converse implication, suppose that F is propositionally satisfiable. Consider a truth-value assignment $\mathfrak{v}$ satisfying F, arbitrarily pick a set $b \neq \emptyset$, and then put

$$MX_k \coloneqq \begin{cases} b & \text{if } \mathfrak{v}(P_k) = \mathsf{false} \\ \emptyset & \text{if } \mathfrak{v}(P_k) = \mathsf{true}, \end{cases} \qquad M\overline{X}_k \coloneqq \begin{cases} \emptyset & \text{if } \mathfrak{v}(P_k) = \mathsf{false} \\ b & \text{if } \mathfrak{v}(P_k) = \mathsf{true}, \end{cases}$$

for each $k = 1, \ldots, n$. Notice that $MX_k \cap M\overline{X}_k = \emptyset$ and $MX_k \cup M\overline{X}_k = b \neq \emptyset$ hold for each k; therefore $M \models \mathcal{C}$ and $M \models X_k \cap \overline{X}_k = \varnothing$.

Moreover, for each $i = 1, \ldots, m$ there exists a $j \in \{1, 2, 3\}$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$. If $L_{ij} = P_k$ for some k, then $T_{ij} = X_k$ and $\mathfrak{v}(P_k) = \mathsf{true}$, hence $T_{ij} = \emptyset$;

if $L_{ij} = \neg P_k$ for some k, then $T_{ij} = \overline{X}_k$, and $\mathfrak{v}(P_k) = \mathsf{false}$, hence $T_{ij} = \emptyset$. We therefore conclude that $M \models \Phi_F$. □

The lemma just seen, and the fact that the above construction yields the conjunction Φ_F in $\mathcal{O}(|F|)$-time, readily yield that

Lemma 3.11. *The satisfiability problem for* BST$(\cup, \cap, =\varnothing, \neq\varnothing)$ *belongs to the complexity class* NP*-complete.*

3.1.5 Summary

In the above discussion we presented four BST fragments whose decision problems belong to the class NP-complete, and also how these results extend to fourteen other fragments. Plainly, any of the above proof can be used also to prove the NP-completeness of both BST and MLS, since the formulae used are naturally also BST, hence MLS, formulae.

Lemma 3.12. *The decision problem for the theory* MLS*, and its fragment* BST*, belongs to class of problems* NP*-complete.*

Gathering the results presented in this section we can begin to compile the table representing the complete complexity taxonomy of BST. The fragments presented in Table 3.2 comprises all the minimal NP-complete fragments of BST, no proof is given to prove that all the sub-theories of this fragments are polynomial however this will become evident after we present the polynomial fragments of BST.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg$Disj	$\subseteq$	$\not\subseteq$	$=$	$\neq$	Complexity
		★								★	NP-complete
		★						★			NP-complete
		★				★					NP-complete
		★		★							NP-complete
★	★									★	NP-complete
★	★							★			NP-complete
★	★		★	★							NP-complete
★	★				★	★					NP-complete
★	★		★			★					NP-complete
★	★			★	★						NP-complete
★					★	★			★		NP-complete
★				★	★				★		NP-complete
★					★	★	★				NP-complete
★					★				★	★	NP-complete
★					★			★	★		NP-complete
★				★	★		★				NP-complete
★					★		★			★	NP-complete
★					★		★	★			NP-complete

Table 3.2: Minimal NP-complete fragments of BST

3.2 The fragments $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cup, =, \neq)$

The previous sections focused on NP-complete fragments, here we will study three polynomial fragments, $\mathsf{BST}(\cup, =, \neq)$, $\mathsf{BST}(\cap, =, \neq)$ and $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$, at the end of the chapter will be clear that this fragments, listed in Table 3.3 are enough to represent all the polynomial fragments of BST, and in turn make also clear that Table 3.2 lists the entirety of minimal NP-complete fragment of the language. In fact, it is easy to check that removing any symbol from a fragment inside Table 3.2 would lead to a sub-language of a maximal polynomial fragment.

We begin with $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$, two fragments that are very close to each other so that in the following we will present a single cubic-time algorithm that can be used to solve the decision problem for both fragments (the results regarding these two fragments can also be found in [CMO21]).

Last we will show that using the expressibility results presented in Chapter 2, it is possible to extend $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$ to two broader fragments, $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ and $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ respectively, which are polynomial maximal inside BST.

Definition 3.6. *The fragment* $\mathsf{BST}(\cup, =, \neq)$ *is the collection of all the conjunctions of literals of types:*

$$\bigcup L = \bigcup R, \quad \bigcup L \neq \bigcup R, \tag{3.11}$$

where L and R are nonempty collections of set variables.

Definition 3.7. *The fragment* $\mathsf{BST}(\cap, =, \neq)$ *is the collection of all the conjunctions of literals of types:*

$$\bigcap L = \bigcap R, \quad \bigcap L \neq \bigcap R, \tag{3.12}$$

where L and R are nonempty collections of set variables.

We recall that given a set assignment M and a collection of set variables V, we follow the standard interpretation

$$MV := \{Mv \mid v \in V\}$$

that in presence of the unary operator $\bigcup$, and $\bigcap$ directly translates into

$$M \bigcup V = \bigcup MV = \bigcup_{v \in V} Mv, \quad M \bigcap V = \bigcap MV = \bigcap_{v \in V} Mv.$$

Throughout this section, since several of the properties and lemmas we are about to present hold true for both fragments $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$, we will use the symbol '$\star$' as a placeholder for either '$\cup$' or '$\cap$', therefore we can refer to both fragments simultaneously $\mathsf{BST}(\star, =, \neq)$. Following the above notation we will have:

$$\bigstar\{x_1,\ldots,x_k\} \;\coloneqq\; x_1\star\cdots\star x_k\,,$$
$$\bigstar\{s_1,\ldots,s_k\} \;\coloneqq\; s_1\star\cdots\star s_k\,,$$

where $x_1,\ldots,x_k$ and $s_1,\ldots,s_k$ denote variables and sets, respectively.

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

Definition 3.8. *Any formula φ of $\mathsf{BST}(\star,=,\neq)$ can be represented by two sets of pairs:*

$$\Phi^{\oplus}_{\varphi} \coloneqq \Big\{\big\{\{x_1,\cdots,x_h\},\{y_1,\cdots,y_k\}\big\} \;/\; x_1\star\cdots\star x_h = y_1\star\cdots\star y_k \text{ is in } \varphi\Big\},$$
$$\Phi^{\ominus}_{\varphi} \coloneqq \Big\{\big\{\{u_1,\cdots,u_m\},\{v_1,\cdots,v_p\}\big\} \;/\; u_1\star\cdots\star u_m \neq v_1\star\cdots\star v_p \text{ is in } \varphi\Big\}.$$

Notice that by representing φ as the pair $\Phi^{\oplus}_{\varphi}$ and $\Phi^{\ominus}_{\varphi}$ two relations between collection of set variables are created. A core component of the satisfiability test we designed is a closure operator built around a certain relation between collection of set variables. Our algorithm will take as input the set $\Phi^{\oplus}_{\varphi}$ in order to build such relation, then use $\Phi^{\ominus}_{\varphi}$ to establish the satisfiability of φ.

Definition 3.9 (Equivalence relation $\sim_\varphi$)**.** *Let $\sim_\varphi$ be the smallest equivalence relation over $\mathscr{P}^+(\mathit{Vars}(\varphi))$ that satisfies the following closure conditions:*

(Cl1) $\Phi^{\oplus}_{\varphi} \subseteq \sim$,

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

When we say $\sim_\varphi$ is the smallest relation we mean that any relation $\sim$ that satisfies both (Cl1) and (Cl2) is included in $\sim_\varphi$ set-theoretically.

Definition 3.10 ($\sim_\varphi$-closure)**.** *The $\sim_\varphi$-closure $\overline{Z}$ of a set $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$ is the collection:*

$$\overline{Z} \coloneqq \bigcup\{W \;/\; W \sim_\varphi Z\}.$$

We will prove that the $\sim_\varphi$-closure $\overline{Z}$ of Z is the largest set of $\mathscr{P}^+(\mathit{Vars}(\varphi))$ that is $\sim_\varphi$-equivalent to Z, then we also provide a quadratic-time algorithm to compute the $\sim_\varphi$ closure of a set and exploit this closure operator to quickly check whether any two sets of variables are $\sim_\varphi$- equivalent, in fact, as we will show, two sets Z_1 and Z_2 are $\sim_\varphi$-equivalent if and only if $\overline{Z}_1 = \overline{Z}_2$.

Lemma 3.13. *Let $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$. Then the closure $\overline{Z}$ of Z, namely the set $\bigcup\{W \;/\; W \sim_\varphi Z\}$, is the largest subset of $\mathit{Vars}(\varphi)$ 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

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

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

Before moving on to the main lemmas upon which by our algorithm is based, we first present some useful properties of the $\sim_\varphi$-closure and the $\sim_\varphi$-equivalence.

Lemma 3.14. *Let $Z, Z_1, Z_2 \in \mathscr{P}^+(\mathit{Vars}(\varphi))$. Then*

(a) $Z \subseteq \overline{Z}$ *and* $Z \sim_\varphi \overline{Z}$;

(b) $\overline{Z} = \overline{\overline{Z}}$;

(c) *if* $Z_1 \sim_\varphi \overline{Z}_2$, *then* $Z_1 \subseteq \overline{Z}_2$;

(d) $Z_1 \sim_\varphi Z_2$ *if and only if* $\overline{Z}_1 = \overline{Z}_2$;

(e) *if* $Z_1 \subseteq Z_2$, *then* $\overline{Z}_1 \subseteq \overline{Z}_2$;

(f) $Z_1 \subseteq \overline{Z}_2$ *if and only if* $\overline{Z}_1 \subseteq \overline{Z}_2$;

(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 3.13.

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

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

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

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

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

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

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

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. □

We are now ready to prove that the equivalence relation $\sim_\varphi$ captures which equalities among terms must be true in every model, if any, of a given formula φ.

Lemma 3.15. *Let φ be any formula in $\mathsf{BST}(\star, =, \neq)$, and let M be any set assignment over $\mathit{Vars}(\varphi)$ satisfying φ. Then*

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

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

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

$$Z_1 \sim_M Z_2 \iff \bigstar M Z_1 = \bigstar M Z_2$$

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

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

As for (Cl2), let $A \sim_M B$ and $C \subseteq \mathit{Vars}(\varphi)$. 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 3.2.3 for computing $\sim_\varphi$-closures.

Lemma 3.16. *Given a $\mathsf{BST}(\star,=,\neq)$-formula φ, let $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$ be such that*

$$L \subseteq Z \iff R \subseteq Z\,, \qquad \text{for every } \{L, R\} \in \Phi^{\oplus}_{\varphi}. \tag{3.13}$$

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

$$W_1 \subseteq Z \iff W_2 \subseteq Z\,. \tag{3.14}$$

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

$$W_1 \sim_Z W_2 \iff \big(W_1 \subseteq Z \iff W_2 \subseteq Z\big) \tag{3.15}$$

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

As for (Cl1), just from the hypothesis it follows that $L \sim_Z R$, for every $\{L, R\} \in \Phi^{\oplus}_{\varphi}$. Concerning (Cl2), let $A \sim_Z B$ and $C \subseteq \mathit{Vars}(\varphi)$, and assume that $A \cup C \subseteq Z$. Then, $A \subseteq Z$ and $C \subseteq Z$, so that by (3.15) 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 (3.15), 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 minimality of $\sim_\varphi$, we have $\sim_\varphi \subseteq \sim_Z$. Therefore, if $W_1 \sim_\varphi W_2$, then $W_1 \sim_Z W_2$, which by (3.15) implies $W_1 \subseteq Z \iff W_2 \subseteq Z$. □

3.2.2 Satisfiability in $\mathsf{BST}(\cup,=,\neq)$ and in $\mathsf{BST}(\cap,=,\neq)$

Below we will present a necessary condition that also suffices to ensure that a given formula in either $\mathsf{BST}(\cup,=,\neq)$ or $\mathsf{BST}(\cap,=,\neq)$ 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 3.17. *Let φ be a $\mathsf{BST}(\cup,=,\neq)$-formula. Then φ is satisfiable if and only if $L \not\sim_\varphi R$ (namely $\overline{L} \neq \overline{R}$) holds for every pair $\{L,R\} \in \Phi_\varphi^\ominus$.*

Proof. (Necessity.) Let M be a model for φ. By way of contradiction, assume that there exists a pair $\{L,R\} \in \Phi_\varphi^\ominus$ such that $L \sim_\varphi R$. Then by Lemma 3.15 we would have $\bigcup ML = \bigcup MR$, a contradiction, since by definition of $\Phi_\varphi^\ominus$, $\bigcup L \neq \bigcap R$ is a literal of φ. Therefore for all pairs $\{L,R\} \in \Phi_\varphi^\ominus$ we must have $L \not\sim_\varphi R$, completing the necessity part of the proof.

(Sufficiency.) Next, let us assume that, for each $\{L,R\} \in \Phi_\varphi^\ominus$, we have $L \not\sim_\varphi R$. We will construct a set assignment M that satisfies φ.

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

$$Mx \coloneqq \{b_{\overline{V}} \mid x \notin \overline{V} \text{ and } V \in \bigcup\Phi_\varphi^\ominus\},$$

so that we have

$$\bigcup MU \coloneqq \{b_{\overline{V}} \mid U \not\subseteq \overline{V} \text{ and } V \in \bigcup\Phi_\varphi^\ominus\}, \tag{3.16}$$

for every $U \subseteq \mathit{Vars}(\varphi)$.

We first show that $\bigcup ML = \bigcup MR$ holds whenever $\{L,R\} \in \Phi_\varphi^\oplus$. Thus, let $\{L,R\} \in \Phi_\varphi^\oplus$ and let $b_{\overline{V}} \in \bigcup ML$, for some $V \in \Phi_\varphi^\ominus$ such that $L \not\subseteq \overline{V}$. Then $\overline{L} \not\subseteq \overline{V}$, by Lemma 3.14(f). Since $\{L,R\} \in \Phi_\varphi^\oplus$, then by (Cl1) we have $L \sim_\varphi R$, so that, by (d) and (f) of Lemma 3.14, $R \not\subseteq \overline{V}$ follows. Hence $b_{\overline{V}} \in \bigcup MR$, and by the arbitrariness of $b_{\overline{V}}$ we have $\bigcup ML \subseteq \bigcup MR$.

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 $\{L,R\} \in \Phi_\varphi^\ominus$. Thus, let $\{L,R\} \in \Phi_\varphi^\ominus$, so that by our assumption we have $L \not\sim_\varphi R$. Hence, by Lemma 3.14(d), $\overline{L} \neq \overline{R}$. W.l.o.g., let us assume that $\overline{L} \not\subseteq \overline{R}$. Since $R \subseteq \overline{R}$ (by Lemma 3.14(a)) and plainly $R \in \bigcup\Phi_\varphi^\ominus$, then $b_{\overline{R}} \notin \bigcup MR$, by (3.16). On the other hand, by Lemma 3.14(f), $L \not\subseteq \overline{R}$, hence $b_{\overline{R}} \in \bigcup ML$, again by (3.16), and therefore $\bigcup ML \neq \bigcup MR$, concluding the proof of the theorem. □

In a dual manner, we prove that the same condition can be used to check the satisfiability of $\mathsf{BST}(\cap,=,\neq)$-formulae.

Theorem 3.18. *Let φ be a $\mathsf{BST}(\cap,=,\neq)$-formula. Then φ is satisfiable if and only if $L \not\sim_\varphi R$ (namely $\overline{L} \neq \overline{R}$) holds for every pair $\{L,R\} \in \Phi_\varphi^\ominus$.*

Proof. (Necessity.) Analogously to Theorem 3.17, the necessary part of this theorem is a direct consequence of Lemma 3.15.

(Sufficiency.) Next, let us assume that we have $L \not\sim_\varphi R$ for each $\{L, R\} \in \Phi^{\ominus}_{\varphi}$. We will construct a model M for φ.

Let us assign again a nonempty set $b_{\overline{V}}$ to each $\overline{V}$ such that $V \in \bigcup \Phi^{\ominus}_{\varphi}$ in such a way that the $b_{\overline{V}}$'s are pairwise distinct. Then we define the set assignment M over $\mathit{Vars}(\varphi)$ by putting, for all $x \in \mathit{Vars}(\varphi)$:

$$Mx := \{b_V \mid x \in \overline{V}\}, \quad \text{thus:} \quad \bigcap ML := \{b_{\overline{V}} \mid L \subseteq \overline{V}\}. \tag{3.17}$$

We observe first that $\bigcap ML = \bigcap MR$ holds when $\{L, R\} \in \Phi^{\oplus}_{\varphi}$. Indeed, let $b_{\overline{V}} \in \bigcup ML$, then $L \subseteq \overline{V}$, then by Lemma 3.14(f), $\overline{L} \subseteq \overline{V}$. Since $\{L, R\} \in \Phi^{\oplus}_{\varphi}$, by (Cl1) $L \sim_\varphi R$, so that by Lemma 3.14(d) and (f) again, $R \subseteq \overline{V}$, thus $b_{\overline{V}} \in \bigcap MR$, i.e. $\bigcap ML \subseteq \bigcap MR$. Analogously $\bigcap ML \subseteq \bigcap MR$, henceforth $\bigcap ML = \bigcap MR$.

Finally we prove that $\bigcap ML \neq \bigcap MR$ holds, for any $\{L, R\} \in \Phi^{\ominus}_{\varphi}$. By hypothesis $L \not\sim_\varphi R$, thus by Lemma 3.14(d) $\overline{L} \neq \overline{R}$, then w.l.o.g. assume $\overline{L} \not\subseteq \overline{R}$. By Lemma 3.14(a), $R \subseteq \overline{R}$, thus $b_{\overline{R}} \in \bigcap MR$, while by Lemma 3.14(f), $L \not\subseteq \overline{R}$, thus $b_{\overline{R}} \notin \bigcap ML$, therefore $\bigcap ML \neq \bigcap MR$. □

Notice that the two above proofs are really close to each other, in particular the two set assignment used to prove the sufficiency part of both theorems are somewhat symmetrical. In fact we can easily prove the following corollary:

Corollary 3.19. *Let φ_1 be a $\mathsf{BST}(\cup,=,\neq)$-formula and let φ_2 be the $\mathsf{BST}(\cap,=,\neq)$-formula obtained by replacing each symbol '$\cup$' within φ_1 with the symbol '$\cap$'. Then the follow holds:*

i) φ_1 and φ_2 are equisatisfiable;

ii) if φ_1 is satisfiable then there exist a model M_1 of it such that the set assignment M_2 such that for each variable $x \in \mathit{Vars}(\varphi)$ $M_2x = \bigcup M_1\,\mathit{Vars}(\varphi) \setminus M_1x$ is a model for φ_2; and analogously

iii) if φ_2 is satisfiable then there exist a model M_2 of it such that the set assignment M_1 such that for each variable $x \in \mathit{Vars}(\varphi)$ $M_1x = \bigcup M_2\,\mathit{Vars}(\varphi) \setminus M_2x$ is a model for φ_1.

3.2.3 A cubic-time satisfiability test for $\mathsf{BST}(\cup,=,\neq)$ and for $\mathsf{BST}(\cap,=,\neq)$

Theorems 3.17 and 3.18 imply that any $\mathsf{BST}(\star,=,\neq)$-formula φ is satisfiable if and only if $\overline{L} \neq \overline{R}$ holds for every pair $\{L, R\} \in \Phi^{\ominus}_{\varphi}$. Hence, they yield straight decision procedures for the theories $\mathsf{BST}(\star,=,\neq)$.

The next step will then be to provide an algorithm for computing the closure $\overline{Z}$ of any input $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$, namely, the largest set in $\mathscr{P}^+(\mathit{Vars}(\varphi))$ 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 $\overline{Z}$ of any given $Z \in \mathscr{P}^+(\mathit{Vars}(\varphi))$, for a $\mathsf{BST}\star,=,\neq$-formula φ. After proving its correctness, we will illustrate a lower-level implementation whose time complexity is quadratic.

Algorithm 1 Satisfiability tester

Require: A $\mathsf{BST}(\star,=,\neq)$-formula φ represented by the sets of pairs $\Phi_\varphi^\oplus$ and $\Phi_\varphi^\ominus$.
Ensure: Is φ satisfiable?
1: **for all** $\{L,R\}$ in $\Phi_\varphi^\ominus$ **do**
2: **if** Closure$(L,\Phi_\varphi^\oplus)$ = Closure$(R,\Phi_\varphi^\oplus)$ **then**
3: **return false**;
4: **return true**;

1: **function** Closure$(Z,\Phi_\varphi^\oplus)$
Input: a set Z and the set $\Phi_\varphi^\oplus$
Output: the $\sim_\varphi$-closure $\mathcal{Z}$ of Z
2: $\mathcal{Z} \leftarrow Z$;
3: **while** there exists $\{L,R\} \in \Phi_\varphi^\oplus$ such that $L \subseteq \mathcal{Z} \iff R \not\subseteq \mathcal{Z}$ **do**
4: $\mathcal{Z} \leftarrow \mathcal{Z} \cup L \cup R$;
5: **return** $\mathcal{Z}$;

We can now prove the correctness of the function Closure.

Lemma 3.20. *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 a collection $\Phi_\varphi^\oplus$ the while-loop of the function Closure plainly terminates within a number $k \leqslant |\Phi_\varphi^\oplus|$ 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 $Z \subseteq Z_i$.

The base case $i = 0$ is trivial.

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

$$L \sim_\varphi R, \quad L \subseteq \mathcal{Z}_{i-1} \iff R \not\subseteq \mathcal{Z}_{i-1}, \quad \text{and} \quad \mathcal{Z}_i = \mathcal{Z}_{i-1} \cup L \cup R.$$

Thus, by inductive hypothesis and Lemma 3.14(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 $\mathcal{Z}^{\mathrm{f}} := \mathcal{Z}_k$ is the final value of the variable $\mathcal{Z}$ returned by the execution of Closure$(Z, \Phi_\varphi^\oplus)$.

In addition, the termination condition for the while-loop yields that $L \subseteq \mathcal{Z}^{\mathrm{f}} \iff R \subseteq \mathcal{Z}^{\mathrm{f}}$, for all $\{L,R\} \in \Phi_\varphi^\oplus$. Thus, from Lemma 3.16, it follows that

$$W_1 \subseteq \mathcal{Z}^{\mathrm{f}} \iff W_2 \subseteq \mathcal{Z}^{\mathrm{f}}, \tag{3.18}$$

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 $\mathcal{Z}^{\mathrm{f}} = \overline{Z}$, we observe that, since $Z \sim_\varphi \overline{Z}$ and $Z \subseteq \mathcal{Z}^{\mathrm{f}}$, by (3.18) we have $\overline{Z} \subseteq \mathcal{Z}^{\mathrm{f}}$. Moreover, by Lemma 3.14(a),(d) and since $Z \sim_\varphi \mathcal{Z}^{\mathrm{f}}$,

we readily get $\mathcal{Z}^{\mathrm{f}} \subseteq \overline{\mathcal{Z}^{\mathrm{f}}} = \overline{Z}$. The latter inclusion, together with the previously established one $\overline{Z} \subseteq \mathcal{Z}^{\mathrm{f}}$, implies $\mathcal{Z}^{\mathrm{f}} = \overline{Z}$, i.e., $\mathcal{Z}^{\mathrm{f}}$ is the closure of Z, proving that the call to $\textsc{Closure}(Z, \Phi^{\oplus}_{\varphi})$ computes the closure $\overline{Z}$ of Z correctly. □

Theorems 3.17 and 3.18, together with Lemmas 3.20 and 3.14(d), readily yield that Algorithm 1 is a valid satisfiability test for formulae in the languages $\mathsf{BST}(\cup,=,\neq)$ and $\mathsf{BST}(\cap,=,\neq)$.

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 $\Phi^{\oplus}_{\varphi}$ and a set $Z \in \mathscr{P}^{+}(\mathit{Vars}(\varphi))$ of which one wants to compute the closure $\overline{Z}$. 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 $\{L, R\} \in \Phi^{\oplus}_{\varphi}$ and $\mathcal{Z}$ is the internal variable whose value will converge to $\overline{Z}$ at termination; a doubly linked list Unripe of pairs of form $\big\langle (L \setminus \mathcal{Z}), (R \setminus \mathcal{Z}) \big\rangle$, with $\{L, R\} \in \Phi^{\oplus}_{\varphi}$; 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 allow fast retrieval of nodes in the lists Ripe and Unripe.

Algorithm 2 Implementation of the function Closure

```
1: function Closure(Z,Φ⊕φ)
2:     Z ← Z;
3:     for all {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 all 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 all index i such that x_i ∈ S do
12:            Z ← Z ∪ {x_i};
13:            for all 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;
```

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 $m = |\mathit{Vars}(\varphi)|$ is the number of distinct set variables in φ, $n = |\varphi|$ is the size of φ, $p = |\Phi^{\oplus}_{\varphi}|$ is the number of literals of type $(=)$ in φ, and $q = |\Phi^{\ominus}_{\varphi}|$ is the number of literals of type $(\neq)$ in φ. Plainly, we have $m, p, q \leqslant n$.

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 $A[i] = 1$. At the end of this chapter we will present an algorithm (Algorithm 9), meant to be executed before any of the satisfiability test we will present, that starting from a raw formula, a simple string, will provide to the satisfiability test any structure it will need as the indexing of the set variables in the formula and the array representing each collection of set variables that appears in $\bigcup\Phi_\varphi^\oplus \cup \bigcup\Phi_\varphi^\ominus$. This precomputation phase will require $\mathcal{O}(n)$ time to obtain the set variable indexing and $\mathcal{O}((p+q)m)$ time to build the arrays.

Having build such index, we can collect the $m \leqslant n$ distinct variables of φ and use them as integers $1, \ldots, m$. By means of such an indexing, in $\mathcal{O}(m(p+q))$ time (where $p = |\Phi_\varphi^\oplus|$ and $q = |\Phi_\varphi^\ominus|$) we can represent as an m-bit array each set of variables in $\bigcup\Phi_\varphi^\oplus \cup \bigcup\Phi_\varphi^\ominus$, and accordingly represent $\Phi_\varphi^\oplus$ and $\Phi_\varphi^\ominus$ as lists of pairs of m-bit arrays. Such preliminary encoding phase can be carried out in $\mathcal{O}(m(p+q)+n)$ 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 Unripe, are maintained as doubly linked lists, both operations can be performed in $\mathcal{O}(1)$ time. The function Add returns also a pointer to the newly inserted node. Finally, we use the function Extract(List) to access in $\mathcal{O}(1)$ time the pointer to the first node of List while removing it.

The function Closure($Z, \Phi_\varphi^\oplus$) 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 $\mathcal{O}(1)$ time. Plainly, unions, set differences, and inclusion tests of sets represented as m-bit arrays can easily be performed in $\mathcal{O}(m)$ time. Also, membership tests and the operations of singleton addition and singleton removal can be performed in $\mathcal{O}(1)$ time.

Thus, the initialization phase of the function Closure (lines 2–8) can be performed in $\mathcal{O}(mp)$ 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 $p = |\Phi_\varphi^\oplus|$. Specifically, the list Ripe contains the set $(L \cup R) \setminus \mathcal{Z}$, for all $\{L, R\} \in \Phi_\varphi^\oplus$ such that $L \cup R \not\subseteq \mathcal{Z}$ 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 $\{L, R\} \in \Phi_\varphi^\oplus$ such that $L \not\subseteq \mathcal{Z}$ and $R \not\subseteq \mathcal{Z}$ both hold.

In view of the assignments at lines 15, 18, and 19, the disjointedness 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, for a total of $\mathcal{O}(mp)$ 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 $\mathcal{O}(p)$. Indeed, (i) at the end of the initialization phase, the list $\textsc{Aux}[i]$ contains at most p pointers to nodes in the lists Ripe and Unripe; (ii) once a pointer in the list $\textsc{Aux}[i]$ 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 $\mathcal{O}(1)$ time and, as observed, the list Ripe has size at most p at the end of the initialization phase, it follows that the **while**-loop at lines 9–24 takes $\mathcal{O}(mp)$ time.

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

Finally, our satisfiability tester (Algorithm 1) checks at most q pairs $\{L, R\} \in \Phi^{\ominus}_{\varphi}$ in $\mathcal{O}(mpq)$ time, by computing the closures $\overline{L}$ and $\overline{R}$ by means of calls to the function Closure and comparing them. By taking into account also the preliminary encoding phase, which has a $\mathcal{O}(m(p+q)+n)$-time complexity, the overall complexity of Algorithm 1 is $\mathcal{O}(mpq+n)$, which is $\mathcal{O}(n^3)$ 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 $\mathcal{O}(mp)$ space, namely $\mathcal{O}(n^2)$ space since $m, p \leqslant n$.

Summarizing, we have proved the following result:

Theorem 3.21. *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.*

In addition 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. In fact by first instantiating Mx to the empty set, for all variable x, and then by generating a fresh set b each time we call the function Closure and putting $Mx = Mx \cup \{b\}$, for each variable in the closure computed we obtain the set assignment of Theorem 3.18. Therefore if the formula φ inputted to our algorithm was a $\mathsf{BST}(\cap,=,\neq)$-formula, M will be a model for such formula, otherwise if φ is a $\mathsf{BST}(\cup,=,\neq)$-formula we can use Remark 3.19 to obtain a model.

3.2.4 Extensions of $\mathsf{BST}(\cup,=,\neq)$ and $\mathsf{BST}(\cap,=,\neq)$

In Chapter 2 we presented the two notions of *existential expressibility* and $\mathcal{O}(f)$-*expressibility*, through which one can reduce, in constant or $\mathcal{O}(f)$ time, a set-theoretic formula of a richer language to an equisatisfiable formula belonging to a smaller language.

Regarding $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$ we can use the following results:

(a) the literal $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cup, =)$ and in $\mathsf{BST}(\cap, =)$;

(b) the literal $x \subseteq y$ is existentially expressible in $\mathsf{BST}(\cup, =)$ and in $\mathsf{BST}(\cap, =)$;

(c) the literal $x \not\subseteq y$ is existentially expressible in $\mathsf{BST}(\cup, \neq)$ and in $\mathsf{BST}(\cap, \neq)$;

(d) the literal $\mathsf{Disj}(x, y)$ is existentially expressible in $\mathsf{BST}(\cap, =\varnothing)$ and therefore, by (a), it is $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cap, =)$;

(e) the literal $\neg\mathsf{Disj}(x, y)$ is existentially expressible in $\mathsf{BST}(\subseteq, \neq)$; therefore, by (a), it is existentially expressible in both of $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$.

We can reduce any formula of the two languages $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ and $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ in linear time into a $\mathsf{BST}(\cup, =, \neq)$ or $\mathsf{BST}(\cap, =, \neq)$-formula respectively, meaning that we can use Algorithm 1 to decide the satisfiability of those two languages.

Lemma 3.22. *The satisfiability decision problem for either one of the languages* $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$, $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ *can be solved in cubic time.*

Remark 3.23. *The two fragments* $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ *and* $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ *are polynomial-maximal inside* BST. *This becomes evident using Table 3.2. In fact, by adding any other symbol of* BST *into them results in a super-language of an* NP*-complete fragment listed in the table.*

3.3 The fragment $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$

The next fragment we analyze is $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$, whose satisfiability test, like those of the previous two languages, can be solved in polynomial-time. In the following section we will describe and prove the correctness of a quadratic-time decision algorithm for $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formulae, then, using our expressibility tools, we show how it is possible for the same decision algorithm to establish satisfiability also in the broader fragment $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$. Alongside with $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ and $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$, the latter fragment comprises the whole collection of maximal-polynomial fragments of BST, allowing us to identify a complete complexity taxonomy of all the fragments of the language BST.

Before we can describe its decision procedure, we need to present the language $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$.

Definition 3.11. *The fragment* $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$ *is the collection of conjuncts of literals of types:*

$$\mathsf{Disj}\left(\bigcup L, \bigcup R\right), \ \neg\mathsf{Disj}\left(\bigcup L, \bigcup R\right), \ \bigcup L \neq \bigcup R, \tag{3.19}$$

where L *and* R *are nonempty collections of set variables.*

We recall that the $\mathsf{Disj}(A, B)$ predicate stands for *the sets A and B are disjoint*, and it is defined as

$$\mathsf{Disj}(A, B) \longleftrightarrow A \cap B = \emptyset.$$

As before we will not directly manipulate set variables but instead collections of set variables. Thus it is convenient to represent $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formulae using sets of pairs of collections of set variables. In particular, a given formula φ will be represented by the following three sets:

$$\begin{aligned}
\Phi_\varphi^{\mathsf{Disj}} &\coloneqq \big\{\{L, R\} \mid \mathsf{Disj}(\bigcup L, \bigcup R) \text{ is in } \varphi\big\},\\
\Phi_\varphi^{\neg\mathsf{Disj}} &\coloneqq \big\{\{L, R\} \mid \neg\mathsf{Disj}(\bigcup L, \bigcup R) \text{ is in } \varphi\big\},\\
\Phi_\varphi^{\ominus} &\coloneqq \big\{\{L, R\} \mid \bigcup L \neq \bigcup R \text{ is in } \varphi\big\}.
\end{aligned}$$

3.3.1 Satisfiability of $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$

The first step in finding a polynomial-time satisfiability test is to find an easy to check property that is equivalent to the satisfiability of a given formula. In the case of $\mathsf{BST}(\cup, =\emptyset, \neq\emptyset, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$ we found two conditions that are related to the two sets $\mathcal{C}_\varphi$ and $\mathcal{N}_\varphi$, that now we are going to define.

The definition of $\mathcal{C}_\varphi$ requires the *unordered Cartesian product* '$A \otimes B$' which we recall is defined as:

$$A \otimes B \coloneqq \{\{s, t\} \mid s \in A \ \wedge t \in B\}.$$

Notice that the pair $\{s, t\}$ is unordered, therefore it can also be written as $\{t, s\}$. In addition when an element z belongs to both sets A and B, the improper pair $\{z, z\}$, which is equivalent to the singleton $\{z\}$, belongs to $A \otimes B$.

We are ready to define $\mathcal{C}_\varphi$.

Definition 3.12. *Given a $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formula φ we define:*

$$\mathcal{C}_\varphi \coloneqq \bigcup_{\{L,R\} \in \Phi_\varphi^{\mathsf{Disj}}} (L \otimes R).$$

The set $\mathcal{C}_\varphi$ is intended to be the collection of all pairs of variables in φ that any model of the formula must assign to disjoint sets. The following lemma, besides proving the correctness of this sentence, will make this sentence clearer.

Lemma 3.24. *Let φ be a satisfiable $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formula. Then for each pair $\{x, y\} \in \mathcal{C}_\varphi$ and all models M of φ, we have $\mathsf{Disj}(Mx, My)$.*

By definition of $\mathcal{C}_\varphi$, there exists a pair $\{L, R\} \in \Phi_\varphi^{\mathsf{Disj}}$ such that $\{x, y\} \in L \otimes R$. W.l.o.g. let $x \in L$ and $y \in R$. Then plainly $Mx \subseteq \bigcup ML$ and $My \subseteq \bigcup MR$. Since $\{L, R\} \in \Phi_\varphi^{\mathsf{Disj}}$ and $M \models \varphi$ we have that $\bigcup ML$ and $\bigcup MR$ are disjoint, therefore $\mathsf{Disj}(Mx, My)$ holds.

The other set we use in our satisfiability conditions is $\mathcal{N}_\varphi$.

Definition 3.13. *Given a* $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$*-formula* φ*, we define:*

$$\mathcal{N}_\varphi \coloneqq \bigcup_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}} (L \cap R).$$

This set is meant to be the collection of all the set variables of φ that any set assignment must assign to the empty set in order to satisfy the formula. The following lemma clarify this statement.

Lemma 3.25. *Let* φ *be a satisfiable* $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$*-formula, then for each variable* $x \in \mathcal{N}_\varphi$ *and all models* M *of* φ*, we have* $Mx = \emptyset$*.*

Proof. By definition of $\mathcal{N}_\varphi$, there exist a pair $\{L, R\} \in \Phi_\varphi^{\mathsf{Disj}}$ such that $x \in L \cap R$. Thus, the singleton $\{x\}$ belongs to $\mathcal{C}_\varphi$. Then, by Lemma 3.24, $\mathsf{Disj}\,(Mx, Mx)$, that is $Mx = \emptyset$. □

Lemmas 3.24 and 3.25 already give us an idea to what condition a $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formula must satisfy in order to admit a model. For example, if the pair $\{\{x\}, \{y\}\}$ belongs to $\Phi_\varphi^{\ominus}$ and both x, y are in $\mathcal{N}_\varphi$, then the formula is unsatisfiable, otherwise we would have $Mx \neq My$ and $Mx = \emptyset = My$.

The set $\mathcal{N}_\varphi$ is actually redundant since it can be entirely deduced from $\mathcal{C}_\varphi$, in fact we have:

Lemma 3.26. *The set* $\mathcal{N}_\varphi$ *is equal to* $\{x \in s \in \mathcal{C}_\varphi \mid |s| = 1\}$*.*

Proof. Let $x \in \mathcal{N}_\varphi$ then there exists a pair $\{L, R\} \in \Phi_\varphi^{\mathsf{Disj}}$ such that $x \in L \cap R$, so that $\{x\} \in L \otimes R \subseteq \mathcal{C}_\varphi$, and therefore $\mathcal{N}_\varphi \subseteq \{u \in s \in \mathcal{C}_\varphi \mid |s| = 1\}$. Now let $x \in \{u \in s \in \mathcal{C}_\varphi \mid |s| = 1\}$. Then there exists a pair $\{L, R\} \in \Phi_\varphi^{\mathsf{Disj}}$ such that $\{x\} \in L \otimes R$ so that $x \in L \cap R \subseteq \mathcal{N}_\varphi$. We can conclude that $\mathcal{N}_\varphi = \{u \in s \in \mathcal{C}_\varphi \mid |s| = 1\}$. □

The reason, however, to maintain the set $\mathcal{N}_\varphi$ is twofold; first to ease notation, but more importantly because the set $\mathcal{N}_\varphi$ is easier to computer then $\mathcal{C}_\varphi$. In Section 3.3.4 we will see that whenever the collection $\Phi_\varphi^{\neg\mathsf{Disj}}$ is empty we only need to compute $\mathcal{N}_\varphi$ actually, lowering the time complexity.

We are now ready to present two conditions that are both necessary an sufficient for satisfiability of $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formulae.As anticipated, both $\mathcal{C}_\varphi$ and $\mathcal{N}_\varphi$ will play a central role.

Theorem 3.27. *A* $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$*-formula* φ *is satisfiable if and only if both conditions:*

$$(L \otimes R) \setminus \mathcal{C} \neq \emptyset \qquad \text{for all } \{L, R\} \in \Phi_\varphi^{\neg\mathsf{Disj}} \tag{3.20}$$

$$L \setminus \mathcal{N} \neq R \setminus N \qquad \text{for all } \{L, R\} \in \Phi_\varphi^{\ominus}, \tag{3.21}$$

holds.

Proof. (Necessity.) Let φ be a satisfiable $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formula, and let M be a set assignment that satisfies φ. To prove the necessity of conditions (3.20) and (3.21) we will proceed by contradiction.

First we assume that there exists a pair $\{L, R\} \in \Phi_{\varphi}^{\neg\mathsf{Disj}}$ such that $(L \otimes R) \subseteq \mathcal{C}_{\varphi}$. By Lemma 3.24, then we have

$$(\forall x \in L)(\forall y \in R) \quad \mathsf{Disj}\,(Mx, My)\,,$$

meaning that $\bigcup ML$ and $\bigcup MR$ are disjoint. But since $M \models \varphi$ we also have $\neg\mathsf{Disj}\,(\bigcup ML, \bigcup MR)$, a contradiction.

Last, assume that there exists a pair $\{L, R\} \in \Phi_{\varphi}^{\ominus}$ such that $L \setminus \mathcal{N}_{\varphi} = R \setminus \mathcal{N}_{\varphi}$. By Lemma 3.25 we have that $\bigcup M(L \cap \mathcal{N}_{\varphi}) = \bigcup M(R \cap \mathcal{N}_{\varphi}) = \emptyset$, so that $\bigcup ML = \bigcup M(L \setminus \mathcal{N}_{\varphi}) \cup \bigcup M(L \cap \mathcal{N}_{\varphi}) = \bigcup M(L \setminus \mathcal{N}_{\varphi})$, and analogously $\bigcup MR = \bigcup M(R \setminus \mathcal{N}_{\varphi}) \cup \bigcup M(R \cap \mathcal{N}_{\varphi}) = \bigcup M(R \setminus \mathcal{N}_{\varphi})$. Thus $\bigcup ML = \bigcup MR$ which is a contradiction since $M \models \varphi$, and so $\bigcup ML \neq \bigcup MR$.

By assuming that either (3.20) or (3.21) are not satisfied, we obtained a contradiction. Therefore both conditions must hold whenever φ is satisfiable.

(Sufficiency.) For sufficiency, let us pick $|\mathit{Vars}(\varphi)|^2 - |\mathit{Vars}(\varphi)|$ pairwise distinct sets b_p, for $p \in \mathit{Vars}(\varphi) \otimes \mathit{Vars}(\varphi)$.

Define the following set assignment:

$$Mx \coloneqq \{b_p \mid x \in p, p \notin \mathcal{C}_{\varphi}\}.$$

Notice that, by the above definition, for every pair of variables x and y, $Mx \cap My = \{b_{\{x,y\}}\}$, if $\{x, y\} \notin \mathcal{C}_{\varphi}$ otherwise $Mx \cap My = \emptyset$.

We will show that if a formula φ satisfies both conditions (3.20) and (3.21), then M is a model of φ, by showing that

$$\begin{array}{ll} \mathsf{Disj}\,(\bigcup ML, \bigcup MR) & \text{holds for all pairs } \{L, R\} \in \Phi_{\varphi}^{\mathsf{Disj}}, \\ \neg\mathsf{Disj}\,(\bigcup ML, \bigcup MR) & \text{holds for all pairs } \{L, R\} \in \Phi_{\varphi}^{\neg\mathsf{Disj}}, \\ \bigcup Ml \neq \bigcup MR & \text{holds for all pairs } \{L, R\} \in \Phi_{\varphi}^{\ominus}. \end{array}$$

First let $\{L, R\}$ be in $\Phi_{\varphi}^{\mathsf{Disj}}$. We have that $(L \otimes R) \subseteq \mathcal{C}_{\varphi}$. Thus for all variables $x \in L$ and $y \in R$, we have $\{x, y\} \in \mathcal{C}_{\varphi}$, so that $Mx \cap My = \emptyset$, and therefore $\mathsf{Disj}\,(\bigcup ML, \bigcup MR)$.

Now, let $\{L, R\} \in \Phi_{\varphi}^{\neg\mathsf{Disj}}$. Then by (3.20) there exists a pair $p = \{x, y\} \in (L \otimes R) \setminus \mathcal{C}_{\varphi}$. Thus, $b_p \in Mx \cap My \subseteq \bigcup ML \cap \bigcup MR$, since either $x \in L \wedge y \in R$ or $x \in R \wedge y \in L$, so that $\neg\mathsf{Disj}\,(\bigcup ML, \bigcup MR)$.

Finally, let $\{L, R\} \in \Phi_{\varphi}^{\ominus}$. By (3.21), $L \setminus \mathcal{N}_{\varphi} \neq R \setminus \mathcal{N}_{\varphi}$. W.l.o.g., assume $L \setminus \mathcal{N}_{\varphi} \not\subseteq R \setminus \mathcal{N}_{\varphi}$ so that there exists an $x \in L$ such that $x \notin R \cup \mathcal{N}_{\varphi}$. Since $x \notin \mathcal{N}_{\varphi}$ $x \notin L' \cap R'$ for each pair $\{L', R'\} \in \Phi_{\varphi}^{\mathsf{Disj}}$ so that the singleton $\{x\}$ does not belong to $\mathcal{C}_{\varphi}$. Then, since $x \in L \setminus R$, we have that $b_{\{x\}} \in \bigcup ML \setminus \bigcup MR$ therefore $\bigcup ML \neq \bigcup MR$. □

3.3.2 An efficient satisfiability test for $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$

Theorem 3.27 readily yields a satisfiability test for $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$. All we have to do is compute $\mathcal{C}_\varphi$ and $\mathcal{N}_\varphi$. Lemma 3.26 actually states that we just need $\mathcal{C}_\varphi$, and then check if both condition (3.20) and (3.21) are satisfied.

Algorithm 3 works in the opposite direction, instead of checking if both condition are satisfied, it checks if at least one of the two condition is not met by the input formula. By Theorem 3.28 this is equivalent to check if either condition (3.22) or (3.23) is met. If so, the algorithm returns false, meaning that the formula is not satisfiable, otherwise the algorithm returns true, meaning that the formula is satisfiable.

Algorithm 3 Satisfiability Tester

Require: a three collection of set variables $\Phi_\varphi^{\mathsf{Disj}}, \Phi_\varphi^{\neg\mathsf{Disj}}$ and $\Phi_\varphi^{\ominus}$,
Ensure: is φ satisfiable?

```
1: Let C_φ be a matrix of size |φ| × |φ| containing only zeroes;
2: for each pair {L, R} ∈ Φ_φ^Disj do
3:     for each x_i ∈ L do
4:         for each x_j ∈ R do
5:             C_φ[i][j] ← C_φ[j][i] ← 1;
6: for each pair {L, R} ∈ Φ_φ^¬Disj do
7:     IsSubset ← true;
8:     for each x_i ∈ L do
9:         for each x_j ∈ R do
10:            if C_φ[i][j] = 0 then
11:                IsSubset ← false;
12:    if IsSubset = true then return false;
13: for each pair {L, R} ∈ Φ_φ^⊖ do
14:    IsSubset ← true
15:    for each x_i ∈ (L ∪ R) \ (L ∩ R) do
16:        if C_φ[i][i] = 0 then
17:            IsSubset ← false;
18:    if IsSubset = true then return false;
    return true;
```

Theorem 3.28. *A $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formula φ admits no satisfying set assignments if and only if at least one of the following conditions*

$$(L \otimes R) \subseteq \mathcal{C}, \qquad \text{for some } \{L, R\} \in \Phi_\varphi^{\neg\mathsf{Disj}} \tag{3.22}$$

$$(L \cup R) \setminus (L \cap R) \subseteq \mathcal{N}_\varphi, \qquad \text{for some } \{L, R\} \in \Phi_\varphi^{\ominus} \tag{3.23}$$

holds.

Proof. To prove the theorem, it is enough to negate the conjunction

$$(\forall\{L, R\} \in \Phi_\varphi^{\neg\mathsf{Disj}})L \otimes R \setminus \mathcal{C}_\varphi \neq \emptyset \ \wedge \ (\forall\{L, R\} \in \Phi_\varphi^{\ominus})L \setminus \mathcal{N}_\varphi \neq R \setminus \mathcal{N}_\varphi.$$

By applying De Morgan's laws, we obtain

$$(\exists\{L,R\} \in \Phi_\varphi^{\neg\mathsf{Disj}})L \otimes R \setminus \mathcal{C}_\varphi = \emptyset \ \vee \ (\exists\{L,R\} \in \Phi_\varphi^{\ominus})L \setminus \mathcal{N}_\varphi = R \setminus \mathcal{N}_\varphi.$$

Then the thesis comes directly since the bi-implications

$$A \setminus B = \emptyset \longleftrightarrow A \subseteq B$$
$$A \setminus C = B \setminus C \longleftrightarrow (A \cup B) \setminus (A \cap B) \subseteq C,$$

hold for all sets A, B, and C. □

Complexity and implementation details

Finally, we focus on the time complexity of Algorithm 3. We will express the time complexity in terms of the size n of the input formula φ, namely $n = \sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L|+|R|)+\sum_{\{L,R\}\in\Phi_\varphi^{\neg\mathsf{Disj}}}(|L|+|R|)+\sum_{\{L,R\}\in\Phi_\varphi^{\ominus}}(|L|+|R|)$. We also define m as the number of different variables in the formula $|\mathit{Vars}(\varphi)|$, and put $p \coloneqq |\Phi_\varphi^{\mathsf{Disj}}|, q \coloneqq |\Phi_\varphi^{\neg\mathsf{Disj}}|$, and $r \coloneqq |\Phi_\varphi^{\ominus}|$.

Before running our algorithm, we first need to compute the sets $\mathit{Vars}(\varphi), \Phi_\varphi^{\mathsf{Disj}}, \Phi_\varphi^{\neg\mathsf{Disj}}, \Phi_\varphi^{\ominus}$ and an indexing $\Pi \coloneqq \{\langle\, x\,,\, i\,\rangle \mid x \in \mathit{Vars}(\varphi)\}$ from the input formula, We can again refer to Algorithm 9, that requires $\mathcal{O}(n + m(p + q + r))$-thus, hence is quadratic so it does not effect the asymptotic complexity of Algorithm 3. However, since we do not need to represent each collection of set variables in $\bigcup\Phi_\varphi^{\mathsf{Disj}} \cup \bigcup\Phi_\varphi^{\neg\mathsf{Disj}} \cup \bigcup\Phi_\varphi^{\ominus}$ as an array of size m, we can entirely forgo the step required to generate those arrays and obtain, as we will see at the end of the chapter, a time complexity of $\mathcal{O}(n)$ for this preliminary phase.

Moving to the actual algorithm, we will implement both $\mathcal{C}_\varphi$ and $\mathcal{N}_\varphi$ using a matrix of bit, C_φ, of size $m \times m$, while each collection of set variables V will be implemented as an ordered list of indexes A_V, so that $\mathsf{C}_\varphi[i][j] = 1$ if and only if the pair $\{x_i, x_j\}$ is in $\mathcal{C}_\varphi$ and A_V is the list of indices of the variables inside V. For simplicity we will use V in place of the list A_V. Doing so, both extracting a new variable index from V and checking if $\mathsf{C}_\varphi[i][j]$ is set to 1 will require constant time. We recall that such lists can be obtained in $\mathcal{O}(n)$ time using Algorithm 9.

The first operation we need to perform is initialize every cell of C_φ to 0. This requires $\Theta(m^2)$-time. The **for**-loop at lines 2-5 cycles exactly $\Theta(\sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L| \times |R|)$ times. Similarly the **for**-loop at lines 6-12 cycles exactly $\Theta(\sum_{\{L,R\}\in\Phi_\varphi^{\neg\mathsf{Disj}}}(|L|\times |R|)$ times.

Finally, the **for**-loop at lines 13-18 needs to check if for each pair $\{L,R\} \in \Phi_\varphi^{\ominus}$ the variables inside the set $(L \cup R) \setminus (L \cap R)$ are also in the main diagonal of C_φ. Algorithm 4 can be used to compute $(L \cup R) \setminus (L \cap R)$ in $\mathcal{O}(|L|+|R|)$ times by assuming that L and R are ordered lists of indexes. This such for-loop takes $\mathcal{O}(\sum_{\{L,R\}\in\Phi_\varphi^{\ominus}}(|(L \cup R) \setminus (L \cap R)| + |L| + |R|)$-time.

Algorithm 4, alongside Algorithms 6 and 8, is variation of the algorithm used to merge ordered lists and is included for completeness.

To simplify the above complexity of $\mathcal{O}(\sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L||R|))$, we have that

$$\sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L||R|) \leqslant \sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L|+|R|)^2 \leqslant \Big(\sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L|+|R|)\Big)^2 \leqslant n^2.$$

Analogously $\sum_{\{L,R\}\in\Phi_\varphi^{\neg\mathsf{Disj}}}(|L||R|) \leqslant n^2$. Thus the time complexity for the two first **for**-loops is $\mathcal{O}(n^2)$. Concerning $\mathcal{O}(\sum_{\{L,R\}\in\Phi_\varphi^{\ominus}}(|(L\cup R)\setminus(L\cap R)|+|L|+|R|)$, we plainly have that $|(L\cup R)\setminus(L\cap R)| \leqslant |L|+|R|$ and $\mathcal{O}(\sum_{\{L,R\}\in\Phi_\varphi^{\ominus}}(|L|+|R|) \leqslant n$. Summing up, we obtain a total time complexity of $\mathcal{O}(\sum_{\{L,R\}\in\Phi_\varphi^{\mathsf{Disj}}}(|L|\times|R|)+\sum_{\{L,R\}\in\Phi_\varphi^{\neg\mathsf{Disj}}}(|L|\times|R|)+\sum_{\{L,R\}\in\Phi_\varphi^{\ominus}}|((L\cup R)\setminus(L\cap R)|+|L|+|R|)$, that can be simplified into $\mathcal{O}(n^2)$ so that we can conclude that

Lemma 3.29. *The satisfiability problem for* $\mathsf{BST}(\cup,\neq,\mathsf{Disj},\neg\mathsf{Disj})$ *can be solved in quadratic time.*

Algorithm 4 Compute $(A\cup B)\setminus(A\cap B)$

Require: two sets A and B represented as ordered lists of indexes,
Ensure: the ordered list representing $(A\cup B)\setminus(A\cap B)$

```
1: Let C be an empty list of indexes;
2: n ← 0
3: i ← POP(A);
4: j ← POP(B);
5: while n < |L| + |R| do
6:     if i < j then ADD(i,C);
7:         i ← POP(A);
8:     else if i > j then ADD(j,C);
9:         j ← POP(B);
10:    else
11:        i ← POP(A);
12:        j ← POP(B);
13:        n ← n + 1;
14:    n ← n + 1;
15: return C;
```

Finally, following the proof of Theorem 3.27, it is easy to modify Algorithm 3 to also generate a model for the input formula. First initialize Mx to the empty set for each variable $x \in \mathit{Vars}(\varphi)$. Then each time the value of $\mathsf{C}_\varphi[i][j]$ changes from 0 to 1, we generate a fresh set b_{ij} and add it to both Mx_i and Mx_j. Then if the formula is satisfiable, the set assignment generated, which is exactly the set assignment used to prove Theorem 3.27, will be a model for the input formula.

3.3.3 Extending $\mathsf{BST}(\cup,\neq,\mathsf{Disj},\neg\mathsf{Disj})$

In the previous section we showed our to use the tools of expressibility to extend the two languages $\mathsf{BST}(\cup,=,\neq)$ and $\mathsf{BST}(\cap,=,\neq)$ into broader languages. Similarly, we can also extend $\mathsf{BST}(\cup,\neq,\mathsf{Disj},\neg\mathsf{Disj})$. Using the following facts it

is in fact possible to reduce any $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$-formula into a $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$-formula in constant time.

(a) the literal $x = \varnothing$ is existential expressible in $\mathsf{BST}(\mathsf{Disj})$;

(b) the literal $x \neq \varnothing$ is existential expressible in $\mathsf{BST}(\neg\mathsf{Disj})$;

(c) the literal $x \not\subseteq y$ is existential expressible in $\mathsf{BST}(\cup, \neq)$.

Hence, we have:

Lemma 3.30. *The satisfiability problem for the language* $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$ *can be solved in quadratic time.*

Remark 3.31. *The language* $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$ *is polynomial-maximal inside* BST. *Using Table 3.2, it is easy to check that all the super-theories of* $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq)$ *are* NP*-complete.*

3.3.4 Emblematic, non-maximal, polynomial-complexity decision algorithms

In view of the realization of a complexity taxonomy of BST, and later to the more comprehensive language MLS, it is enough to find those sub-languages that are *minimal* NP*-complete* or *maximal polynomial.* There is a limited interest in further investigating the non-minimal NP-complete fragments, the satisfiability problem for MLS is NP therefore any extension inside the language of an NP-complete fragment will be another NP-complete fragment. One of the rationals behind the study of a complete taxonomy of MLS is to find fast specialized satisfiability test that can be used by automated proof verifies, such as $\mathsf{\text{Ætna}Nova}$, instead of the more complex and general ones. That is why it is of a great interest to study non-maximal polynomial fragments; the latter may, in fact, admit decision tests outperforming any of the maximal polynomial fragments extending them. So far we have found two of such languages, $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$ and $\mathsf{BST}(\cap, =\varnothing, \neq)$. In this section, we will present their satisfiability tests.

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

In Section 3.3 we stated that by removing literals of type $\neg\mathsf{Disj}\,(\bigcup L, \bigcup R)$ from $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$ we can solve in linear time the satisfiability test for the fragment $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$. First we give the definition of $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$

Definition 3.14. *The fragment* $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$ *is the collection of all the conjunctions of literals of types:*

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

where L *and* R *are nonempty collections of set variables.*

As expected, all that applies to the fragment $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$ also applies to $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$ too. We will reuse the same sets $\Phi_{\varphi}^{\mathsf{Disj}}$ and $\Phi_{\varphi}^{\ominus}$, as before, removing the set $\Phi_{\varphi}^{\neg\mathsf{Disj}}$ defined in Section 3.3. Again, the set $\mathcal{N}_{\varphi}$ will have a

central role in our satisfiability test, while we will not need the set $\mathcal{C}_\varphi$, as condition (3.20) will be met vacuously by all $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-formulae because of the absence of literals of type $\neg\mathsf{Disj}\,(\bigcup L, \bigcup R)$. Taking the above in consideration, Theorem 3.27 becomes:

Theorem 3.32. *A $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-formula φ is satisfiable if and only if*

$$L \setminus \mathcal{N}_\varphi \neq R \setminus \mathcal{N}_\varphi, \quad \textit{for all } \{L, R\} \in \Phi^{\ominus}_\varphi. \tag{3.24}$$

The proof of the above theorem is the same as for Theorem 3.27, so we omit it.

Satisfiability Test At this point, it is quite clear that Algorithm 3 can be used to check $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-formulae for satisfiability. Its complexity is $\mathcal{O}(n^2 + n)$, where n is the length of the input formula. The quadratic part, however, is just due to the computation of C_φ. Since the **for**-loop at lines 6-12 will never be entered, and as we observed in the above discussion here we just need the set $\mathcal{N}_\varphi$ in place of the whole set $\mathcal{C}_\varphi$, we can modify Algorithm 3 as follows.

Algorithm 5 Satisfiability Tester

Require: a three collection of set variables $\Phi^{\mathsf{Disj}}_\varphi$, and $\Phi^{\ominus}_\varphi$,
Ensure: is φ satisfiable?

```
 1: Let N_φ be an array of size |φ| initialized to zeroes;
 2: for each pair {L, R} ∈ Φ^Disj_φ do
 3:     for each x_i ∈ L ∩ R do
 4:         C_φ[i] ← 1;
 5: for each pair {L, R} ∈ Φ^⊖_φ do
 6:     IsSubset ← true
 7:     for each x_i ∈ (L ∪ R) \ (L ∩ R) do
 8:         if C_φ[i][i] = 0 then
 9:             IsSubset ← false;
10:     if IsSubset = true then return false;
    return true;
```

As in the previous section we represent a collection of set variable as an ordered lists of indexes. Again, using Algorithm 9 we can obtain all the structure needed by the algorithm in linear time. Set m to $|\mathit{Vars}(\varphi)|$. Then concerning the time complexity of Algorithm 5, we will need $\mathcal{O}(m)$-time to initialize the array $\mathcal{N}_\varphi$ and, as in the previous section, the **for**-loop at lines 5-10 will take $\mathcal{O}(n)$-time.

Moving to the **for**-loop at lines 2-4, we can use Algorithm 6 to compute $L \cap R$ in $\mathcal{O}(|L| + |R|)$-time, by assuming that L and R are maintained as ordered lists. The **for**-loop 2-4 will take $\mathcal{O}\big(\sum_{\{L,R\}\in\Phi^{\mathsf{Disj}}_\varphi}(|L| + |R|)\big)$-time, that is $\mathcal{O}(n)$-time.

We can conclude:

Lemma 3.33. *The satisfiability problem for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$-conjunctions can be solved in linear time.*

Algorithm 6 Compute $A \cap B$

Require: two sets A and B represented as ordered lists of indexes,
Ensure: the ordered list representing $A \cap B$

```
1: Let C be an empty list of indexes;
2: n ← 0
3: i ← POP(A);
4: j ← POP(B);
5: while n < |A| + |B| do
6:     if i < j then
7:         i ← POP(A);
8:     else if i > j then
9:         j ← POP(B);
10:    else
11:        ADD(i,C);
12:        i ← POP(A);
13:        j ← POP(B);
14:        n ← n + 1;
15:    n ← n + 1;
16: return C;
```

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

The other significant non-maximal polynomial fragments is $\mathsf{BST}(\cap, =\varnothing, \neq)$, whose satisfiability problem can be solved in quadratic time, thus improving upon the cubic time required for $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$.

Definition 3.15. *The fragment $\mathsf{BST}(\cap, =\varnothing, \neq)$ is the collection of all conjunctions of literals of types*

$$\bigcap L \neq \bigcap R, \quad \bigcap D = \varnothing,$$

where L, R, and D are nonempty collections of set variables.

As usual, we represent any formula φ using the following sets:

$$\Phi^{\ominus}_{\varphi} := \{\{L, R\} \mid \bigcap L \neq \bigcap R \text{ is in } \varphi\}$$
$$\Phi^{\varnothing}_{\varphi} := \{D \mid D = \varnothing \text{ is in } \varphi\}.$$

Theorem 3.34. *A $\mathsf{BST}(\cap, =\varnothing, \neq)$-formula φ is satisfiable if and only if both the following conditions are satisfied:*

$$(\forall\{L, R\} \in \Phi^{\ominus}_{\varphi})\ L \neq R; \tag{3.25}$$
$$(\forall\{D, D'\} \in \Phi^{\varnothing}_{\varphi} \otimes \Phi^{\varnothing}_{\varphi})(\forall\{L, R\} \in \Phi^{\ominus}_{\varphi})\ D \not\subseteq L \ \vee\ D \not\subseteq R. \tag{3.26}$$

Proof. (Necessity.) Let φ be a satisfiable $\mathsf{BST}(\cap, =\varnothing, \neq)$-formula and let M be one of its models. Plainly, (3.25) must be satisfied otherwise there would exist a pair $\{L, R\} \in \Phi^{\ominus}_{\varphi}$ such that $L = R$. Thus, $\bigcap ML = \bigcap MR$, and since M models φ we will also have $\bigcap ML \neq \bigcap MR$. Arguing by contradiction, suppose that

condition (3.26) is not satisfied, so that there exists a pair $\{D, D'\} \in \Phi^{\varnothing}_{\varphi} \otimes \Phi^{\varnothing}_{\varphi}$ and a pair $\{L, R\} \in \Phi^{\ominus}_{\varphi}$ such that $D \subseteq L$ and $D' \subseteq R$. Recalling that $\bigcap MD = \emptyset$ for all $D \in \Phi^{\varnothing}_{\varphi}$, we have:

$$\bigcap ML = \bigcap M(L \setminus D) \cap \bigcap M(D) = \bigcap M(L \setminus D) \cap \emptyset = \emptyset = \emptyset.$$
$$\bigcap MR = \bigcap M(R \setminus D') \cap \bigcap M(D') = \bigcap M(R \setminus D') \cap \emptyset = \emptyset = \emptyset.$$

Thus $\bigcap ML = \bigcap MR$, but again we must have $\bigcap ML \neq \bigcap MR$.

(Sufficiency.) Let $\mathcal{D}$ be the collection of all the sets in $\bigcup \Phi^{\ominus}_{\varphi}$ that contains a set of $\Phi^{\varnothing}_{\varphi}$, namely

$$\mathcal{D} \coloneqq \{V \in \bigcup \Phi^{\ominus}_{\varphi} \mid (\exists D \in \Phi^{\varnothing}_{\varphi}) D \subseteq V\}.$$

Assign to each element V of $\bigcup \Phi^{\ominus}_{\varphi}$ a set b_V, so that $V \neq V'$ if and only if $b_V \neq b_{V'}$. Then define for each variable in $Vars(\varphi)$ the set assignment

$$Mx \coloneqq \{b_V \mid x \in V \in (\bigcup \Phi^{\ominus}_{\varphi}) \setminus \mathcal{D}\};$$

then it is easy to show that for each collection of set variables S:

$$\bigcap MS \coloneqq \{b_V, \mid S \subseteq V \ \wedge \ V \in (\bigcup \Phi^{\ominus}_{\varphi}) \setminus \mathcal{D}\}.$$

To show that $M \models \varphi$, we prove that $\bigcap MD = \emptyset$, for all $D \in \Phi^{\varnothing}_{\varphi}$, and $\bigcap ML \neq \bigcap MR$, for all $\{L, R\} \in \Phi^{\ominus}_{\varphi}$.

Assume by contradiction that there exists a $D \in \Phi^{\varnothing}_{\varphi}$ such that $\bigcap MD \neq \emptyset$. Then there must exists a $V \in (\bigcap \Phi^{\ominus}_{\varphi}) \setminus \mathcal{D}$ such that $D \subseteq V$, hence $V \in \mathcal{D}$, a contradiction; thus $\bigcap MD = \emptyset$.

Now let $\{L, R\} \in \Phi^{\ominus}_{\varphi}$. by (3.25) we have $L \neq R$. w.l.o.g., assume $L \nsubseteq R$. then as long as $R \notin \mathcal{D}$ we have $b_R \in \bigcap MR \setminus \bigcap ML$. If $R \in \mathcal{D}$, then there exists a $D \in \Phi^{\varnothing}_{\varphi}$ such that $D \subseteq R$. Hence, by condition (3.26), $L \notin \mathcal{D}$ and $R \nsubseteq L$; in fact if $R \subseteq L$ then $D \subseteq L$, contradicting (3.26). Thus $b_L \in \bigcap ML \setminus \bigcap MR$. In both cases $\bigcap ML \neq \bigcap MR$ therefore we can conclude that if a formula satisfies both conditions (3.25) and (3.26) then it is satisfiable. □

Satisfiability Test

Theorem 3.34 states that, in order to establish whether or not a $\mathsf{BST}(\cap, =\varnothing, \neq)$-conjunction is satisfiable, it suffices to check whether the conjunction satisfies both conditions (3.25) and (3.26). Algorithm 7 is a straightforward implementation of this check. Through its analysis, we will now show that such task can be performed in quadratic time.

Correctness of Algorithm 7 is easy to prove; just notice that the **for**-loop at lines 1-3 ensures that (3.25) is satisfied, otherwise false is returned; analogously the second **for**-loop, at line 4-12 ensures that condition (3.26) is satisfied.

To analyze the complexity of this algorithm, first we put n as the length of the input formula interpreted as a string. Using Algorithm 9, we will obtain, in

Algorithm 7 Satisfiability test for $\mathsf{BST}(\cap, =\varnothing, \neq)$-conjunctions

Require: The collections $\Phi_\varphi^\ominus$ and $\Phi_\varphi^\varnothing$.
Ensure: Is φ satisfiable?
1: **for** $\{L, R\} \in \Phi_\varphi^\ominus$ **do**
2: **if** $L_i = R_i$ **then**
3: **return** false;
4: **for** $\{L, R\} \in \Phi_\varphi^\ominus$ **do**
5: $B_L \leftarrow B_R \leftarrow$ false;
6: **for** $D \in \Phi_\varphi^\varnothing$ **do**
7: **if** $D \subseteq L$ **then**
8: $B_L \leftarrow$ true;
9: **if** $D \subseteq R$ **then**
10: $B_R \leftarrow$ true;
11: **if** $B_L \wedge B_R$ **then**
12: **return** false;
13: **return** true;

linear time, both sets $\Phi_\varphi^\ominus$ and $\Phi_\varphi^\varnothing$ along with the representation of each collection of set variables that appears in φ as an ordered list of set variables, according to some ordering. Doing so we can use Algorithm 8 to check if $S \subseteq T$ in time $\mathcal{O}(|S| + |T|)$.

Then, it is an easy matter to show that the **for**-loop at lines 1-3 runs in $\mathcal{O}\big(\sum_{\{L,R\}\in\Phi_\varphi^\ominus}(|L| + |R|)\big)$-time, that can be quickly simplified into $\mathcal{O}(n)$ time.

Similarly, the **for**-loop at lines 4-12 runs in $\mathcal{O}\big(\sum_{\{L,R\}\in\Phi_\varphi^\ominus}\sum_{D\in\Phi_\varphi^\varnothing}(|D| + |L| + |R|)\big)$. We can rewrite the last expression as $\mathcal{O}\big(|\Phi_\varphi^\varnothing|\sum_{\{L,R\}\in\Phi_\varphi^\ominus}(|L| + |R|) + |\Phi_\varphi^\ominus|\sum_{D\in\Phi_\varphi^\varnothing}|D|\big)$. Plainly, the terms $|\Phi_\varphi^\varnothing|, |\Phi_\varphi^\ominus|, \sum_{\{L,R\}\in\Phi_\varphi^\ominus}(|L|+|R|)$ and $\sum_{D\in\Phi_\varphi^\varnothing}|D|$ are smaller then n. Thus we can simplify the above expression to $\mathcal{O}(n^2)$.

Summing up, we have:

Lemma 3.35. *The satisfiability problem for* $\mathsf{BST}(\cap, =\varnothing, \neq)$ *can be solved in quadratic time.*

3.3.5 Managing the input formulae

In sections 3.2, 3.3, and 3.3.4, we presented several polynomial-time satisfiability tests. Neither of these algorithms, however, take as input the raw formula, as a string of length n, but instead requires a specific representation of the formula, i.e., using sets such as $\Phi_\varphi^\oplus$.

We need a precomputation phase that can provide such sets to the various satisfiability tests by parsing the input formula. It is important for this precomputation phase to not overcome the asymptotic time complexity of the actual satisfiability test. Some of the strategies we used for the polynomial satisfiability test are chosen to reduce time complexity. For example a naive implementation of the check required to ensure that two sets A and B are equal could require

Algorithm 8 Is $A \subseteq B$

Require: two sets A and B represented as ordered lists of indexes,
Ensure: is $A \subseteq B$ true?

```
1: i ← POP(A);
2: j ← POP(B);
3: n ← 1
4: c ← 1
5: while n ⩽ |R| do
6:     if ( then i < j)
7:         return false;
8:     else if ( then i > j)
9:         j ← POP(B);
10:    else
11:        i ← POP(A);
12:        j ← POP(B);
13:        c ← n + 1;
14:    n ← n + 1;
15: if c = |A| then
16:    return true;
17: else
18:    return false;
```

$\mathcal{O}(|A||B|)$-time, while a more accurate implementation requires $\mathcal{O}(|A| + |B|)$-time. For Algorithm 9 we target linear time complexity, although failing since Step 5 requires quadratic time. Luckily Step 5 is only required by the satisfiability test for $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$, whose time complexity is cubic, while forgoing this step the algorithm can be executed in linear time.

From the above discussion, we can underline the specification for this precomputation algorithm:

- The input of Algorithm 9 is a string of characters representing a BST formula;
- the output of Algorithm 9 is a collection of sets Φ, one for each relation symbol inside the formula; and an indexing of the variables of the formula;
- Φ is a set of pairs $\{L, R\}$, where L, R are collection of set variables represented either by ordered lists of set variables or by arrays of bits.

To implement the set Φ we will use simple linked lists. Likewise, the two support structures F and V, to be discussed later, will also be implemented as linked lists, while set variables will be represented by pairs (x, i) where x is a string containing the name of the variable (the same string found inside the raw formula) while i is the indexing of the variable in the index we are building.

A requirement to support the linear-time complexity of the satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$ is that the five step algorithm, presented in Algorithm 9, can be performed also in linear-time. As we will show, this requirement is met by the

first four steps of the algorithm, while Step 5 will take quadratic-time. Luckily this last step is only required by the satisfiability test for both $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$, which have a cubic time complexity, and can be entirely skipped in the other cases.

Obtaining a linear time complexity for the first four step is not trivial however. In particular Step 2 will require a boundedness condition, which is quite natural.

Definition 3.16. *A (n infinite) collection $\mathcal{C}$ of finite lists of finite sets of natural numbers is said to be*

(a) c-bounded *(for some constant $c \geqslant 1$), if each list $[S_1, \ldots, S_k]$ in $\mathcal{C}$ satisfies the following condition:*

- if $S_i \neq \emptyset$, for some $i = 1, \ldots, k$, then

$$\max \bigcup_{i=1}^{k} S_i - \min \bigcup_{i=1}^{k} S_i + 1 \leqslant c \cdot \sum_{i=1}^{k} |S_i| \,;$$

(b) bounded*, if it is c-bounded for some $c \geqslant 1$.*

Let $\mathcal{L} \coloneqq [S_1, \ldots, S_k]$ be a list of finite subsets of $\mathbb{N}$, drawn from an *infinite* bounded collection $\mathcal{C}$. Without loss of generality, we assume that at least one set in $\mathcal{L}$ is nonempty. Put

$$n \coloneqq \sum_{i=1}^{k} |S_i| \qquad \text{(input size)}$$

and

$$\sigma \coloneqq \max \bigcup_{i=1}^{k} S_i - \min \bigcup_{i=1}^{k} S_i + 1 \qquad \text{(range size)},$$

so that $\sigma = \mathcal{O}(n)$, by the boundedness assumption.

Algorithm 9 Precomputation algorithm

Step 1 Instantiate three empty lists Φ, F, and V, then parse the formula φ to populate the list Φ, while doing so put the pointer to each collection of set variables inside F and the pointer to each set variables to V;

Step 2 . sort the two lists Φ and V and also each list pointed by the elements of F;

Step 3 . following the order inside V assign to each distinct variable an index;

Step 4 . remove all the duplicate variables inside the collections pointed by the elements of F;

Step 5 . Associate to each list V pointed by an element of F an array of bit A_V representing the same collection of set variables as V;

Algorithm 9 provides a quick overview of the precomputation algorithm. What follows is a detailed description alongside an analysis of the complexity for each of its steps.

Step 1. To parse the formula we do as follows: collect characters from the formula inside the current string x until either an operator, or a relation symbol, or a logical connective is found. Then add the pair (x, i), where i is an uninitialized integer, to the current list V, that represents a collection of set variables. A pointer to this pair is also added to the list V. Then initialize a new current string x. Whenever a relation symbol, or a logical connective '$\wedge$' is found, add the current list V to the current triple $\{V, U, \sim\}$, where $\sim$ is a relation symbol, also add the pointer to V to the list $\oplus$, and initialize a new current list V. When a relation symbol $\sim$ is found, add it to the current triple $\{V, U, \sim\}$. Finally, whenever a logical connective is found, add the current triple $\{V, U, \sim\}$ to the list Φ, and then initialize a new current triple $\{V, U, \sim\}$. Once the entire formula φ is parsed, V will represent the collection $\mathit{Vars}(\varphi)$, containing multiple instance of the same variable. Moreover, the collections V's are unordered and can contain duplicate set variables. In order to complete this step, the input formula must be parsed once. For each character scanned, a constant number of basic operation must be performed: initialize a pair, add a character at the end of a string, add a pointer to a list, thus this step can be performed in $\mathcal{O}(n)$-time.

Step 2. During this step, we need to sort both lists Φ and V, alongside all the list pointed by elements of F. To order the triples inside Φ, we give whatever order to the different relation symbols and order the triples only referring to their relation symbol. Doing so we can easily split Φ by gathering each triple by their relation symbol, obtaining the various sets $\Phi^{\oplus}_{\varphi}, \Phi^{\ominus}_{\varphi}$ and so on. We will treat the list V as the list of set variables pointed by its elements, and sort each list of set variables according to the lexicographic order of the strings x representing the set variable.

Since we are targeting linear-time complexity, we can not use sorting algorithms based on comparisons, such as *Quick-Sort*, otherwise we would have a lower bound of $\Omega(n \log n)$. We recall that Φ, $\mathcal{V}$ and $\bigcup\mathsf{F}$, that is the concatenation of all the lists pointed by the elements of $\mathcal{F}$, have all size $\mathcal{O}(n)$. The sorting algorithm we choose is *Bucket-Sort*. However, if during step Step 2, the lists V's pointed by the elements of F were sorted separately by means of k distinct runs of the Bucket-Sort algorithm, then this step would take $\mathcal{O}\big(k(n+\sigma)\big)$-time, namely $\mathcal{O}(n+(k+1)\sigma) = \mathcal{O}((k+1)n)$-time, by the boundedness condition, where k is the size of F, ruling out from the very outset the possibility to attain an overall $\mathcal{O}(n)$-time complexity. However, the following trick will allow us to sort the sets V's by a single run of Bucket-Sort. The collections V are naturally indexed by their position inside F, so that we can call V_i the i-th element of $\oplus$, we can scan F, in $\mathcal{O}(n)$-time, to build the collection $\mathsf{P} \coloneqq \{(x, i) \mid x \in V_i\}$ and call *Bucket-Sort* over this set. Specifically, each set V_i, for $i = 1, \ldots, k$, is scanned and, for each $x \in V_i$, the pair (x, i) (rather than just the plain string x, as would happen in a standard execution of Bucket-Sort) is placed in bucket x (previously initialized to empty). Afterwards, k initially empty tuples T_i, with $i = 1, \ldots, k$, are allocated and the buckets are inspected in increasing order. For each pair (x, i) in bucket x, the pair

(x, i) is appended to the tuple T_i. After all buckets have been scanned, each tuple will contain the elements of the corresponding set in sorted order. We can then simply order V with a single additional run of the *Bucket-Sort* algorithm; the total complexity of phase Step 2 is easily seen to be $\mathcal{O}(n + \sigma)$, which, by the boundedness condition, reduces to $\mathcal{O}(n)$.

Step 3. To implement this step, first set a counter c to 0 and then for each pair $(x, i) \in \mathsf{V}$, increment c if the string x is different to that of its predecessor inside V, then set $i = c$. It is important that the first index is 0. During this step, we need to reed the string x of each set variables that appears in the formula twice, one time when we are comparing x with its predecessor and another time when we are comparing x with its successor, since the combined length of each string x, for all the set variables, is less then n, we can conclude that this step requires $\mathcal{O}(n)$-time.

Step 4. This step is easy to implement. Once the collection V inside F are internally ordered, we can just scan each V and remove from them each pair (x, i) that is equal to its predecessor. Also this step plainly requires $\mathcal{O}(n)$-time to be completed.

Step 5. To generate the arrays of this step, we proceed as follows: for each collection of set variables V pointed by an element of F, initialize an array of A_V, whose size is equal to the index of the variable pointed by the last element of V plus one, that is $|\mathit{Vars}(\varphi)|$. Initially all the array components are set to 0; then for each set variable (x, i) in V $A_V[i]$ is set to 1. For each list V, we need to initialize each cell of an array of $|\mathit{Vars}(\varphi)|$ to 0, then scan the list to set to 1 the proper cells. This requires $\mathcal{O}(|\mathit{Vars}(\varphi)|n)$-time namely quadratic time.

From the above discussion, it should be clear that to execute Algorithm 9 we need quadratic-time. However, by not executing Step 5, this complexity reduces to linear-time allowing us to confirm the linear time complexity for the satisfiability test for $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$.

3.3.6 Summary

We have concluded our analysis of the polynomial fragments of BST. We can gather the results of the above sections and add them to our table representing the taxonomy of BST.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg\mathsf{Disj}$	$\subseteq$	$\not\subseteq$	$=$	$\neq$	Complexity
★			★	★	★	★		★		★	$\mathcal{O}(n^2)$
★			★	★		★	★	★	★	★	$\mathcal{O}(n^3)$
	★		★	★	★	★	★	★	★	★	$\mathcal{O}(n^3)$

TABLE 3.3: Maximal polynomial fragments of BST

3.4 A complete complexity taxonomy for BST

Table 3.3 claims to be the table of all the maximal polynomial polynomial fragments of BST, and it is true indeed. By combining this table with Table 3.2 we should cover every possible fragment of BST. Something is missing however: what about $\mathsf{BST}(\setminus, =)$? This fragment is neither a super-theory of a minimal NP-complete fragment nor a sub-theory of a maximal polynomial fragment.

3.4.1 Constant time satisfiable fragments of BST

There are actually quite a few other fragments that share this property with $\mathsf{BST}(\setminus, =)$, those are the fragment that only contains satisfiable formulae, therefore their satisfiability problem can be solved in constant time.

Like polynomial fragments, constant fragments can be represented by the collection of maximal constant fragments, we have two such fragments:

$$\mathsf{BST}(\cup, \cap, \setminus, =\varnothing, \mathsf{Disj}, \subseteq, =), \qquad \mathsf{BST}(\cup, \cap, =, \neq\varnothing, \neg\mathsf{Disj}, \subseteq).$$

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

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

After adding this last small piece to our puzzle, we can finally compile Table 3.4 reporting over the complete complexity taxonomy of the language BST.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg$Disj	$\subseteq$	$\not\subseteq$	$=$	$\neq$	Complexity
		★								★	NP-complete
		★						★			NP-complete
		★				★					NP-complete
		★		★							NP-complete
★	★									★	NP-complete
★	★							★			NP-complete
★	★		★	★							NP-complete
★	★				★	★					NP-complete
★	★		★			★					NP-complete
★	★			★	★						NP-complete
★					★	★			★		NP-complete
★				★	★				★		NP-complete
★					★	★	★				NP-complete
★					★				★	★	NP-complete
★					★			★	★		NP-complete
★				★	★		★				NP-complete
★					★		★			★	NP-complete
★					★		★	★			NP-complete
★			★	★	★	★		★		★	$\mathcal{O}(n^3)$
★			★	★		★	★	★	★	★	$\mathcal{O}(n^3)$
	★		★	★	★	★	★	★	★	★	$\mathcal{O}(n^4)$
★	★	★	★		★		★		★		$\mathcal{O}(1)$
★	★			★		★	★		★		$\mathcal{O}(1)$

TABLE 3.4: Complete taxonomy of theory BST

Chapter 4

A complete complexity taxonomy of MST

The language **Membership Set Theory** (MST) can be viewed as the dual fragment of BST inside MLS, in fact it is obtained by removing the equality relation symbol ' = '. The constructs of MST refer to a universe of nested (as opposed to 'flat') sets: this is the domain which the variables of MST range over. Albeit MST is devoid of quantifiers. In the ongoing we will prove that the decision problem for MST is NP-complete, proving once again the NP-completeness of MLS. Furthermore, like BST, we will show that some of the fragments of MLS have a polynomial-time solvable decision problem. We will provide a complete taxonomy of MST that can be useful in real life scenarios, such as the implementation of a proof verifier, this time we have to analyze far less fragments then for BST. In fact, MST admits 24 distinct fragments, of which 14 are polynomial and 10 are NP-complete, nevertheless we only study the minimal NP-complete and the maximal polynomial fragments inferring the complexity of the other fragments from them. The following results where already shown in [CMO20].

Before moving on to the various results, we provide an overview of the syntax of the language MST.

Syntax of MST

The fragments of set theory whose satisfiability problems are addressed in this chapter are parts, syntactically delimited, of a specific quantifier-free language MST.

The symbols appearing inside MST are:

- infinitely many set variables $x, y, z, \ldots$;
- the constant symbol '$\varnothing$';
- the set operators symbols ' $\setminus$ ', ' $\cup$ ', ' $\cap$ ';
- the set membership relation symbol ' $\in$ ';
- the logical connective ' $\wedge$ ', '$\neg$'.

Then we define the MST-terms as the smallest set of expressions such that:

- all the set variables and the constant '$\varnothing$' are MST-terms;
- if s and t are MST-terms, so are $s \setminus t$, $s \cup t$, and $s \cap t$.

The MST-atoms are the smallest set of expressions of type

$$s \in t,$$

where both s and t are MST-terms; MST-literals are simply atoms A and their negation $\neg A$ and finally MST-formulae are conjunctions of literals be means of the logical connective $\wedge$.

We will forgo the symbol '$\neg$' in our formulae replacing each literal $\neg(s \in t)$ with the atom $s \notin t$.

As before, it is needless to define a semantic for MST since we will use the same interpretation used for MLS.

In the following section we will build, in a similar fashion to Table 3.4, a table listing all the fragments of MST alongside their complexity.

$\cup$	$\cap$	$\setminus$	$\in$	$\notin$	Complexity

TABLE 4.1: Empty complexity table for MLS

4.1 The minimal NP-complete fragments of MST

Beginning with NP-complete fragments, we will study the two fragments MST$(\cup, \cap, \in)$ and MST$(\setminus, \in)$ which are minimal NP-complete. For the time being, their minimality in only suggested, but it will become evident once we present the polynomial fragments of MST.

4.1.1 The fragment MST$(\cup, \cap, \in)$

Definition 4.1. *The fragment* MST$(\cup, \cap, \in)$ *consists of all the conjunctions of literals of type:*

$$t_1 \in t_2,$$

where t_1 *and* t_2 *are terms involving only set variables and the set operators* '$\cup$' *and* '$\cap$'.

We shall prove that the satisfiability problem for MST$(\cup, \cap, \in)$-formulae is NP-complete by reducing the problem 3SAT to it.

Let F be an instance of 3SAT, defined as in Definition 3.1, and let $x, X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n$ be $2n + 1$ distinct set variables.

For all $i \in \{1, \ldots, m\}$, $j \in \{1, 2, 3\}$, and $k \in \{1, \ldots, n\}$, let us put

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

Finally, put

$$\Phi_F \coloneqq \bigwedge_{i=1}^{m} \big(x \in T_{i1} \cup T_{i2} \cup T_{i3}\big) \ \wedge \bigwedge_{k=1}^{n} \big(x \in X_k \cup \overline{X}_k \ \wedge \ X_k \cap \overline{X}_k \in x\big).$$

Theorem 4.1. *A 3SAT instance F is propositionally satisfiable if and only if the corresponding $\mathsf{MST}(\cup,\cap,\in)$-conjunction Φ_F is satisfied by some set assignment.*

Proof. (Necessity). Assume first that F is propositionally satisfiable, and let $\mathfrak{v}$ be a Boolean valuation that satisfies it. Let $M_{\mathfrak{v}}$ be the set-assignment induced over $\mathit{Vars}(\Phi_F)$ by $\mathfrak{v}$ according to the following rules: for $k = 1, \ldots, n$,

$$M_{\mathfrak{v}} X_k \coloneqq \begin{cases} \{\{\emptyset\}\} & \text{if } \mathfrak{v}(P_k) = \mathsf{true}\,, \\ \emptyset & \text{otherwise;} \end{cases}$$
$$M_{\mathfrak{v}} \overline{X}_k \coloneqq \{\{\emptyset\}\} \setminus M_{\mathfrak{v}} X_k\,; \quad M_{\mathfrak{v}} x \coloneqq \{\emptyset\}\,.$$

Hence, $M_{\mathfrak{v}}(X_k \cup \overline{X}_k) = \{\{\emptyset\}\}$ and $M_{\mathfrak{v}}(X_k \cap \overline{X}_k) = \emptyset$ for each k, so that

$$M_{\mathfrak{v}} \models \bigwedge_{k=1}^{n} \big(x \in X_k \cup \overline{X}_k \ \wedge \ X_k \cap \overline{X}_k \in x\big)\,. \tag{4.1}$$

Let now $i \in \{1, \ldots, m\}$. Since $\mathfrak{v}$ satisfies F by hypothesis, $\mathfrak{v}(L_{i1} \vee L_{i2} \vee L_{i3}) = \mathsf{true}$, and hence $\mathfrak{v}(L_{ij_i}) = \mathsf{true}$ for some $j_i \in \{1, 2, 3\}$. Let $k \in \{1, \ldots, n\}$ be such that $L_{ij_i} \in \{P_k, \neg P_k\}$. If $L_{ij_i} = P_k$, then $\mathfrak{v}(P_k) = \mathsf{true}$ and $T_{ij_i} = X_k$, and hence $M_{\mathfrak{v}} T_{ij_i} = \{\{\emptyset\}\}$. On the other hand, if $L_{ij_i} = \neg P_k$, then $\mathfrak{v}(P_k) = \mathsf{false}$ and $T_{ij_i} = \overline{X}_k$; hence, again, $M_{\mathfrak{v}} T_{ij_i} = \{\{\emptyset\}\}$. Thus, $M_{\mathfrak{v}} x = \{\emptyset\} \in \{\{\emptyset\}\} = M_{\mathfrak{v}}(T_{ij_i}) \subseteq M_{\mathfrak{v}}(T_{i1} \cup T_{i2} \cup T_{i3})$, proving that $M_{\mathfrak{v}}$ satisfies the literal $x \in T_{i1} \cup T_{i2} \cup T_{i3}$.

This fact, true of each i, along with (4.1), yields that $M_{\mathfrak{v}}$ rightly models Φ_F.

(Sufficiency). Assume next that the $\mathsf{MST}(\cup,\cap,\in)$-conjunction Φ_F is satisfiable, and consider a set assignment M modeling it. Then, for each $k = 1, \ldots, n$, we have

$$Mx \in MX_k \cup M\overline{X}_k \ \wedge \ MX_k \cap M\overline{X}_k \in Mx,$$

whence $Mx \notin MX_k \cap M\overline{X}_k$. Thus, either $Mx \in MX_k$ or $Mx \in M\overline{X}_k$ holds, but not both. A Boolean valuation $\mathfrak{v}_M$ is induced on $P_1, \ldots, P_n$ by the rule

$$\mathfrak{v}_M(P_k) \coloneqq \begin{cases} \mathsf{true} & \text{if } Mx \in MX_k\,, \\ \mathsf{false} & \text{otherwise}\,. \end{cases}$$

Since $M \models \Phi_F$, then $M \models \bigwedge_{i=1}^{m} x \in T_{i1} \cup T_{i2} \cup T_{i3}$ holds. Thus, for each $i \in \{1, \ldots, m\}$, $Mx \in MT_{ij_i}$ holds for some $j_i \in \{1, 2, 3\}$. Let $k \in \{1, \ldots, n\}$ be such that $T_{ij_i} \in \{X_k, \overline{X}_k\}$. If $T_{ij_i} = X_k$, then $L_{ij_i} = P_k$ and $\mathfrak{v}_M(P_k) = \mathsf{true}$. If, on the contrary, $T_{ij_i} = \overline{X}_k$, then $L_{ij_i} = \neg P_k$ and $\mathfrak{v}_M(P_k) = \mathsf{false}$. In any case,

$\mathfrak{v}_M(L_{ij_i}) = \mathsf{true}$, and so $\mathfrak{v}_M(L_{i1} \vee L_{i2} \vee L_{i3}) = \mathsf{true}$. This fact, true of each i, yields that the Boolean valuation $\mathfrak{v}$ satisfies F. □

Since Φ_F gets constructed from F in $\mathcal{O}(|F|)$-time, from Theorem 4.1 we get:

Lemma 4.2. *The satisfiability problem for* $\mathsf{MST}(\cup, \cap, \in)$*-conjunctions is* NP*-complete.*

4.1.2 The fragment $\mathsf{MST}(\setminus, \in)$

Definition 4.2. *The fragment* $\mathsf{MST}(\setminus, \in)$ *consists of all the conjunctions of literals of type:*

$$t_1 \in t_2,$$

where t_1 *and* t_2 *are terms involving only set variables and the set operator* ' $\setminus$ '.

We shall prove that the satisfiability problem for $\mathsf{MST}(\setminus, \in)$-conjunctions is NP-complete by reducing the problem 3SAT to it.

Again let F be an instance of the 3SAT problem, and let $x, X_1, \ldots, X_n, X^*$ be $n+2$ distinct set variables. For $i = 1, \ldots, m$ and $j = 1, 2, 3$, put

$$T_{ij} \coloneqq \begin{cases} X_k & \text{if } L_{ij} = \quad P_k \text{ for some } k\,, \\ X^* \setminus X_k & \text{if } L_{ij} = \neg P_k \text{ for some } k\,; \end{cases}$$

then define

$$\mathcal{C}_i \coloneqq (((X^* \setminus T_{i1}) \setminus T_{i2}) \setminus T_{i3})\,,$$

and finally put

$$\Psi_F \coloneqq x \in (\cdots((X^* \setminus \mathcal{C}_1) \setminus \mathcal{C}_2)\cdots) \setminus \mathcal{C}_m\,. \tag{4.2}$$

Theorem 4.3. *A 3SAT instance* F *is propositionally satisfiable if and only if the corresponding* $\mathsf{MST}(\setminus, \in)$*-conjunction* Ψ_F *is satisfied by some set assignment.*

Proof. (Necessity). First assume that F is propositionally satisfiable, and let $\mathfrak{v}$ be a Boolean valuation satisfying it; then define, for $k = 1, \ldots, n$:

$$M^*X_k \coloneqq \begin{cases} \{\{\emptyset\}\} & \text{if } \mathfrak{v}(P_k) = \mathsf{true}\,, \\ \emptyset & \text{otherwise;} \end{cases}$$
$$M^*x \coloneqq \{\emptyset\}\,; \quad M^*X^* \coloneqq \{\{\emptyset\}\}\,.$$

Since $\mathfrak{v}$ satisfies F, for each $i \in \{1, \ldots, m\}$ we have $\mathfrak{v}(C_i) = \mathsf{true}$, and hence to each i there corresponds a $j = 1, 2, 3$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$. If $L_{ij} = P_k$ for some k, then $M^*T_{ij} = M^*X_k = \{\{\emptyset\}\} = M^*X^*$ follows from $\mathfrak{v}(P_k) = \mathsf{true}$; if, on the contrary, $L_{ij} = \neg P_k$ for some k, then $M^*T_{ij} = M^*X^* \setminus M^*X_k = M^*X^* \setminus \emptyset = M^*X^*$ follows from $\mathfrak{v}(P_k) = \mathsf{false}$. These remarks make it clear that $M^*\mathcal{C}_i = \emptyset$ holds for $i = 1, \ldots, m$, so that

$$(\cdots(M^*X^* \setminus M^*\mathcal{C}_1)\cdots) \setminus M^*\mathcal{C}_m = M^*X^*\,;$$

therefore, since $M^*x \in M^*X^*$, we conclude that M^* models Ψ_F.

(Sufficiency). Assume next that the $\mathsf{MST}(\setminus, \in)$-conjunction Ψ_F is satisfiable, and consider a set-model M for it. Define the Boolean evaluation $\overline{\mathfrak{v}}$ by the rule

$$\overline{\mathfrak{v}}(P_k) \coloneqq \begin{cases} \mathsf{true} & \text{if } Mx \in MX_k\,, \\ \mathsf{false} & \text{otherwise.} \end{cases}$$

Since $M \models \Psi_F$, it must hold that $Mx \in MX^*$ and $Mx \notin M\mathcal{C}_i$ for any $i \in \{1, \ldots, m\}$. Thus, by the definition of $\mathcal{C}_i$, to each i there must correspond a $j \in \{1, 2, 3\}$ such that $Mx \in MT_{ij}$: if $L_{ij} = P_k$ for some k, then $Mx \in MT_{ij} = MX_k$, hence $\overline{\mathfrak{v}}(P_k) = \overline{\mathfrak{v}}(L_{ij}) = \mathsf{true}$; if, on the contrary, $L_{ij} = \neg P_{ij}$, then $Mx \in MT_{ij} = MX^* \setminus MX_k$, hence $Mx \notin MX_k$, $\overline{\mathfrak{v}}(P_k) = \mathsf{false}$, and $\overline{\mathfrak{v}}(L_{ij}) = \mathsf{true}$ again. Summing up, for each $i = 1, \ldots, m$ there exists a $j = 1, 2, 3$ such that $\overline{\mathfrak{v}}(L_{ij}) = \mathsf{true}$, hence $\overline{\mathfrak{v}}$ satisfies F. □

Since Ψ_F gets constructed from F in $\mathcal{O}(|F|)$-time, from Theorem 4.3 we get:

Lemma 4.4. *The satisfiability problem for* $\mathsf{MST}(\setminus, \in)$ *is* NP*-complete.*

Remark 4.5. *In Theorems 4.1 and 4.3 concerning* NP*-hardness, we have built models of rank at most 2. It is often interesting, in connection with more demanding fragments of set theory (see, e.g., [Can91, CUO02]), to determine the maximum rank needed to satisfy a formula as a function of its length (or just the number of relators, in case of normalized conjunctions), so as to obtain a small model theorem of some sort.*

4.2 The maximal polynomial fragments of MST

The maximal polynomial fragments of MST are $\mathsf{MST}(\cup, \in, \notin)$, $\mathsf{MST}(\cap, \in, \notin)$, and $\mathsf{MST}(\cup, \cap, \setminus, \notin)$, whose satisfiability problems can be solved in linear, quadratic, and constant time, respectively, as we are about to show.

In sight of the results we will present in this section, it is useful to analyze some easy, yet crucial, examples of unsatisfiable conjunctions in the aforementioned fragments of MST.

4.2.1 Introductory examples

Example 4.1. *The* $\mathsf{MST}(\cup, \in)$*-conjunction*

$$x_0 \cup x_1 \in x_2 \cup x_3 \quad \wedge \quad x_2 \cup x_3 \in x_0 \cup x_2$$

is unsatisfiable.

Notice that if the above conjunction were satisfied by some set assignment M, then $Mx_2 \cup Mx_3 \in Mx_0 \cup Mx_2$ should hold. However, $Mx_2 \cup Mx_3$ cannot belong to Mx_2, else we would have $Mx_2 \cup Mx_3 \in Mx_2 \cup Mx_3$. On the other, Mx_2 cannot belong to Mx_0 either, else the first conjunct would lead us into

$Mx_2 \cup Mx_3 \in Mx_0 \cup Mx_1 \in Mx_2 \cup Mx_3$, a membership cycle again. Summing up, the conjunction under consideration is unsatisfiable.

Example 4.2. *The* $\mathsf{MST}(\cup, \in, \notin)$*-conjunction*

$$\begin{aligned} &x_0 \cup x_1 \in x_2 \cup x_3 \ \wedge \\ &x_2 \cup x_3 \in x_0 \cup x_2 \cup x_4 \cup x_5 \ \wedge \\ &x_2 \cup x_3 \notin x_4 \cup x_5 \end{aligned}$$

is unsatisfiable.

Since the last two conjuncts can be simplified into the single conjunct $x_2 \cup x_3 \in x_0 \cup x_2$, the above conjunction yields the one discussed in Example 4.1; therefore it is unsatisfiable in its turn.

Example 4.3. *The* $\mathsf{MST}(\cap, \in, \notin)$*-conjunction*

$$x_0 \cap x_1 \in x_2 \cap x_3 \cap x_4 \quad \wedge \quad x_0 \cap x_1 \notin x_2 \cap x_4$$

is unsatisfiable.

The conjunction is plainly unsatisfiable since it requires that $x_0 \cap x_1$ belongs and does not belong to $x_2 \cap x_4$.

Example 4.4. *The* $\mathsf{MST}(\cap, \in)$*-conjunction*

$$x_0 \cap x_1 \in x_2 \cap x_3 \cap x_4 \quad \wedge \quad x_2 \cap x_3 \in x_0 \cap x_1 \cap x_5$$

is unsatisfiable.

In this case, if the conjunction were satisfied by some set assignment M, then $Mx_2 \cap Mx_3 \in Mx_0 \cap Mx_1 \in Mx_2 \cap Mx_3$ should hold, conflicting with the axiom of regularity.

4.2.2 The fragment $\mathsf{MST}(\cup, \in, \notin)$, decidable in linear time

In order to see that the satisfiability problem for $\mathsf{MST}(\cup, \in, \notin)$-conjunctions admits a linear-time solution, we first prove an analogous statement for $\mathsf{MST}(\cup, \in)$ and then will prove that the satisfiability problem for $\mathsf{MST}(\cup, \in, \notin)$ can be reduced in linear time to the one for $\mathsf{MST}(\cup, \in)$.

A linear-time satisfiability test for $\mathsf{MST}(\cup, \in)$**.**

To begin with, we provide a decision procedure for $\mathsf{MST}(\cup, \in)$ and then prove that it runs in linear time in the size of the input formula.

Definition 4.3. *The fragment* $\mathsf{MST}(\cup, \in)$ *is the collection of all the conjunctions of literals of types*

$$\bigcup L \in \bigcup R,$$

where L *and* R *are nonempty collections of set variables.*

In order to simplify our proof, we can all the literals of a given $\mathsf{MST}(\cup,\in)$-formula φ so that we have

$$\varphi \coloneqq \bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R_i. \tag{4.3}$$

Theorem 4.6. *Let φ be an $\mathsf{MST}(\cup,\in)$-conjunction of the form (4.3). Then:*

(a) *if $\prec$ is a linear ordering of $Vars(\varphi)$ satisfying the condition*

$$\max(L_i,\prec) \prec \max(R_i,\prec), \text{ for } i = 1,\dots,p, \tag{4.4}$$

then φ has a model M such that

(a_1) *$Mx = \{b_x\} \cup \{\, \bigcup ML_i \mid i = 1,\dots,p \ \wedge\ x = \max(R_i,\prec)\,\}$ for each $x \in Vars(\varphi)$, where the b_x's are pairwise distinct sets of the same rank α, and*

(a_2) *$\bigcup ML \notin Mx$, for any $L \subseteq Vars(\varphi)$ and any $x \in Vars(\varphi)$ such that*

$$\bigwedge_{i=1}^{p} \big(L = L_i \longrightarrow x \notin R_i\big); \tag{4.5}$$

(b) *if φ is satisfiable, then there is a linear ordering $\prec$ of $Vars(\varphi)$ such that condition (4.4) holds.*

Hence, the satisfiability problem for $\mathsf{MST}(\cup,\in)$-conjunctions is solvable.

Proof. Let us first assume that there is a linear ordering $\prec$ of $Vars(\varphi)$ such that (4.4) holds. Following the ordering $\prec$, for $x \in Vars(\varphi)$ we put

$$Mx \coloneqq \{b_x\} \cup \Big\{\ \bigcup ML_i \mid i = 1,\dots,p \ \wedge\ x = \max(R_i,\prec)\ \Big\}, \tag{4.6}$$

where the b_x's are pairwise distinct sets of the same rank α. For any literal $\bigcup L_i \in \bigcup R_i$ in φ, with $i \in \{1,\dots,p\}$, by setting $x_i \coloneqq \max(R_i,\prec)$ we enforce $\bigcup ML_i \in Mx_i \subseteq \bigcup MR_i$, so that $M \models \bigcup L_i \in \bigcup R_i$. Thus, M models correctly all conjuncts of φ, and it plainly satisfies condition (a_1).

To see that also condition (a_2) is true of M, we argue as follows. The biimplication

$$\bigcup ML = \bigcup ML' \text{ if and only if } L = L' \tag{4.7}$$

holds for $L, L' \subseteq Vars(\varphi)$, as the sets b_x in $\bigcup ML$ are the same as those in $\bigcup ML'$ if and only if $L = L'$. Let $L \subseteq Vars(\varphi)$ and $x \in Vars(\varphi)$ be such that $\bigwedge_{i=1}^{p}(L = L_i \longrightarrow x \notin R_i)$ holds, but assume, by way of contradiction, $\bigcup ML \in Mx$. Then, by (4.7) and (4.6), $L = L_{i_0}$ for some $i_0 \in \{1,\dots,p\}$ such that $x = \max(R_{i_0},\prec) \in R_{i_0}$, contradicting (4.5). Hence M satisfies condition (a_2) too.

Concerning claim (b), assume now that $\overline{M}$ is a model of our conjunction φ. Also, let $\prec$ be any linear ordering of $\mathit{Vars}(\varphi)$ such that

$$\mathsf{rk}\left(\overline{M}x\right) < \mathsf{rk}\left(\overline{M}y\right) \longrightarrow x \prec y, \quad \text{for } x, y \in \mathit{Vars}(\varphi). \tag{4.8}$$

To check that the ordering $\prec$ satisfies (4.4), consider an $i \in \{1, \ldots, p\}$. Since $\bigcup\overline{M}L_i \in \bigcup\overline{M}R_i$, we have

$$\max\{\mathsf{rk}\left(\overline{M}x\right) \mid x \in L_i\} = \mathsf{rk}\left(\bigcup\overline{M}L_i\right) < \mathsf{rk}\left(\bigcup\overline{M}R_i\right) = \max\{\mathsf{rk}\left(\overline{M}x\right) \mid x \in R_i\},$$

so that, by (4.8), $\max(L_i, \prec) \prec \max(R_i, \prec)$ holds also in this case. Therefore (4.4) holds, completing the proof of (b). □

Claims (a) and (b) of the above theorem entail that the formula φ is satisfiable if and only if there is a linear ordering $\prec$ of $\mathit{Vars}(\varphi)$ satisfying (4.4). From this, the decidability of the satisfiability problem for $\mathsf{MST}(\cup, \in)$ readily follows.

Towards a linear-time satisfiability test for $\mathsf{MST}(\cup, \in)$, we next derive two conditions which can be tested in linear time and whose conjunction is equivalent to (4.4). To begin with, in connection with any formula φ of $\mathsf{MST}(\cup, \in)$, we define the *left-variables* (resp., *right-variables*) of φ to be those variables in φ occurring in the left-hand side L (resp., right-hand side R) of some literal $\bigcup L \in \bigcup R$ in φ. We call *pure right-variables* the right-variables of φ that are not left-variables.

Now let φ be any satisfiable $\mathsf{MST}(\cup, \in)$-formula and let $\prec$ be any linear ordering of $\mathit{Vars}(\varphi)$ fulfilling condition (4.4). Plainly, no left-variable in φ can be $\prec$-maximal. Hence

(A) φ has some pure right-variable.

In addition, by letting φ^- be the result of dropping from φ all conjuncts that involve some pure right-variable, we clearly have

(B) φ^- is satisfiable.

It turns out that conditions (A) and (B) are also sufficient for the satisfiability of φ. Indeed, assume that conditions (A) and (B) hold for a given $\mathsf{MST}(\cup, \in)$-formula φ, and let $\prec^-$ be a linear ordering of $\mathit{Vars}(\varphi^-)$ fulfilling condition (4.4) of Theorem 4.6 as applied to φ^-. Then, in order to fulfil condition (4.4) of Theorem 4.6 as applied to φ, it will suffice to take any extension $\prec$ of $\prec^-$ to a linear ordering of $\mathit{Vars}(\varphi)$ such that $x \prec y$ holds for every left-variable x and every pure right-variable y of φ; hence the conjunction φ is satisfiable.

The above considerations readily yield the following satisfiability test for $\mathsf{MST}(\cup, \in)$:

A straightforward implementation of Algorithm 10 is quadratic. An alternative implementation consists in counting the number of left-occurrences of each variable $x \in \mathit{Vars}(\varphi)$, while also maintaining, for each right-variable y in φ, the list of the conjuncts where y occurs. Specifically, we maintain two arrays $\mathcal{A}$ and

Algorithm 10 Satisfiability tester for $\mathsf{MST}(\cup, \in)$

```
while φ contains some pure right-variable do
    drop from φ all of the conjuncts involving some pure right-variable;
if φ is the empty conjunction then
    return SATISFIABLE;
else
    return UNSATISFIABLE;
```

$\mathcal{B}$, both indexed by set variables. The former array consists of counters $\mathcal{A}[x]$, each indicating the number of conjunctions in which the variable x appears in the left-hand-side term; the latter array consists of pointers $\mathcal{B}[x]$, each of which is the list of pointers to the conjuncts where x appears in the right-hand-side term. Variables with a zero counter are exactly the pure right-variables. As long as the conjunction φ contains right-variables with a zero counter, drop all conjuncts containing some right-variable with a zero counter from it and, accordingly, decrement the counters of the left-occurrences in the dropped conjuncts. If, eventually, one ends up with the empty conjunction, then the initial conjunction φ is declared to be satisfiable, else unsatisfiable.

Algorithm 11 shows a linear-time complexity implementation for the above satisfiability test. Hence, we have:

Lemma 4.7. *The satisfiability problem for* $\mathsf{MST}(\cup, \in)$ *is solvable in linear time.*

Remark 4.8. *Extending the signature of* $\mathsf{MST}(\cup, \in)$ *with a constant* $\varnothing$ *designating the empty set only calls for marginal retouches to the decision algorithm. In essence, we must refer the satisfiability problem to conjunctions of the form* $\bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R_i$*, where some of the collections* L_i, R_i *of variables can be empty. This slightly richer situation readily leads to an unsatisfiability response should* $\emptyset$ *occur among* $R_1, \ldots, R_p$*; otherwise, it can be handled by adding a symbol* $\perp$ *distinct from any set-variable to the domain* $Vars(\varphi)$ *of the orderings* $\prec$ *which interest us (see Theorem 4.6). We must stipulate that* $\perp \prec x$ *holds for every set variable* x*, and shall consequently take* $\max(L_i, \prec) = \perp$ *when* L_i *is empty.*

A linear-time reduction of $\mathsf{MST}(\cup, \in, \notin)$ to $\mathsf{MST}(\cup, \in)$.

We shall now prove that the satisfiability problem for $\mathsf{MST}(\cup, \in, \notin)$ can be reduced in linear time to the one for $\mathsf{MST}(\cup, \in)$. This will yield a linear-time decision procedure for the satisfiability problem of the extended fragment $\mathsf{MST}(\cup, \in, \notin)$.

Definition 4.4. *The fragment* $\mathsf{MST}(\cup, \in, \notin)$ *is the collection of all conjunctions of literals of following types*

$$\bigcup L \in \bigcup R, \quad \bigcup L \notin \bigcup R;$$

where L *and* R *are nonempty collection of set variables.*

Algorithm 11 Linear satisfiability test for $\mathsf{MST}(\cup, \in)$

Require: An $\mathsf{MST}(\cup, \in)$-formula φ
Ensure: is φ satisfiable

```
1: for each conjunct ℓ in φ do
2:     for each left-variable l in ℓ do
3:         A[l] ← A[l] + 1;
4:     for each right-variable r in ℓ do
5:         add a pointer to ℓ to B[r];
6: for each set variable x in φ do
7:     if A[x] = 0 then
8:         R ← R ∪ {x};
9: while φ is nonempty do
10:    if R = ∅ then
11:        return UNSATISFIABLE;
12:    extract a set variable x from R;
13:    for each conjunct ℓ in B[x] do
14:        for each left-variable l in ℓ do
15:            A[l] ← A[l] − 1;
16:            if A[l] = 0 then
17:                R ← R ∪ {l};
18:        for each right-variable r in ℓ do
19:            remove the pointer to ℓ from B[r];
20:        remove ℓ from φ
21: return SATISFIABLE;
```

Likewise $\mathsf{MST}(\cup,\in)$-formulae, we can index all the literals inside a given $\mathsf{MST}(\cup,\in,\notin)$-formula φ so that

$$\varphi \coloneqq \bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R_i \ \wedge \ \bigwedge_{j=p+1}^{n} \bigcup L_j \notin \bigcup R_j \tag{4.9}$$

Lemma 4.9. *The satisfiability problem for* $\mathsf{MST}(\cup,\in,\notin)$ *can be reduced in linear time to that for* $\mathsf{MST}(\cup,\in)$.

Proof. Let φ be a $\mathsf{MST}(\cup,\in,\notin)$-formula of type (4.9). For each $i = 1,\ldots,p$, let

$$R'_i \coloneqq R_i \setminus \bigcup\{\, R_j \mid j = p+1,\ldots,n \ \wedge \ L_j = L_i \,\}, \tag{4.10}$$

and put

$$\varphi' \coloneqq \begin{cases} \bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R'_i & \text{if no } R'_i \text{ is empty,} \\ x_0 \in x_0 & \text{otherwise,} \end{cases}$$

for a fixed, but otherwise arbitrary, set variable x_0. Notice that $x_0 \in x_0$ is unsatisfiable. It is an easy matter to verify that if φ is satisfiable, then ($\varphi' \equiv \bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R'_i$ and) any model for φ is also a model for φ', i.e., $\models \varphi \longrightarrow \varphi'$.

Conversely, if φ' is satisfiable (and therefore $\varphi' \equiv \bigwedge_{i=1}^{n} \bigcup L_i \in \bigcup R'_i$), then, by Theorem 4.6(b), there is a linear ordering $\prec$ of $\mathit{Vars}(\varphi')$ fulfilling condition (4.4) and in addition, by Theorem 4.6(a), the conjunction φ' has a model M satisfying conditions (a_1) and (a_2) of the same theorem. We extend M to the variables $x \in \mathit{Vars}(\varphi) \setminus \mathit{Vars}(\varphi')$, if any, by setting $Mx \coloneqq \{b_x\}$, where the b_x's are pairwise distinct sets of the same rank α new to the assignment M. Since $\bigcup MR'_i \subseteq \bigcup MR_i$, for $i = 1,\ldots,p$, we plainly have $M \models \bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R_i$, namely M satisfies all the positive conjuncts of φ.

Let us now show that M satisfies the negative part $\bigwedge_{j=p+1}^{n} \bigcup L_j \notin \bigcup R_j$ of φ as well. Indeed, if this were not the case, there would exist a $j \in \{p+1,\ldots,n\}$ and a variable $x \in R_j$ such that $\bigcup ML_j \in Mx$. Since $\bigcup ML_j$ is not a set of rank α, Mx cannot be of the form $\{b_x\}$ and therefore $x \in \mathit{Vars}(\varphi')$. Thus, by condition ($\mathrm{a}_1$) of Theorem 4.6, $Mx = \{b_x\} \cup \big\{\, \bigcup ML_i \mid i = 1,\ldots,p \ \wedge \ x = \max(R'_i, \prec) \big\}$, and so we would have $\bigcup ML_j = \bigcup ML_i$ for some $i \in \{1,\ldots,p\}$ such that $x = \max(R'_i, \prec)$, so that $x \in R'_i$. The pairwise distinctness of the urelements would yield $L_j = L_i$. Hence, by (4.10), $x \notin R'_i$, a contradiction.

In conclusion, M must also satisfy the negative part $\bigwedge_{j=p+1}^{n} \bigcup L_j \notin \bigcup R_j$ of φ, and thus, in view of $M \models \bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R_i$ seen above, M satisfies φ.

Summing up, we have proved that if φ' is satisfiable, so is φ, which, in view of $\models \varphi \longrightarrow \varphi'$ observed above, yields the equisatisfiability between φ and φ'.

To complete the proof, it suffices to observe that the conjunction φ' can be constructed in $\mathcal{O}(|\varphi|)$ time, where $|\varphi|$ denotes the size of φ, by means of a suitable linear-time algorithm for detecting all duplicates in the list $L_1,\ldots,L_n$ of sets (see below). □

Linear-time duplicate detection.

We briefly outline how to detect duplicate sets in linear time in *bounded* lists of sets of natural numbers, where the notion of boundedness is defined as follows.

To detect duplicates in $\mathcal{L}$, one can utilize the following three-phase algorithm:

Algorithm 12 Duplicate detection

1: sort each one of the sets S_i's in $\mathcal{L}$; let $\mathcal{L}' \coloneqq [T_1, \ldots, T_k]$ be the list of the resulting sorted tuples;
2: sort lexicographically the list $\mathcal{L}'$; let $\mathcal{L}'' \coloneqq [T_{i_1}, \ldots, T_{i_k}]$ be the resulting list;
3: find the duplicate tuples in $\mathcal{L}''$ (these correspond to the duplicate sets in $\mathcal{L}$).

Completing phase 1 in $\mathcal{O}(n)$ is possible by simply running the first four steps of Algorithm 9.

By means of Algorithm 3.2 of [AHU74, pp. 80–84] for sorting tuples of varying length in lexicographic order, phase 2 can be accomplished in time $\mathcal{O}(n + \sigma)$ that is by the boundedness condition Definition 3.16 equal to $\mathcal{O}(n)$.

Finally, duplicate tuples in $\mathcal{L}''$ (corresponding to duplicate sets in $\mathcal{L}$) can be spotted during phase 3 just by checking all adjacent pairs $T_{i_\ell}, T_{i_{\ell+1}}$ of ordered tuples. In this last phase, each component of the tuples $T_{i_1}, \ldots, T_{i_k}$ is accessed at most twice, yielding a $\mathcal{O}(n)$-time complexity.

In conclusion, each of the above phases takes $\mathcal{O}(n)$ time, which results in an overall $\mathcal{O}(n)$-time complexity for our three-phase algorithm. In view of Lemma 4.7, Lemma 4.9 yields the following complexity result:

Lemma 4.10. *The satisfiability problem for* $\mathsf{MST}(\cup, \in, \notin)$ *can be solved in linear time.*

Remark 4.11. *Much as with* $\mathsf{MST}(\cup, \in)$ *(see Remark 4.8), extending the signature of* $\mathsf{MST}(\cup, \in, \notin)$ *with a constant* $\varnothing$ *designating the empty set affects but marginally the decision algorithm. We must refer the satisfiability problem to conjunctions* φ *of the form* $\bigwedge_{i=1}^{p} \bigcup L_i \in \bigcup R_i \wedge \bigwedge_{j=p+1}^{n} \bigcup L_j \notin \bigcup R_j$*, where some of the collections* L_h, R_h *of variables can be empty. A negative literal whose* R_j *is empty is obviously true and, as a matter of fact, it has no bearing on the construction* (4.10) *of* $R'_1, \ldots, R'_p$ *carried out by the satisfiability-preserving translation* $\varphi \mapsto \varphi'$ *discussed in Lemma 4.9. Also in the case when* $L_j = \emptyset$ *but* $R_j \neq \emptyset$ *for some* $j \in \{p+1, \ldots, n\}$*, the translation can treat* L_j, R_j *precisely in the same manner as the pair* $L_{j'}, R_{j'}$ *involved in any other negative literal of* φ*.*

4.2.3 The fragment $\mathsf{MST}(\cap, \in, \notin)$, decidable in quadratic time

Definition 4.5. *The fragment* $\mathsf{MST}(\cap, \in, \notin)$ *is the collection of all the conjunctions of literals of the following types*

$$\bigcap L \in \bigcap R, \quad \bigcap L \notin \bigcap R',$$

where L and R are nonempty collections of set variables.

Much like $\mathsf{MST}(\cup,\in)$ and $\mathsf{MST}(\cup,\in,\notin)$, we can index each literal of a given $\mathsf{MST}(\cap,\in,\notin)$-formula φ so that

$$\varphi \coloneqq \bigwedge_{i=1}^{p} \bigcap L_i \in \bigcap R_i \ \wedge \ \bigwedge_{j=p+1}^{n} \bigcap L_j \notin \bigcap R_j. \tag{4.11}$$

We shall denote by φ^+ and φ^- the *positive part* $\bigwedge_{i=1}^{p} \bigcap L_i \in \bigcap R_i$ and the *negative part* $\bigwedge_{j=p+1}^{n} \bigcap L_j \notin \bigcap R_j$ of φ, respectively. Without loss of generality, as regards decidability, we may assume that the following condition holds:

(C1) $L_i \neq L_h$ for any two distinct $i, h \in \{1,\dots,p\}$.

In fact, condition (C1) can be enforced, without affecting satisfiability, by replacing, for each set $L \in \{L_1,\dots,L_p\}$ of variables, the collection of conjuncts $\bigcap L_i \in R_i$ in φ^+ such that $L_i = L$ by the single conjunct $\bigcap L \in \bigcap R$, where $R \coloneqq \bigcup\{R_i \mid i = 1,\dots,p \ \wedge \ L_i = L\}$, since $\models \bigcap\{\bigcap R_i \mid i = 1,\dots,p \ \wedge \ L_i = L\} = \bigcap R$.

As seen earlier, the duplicates in the list $L_1,\dots,L_p$ of sets can be detected in linear time, and therefore condition (C1) can be enforced in time $\mathcal{O}(|\varphi|)$.

Theorem 4.12. *Let φ be an $\mathsf{MST}(\cap,\in,\notin)$-formula of the form (4.11), fulfilling condition (C1). Then φ is satisfiable if and only if the following two conditions hold:*

(a) *$L_j = L_i$ implies $R_j \not\subseteq R_i$, for $i = 1,\dots,p$ and $j = p+1,\dots,n$;*

(b) *the relation*

$$\{(i,h) \mid i,h = 1,\dots,p \ \wedge \ L_i \subseteq R_h\}$$

among subscripts of the conjuncts of φ^+ is acyclic.

Proof. (Necessity). Assume first that φ is satisfiable, and let $M \models \varphi$.

Concerning condition (a), suppose that $L_j = L_i$, for some $i \in \{1,\dots,p\}$ and $j \in \{p+1,\dots,n\}$. Then we have $\bigcap ML_j = \bigcap ML_i \in \bigcap MR_i \setminus \bigcap MR_j$, so $R_j \not\subseteq R_i$ must hold.

As for condition (b), by way of contradiction suppose that

$$i_0,\dots,i_n \in \{1,\dots,p\} \quad \text{are such that} \quad R_{i_0} \supseteq L_{i_1},\dots,R_{i_{n-1}} \supseteq L_{i_n}, \quad \text{and} \quad R_{i_n} \supseteq L_{i_0}$$

Then φ^+ requires that

$$\bigcap ML_{i_0} \in \bigcap MR_{i_0} \subseteq \bigcap ML_{i_1} \in \cdots \in MR_{i_{n-1}} \subseteq \bigcap ML_{i_n} \in \bigcap MR_{i_n} \subseteq \bigcap ML_{i_0},$$

leading to the untenable cycle $\bigcap ML_{i_0} \in \bigcap ML_{i_1} \in \cdots \in \bigcap ML_{i_n} \in \bigcap ML_{i_0}$.

(Sufficiency). Conversely, assume that φ complies with conditions (a), (b).

We associate a distinct set b_k with each conjunct in φ such that all the sets b_k's have the same rank α and, for each $i = 1, \ldots, p$, make the following technical definition:

$$\mathcal{I}_i \coloneqq \{b_k \mid k = 1, \ldots, n \ \wedge \ L_i \subseteq L_k\} \ \cup \ \{\mathcal{I}_h \mid h = 1, \ldots, p \ \wedge \ L_i \subseteq R_h\}\,. \quad (4.12)$$

Thanks to condition (b), this definition of the $\mathcal{I}_i$'s is well-posed. In what follows, we will manage to define a model M of φ so that, for $i = 1, \ldots, p$ and $k = 1, \ldots, n$, $\mathcal{I}_i$ will equal the intersection of all $M\ell$'s with $\ell \in L_k$ if and only if $i = k$.

We put, for $x \in \mathit{Vars}(\varphi)$:

$$Mx \coloneqq \{b_k \mid k = 1, \ldots, n \ \wedge \ x \in L_k\} \ \cup \ \{\mathcal{I}_i \mid i = 1, \ldots, p \ \wedge \ x \in R_i\}\,. \quad (4.13)$$

The rest of the proof is devoted to showing that M, as just defined, satisfies φ. We shall make use of the following claims.

Claim 4.13. *$\mathcal{I}_i \neq \mathcal{I}_h$ holds for all $i, h = 1, \ldots, p$ such that $i \neq h$.*

By (C1), $L_i \neq L_h$ follows from $i \neq h$. Assuming, w.l.o.g., $L_i \not\subseteq L_h$, we get $b_i \in \mathcal{I}_i \setminus \mathcal{I}_h$ from (4.12). □

Claim 4.14. *$\mathcal{I}_i = \bigcap ML_i$ holds for all $i = 1, \ldots, p$.*

By (4.13) and (4.12) every member of $\bigcap ML_i$, as well as of $\mathcal{I}_i$, either is a sat b_k of rank α, with $k \in \{1, \ldots, n\}$, or is a set $\mathcal{I}_h$, with rank strictly greater then α, with $h \in \{1, \ldots, p\}$. We also have

$$\begin{aligned}
&b_k \in \textstyle\bigcap ML_i && \iff\\
&b_k \in Ml, \text{ for all } l \in L_i && \iff\\
&b_i \subseteq L_k && \text{by (4.13)} \iff\\
&b_k \in \mathcal{I}_i && \text{by (4.12);}
\end{aligned}$$

and

$$\begin{aligned}
&\mathcal{I}_h \in \textstyle\bigcap ML_i && \iff\\
&\mathcal{I}_h \in Ml, \text{ for all } l \in L_i && \iff\\
&L_i \subseteq R_h && \text{by (4.13) and Claim 4.13} \iff\\
&\mathcal{I}_h \in \mathcal{I}_i && \text{by (4.12).}
\end{aligned}$$

□

Claim 4.15. *For every $S \subseteq \mathit{Vars}(\varphi)$ and $i = 1, \ldots, p$, if $\mathcal{I}_i \in \bigcap MS$ then $S \subseteq R_i$.*

$$\begin{array}{ll} \mathcal{I}_i \in \bigcap MS & \Longrightarrow \\ \mathcal{I}_i \in Mv, \text{ for all } v \in S & \Longrightarrow \\ v \in R_i, \text{ for all } v \in S \qquad \text{by (4.13) and Claim 4.13} & \Longrightarrow \\ S \subseteq R_i. & \end{array}$$

□

Claim 4.16. *For $i = 1, \ldots, p$ and $k = 1, \ldots, n$, if $\mathcal{I}_i = \bigcap ML_k$ then $L_i = L_k$.*

Suppose that $\mathcal{I}_i = \bigcap ML_k$. Then, by (4.13), $b_k \in \mathcal{I}_i$; hence, by (4.12), $L_i \subseteq L_k$. Again by (4.13), thanks to Claim 4.14, $b_i \in \bigcap ML_k$; therefore, in analogy with the proof of Claim 4.14, $L_k \subseteq L_i$. □

Now, in order to prove that the positive conjuncts of φ are satisfied by M, consider a literal $\bigcap L_i \in \bigcap R_i$ with $i \in \{1, \ldots, p\}$. By Claim 4.14, $\bigcap ML_i = \mathcal{I}_i$ holds. In addition, by (4.13), $\mathcal{I}_i \in Mx$ for all $x \in R_i$, and therefore $\bigcap ML_i \in \bigcap MR_i$, which proves that M models correctly the conjunct $\bigcap L_i \in \bigcap R_i$.

Next we show that also the negative conjuncts of φ are satisfied by M. To see this, consider a literal $\bigcap L_j \notin \bigcap R_j$ with $j \in \{p+1, \ldots, n\}$. By way of contradiction, suppose that $\bigcap ML_j \in \bigcap MR_j$; then, by (4.13) and since $\bigcap ML_j$ is a set of rank grather then α, since $b_j \in \bigcap ML_k$, $\bigcap ML_j = \mathcal{I}_i$ for some $i \in \{1, \ldots, p\}$. Then $L_j = L_i$ holds by Claim 4.16, and $R_j \subseteq R_i$ holds by Claim 4.15, contradicting condition (a) of the theorem. Thus $\bigcap ML_j \notin \bigcap MR_j$, which proves that M models correctly the conjunct $\bigcap L_i \notin \bigcap R_i$. □

Complexity issues

We can assess the complexity of the satisfiability problem for $\mathsf{MST}(\cap, \in, \notin)$ as follows. Given an $\mathsf{MST}(\cap, \in, \notin)$-formula of the form (4.11), condition (a) of Theorem 4.12 can be tested in $\mathcal{O}(\varphi)$ time by detecting the duplicates in the list $L_1, \ldots, L_p$ of sets.

Next, we observe that condition (b) of Theorem 4.12 is equivalent to the acyclicity of the oriented graph $G_{\varphi^+} = (V_{\varphi^+}, E_{\varphi^+})$, where $V_{\varphi^+} \coloneqq \{1, \ldots, p\}$ and $E_{\varphi^+} \coloneqq \{(i, h) \mid L_i \subseteq R_h, \text{ for } i, h \in V_{\varphi^+}\}$. The graph G_{φ^+} can be constructed in time $\mathcal{O}(p \cdot |\varphi^+|)$ and its acyclicity tested in time $\mathcal{O}(p^2)$. Hence, condition (b) can be tested in time $\mathcal{O}(p \cdot |\varphi^+|)$, yielding an overall time complexity $\mathcal{O}(p \cdot |\varphi^+| + |\varphi|)$ for the satisfiability problem of $\mathsf{MST}(\cap, \in, \notin)$.

Summing up:

Lemma 4.17. *The satisfiability problem for $\mathsf{MST}(\cap, \in, \notin)$-conjunctions can be solved in quadratic time.*

4.2.4 The trivial fragment $\mathsf{MST}(\cup, \cap, \setminus, \notin)$

Finally we quickly disclose the fragment $\mathsf{MST}(\cup, \cap, \setminus, \notin)$.

Definition 4.6. *The fragment* $\mathsf{MST}(\cup, \cap, \setminus, \notin)$ *is the collection of all the conjunctions of literals of type*

$$t_1 \notin t_2,$$

where t_1 *and* t_2 *are terms involving the set operators* '$\cup$', '$\cap$', *and* '$\setminus$' *alongside set variables.*

Since any $\mathsf{MST}(\cup, \cap, \setminus, \notin)$-formula φ is trivially satisfied by the assignment $M_\emptyset$ over $Vars(\varphi)$, defined by

$$M_\emptyset x = \emptyset \text{ for every } x \in Vars(\varphi),$$

we readily get:

Lemma 4.18. *The satisfiability problem for the fragment* $\mathsf{MST}(\cup, \cap, \setminus, \notin)$ *can be solved in constant time.*

Remark 4.19. *By adding any symbol of the language* MST *to either* $\mathsf{MST}(\cup, \in, \notin)$, *or* $\mathsf{MST}(\cap, \in, \notin)$, *or* $\mathsf{MST}(\cup, \cap, \setminus, \notin)$, *we will obtain a fragment that contains one of the two* NP*-complete fragments* $\mathsf{MST}(\cup, \cap, \in)$ *or* $\mathsf{MST}(\setminus, \in)$. *On the other end by removing any symbol from the two aforementioned* NP*-complete fragments, we obtain a sub-language of a polynomial fragment.*

So $\mathsf{MST}(\cup, \in, \notin)$, $\mathsf{MST}(\cap, \in, \notin)$, *and* $\mathsf{MST}(\cup, \cap, \setminus, \notin)$ *are all the maximal polynomial fragments of* MST *and* $\mathsf{MST}(\cup, \cap, \in)$ *and* $\mathsf{MST}(\setminus, \in)$ *are all the minimal* NP*-complete fragments of* MST.

Before compiling the table listing the complete complexity taxonomy of MST we analyze one last fragment of the theory, that can be viewed either as a restriction over $\mathsf{MST}(\cup, \cap, \in)$ or an extension of $\mathsf{MST}(\cup, \in)$.

4.3 The fragment $\mathsf{MST}(\cup, \cap^R, \in)$, decidable in linear time

In Section 4.1.1 we proved that the fragment $\mathsf{MST}(\cup, \cap, \in)$ is NP-complete; now we shall prove that if we forbid the set intersection operator '$\cap$' to occur in the left-hand sides of membership conjuncts, the resulting fragment, denoted $\mathsf{MST}(\cup, \cap^R, \in)$, admits a linear-time satisfiability test.

Definition 4.7. *The fragment* $\mathsf{MST}(\cup, \cap^R, \in)$ *is the collection of all the conjunctions of literals of type*

$$t_1 \in t_2,$$

where t_1 *is a term that comprises only set variables and the set operator* '$\cup$' *while* t_2 *is a term comprising set variables and both the set operators* '$\cup$' *and* '$\cap$'.

Towards this direction, a preliminary immediate result is a consequence of the following straightforward equivalence

$$\models s \in s_1 \cap \cdots \cap s_k \longleftrightarrow s \in s_1 \wedge \cdots \wedge s \in s_k \tag{4.14}$$

that holds for any set terms $s, s_1, \ldots, s_k$. In the specific case in which all of $s, s_1, \ldots, s_k$ are $\cup$-terms, the equivalence (4.14) allows one to rewrite the literal $s \in s_1 \cap \cdots \cap s_k$ as the $\mathsf{MST}(\cup, \in)$-formula $s \in s_1 \wedge \cdots \wedge s \in s_k$. Thus, by repeatedly applying the equivalence (4.14), a conjunction φ of the form

$$\varphi = \bigwedge_{i=1}^{p} s_i \in \bigcap U_i, \tag{4.15}$$

where the s_i's are $\cup$-terms and the U_i's are finite nonempty collections of $\cup$-terms, can be rewritten as the equivalent $\mathsf{MST}(\cup, \in)$-formula

$$\varphi' \coloneqq \bigwedge_{i=1}^{p} \bigwedge_{t \in U_i} s_i \in t\,.$$

By recalling that the satisfiability problem for $\mathsf{MST}(\cup, \in)$ can be solved in linear time by the decision test reported in Algorithm 11 (see also Lemma 4.7), and by observing that $|\varphi'| = \mathcal{O}(|\varphi|^2)$, we conclude that the rewriting approach just described provides us with a quadratic decision procedure for the sub-collection $\mathsf{MST}'(\cup, \cap^R, \in)$ of $\mathsf{MST}(\cup, \cap^R, \in)$ consisting of all conjunctions φ of the form $\varphi = \bigwedge_{i=1}^{p} s_i \in \bigcap U_i$, where the s_i's are $\cup$-terms and the U_i's are finite nonempty collections of $\cup$-terms.

To further cement the relationship between this fragment and $\mathsf{MST}(\cup, \in)$, we provide the following example of an unsatisfiable conjunction.

Example 4.5. *The* $\mathsf{MST}(\cup, \cap^R, \in)$*-formula*

$$x_0 \cup x_1 \in (x_2 \cup x_3) \cap (x_4 \cup x_5) \quad \wedge \quad x_2 \cup x_3 \in x_0 \cup x_2,$$

is unsatisfiable.

Since the first conjunct can be rewritten as $x_0 \cup x_1 \in x_2 \cup x_3 \wedge x_0 \cup x_1 \in x_4 \cup x_5$, the above conjunction can be reduced to the unsatisfiable $\mathsf{MST}(\cup, \in)$-conjunction discussed in Example 4.1.

In the rest of this section, we shall provide a *linear* satisfiability test for $\mathsf{MST}(\cup, \cap^R, \in)$.

Given a nonempty $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ, we let $\mathcal{L}_\varphi$ denote the collection of the left-variables of φ, namely the variables occurring in the left-hand side s of any membership conjunct $s \in t$ in φ. Analogously, we let $\mathcal{R}_\varphi$ denote the collection of the right-variables of φ. More precisely, we put:

$$\mathcal{L}_\varphi \coloneqq \bigcup_{s \in t \text{ in } \varphi} \mathit{Vars}(s) \qquad \text{and} \qquad \mathcal{R}_\varphi \coloneqq \bigcup_{s \in t \text{ in } \varphi} \mathit{Vars}(t).$$

Plainly, $Vars(\varphi) = \mathcal{L}_\varphi \cup \mathcal{R}_\varphi$.

Then a necessary condition for the satisfiability of any $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ is that φ must have at least one *pure* right-variable, namely a right-variable that is not a left-variable:

Lemma 4.20. *Every nonempty satisfiable* $\mathsf{MST}(\cup, \cap^R, \in)$*-formula* φ *has a pure right-variable, namely* $\mathcal{R}_\varphi \setminus \mathcal{L}_\varphi \neq \emptyset$.

Proof. Let M be a model for φ, and let $\overline{x} \in Vars(\varphi)$ be such that $\mathsf{rk}(M\overline{x}) = \max\{\mathsf{rk}(Mx) \mid x \in Vars(\varphi)\}$. We claim that $\overline{x}$ is a pure right-variable. By way of contradiction, assume that $\overline{x}$ is a left-variable, and let $s \in t$ be any conjunct in φ such that $\overline{x}$ occurs in s. Then,

$$\mathsf{rk}(M\overline{x}) \leqslant \mathsf{rk}(Ms) < \mathsf{rk}(Mt) = \max\{\mathsf{rk}(Mx) \mid x \in Vars(t)\} \leqslant \mathsf{rk}(M\overline{x}),$$

a contradiction. Thus, $\overline{x}$ is a pure right-variable. □

Next, we describe a simplification procedure that, given an $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ with some pure right-variables and any variable $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$, produces a shorter $\mathsf{MST}(\cup, \cap^R, \in)$-formula $\varphi_{\overline{x}}$, equisatisfiable with φ, such that $Vars(\varphi_{\overline{x}}) \subseteq Vars(\varphi) \setminus \{\overline{x}\}$ holds. By repeatedly applying such simplification procedure till possible, one generates in a linear number of steps an $\mathsf{MST}(\cup, \cap^R, \in)$-formula $\overline{\varphi}$ of a special form which is equisatisfiable with the initial conjunction φ. Specifically, two cases are possible for $\overline{\varphi}$: either $\overline{\varphi}$ is just the empty conjunction, in which case $\overline{\varphi}$ is vacuously satisfiable[1] (and so φ can be declared satisfiable), or $\overline{\varphi}$ has no pure right-variable, namely $\mathcal{R}_{\overline{\varphi}} \subseteq \mathcal{L}_{\overline{\varphi}}$, in which case, by Lemma 4.20, $\overline{\varphi}$ is unsatisfiable (and therefore φ can accordingly be declared unsatisfiable). We shall also argue that the satisfiability test for $\mathsf{MST}(\cup, \cap^R, \in)$ just outlined admits an efficient linear-time implementation.

We shall make use of the following simple monotonicity property for set-assignments, in connection with $\{\cup, \cap\}$-terms. Specifically, given a collection V of set variables and two set assignments M_1 and M_2 over V, we write "$M_1 \subseteq_V M_2$" to mean that

$$M_1 x \subseteq M_2 x, \qquad \text{for every } x \in V. \tag{4.16}$$

As expected, the monotonicity property (4.16) extends to all $\{\cup, \cap\}$-terms over V, as proved in the next lemma.

Lemma 4.21 (Monotonicity)**.** *Let* M_1 *and* M_2 *be set assignments over a given collection* V *of set variables such that* $M_1 \subseteq_V M_2$*, and let* t *be any* $\{\cup, \cap\}$*-term over* V*. Then* $M_1 t \subseteq M_2 t$.

Proof. We proceed by structural induction on t. The base case in which t is just a variable is checked out by the very definition of $M_1 \subseteq_V M_2$.

Next, if $t = t' \cup t''$, for some $\{\cup, \cap\}$-terms t' and t'' over V, we have

$$M_1 t = M_1(t' \cup t'') = M_1 t' \cup M_1 t'' \subseteq M_2 t' \cup M_2 t'' = M_2(t' \cup t'') = M_2 t,$$

[1] We recall that the empty conjunction is satisfied by every set assignment.

as by inductive hypothesis $M_1 t' \subseteq M_2 t'$ and $M_1 t'' \subseteq M_2 t''$.

Analogously, if $t = t' \cap t''$, for some $\{\cup, \cap\}$-terms t' and t'' over V, then

$$M_1 t = M_1(t' \cap t'') = M_1 t' \cap M_1 t'' \subseteq M_2 t' \cap M_2 t'' = M_2(t' \cap t'') = M_2 t,$$

since, as before, by inductive hypothesis $M_1 t' \subseteq M_2 t'$ and $M_1 t'' \subseteq M_2 t''$. □

Corollary 4.22. *Let M be a set assignment over a given collection V of set variables, and let S be any set such that $Mx \subseteq S$, for every $x \in V$. Then, $Mt \subseteq S$, for every $\{\cup, \cap\}$-term t over V.*

Proof. Let M_S be the S-constant set assignment over V, namely the one such that $M_S x = S$, for each $x \in V$. Plainly, we have $M \subseteq_V M_S$. Hence, the claim follows at once from Lemma 4.21. □

Let φ be a (nonempty) satisfiable $\mathsf{MST}(\cup, \cap^R, \in)$-formula. By Lemma 4.20, φ has some pure right-variable $\overline{x}$. Let M be any set assignment over $\mathit{Vars}(\varphi)$ such that $M \models \varphi$, and let $D_M \coloneqq \bigcup_{x \in \mathit{Vars}(\varphi)} Mx$ be its 'active domain'. Consider the $\overline{x}$-variant $\overline{M}$ of M defined by

$$\overline{M}x \coloneqq \begin{cases} Mx & \text{if } x \in \mathit{Vars}(\varphi) \setminus \{\overline{x}\}, \\ D_M & \text{if } x = \overline{x}. \end{cases}$$

By Corollary 4.22, $\overline{M}t \subseteq D_M = \overline{M}\overline{x}$, for every $\{\cup, \cap\}$-term t over $\mathit{Vars}(\varphi)$. In addition, for every literal $s \in t$ in φ, we have:

- $Ms = \overline{M}s$ (since $\overline{x} \notin \mathit{Vars}(s)$),
- $Mt \subseteq \overline{M}t$ (by Lemma 4.21, since $M \subseteq_{\mathit{Vars}(\varphi)} \overline{M}$).

Hence, $\overline{M}s = Ms \in Mt \subseteq \overline{M}t$, so that $\overline{M} \models s \in t$, and therefore $\overline{M} \models \varphi$, in view of the arbitrariness of the conjunct $s \in t$ in φ.

Each subterm in φ of the form $\overline{x} \cup t$ or $t \cup \overline{x}$, for some term t, can be replaced by the variable $\overline{x}$ without disrupting the satisfiability by $\overline{M}$, since

$$\overline{M}(\overline{x} \cup t) = \overline{M}(t \cup \overline{x}) = \overline{M}\overline{x} \cup \overline{M}t = \overline{M}\overline{x}$$

($\cup$-*absorption rule* w.r.t. $\overline{x}$).

Dually, each subterm in φ of the form $\overline{x} \cap t$ or $t \cap \overline{x}$, for some term t, can be replaced by the term t without disrupting the satisfiability by $\overline{M}$, since

$$\overline{M}(\overline{x} \cap t) = \overline{M}(t \cap \overline{x}) = \overline{M}\overline{x} \cap \overline{M}t = \overline{M}t$$

($\cap$-*absorption rule* w.r.t. $\overline{x}$).

By repeatedly applying the $\cup$- and $\cap$-absorption rules to φ, in any order and until possible, one ends up with an $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ' that is satisfied

by $\overline{M}$ and whose conjuncts have one of the forms

$$s \in t \qquad \text{and} \qquad s \in \overline{x},$$

where s and t are terms over $\mathit{Vars}(\varphi) \setminus \{\overline{x}\}$.

Next, all literals of the form $s \in \overline{x}$ are dropped from φ' (*dropping rule* w.r.t. $\overline{x}$). We refer to the resulting (possibly empty) $\mathsf{MST}(\cup, \cap^R, \in)$-formula, denoted $\varphi_{\overline{x}}$, as *the $\overline{x}$-reduction* of φ[2]. Trivially, the inclusion $\mathit{Vars}(\varphi_{\overline{x}}) \subseteq \mathit{Vars}(\varphi) \setminus \{\overline{x}\}$ holds.

Summing up, we have:

(A) the $\overline{x}$-reduction $\varphi_{\overline{x}}$ of a satisfiable $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ with respect to any $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$ is also satisfiable, and

(B) $\mathit{Vars}(\varphi_{\overline{x}}) \subsetneq \mathit{Vars}(\varphi)$ (and $|\varphi_{\overline{x}}| < |\varphi|$).

The converse property holds too, namely for every $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ such that $\mathcal{R}_\varphi \setminus \mathcal{L}_\varphi \neq \emptyset$ and every variable $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$, if the $\overline{x}$-reduction $\varphi_{\overline{x}}$ of φ is satisfiable, so is φ.

In order to prove it, it is enough to show that the above property holds after each single application of any $\cup$- and $\cap$-absorption rule and each single removal of any literal of the form $s \in \overline{x}$, which we do next.

Let then φ be an $\mathsf{MST}(\cup, \cap^R, \in)$-formula and let $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$. Also let $\varphi_\cup$ be the conjunction resulting from replacing, in φ, a subterm $\bar{t}$ of the form $\overline{x} \cup u$ or $u \cup \overline{x}$ (for some term u) by the set variable $\overline{x}$. Thus, $\varphi = \varphi^z\{z \mapsto \bar{t}\}$ and $\varphi_\cup = \varphi^z\{z \mapsto \overline{x}\}$ for a suitable $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ^z, where z is any set variable not occurring in φ, $\{z \mapsto \bar{t}\}$ is the substitution mapping the variable z to the term $\bar{t}$, and $\{z \mapsto \overline{x}\}$ is the substitution mapping the variable z to the variable $\overline{x}$. Notice that φ^z contains just a single occurrence of the variable z and this is placed within a term in the right-hand side of some of its membership literals, so that $z \in \mathcal{R}_{\varphi^z} \setminus \mathcal{L}_{\varphi^z}$.

We claim that if $\varphi_\cup$ is satisfiable, so is φ. Indeed, let $M_\cup$ be any set assignment over $\mathit{Vars}(\varphi) \cup \{z\}$ such that $M_\cup \models \varphi_\cup$, and let $M^z_\cup$ and $M^{z,u}_\cup$ be its z-variants defined by

$$M^z_\cup x := \begin{cases} M_\cup x & \text{if } x \in \mathit{Vars}(\varphi)\,, \\ M_\cup \overline{x} & \text{if } x = z \end{cases}$$

and

$$M^{z,u}_\cup x := \begin{cases} M_\cup x & \text{if } x \in \mathit{Vars}(\varphi)\,, \\ M_\cup \overline{x} \cup M_\cup u & \text{if } x = z. \end{cases}$$

[2] In fact, it can be shown that the $\cup$- and $\cap$-absorption rules form a *confluent* system and therefore the $\overline{x}$-reduction of φ does not depend on the order in which such rules are applied to φ.

Plainly, $M^z_\cup \models \varphi^z$. Since $M^z_\cup \subseteq_{Vars(\varphi)\cup\{z\}} M^{z,u}_\cup$, then Lemma 4.21 yields $M^z_\cup t \subseteq M^{z,u}_\cup t$, for every $\{\cup, \cap\}$-term t over $Vars(\varphi) \cup \{z\}$. In addition, we plainly have $M^z_\cup s = M^{z,u}_\cup s$, for every $\cup$-term s over $Vars(\varphi)$. Thus, $M^{z,u}_\cup \models \varphi^z$, since

$$M^{z,u}_\cup s = M^z_\cup s \in M^z_\cup t \subseteq M^{z,u}_\cup t,$$

for every literal $s \in t$ in φ^z. But then $M^{z,u}_\cup \models \varphi$, since

$$M^{z,u}_\cup z = M_\cup \overline{x} \cup M_\cup u = M^{z,u}_\cup \overline{x} \cup M^{z,u}_\cup u = M^{z,u}_\cup \overline{t},$$

proving that φ is satisfiable.

Analogously, letting $\varphi_\cap$ be the conjunction resulting from replacing in φ a subterm $\overline{t}$ of the form $\overline{x} \cap u$ or $u \cap \overline{x}$ (for some term u) by the set term u, it can be proved that if $\varphi_\cap$ is satisfiable, so is φ.

Finally, let φ^- be the conjunction resulting after dropping from φ any literal of the form $s \in \overline{x}$. We claim that if φ^- is satisfiable, so is φ. Indeed, let M^- be any set assignment over $Vars(\varphi)$ satisfying φ^-, and let M^+ be its $\overline{x}$-variant defined by

$$M^+ x := \begin{cases} Mx & \text{if } x \in Vars(\varphi) \setminus \{\overline{x}\}, \\ M\overline{x} \cup Ms & \text{if } x = \overline{x}. \end{cases} \tag{4.17}$$

Since $Vars(s) \subseteq Vars(\varphi) \setminus \{\overline{x}\}$, we plainly have

$$M^+ s = Ms \in M\overline{x} \subseteq M^+ \overline{x},$$

proving that M^+ models correctly the literal $s \in \overline{x}$. In addition, since $M^- \subseteq_{Vars(\varphi)} M^+$ and $\overline{x} \notin Vars(s')$, for every conjunct $s' \in t'$ in φ^-, Lemma 4.21 yields

$$M^+ s' = M^- s' \in M^- t' \subseteq M^+ t',$$

and therefore, from the arbitrariness of $s' \in t'$ it follows that $M^+ \models \varphi^- \wedge s \in \overline{x}$, i.e., $M^+ \models \varphi$.

Having thus proved that an $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ is satisfiable if so is the result of either applying to it any $\cup$- or $\cap$-absorption rule w.r.t. to any variable $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$, or by dropping from it any literal of the form $s \in \overline{x}$, by a simple inductive argument we get that φ is satisfiable if so is its $\overline{x}$-reduction $\varphi_{\overline{x}}$.

We hence conclude:

(C) if the $\overline{x}$-reduction $\varphi_{\overline{x}}$ of φ w.r.t. to a variable $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$ is satisfiable, so is the conjunction φ.

Together with the previously proved assertions (A) and (B), assertion (C) yields the following result:

Lemma 4.23. *Let φ be an $\mathsf{MST}(\cup, \cap^R, \in)$-formula and $\overline{x}$ be any pure right-variable occurring in φ. Then φ and its $\overline{x}$-reduction $\varphi_{\overline{x}}$ are equisatisfiable. In addition, $Vars(\varphi_{\overline{x}}) \subseteq Vars(\varphi) \setminus \{\overline{x}\}$ (and $|\varphi_{\overline{x}}| < |\varphi|$).*

As argued above, Lemmas 4.20 and 4.21 allow us to conclude that the satisfiability test for $\mathsf{MST}(\cup, \cap^R, \in)$ in Algorithm 13 below is correct.

Algorithm 13 Satisfiability test for $\mathsf{MST}(\cup, \cap^R, \in)$-formula

Require: An $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ.
Ensure: Is φ satisfiable?
1: **while** φ is nonempty **do**
2: **if** $\mathcal{R}_\varphi \setminus \mathcal{L}_\varphi = \emptyset$ **then return** UNSATISFIABLE;
3: let $\overline{x} \in \mathcal{R}_\varphi \setminus \mathcal{L}_\varphi$;
4: $\varphi \leftarrow \varphi_{\overline{x}}$;
5: **return** SATISFIABLE

Algorithm 13 can easily be implemented to run in time $\mathcal{O}(|\varphi|^2)$. However, in the next section we shall present a more careful implementation that runs in linear time.

4.3.1 A linear-time implementation of Algorithm 13

For any $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ, Algorithm 14 below will perform on φ a sequence of satisfiability-preserving $\cup$- and $\cap$-absorption rules till either the current simplified conjunction, *which is maintained only implicitly by Algorithm 14 and is denoted* φ_c in this explanation, is either the empty conjunction or contains no pure right-variable.

To efficiently apply such rules, Algorithm 14 makes use of the following variables and data structures:

- a variable V, containing at each step of the computation the collection of the set variables occurring in the current conjunction φ_c, initialized to $\mathit{Vars}(\varphi)$ at line 1;
- a variable $\mathsf{pureRightOcc}$, containing at each step the collection of the pure right-variables in the current conjunction φ_c, initialized at lines 13–16;
- an array $\mathsf{MembCount}$ of counters (indexed by the set variables in $\mathit{Vars}(\varphi)$), intended to count for each $x \in \mathit{Vars}(\varphi)$ the distinct occurrences of the set variable x within the left-hand terms of the literals in φ_c; such counters are initialized at line 3 of the **for-loop** 2–4 and in the **for-loop** 9–10, while the literals of φ are scanned in the outer **for-loop** 5–12;
- an array $\mathsf{RightOccSet}$ of sets of pointers (indexed by the set variables in $\mathit{Vars}(\varphi)$), intended to group for each $x \in \mathit{Vars}(\varphi)$ all the occurrences of the set variable x within the right-hand terms of the literals in φ_c (via pointers); such sets are initialized at line 4 of the **for-loop** 2–4 and in the **for-loop** 6–8, while the literals of φ are scanned in the outer **for-loop** 5–12.

During the computation, the current simplified conjunction φ_c is internally maintained by Algorithm 14 as the implicit collection of the syntax trees of the right-hand terms of the literals in φ_c, which can be accessed only through the pointers to their variables contained in the set $\mathsf{RightOccSet}[x]$, for each $x \in \mathit{Vars}(\varphi_c)$. In addition, for each literal $s \in t$ in the current conjunction

φ_c, the set $Vars(s)$ (and, indirectly, the term s) is assigned to the attribute $\mathsf{MembTerm}$ of the syntax tree root of t. Such attributes are properly initialized at lines 11–12, within the **for-loop** 5–12, while the literals of the input conjunction φ are scanned.

Notice that the initialization phase (at lines 1–16 of Algorithm 14) takes linear time in $|\varphi|$, as can easily be checked.

The outer **while-loop** at lines 17–41 of Algorithm 14 is the core of the satisfiability test for the theory $\mathsf{MST}(\cup, \cap^R, \in)$. It performs a sequence of applications of the $\cup$-absorption rule (lines 31–37), of the $\cap$-absorption rule (lines 38–40), and of dropping rule (lines 22–30) till the current conjunction φ_c either reduces to the empty conjunction, namely when $\mathsf{V} = \emptyset$ holds (in which case the initial conjunction φ is declared SATISFIABLE at line 42), or has no pure right-variables despite being nonempty, namely when $\mathsf{V} \neq \emptyset$ and $\mathsf{pureRightOcc} = \emptyset$ hold (in which case φ is declared UNSATISFIABLE at line 19).

Specifically, for each pure right-variable x in the current conjunction φ_c (line 20), and while there are occurrences ν of x in the (implicit) syntax tree of φ_c (line 21), one of the following three possibilities must hold:

- The occurrence x is the right-hand term of a literal in φ_c of the form $s \in x$: in this case, ν is the root of the right-hand term of $s \in x$ and $\nu.\mathsf{MembTerm} = Vars(s)$. Then, the literal $s \in x$ is implicitly dropped from φ_c by executing the instructions at lines 23–30, which are intended to remove consistently the left-hand term s and the occurrence ν of x from the algorithm's data structures.
- The occurrence ν of x in φ_c lies within a $\cup$-term of the form $t \cup x$ or $x \cup t$, namely the father μ of ν has label '$\cup$' (cf. line 31): in this case the occurrence $t \cup x$ or $x \cup t$ in question is replaced by the occurrence ν of x (lines 32–37).
- The occurrence ν of x in φ_c lies within a $\cap$-term of the form $t \cap x$ or $x \cap t$, namely the father μ of ν has label '$\cap$' (cf. line 38): in this case the occurrence $t \cap x$ or $x \cap t$ in question is replaced by that of the subterm t, i.e., the sibling of ν (lines 39–40).

It is an easy matter to check that, during the execution of the outer **while-loop** at lines 17–41, the syntax tree of the input conjunction φ (implemented as a parent pointer tree) is read just once. In addition, all the sets present in Algorithm 14 (namely V, $\mathsf{pureRightOcc}$, $\mathsf{RightOccSet}[x]$, for $x \in Vars(\varphi)$, and the attributes $\mathsf{MembTerm}$ of the roots of the right-hand terms) can just be implemented as doubly-linked lists, so that each insertion, deletion, and access (through pointers) to their elements can be performed in constant time. Hence, the outer **while-loop** at lines 17–41 takes linear time in $|\varphi|$, as made clear in the below Remark 4.24, just as the initialization phase at lines 1–16, resulting in an overall linear time complexity of Algorithm 14.

Remark 4.24. *It is immaterial, for the assessment of the algorithmic complexity results which we will present, whether one measures the input formula as the*

length of the string of symbols representing it in the concrete syntax, or as the size of the labeled tree representing it in the abstract syntax. As a matter of fact, the decidable fragments which we will consider consist of formulae belonging to a language specifiable through a simple operator-precedence grammar. In view of the simplicity of that language, the translation of formulae from their external syntax to their internal syntax (e.g. by means of Edsger Dijkstra's shunting yard algorithm) and the converse translation can both be carried out in linear time.

4.4 A complete complexity taxonomy of MST

We have identified the maximal polynomial and the minimal NP-complete sub-languages of the fragment MST of set theory, having in mind applications in automated proof checking with verifiers rooted in the set-theoretic formalism. On the basis of the reported results, summarized in Table 4.2, one easily pin-points the 14 polynomial fragments (of which 2 are quadratic, 4 are linear, and 8 are constant) and the 10 NP-complete fragments among the 24 sub-languages of MST.

$\cup$	$\cap$	$\setminus$	$\in$	$\notin$	Complexity
★			★	★	$\mathcal{O}(n)$
★			★		$\mathcal{O}(n)$
			★	★	$\mathcal{O}(n)$
			★		$\mathcal{O}(n)$
	★		★	★	$\mathcal{O}(n^2)$
	★		★		$\mathcal{O}(n^2)$
★	★	★		★	constant
★		★		★	constant
★	★			★	constant
	★	★		★	constant
★				★	constant
		★		★	constant
	★			★	constant
				★	constant

$\cup$	$\cap$	$\setminus$	$\in$	$\notin$	Complexity
		★	★		NP-complete
★		★	★		NP-complete
	★	★	★		NP-complete
★	★	★	★		NP-complete
		★	★	★	NP-complete
★		★	★	★	NP-complete
	★	★	★	★	NP-complete
★	★	★	★	★	NP-complete
★	★		★		NP-complete
★	★		★	★	NP-complete

TABLE 4.2: Complete taxonomy of MST, maximal polynomial and minimal NP-complete fragments are in black

Algorithm 14 Linear-time implementation of a satisfiability test for $\mathsf{MST}(\cup, \cap^R, \in)$-formulae

Require: An $\mathsf{MST}(\cup, \cap^R, \in)$-formula φ.
Ensure: Is φ satisfiable?

```
1: V ← Vars(φ);
2: for each variable x in V do
3:     MembCount[x] ← 0;
4:     RightOccSet[x] ← ∅;
5: for each conjunct s ∈ t in φ do
6:     for each leaf λ in the syntax tree of the term t do
7:         let z be the variable labeling λ;
8:         insert λ into the set RightOccSet[z];
9:     for each set variable x in the term s do
10:        MembCount[x] ← MembCount[x] + 1;
11:    let ρ_t be the root of the syntax tree of the term t;
12:    assign the set Vars(s) to ρ_t.MembTerm, namely to ρ_t's attribute
       MembTerm;
13: pureRightOcc ← ∅;
14: for each variable x in V do
15:    if MembCount[x] = 0 then
16:        insert x into the set pureRightOcc;
17: while V ≠ ∅ do
18:    if pureRightOcc = ∅ then
19:        return UNSATISFIABLE;
20:    for each variable x in pureRightOcc do
21:        while there is some node ν in the set RightOccSet[x] do
22:            if ν is a root then
23:                for each variable y in ν.MembTerm do
24:                    MembCount[y] ← MembCount[y] − 1;
25:                    if MembCount[y] = 0 then
26:                        if RightOccSet[y] ≠ ∅ then
27:                            insert y into the set pureRightOcc;
28:                        else
29:                            remove y from V;
30:                remove ν from the set RightOccSet[x];
31:            else if the father μ of ν has label '∪' then
32:                for each leaf λ in the subtree rooted at the sibling of ν do
33:                    let z be the variable labeling λ;
34:                    remove λ from the set RightOccSet[z];
35:                    if RightOccSet[z] = ∅ and MembCount[z] = 0 then
36:                        remove z from V;
37:                replace μ by ν (if μ was a root, ν inherits its MembTerm
                   attribute);
38:            else if the father μ of ν has label '∩' then
39:                remove ν from the set RightOccSet[x];
40:                replace μ by the sibling ν′ of ν (if μ was a root, ν′ inherits
                   the MembTerm attribute of μ);
41:        remove x from the sets pureRightOcc and V;
42: return SATISFIABLE;
```

Chapter 5

The complete taxonomy of Multi-Level Syllogistic

The ultimate goal of our analysis of the two taxonomies of BST and MST is to lay strong foundations to a complete taxonomy of their mother theory Multi-Level Syllogistic, MLS.

Following the same approach as before, we forgo to study the thousands of possible fragments comprising both equality ' = ', and membership ' ∈ ' symbols, and instead focus to find the *minimal* NP-*complete* and *maximal polynomial* fragments of the language. By their very nature, minimal NP-complete fragments of MST or BST will still be minimal NP-complete fragments inside MLS, and, as we will see, so far our investigation lacks just one minimal NP-complete fragment of MLS. However the same can not be said for maximal polynomial fragments, by extending the language, it is possible, and in fact it is, to extend those fragments with symbols that where not available in the previous language and still obtain a polynomial fragment.

At the end of this chapter we will have filled the gaps between the two taxonomies of BST and MST and obtained the taxonomy of the more comprehensive MLS.

First we will present the last minimal NP-complete, then we extend the maximal polynomial fragment presented so far, merging together the results for BST and MST, obtaining a complete complexity taxonomy of MLS.

The syntax and semantics of MLS have already been presented in Section 1.1, and will not be repeated here. In the same fashion of the tables presented for MST and BST, we will assemble a table to store the minimal NP-complete and maximal polynomial fragments of MLS that represent the whole taxonomy of the language.

∪	∩	\	=∅	≠∅	Disj	¬Disj	⊆	⊈	=	≠	∈	∉	Complexity

Table 5.1: Early look at the table of containing the MLS taxonomy

5.1 The NP-complete fragment MLS(∪, Disj, ∈)

Definition 5.1. *The fragment* $\mathsf{MLS}(\cup, \mathsf{Disj}, \in)$ *is the collection of all the conjunctions of literals of the following two types*

$$t_1 \in t_2, \quad \mathsf{Disj}\,(t_1, t_2)\,,$$

where t_1 *and* t_2 *are terms involving only the set operator '* $\cup$ *' and set variables.*

As before we shall prove that it is possible to reduce the satisfiability problem 3SAT into of the satisfiability for $\mathsf{MLS}(\cup, \mathsf{Disj}, \in)$, proving so the NP-complete the latter problem.

Let F be a generic instance of 3SAT, and let $x, X_1, \overline{X}_1, \ldots, X_n, \overline{X}_n$ be $2n+1$ distinct set variables.

For all $i \in \{1, \ldots, m\},\ j \in \{1, 2, 3\}$, and $k \in \{1, \ldots, n\}$, put

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

then put

$$\Phi_F \coloneqq \bigwedge_{i=1}^{m} (x \in T_{i1} \cup T_{i2} \cup T_{i3}) \ \wedge \ \bigwedge_{k=1}^{n} \left(x \in X_k \cup \overline{X}_k \wedge \mathsf{Disj}\left(X_k, \overline{X}_k\right)\right).$$

Lemma 5.1. *A 3SAT instance* F *is propositionally satisfiable if and only if the corresponding* $\mathsf{MLS}(\cup, \mathsf{Disj}, \in)$*-formula* Φ_F *is satisfiable.*

Proof. (Sufficiency). Let M be a set assignment that satisfies Φ_F. Then for each $k \in \{1, \ldots, n\}$ we have that:

$$Mx \in MX_k \cup M\overline{X}_k \wedge \mathsf{Disj}\left(MX_k, M\overline{X}_k\right).$$

Thus, either we have $Mx \in MX_k \wedge Mx \notin M\overline{X}_k$ or $Mx \notin MX_k \wedge Mx \in M\overline{X}_k$.

We can define the following propositional valuation $\mathfrak{v}$:

$$\mathfrak{v}(P_k) \coloneqq \begin{cases} \mathsf{true} & \text{if } Mx \in MX_k \\ \mathsf{false} & \text{if } Mx \notin MX_k. \end{cases}$$

To prove that $\mathfrak{v}$ satisfies F, it is enough to prove that for each $i \in \{1, \ldots, m\}$ there exists a $j \in \{1, 2, 3\}$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$.

For each $i \in \{1, \ldots, m\}$ we have $Mx \in MT_{i1} \cup MT_{i2} \cup MT_{i3}$. Therefore, there must exists a $j \in \{1, 2, 3\}$ such that $Mx \in MT_{ij}$. If $T_{ij} = X_k$ for some k, so that $L_{ij} = P_k$, then $Mx \in MX_k$ and therefore $\mathfrak{v}(L_{ij}) = \mathsf{true}$. Otherwise, if $T_{ij} = \overline{X}_k$, so that $L_{ij} = \neg P_k$, then $Mx \in M\overline{X}_k$ and therefore $Mx \notin MX_k$. Hence $\mathfrak{v}(P_k) = \mathsf{false}$ and so $\mathfrak{v}(L_{ij}) = \mathsf{true}$, concluding the sufficient part of our proof.

(Necessity). Let $\mathfrak{v}$ be a propositional valuation that satisfies F and let s be any sets. Define the following set assignment:

$$Mx \coloneqq s, \quad MX_k \coloneqq \begin{cases} \{s\} & \text{if } \mathfrak{v}(P_k) = \mathsf{true} \\ \emptyset & \text{otherwise,} \end{cases} \quad M\overline{X}_k = \{s\} \backslash MX_k \text{ for } k = 1, \ldots, n.$$

Plainly, by the very definition of M, $Mx \in MX_k \cup M\overline{X}_k$ and $\mathsf{Disj}\left(MX_k, M\overline{X}_k\right)$ hold for all $k \in \{1, \ldots, n\}$.

It remains to prove that for all $i \in \{1, \ldots, m\}$, $Mx \in MT_{1j} \cup MT_{i2} \cup MT_{i3}$, that is there exists a $j \in \{1, 2, 3\}$ such that $Mx \in MT_{ij}$. For all $i \in \{1, \ldots, m\}$ there exists a $j \in \{1, 2, 3\}$ such that $\mathfrak{v}(L_{ij}) = \mathsf{true}$. If $L_{ij} = P_k$ for some k, then $T_{ij} = X_k$ and $MX_k = \{s\}$ so that $Mx \in MT_{ij}$. If otherwise $L_{ij} = \neg P_k$ then, $T_{ij} = \overline{X}_k$, and $\mathfrak{v}(P_k) = \mathsf{false}$. Hence $MX_k = \emptyset$ and $M\overline{X}_k = \{s\}$ thus $Mx \in MT_{ij}$, proving that $M \models \Phi_F$ and in turn concluding the proof. □

The formula Φ_F can be easily built in $\mathcal{O}(|F|)$-time. Thus Lemma 5.1 readily yields:

Lemma 5.2. *The satisfiability problem for* $\mathsf{MLS}(\cup, \mathsf{Disj}, \in)$ *belongs to the class of* NP*-complete problems.*

5.2 Maximal polynomial fragments of MLS

The maximal polynomial fragments of BST are those fragments that only admit NP-complete super theories inside BST, or in other words all the polynomial super theories of those fragments must contain symbols outside BST. Following this reasoning to find the maximal polynomial fragments of MLS it is enough to take the maximal polynomial fragments of BST and add to them ' $\in$ ' or ' $\notin$ '. Of course we could take maximal MST fragments and add to them the symbols that MST lacks, but in such study we would consider adding more symbols than just ' $\in$ ' and ' $\notin$ '.

Moreover all the existential expressibility result for a given fragment are also true for all its super-languages, and although the same is not true also for $\mathcal{O}(f)$-expressibility results, for what concern us we can also extend the $\mathcal{O}(f)$-expressibility of the atom $x = \varnothing$ to all the extension inside MLS of both $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$. Summing up we just need to study the fragments $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$, $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$, and $\mathsf{BST}(\cup, \mathsf{Disj}, \neg\mathsf{Disj}, \neq, \notin)$.[1]

5.2.1 The theory BST ∪ {' ∉ '}

The theory BST ∪ {' ∉ '}, is obtained by adding the symbol ' ∉ ' to BST or alternatively is the collection of all the conjunctions of literals ℓ that are either a BST-literal or a literal of type $s \notin t$ where s and t are both terms involving only set variables and the set operators symbols ' $\setminus$ ', ' $\cup$ ', and ' $\cap$ '.

[1] ' $\in$ ' must be taken out since $\mathsf{MLS}(\cup, \mathsf{Disj}, \in)$ was already proven NP-complete in Section 5.1

In order to prove that the atom $x \in y$ is not existentially expressible in BST, in Chapter 2 we proved Lemma 2.11 that states that any satisfiable BST-formula admits a ρ-flat model. As a direct consequence of that lemma we have:

Corollary 5.3. *Any* $\mathsf{BST} \cup \{`\notin'\}$*-formula* φ *is satisfiable if and only if its* BST*-component* φ^-*, that is the conjunction of all its* BST*-literals, is satisfiable.*

Proof. The necessity part of the corollary is trivial. Concerning the sufficient part assume φ^- is satisfiable. Then by Lemma 2.11 it has a ρ-flat model M, that is a set assignment such that $s \in Mv \longrightarrow \mathsf{rk}(s) = \rho$ for all $v \in Vars(\varphi^-)$. Notice that extending M to all the variables $x \in Vars(\varphi) \setminus Vars(\varphi^-)$ by setting $Mx = \emptyset$ for all $x \in Vars(\varphi) \setminus Vars(\varphi^-)$, M remains a ρ-flat assignment. It is a simple matter to prove by structural induction that, for each term t of φ, Mt is either empty or as rank $\rho+1$ and contains sets of rank ρ only. Then for each literal of type $s \notin t$ we have that $\mathsf{rk}(s) \in \{0, \rho+1\}$ and either $Mt = \emptyset$ or Mt has only set of rank ρ meaning that $Ms \notin Mt$, therefore $M \models \varphi$. □

The above corollary states that when checking the satisfiability of any $\mathsf{BST} \cup \{`\notin'\}$-formula, we can forgo all the literals of type $s \notin t$ and solve the satisfiability problem just for its BST-component. In particular any formula of $\mathsf{MLS}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq, \notin)$ is satisfiable if and only if its $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \mathsf{Disj},$ component is satisfiable. Since the satisfiability problem for $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq$ can be solved in cubic time, we have:

Lemma 5.4. *The satisfiability problem for* $\mathsf{MLS}(\cup, =\varnothing, \neq\varnothing, \mathsf{Disj}, \mathsf{Disj}, \neg\mathsf{Disj}, \not\subseteq, \neq, \notin)$ *can be solved in cubic time.*

By the very same reasoning we also have that

Lemma 5.5. *The satisfiability problem for both* $\mathsf{MLS}(\cup, \cap, \setminus, =\varnothing, \mathsf{Disj}, \subseteq, =, \notin)$ *and* $\mathsf{MLS}(\cup, \cap, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, =, \notin)$ *can be solved in constant time.*

5.3 The fragment $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$

Definition 5.2. *The fragment* $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ *is the collection of all the conjunctions of literals of the following types*

$$\bigcup L \in \bigcup R, \quad \bigcup L \notin \bigcup R, \quad \bigcup L = \bigcup R, \quad \bigcup L \neq \bigcup R,$$

where L*, and* R *are nonempty collections of set variables.*

As before, for every given $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula φ, it is convenient to define the collections

$$\begin{aligned}
\Phi^{=}_{\varphi} &:= \{\{L, R\} \mid \bigcup L = \bigcup R \text{ is in } \varphi\},\\
\Phi^{\in}_{\varphi} &:= \{\langle L, R\rangle \mid \bigcup L \in \bigcup R \text{ is in } \varphi\},\\
\Phi^{\neq}_{\varphi} &:= \{\{L, R\} \mid \bigcup L \neq \bigcup R \text{ is in } \varphi\},\\
\Phi^{\notin}_{\varphi} &:= \{\langle L, R\rangle \mid \bigcup L \notin \bigcup R \text{ is in } \varphi\},
\end{aligned}$$

where the $\{L, R\}$'s and $\langle L\,,\ R\rangle$'s are respectively unordered and ordered pairs of collections of nonempty set variables.

To prove that the satisfiability problem for $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ is solvable in polynomial time, we present a polynomial-time reduction to the satisfiability problem for $\mathsf{MLS}(\cup, =, \neq, \notin)$, which is equivalent to the satisfiability problem of $\mathsf{BST}(\cup, =, \neq)$ by Corollary 5.3.

Starting from $\Phi^{=}_{\varphi}$, we define the equivalence relation $\sim_{\varphi}$ in the same fashion as in Definition 3.9 and the $\sim_{\varphi}$-closure $\overline{V}$ of a given collection of set variables $V \subseteq \mathit{Vars}(\varphi)$, as $\overline{V} \coloneqq \bigcup\{W \mid W \sim_{\varphi} V\}$. plainly all the properties of the relation $\sim_{\varphi}$ and the closure operator $\bar{\cdot}$ seen in Section 3.2.1 still holds.

For each collection of set variables L such that $\langle L\,,\ R\rangle \in \Phi^{\in}_{\varphi}$ define the collection $\mathcal{N}_L$ as

$$\mathcal{N}_L \coloneqq \{x \mid (\exists\, \langle U\,,\ V\rangle \in \Phi^{\notin}_{\varphi})\ x \in \overline{V} \wedge \overline{L} = \overline{U}\}. \tag{5.1}$$

Retracing the proof we used to prove that the satisfiability problem for $\mathsf{MST}(\cup, \in)$ can be solved in linear-time, we can define the collection of *non-left* set variables similarly to the definition of *pure right variable* of Section 4.2.2.

Definition 5.3. *Given an $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula φ, we say that a set variable $x \in \mathit{Vars}(\varphi)$ is* ***non-left*** *if*

$$x \notin L, \quad \textit{for all } \langle L\,,\ R\rangle \in \Phi^{\in}_{\varphi}.$$

Then we denote by $\mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi}$, the largest collection of φ such that

(A1) all the variables inside $\mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi}$ are non-left;

(A2) $\mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi} \neq \emptyset \longrightarrow R \cap \mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi} \neq \emptyset$ and $\mathsf{Disj}\left(\mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi}, \mathcal{N}_L\right)$ hold;

(A3) $(\forall \{U, V\} \in \Phi^{=}_{\varphi})\ U \cap \mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi} \neq \emptyset \longleftrightarrow V \cap \mathcal{A}^{\langle L\,,\,R\rangle}_{\varphi} \neq \emptyset$.

Lemma 5.6. *If an $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula φ is satisfiable, then there exists a pair $\left\langle \hat{L}\,,\ \hat{R}\right\rangle$ such that $\mathcal{A}^{\left\langle \hat{L}\,,\,\hat{R}\right\rangle}_{\varphi}$ is not empty.*

Proof. Let M be a model for φ and let the pair $\left\langle \hat{L}\,,\ \hat{R}\right\rangle \in \Phi^{\in}_{\varphi}$ be such that

$$(\forall\, \langle L\,,\ R\rangle \in \Phi^{\in}_{\varphi})\ \mathsf{rk}\,(\bigcup ML) \leqslant \mathsf{rk}\left(\bigcup M\hat{L}\right). \tag{5.2}$$

Also let $\mathcal{M} \coloneqq \{x \mid \bigcup M\hat{L} \in Mx\}$. Notice that $\mathcal{M}$ is not empty since $M \models \bigcup \hat{L} \in \bigcup \hat{R}$.

First assume that there exists a variable $x \in \mathcal{M}$ that is not non-left. Then there must exist a pair $\langle L'\,,\ R'\rangle \in \Phi^{\in}_{\varphi}$ such that $x \in L'$. But then $\mathsf{rk}\left(\bigcup M\hat{L}\right) < \mathsf{rk}\,(Mx) \leqslant \mathsf{rk}\,(\bigcup ML')$, contradicting (5.2), thus proving that $\mathcal{M}$ satisfies (A1).

Concerning (A3), let $\{L, R\} \in \Phi^{=}_{\varphi}$ and assume $\mathcal{M} \cap L \neq \emptyset$. Then there exists an $x \in L$ such that $\bigcup M\hat{L} \in Mx \subseteq \bigcup ML$. Since $M \models \varphi$, we have $\bigcup ML = \bigcup MR$.

Thus, there exists a variable $y \in R$ such that $\bigcup M\hat{L} \in My$ so that $R \cap \mathcal{M} \neq \emptyset$. By the generality of $\{L, R\}$ in $\Phi^{=}_{\varphi}$ and of L in $\{L, R\}$, we have that $\mathcal{M}$ fulfills (A3).

Finally, since $\bigcup M\hat{L} \in \bigcup M\hat{R}$, there exists $R \cap \mathcal{M} \neq \emptyset$. By contradiction assume that there exists a variable $x \in \mathcal{N}_L \cap \mathcal{M}$. Then there exists a pair $\langle U\,,\, V\rangle \in \Phi^{\notin}_{\varphi}$ such that $x \in \overline{V}$ and $\overline{U} = \hat{\overline{L}}$.

Since $M \models \varphi$, then M also models its $\mathsf{BST}(\cup, =, \neq)$-component. Thus, by Lemma 3.15, if $A \sim_{\varphi} B$ then $\bigcup MA = \bigcup MB$. Since $M \models \varphi$, then we also have that $\bigcup M\hat{L} = \bigcup MU \notin \bigcup MV = \bigcup M\overline{V}$. Therefore $\bigcup ML \notin y$ for each $y \in \overline{V}$. But $x \in \overline{V}$ and $\bigcup M\hat{L} \in Mx$, a contradiction. Therefore $\mathcal{M}$ fulfills also (A2).

$\mathcal{M}$ satisfies all (A1), (A3), and (A2) hence is a subset of $\mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$, proving that $\mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ is nonempty. □

Let $\langle L\,,\, R\rangle \in \Phi^{\in}_{\varphi}$ be such that $\mathcal{A}_{\varphi}^{\langle L\,,\, R\rangle}$ is nonempty. We define φ^{-} as the result of deleting from φ the literal $\bigcup L \in \bigcup R$.

Theorem 5.7. *Let φ be a $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula. Then φ is satisfiable if and only if there exists a pair $\left\langle \hat{L}\,,\, \hat{R}\right\rangle \in \Phi^{\in}_{\varphi}$ such that $\mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ is nonempty and φ^{-} is satisfiable.*

Proof. (Sufficiency). Let $\left\langle \hat{L}\,,\, \hat{R}\right\rangle \in \Phi^{\in}_{\varphi}$ such that $\mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ is nonempty and let φ^{-} be satisfiable. Also let M_0 be a model for φ. First we will provide another model for φ^{-} imbued with useful properties.

Let α be a rank large enough so that we can assign to each collection $V \in \mathscr{P}(\mathit{Vars}(\varphi))$ a distinct nonempty set b_V of rank α. For each $v \in \mathit{Vars}(\varphi)$, define

$$\hat{M}v := \{b_{\overline{V}} \mid v \notin \overline{V}\} \cup \{\bigcup \hat{M}L \mid \bigcup M_0L \in M_0x\}.$$

We preliminarily prove that $\hat{M} \models \varphi^{-}$.

First notice that $\emptyset \in \mathscr{P}(\mathit{Vars}(\varphi))$ and it is actually easy to prove that the $\sim_{\varphi}$-closure of the $\emptyset$ is the emptyset itself. Thus, we have $b_{\emptyset} \in Mx$, for all $x \in \mathit{Vars}(\varphi)$, so that $\mathsf{rk}\,(Mx) > \alpha$.

Let $A, B \subseteq \mathit{Vars}(\varphi)$ and $\overline{A} \neq \overline{B}$. Then $\bigcup \hat{M}A \neq \bigcup \hat{M}B$. In fact, w.l.o.g., we may assume that $\overline{A} \not\subseteq \overline{B}$. Then, by Lemma 3.14(f), $A \not\subseteq \overline{B}$, and by definition of $\sim_{\varphi}$-closure $B \subseteq \overline{B}$, $b_{\overline{B}} \in \bigcup MA \setminus \bigcup MB$.

Let $\{L, R\} \in \Phi^{\neq}_{\varphi^{-}}$. Since $M_0 \models \varphi^{-}$, $\overline{L} \neq \overline{R}$ by Theorem 3.17. Thus $\bigcup \hat{M}L \neq \bigcup \hat{M}R$.

Let $\langle L\,,\, R\rangle \in \Phi^{\in}_{\varphi^{-}}$. Then there exists an $x \in R$ such that $\bigcup M_0L \in M_0x$, so that $\bigcup \hat{M}L \in \hat{M}x \subseteq \bigcup \hat{M}R$.

Let $\{L, R\} \in \Phi^{=}_{\varphi^{-}}$. We prove that $\bigcup \hat{M}L \subseteq \bigcup \hat{M}R$ and analogously $\bigcup \hat{M}R \subseteq \bigcup \hat{M}L$, so that $\bigcup \hat{M}L = \bigcup \hat{M}R$.

Let $s \in \bigcup \hat{M}L$. Then either $\mathsf{rk}(s) = \alpha$ or $\mathsf{rk}(s) > \alpha$. If $\mathsf{rk}(s) = \alpha$, then there exists a $V \subseteq \mathit{Vars}(\varphi)$ such that $L \not\subseteq \overline{V}$. Thus, by Lemma 3.14 (c) and (f), $\overline{L} = \overline{R} \not\subseteq \overline{V}$ and $R \not\subseteq \overline{V}$, so that $s = b_{\overline{V}} \in \bigcup \hat{M}R$. If $\mathsf{rk}(s) > \alpha$, then there exists a $V \in \mathit{Vars}(\varphi)$ such that $s = \bigcup \hat{M}V$, and $\bigcup M_0 V \in \bigcup M_0 L = \bigcup M_0 R$ thus $s = \bigcup \hat{M}V \in \bigcup \hat{M}R$.

Let $\langle L\,,\, R\rangle \in \Phi^{\notin}_{\varphi^-}$, and assume by contradiction that $\bigcup \hat{M}L \in \bigcup \hat{M}R$. Then there exists an $x \in R$ such that $\bigcup \hat{M}L \in \hat{M}x$, so that there exists a $V \subseteq \mathit{Vars}(\varphi)$ such that $\bigcup \hat{M}V = \bigcup \hat{M}L$ and $\bigcup M_0 V \in M_0 x$. But then we have $\overline{L} = \overline{V}$, so that $\bigcup M_0 V = \bigcup M_0 L \notin \bigcup M_0 R$. Thus $\bigcup M_0 V \notin M_0 x$, a contradiction, and therefore $\bigcup \hat{M}L \notin \bigcup \hat{M}R$.

We will now modify the set assignment $\hat{M}$ in order to obtain a model M for φ. For each $x \in \mathit{Vars}(\varphi)$, let us put

$$Mx \coloneqq \hat{M}x \cup \{\bigcup M\hat{L} \mid x \in \mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}\}.$$

Let $\{L, R\} \in \Phi^{\neq}_{\varphi}$. Then following, the same reasoning used for $\hat{M}$, we have $\bigcup ML \neq \bigcup MR$.

Let $\langle L\,,\, R\rangle \in \Phi^{\in}_{\varphi}$. Notice that by (A1) we have $\mathsf{Disj}\left(L, \mathcal{A}_{\varphi}^{\langle L\,,\, R\rangle}\right)$. Thus $\bigcup ML = \bigcup \hat{M}L$. If $\langle L\,,\, R\rangle \in \varphi^-$, then $\bigcup \hat{M}L \in \bigcup \hat{M}R \subseteq \bigcup MR$. Otherwise if $\langle L\,,\, R\rangle = \left\langle \hat{L}\,,\, \hat{R}\right\rangle$ then there exists an $x \in \hat{R} \cap \mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ and therefore $\bigcup M\hat{L} \in \bigcup Mx \subseteq \bigcup M\hat{R}$.

Let $\{L, R\} \in \Phi^{=}_{\varphi}$. Then, by $\bigcup \hat{M}L = \bigcup \hat{M}R$ and (A3), $\bigcup ML = \bigcup MR$.

Let $\langle L\,,\, R\rangle \in \Phi^{\notin}_{\varphi}$. By way of contradiction assume that $\bigcup ML \in \bigcup MR$. Then there exists an $x \in R$ such that $\bigcup ML \in Mx$. Either we have $\bigcup ML \in \hat{M}x$ or we have $x \in \mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ and $\bigcup ML = \bigcup M\hat{L}$. In the first case $\bigcup ML \in \hat{M}x$, there exists a $V \subseteq \mathit{Vars}(\varphi)$ such that $\bigcup ML = \bigcup \hat{M}V$. If $\overline{L} \neq \overline{V}$ then the sets $\bigcup \hat{M}L$ and $\bigcup \hat{M}V$ would be differentiated by a set of rank α. Thus, also $\bigcup ML$ and $\bigcup \hat{M}V$ are differentiated by the same set of rank α. Notice that the rank of $\bigcup M\hat{L}$ is greater then α. Thus we have that $\overline{L} = \overline{V}$.

Recalling that $\hat{M} \models \varphi^-$, then $\bigcup \hat{M}L = \bigcup \hat{M}V$, since $L \sim_{\varphi} V$. Thus $\bigcup \hat{M}L \in \hat{M}x \subseteq \bigcup \hat{M}R$, a contradiction.

In the case in which $x \in \mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ and $\bigcup ML = \bigcup M\hat{L}$, as before we have $\overline{L} = \overline{\hat{L}}$ and thus plainly $x \in \mathcal{N}_{\hat{L}}$, contradicting (A2). Therefore we must have $\bigcup ML \notin \bigcup MR$, completing the sufficient part of the theorem.

(Necessity). Let φ be satisfiable. Then, by Lemma 5.6, there exists a pair $\left\langle \hat{L}\,,\, \hat{R}\right\rangle \in \Phi^{\in}_{\varphi}$ such that $\mathcal{A}_{\varphi}^{\langle \hat{L}\,,\, \hat{R}\rangle}$ is nonempty. Then trivially φ^- is satisfied by any model of φ. □

Remark 5.8. *The necessity part of Theorem 5.7 can be strengthened. Trivially, if a formula φ is satisfiable then so are also all of its sub-formulae, so that for*

all the pairs $\langle L, R \rangle \in \Phi_\varphi^\in$ *such that* $\mathcal{A}_\varphi^{\langle L, R \rangle}$ *is nonempty we have that* φ^- *is satisfiable. Conversely if we found a single pair* $\langle L, R \rangle$ *such that* $\mathcal{A}_\varphi^{\langle L, R \rangle}$ *is nonempty but* φ^- *is unsatisfiable then the whole formula* φ *would be unsatisfiable.*

5.3.1 A polynomial satisfiability test

The goal of our satisfiability test is that of trying to reduce, using Theorem 5.7, any $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula to an equisatisfiable $\mathsf{MLS}(\cup, =, \neq, \notin)$-formula. Then, in view of Corollary 5.3, we can simply check the satisfiability of the $\mathsf{BST}(\cup, =, \neq)$-component of the reduced formula. This strategy leads to the fairly easy approach described below:

Algorithm 15 Strategy for $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ satisfiability

step 1 Find a pair $\langle L, R \rangle \in \Phi_\varphi^\in$ such that $\mathcal{A}_\varphi^{\langle L, R \rangle} \neq \emptyset$;
step 2 If no such pair is found, return false;
step 3 Remove $\bigcup L \in \bigcup R$ from φ; if $\Phi_\varphi^\in$ is not empty go to **step 1**;
step 4 Check for satisfiability of the $\mathsf{BST}(\cup, =, \neq)$ part of φ;

Algorithm 16 is an instance of the strategy illustrated in Algorithm 15. Although much more implementation details can be added, we can use it to obtain an upper bound to the time complexity for the satisfiability test of $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$.

Let n be the length of a given $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula φ, m be the number of literals of φ, and $p = |\mathit{Vars}(\varphi)|$. We will express our upper bound to the time complexity of Algorithm 16 in terms of n, since plainly $p, m \leqslant n$.

Preliminarily, we observe that, by using Algorithm 9, in time $\mathcal{O}(n^2)$ we can generate the collections $\Phi_\varphi^\in$, $\Phi_\varphi^\notin$, and $\Phi_\varphi^=$, an index of all the variables of φ, and we can associate to all the collection of variables that appears in φ an array of size p such that $A[x] = 1 \iff x \in A$.

By representing any collection of set variables V as an array of size p, will can perform each relevant operation involving collection of set variable in $\mathcal{O}(p)$-time, and each operation involving a set variable an a collection of set variables in constant time.

The first phase of Algorithm 16 consists in computing the collection $\mathcal{N}_L$ for each pair $\langle L, R \rangle \in \Phi_\varphi^\in$ (lines 1 to 5). Two nested **for**-loops are used, each one iterated at most m-times. The initialization of each collection $\mathcal{N}_L$ requires $\mathcal{O}(p)$-time, and computing the $\sim_\varphi$-closure of any given collection requires $\mathcal{O}(n^2)$ time by using Algorithm 2. Thus, this phase takes at most $\mathcal{O}(n^4)$ time.

Then the collection NON-LEFT of the non-left variables of φ is built. In particular, NON-LEFT$[x]$ is the number of pairs $\langle L, R \rangle \in \Phi_\varphi^\in$ such that $x \in L$. Initializing NON-LEFT requires $\mathcal{O}(p)$-time. To compute NON-LEFT$[x]$ for each x, we scan each variable $y \in L$ for all pairs $\langle L, R \rangle \in \Phi_\varphi^\in$ and increment

Algorithm 16 Satisfiability test for $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$

Require: An $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula φ
Ensure: is φ satisfiable

for each pair $\langle L, R \rangle \in \Phi_\varphi^\in$ **do**
  $\mathcal{N}_L \longleftarrow \emptyset$;
  for each pair $\langle U, V \rangle \in \Phi_\varphi^\notin$ **do**
    if $\overline{U} = \overline{L}$ **then**
      $\mathcal{N}_L \longleftarrow \mathcal{N}_L \cup \overline{T}$;
Let NON-LEFT be an array of size $|Vars(\varphi)|$;
Initialize NON-LEFT so that for each $x \in Vars(\varphi)$ NON-LEFT$[x] = 0$;
for each pair $\langle L, R \rangle \in \Phi_\varphi^\in$ **do**
  for each $x \in L$ **do**
    NON-LEFT$[x] \longleftarrow$ NON-LEFT$[x] + 1$;
while $\Phi_\varphi^\in \neq \emptyset$ **do**
  $m \longleftarrow |\Phi_\varphi^\in|$;
  for each pair $\langle L, R \rangle \in \Phi_\varphi^\in$ **do**
    $\mathcal{A}_\varphi^{\langle L, R \rangle} \longleftarrow \emptyset$
    for all $x \in Vars(\varphi)$ **do**
      if NON-LEFT$[x] = 0$ and $x \notin \mathcal{N}_L$ **then**
        $\mathcal{A}_\varphi^{\langle L, R \rangle} \longleftarrow \mathcal{A}_\varphi^{\langle L, R \rangle} \cup \{x\}$;
    while exists a pair $\{U, V\} \in \Phi_\varphi^=$ such that $\mathsf{Disj}\left(U, \mathcal{A}_\varphi^{\langle L, R \rangle}\right) \wedge \neg\mathsf{Disj}\left(V, \mathcal{A}_\varphi^{\langle L, R \rangle}\right)$ or $\mathsf{Disj}\left(V, \mathcal{A}_\varphi^{\langle L, R \rangle}\right) \wedge \neg\mathsf{Disj}\left(U, \mathcal{A}_\varphi^{\langle L, R \rangle}\right)$ **do**
      $\mathcal{A}_\varphi^{\langle L, R \rangle} \longleftarrow \mathcal{A}_\varphi^{\langle L, R \rangle} \setminus U \cup V$;
    if R and $\mathcal{A}_\varphi^{\langle L, R \rangle}$ are not disjoint **then**
      for each $x \in L$ **do**
        NON-LEFT$[x] \longleftarrow$ NON-LEFT$[x] - 1$;
      remove $\langle L, R \rangle$ from $\Phi_\varphi^\in$;
  if $m = |\Phi_\varphi^\in|$ **then**
    return false;
remove each literal $\bigcup L \notin \bigcup R$ from φ;
return SATISFIABILITY(φ);

NON-LEFT$[y]$ each time y is found. Hence the collection NON-LEFT can be built in time $\mathcal{O}(mp)$.

The **while**-loop at lines 11-25 reduces the formula φ into an equisatisfiable $\mathsf{MLS}(\cup, =, \neq, \notin)$ formula. At each iteration, if no pair of $\Phi^{\in}_{\varphi}$ is discarded the algorithm halts. Thus the **while**-loop can be iterated at most m times. Nested inside this **while**-loop we find a **for**-loop that is iterated once for each pair of $\Phi^{\in}_{\varphi}$, for a total of m times. During each of these m iterations, we initialize the collection $\mathcal{A}^{\langle L,\, R\rangle}_{\varphi}$ so that it contains all the non-left variables that are also not in $\mathcal{N}_L$. This step requires $\mathcal{O}(mp)$-time. Subsequently, we must ensure that $\mathcal{A}^{\langle L,\, R\rangle}_{\varphi}$ fulfills condition (A3). To do so, we look for a pair $\{U, V\} \in \Phi^{=}_{\varphi}$ that does not satisfy condition (A3) and remove all the variables of $U \cup V$ from $\mathcal{A}^{\langle L,\, R\rangle}_{\varphi}$, repeating until (A3) is fulfilled. Since $\mathcal{A}^{\langle L,\, R\rangle}_{\varphi}$ does not get any new variable, once a pair $\{U, V\}$ is chosen during this cycle, it will never be chosen again. Thus in order to fulfill (A3) we need to iterate this loop at most $\mathcal{O}(m^2)$-times, where each iteration requires $\mathcal{O}(p)$ time. Thus, the **while**-loop at lines 11-25 can be completed in $\mathcal{O}(n^5)$ steps.

Finally, each literal of type $\bigcup L \notin \bigcup R$ can be removed in constant time, by discarding $\Phi^{\notin}_{\varphi}$ and then checking if the $\mathsf{BST}(\cup, =, \neq)$-formula so obtained is satisfiable. Using Algorithm 1, such a check can be performed in $\mathcal{O}(n^3)$ time.

Lemma 5.9. *The satisfiability problem for* $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ *can be solved in* $\mathcal{O}(n^5)$ *time.*

5.4 The fragment $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$

Definition 5.4. *The fragment* $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$ *is the collection of all the conjunctions of literals of the following types*

$$\bigcap L \in \bigcap R, \quad \bigcap L \notin \bigcap R, \quad \bigcap L = \bigcap R, \quad \bigcap L \neq \bigcap R,$$

where L and R are nonempty collections of set variables.

The satisfiability problem for any $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$-formula φ is easily reducible to the satisfiability of both $\mathsf{BST}(\cap, =, \neq)$ and $\mathsf{MST}(\cap, \in, \notin)$ by exploiting the properties of the $\sim_{\varphi}$-equivalence and its associated closure operator.

First, we define the collection of pairs:

$$\begin{aligned}
\Phi^{=}_{\varphi} &:= \{\{L, R\} \mid \textstyle\bigcap L = \bigcap R \text{ is in } \varphi\},\\
\Phi^{\in}_{\varphi} &:= \{\langle L\,,\, R\rangle \mid \textstyle\bigcap L \in \bigcap R \text{ is in } \varphi\},\\
\Phi^{\neq}_{\varphi} &:= \{\{L, R\} \mid \textstyle\bigcap L \neq \bigcap R \text{ is in } \varphi\},\\
\Phi^{\notin}_{\varphi} &:= \{\langle L\,,\, R\rangle \mid \textstyle\bigcap L \notin \bigcap R \text{ is in } \varphi\},
\end{aligned}$$

where as always the $\{L, R\}$'s and $\langle L\,,\, R\rangle$'s are respectively unordered and ordered pairs of nonempty collections of set variables, and define the $\sim_{\varphi}$ equivalence and its closure operator as in Definitions 3.9 and 3.10.

Using the $\sim_\varphi$-closure operator, we generate from any $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$-formula φ the following $\mathsf{MST}(\cap, \in, \notin)$-formula $\overline{\varphi}$

$$\overline{\varphi} \coloneqq \bigwedge_{\langle L\,,\,R\rangle \in \Phi_\varphi^{\in}} \overline{L} \in \overline{R} \;\wedge \bigwedge_{\langle L\,,\,R\rangle \in \Phi_\varphi^{\notin}} \overline{L} \notin \overline{R}.$$

Finally we define φ^- as the $\mathsf{BST}(\cup, =, \neq)$ component of φ.

Theorem 5.10. *Any given* $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$*-formula is satisfiable if and only if the* $\mathsf{MST}(\cap, \in, \notin)$*-formula* $\overline{\varphi}$ *and the* $\mathsf{BST}(\cap, =, \neq)$*-component* φ^- *of* φ *are satisfiable.*

Proof. (Sufficiency). Let both $\overline{\varphi}$ and φ^- be satisfiable. Then by Theorem 3.18 $\overline{L} \neq \overline{R}$, for all $\{L, R\} \in \Phi_\varphi^{\neq}$. Let $\overline{M}$ be a model for $\overline{\varphi}$, and assign to each collection in $\mathscr{P}(\mathit{Vars}(\varphi))$ a different set b_V of rank α. Then define the following set assignment:

$$Mx \coloneqq \{b_{\overline{V}} \mid x \in \overline{V}\} \cup \{\bigcap ML \mid (\exists\, \langle\, L\,,\, R\,\rangle \in \Phi_\varphi^{\in}) \bigcap \overline{ML} \in \bigcap \overline{MR} \wedge x \in \overline{R}\},$$

for each $x \in \mathit{Vars}(\varphi)$.

First notice that, for every $V \subseteq \mathit{Vars}(\varphi)$ we have $V \subseteq \overline{\mathit{Vars}(\varphi)}$. Thus $b_{\overline{\mathit{Vars}(\varphi)}} \in \bigcap MV$, so that $\mathsf{rk}\,(\bigcap MV) > \alpha$. We will now show that $M \models \varphi$.

Let $\{L, R\} \in \Phi_\varphi^{\neq}$i. Then we have $\overline{L} \neq \overline{R}$. W.l.o.g., we may assume that $\overline{R} \not\subseteq \overline{L}$. Thus, $L \subseteq \overline{L}$ by definition of $\sim_\varphi$-closure, and therefore $b_{\overline{L}} \in \bigcap ML \setminus \bigcap MR$, therefore $M \models \bigcap L \neq \bigcap R$.

Let $\langle\, L\,,\, R\,\rangle \in \Phi_\varphi^{\in}$. Then $\bigcap \overline{ML} \in \bigcap \overline{MR}$ and $R \subseteq \overline{R}$, and so $\bigcap ML \in \bigcap MR$.

Let $\{L, R\} \in \Phi_\varphi^{=}$. We will prove that $\bigcap ML \subseteq \bigcap MR$ and similarly $\bigcap MR \subseteq \bigcap ML$ hold, so that $M \models \bigcap L = \bigcap R$. Let $s \in \bigcap ML$. Then either $\mathsf{rk}\,(s) = \alpha$ or $\mathsf{rk}\,(s) > \alpha$. If $\mathsf{rk}\,(s) = \alpha$, there exists a $V \subseteq \mathit{Vars}(\varphi)$ such that $L \subseteq \overline{V}$. Then, by Lemma 3.14(f), we have $\overline{L} \subseteq \overline{V}$, and since $\overline{R} = \overline{L}$, $R \subseteq \overline{V}$ so that $b_{\overline{V}} = s \in \bigcap MR$ holds.

On the other hand, if $\mathsf{rk}\,(s) > \alpha$ then there exists a pair $\langle\, L'\,,\, R'\,\rangle \in \Phi_\varphi^{\in}$ such that $s = \bigcap ML'$, $L \subseteq \overline{R'}$, and $\bigcap \overline{ML'} \in \bigcap \overline{MR'}$. Then, as before, we have that $R \subseteq \overline{R'}$, and so $s = \bigcap ML' \in \bigcap MR$.

Finally, let $\langle\, L\,,\, R\,\rangle \in \Phi_\varphi^{\notin}$. By way of contradiction, assume $\bigcap ML \in \bigcap MR$. Then there must exist a pair $\langle\, L'\,,\, R'\,\rangle \in \Phi_\varphi^{\in}$ such that: $\bigcap ML = \bigcap ML'$, $R \subseteq \overline{R'}$, and $\bigcap \overline{ML'} \in \bigcap \overline{MR'}$. First notice that since $\bigcap ML = \bigcap ML'$ we have $b_{\overline{L}} \in \bigcap ML'$. Thus $L' \subseteq \overline{L}$, and by Lemma 3.14(f) $\overline{L'} \subseteq \overline{L}$. Analogously, $\overline{L} \subseteq \overline{L'}$ since $b_{\overline{L'}} \in \bigcap ML$, so that $\overline{L} = \overline{L'}$. Then we must have that $\bigcap \overline{ML} = \bigcap \overline{ML'} \in \bigcap \overline{MR'}$, and since $R \subseteq \overline{R'}$ we have $\overline{R} \subseteq \overline{R'}$ so that $\bigcap \overline{MR'} \subseteq \overline{R}$, and therefore $\bigcap \overline{ML} \in \bigcap \overline{MR}$. Since $\langle\, L\,,\, R\,\rangle \in \Phi_\varphi^{\notin}$, then $\bigcap \overline{L} \notin \bigcap \overline{R}$ in $\overline{\varphi}$, thus we must have $\bigcap \overline{ML} \notin \bigcap \overline{MR}$, a contradiction, and therefore we must have $\bigcap ML \notin \bigcap MR$.

(Necessity). Let φ be satisfiable and let M be a model for it. Plainly, $M \models \varphi^-$. Thus, by Lemma 3.15, we have that $\bigcap MV = \bigcap M\overline{V}$ for each $V \subseteq \mathit{Vars}(\varphi)$.

Therefore, for each $\langle L\,,\ R\rangle \in \Phi^{\in}_{\varphi}$, we have

$$\bigcap M\overline{L} = \bigcap ML \in \bigcap MR = \bigcap M\overline{R},$$

and analogously, for every pair $\langle L\,,\ R\rangle \in \Phi^{\notin}_{\varphi}$, we have

$$\bigcap M\overline{L} = \bigcap ML \notin \bigcap MR = \bigcap M\overline{R}.$$

Thus $M \models \overline{\varphi}$, concluding the proof of the theorem. □

Using Algorithm 2, it is possible to compute the $\sim_{\varphi}$-closure of any given collection of set variables in $\mathcal{O}(n^2)$ time, where n is the length of the formula φ. We can assemble the $\mathsf{MST}(\cap, \in, \notin)$-formula $\overline{\varphi}$ in $\mathcal{O}(n^3)$-time, then we can check the satisfiability of $\overline{\varphi}$ and φ^{-} using the satisfiability tests for $\mathsf{MST}(\cap, \in, \notin)$ and $\mathsf{BST}(\cap, =, \neq)$, which take respectively $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$ time. We can then conclude:

Lemma 5.11. *The satisfiability test for* $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$ *can be solved in cubic time.*

5.5 Extending $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ and $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$

At the end of Section 3.2 we extended $\mathsf{BST}(\cup, =, \neq)$ and $\mathsf{BST}(\cap, =, \neq)$ to $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ and $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$ respectively using the expressibility results presented in Chapter 2. Similarly, in this section, we first prove that the literal $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ and $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$.

Lemma 5.12. *The literal* $x = \varnothing$ *is* $\mathcal{O}(n)$*-expressible in* $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$.

Proof. Following Definition 2.2, we need to find a map from $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ into another fragment that fulfills conditions (a) to (c). We will use the same map seen in Lemma 2.7 thus letting φ be any $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula, we put

$$\langle \varphi\,,\ x = \varnothing \rangle \mapsto \bigwedge_{v \in Vars(\varphi)} x \subseteq v.$$

Much as in Lemma 2.7, such map can be computed in $\mathcal{O}(n)$-time. For short, we put

$$\Xi^{x}_{\varphi} \coloneqq \bigwedge_{v \in Vars(\varphi)} x \subseteq v.$$

Concerning (b), let $\varphi \wedge \Xi^{x}_{\varphi}$. Notice that the literal $x \subseteq y$ is existentially expressible in $\mathsf{BST}(\cup, =, \neq)$ and a fortiori in $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$. Thus, there exists a $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$-formula φ' such that

$$\models \varphi \wedge \Xi^{x}_{\varphi} \longleftrightarrow \varphi',$$

proving that φ' is satisfiable.

In Theorem 5.7 we showed that any satisfiable $\mathsf{MLS}(\cup,=,\neq,\in,\notin)$ formula admits a model M' such that, for each variable v, $M'v$ as rank strictly grater then a certain rank α, so that for every $s \in M'v$, either $\mathsf{rk}(s) = \alpha$ or there exists a collection V of variables such that $s = \bigcup M'V$. Moreover we have that $\overline{V} = \overline{V'} \longleftrightarrow \bigcup M'V = \bigcup M'V'$, for all collections of variables V and V'.

Let M' be such a model for φ', so that $M' \models \varphi \wedge \Xi^x_\varphi$. First notice that $\mathsf{rk}(M'x) = \alpha + 1$, otherwise we would have a collection of variables V such that $\bigcup M'V \in M'x$ and since $M' \models \Xi^x_\varphi\ M'x \subseteq \bigcup M'V$ we would contradict the axiom of regularity.

Now, define the set assignment M by putting

$$Mv \coloneqq M'v \setminus M'x \cup \{\bigcup ML \mid \bigcup L \in \bigcup R \in \varphi' \wedge \bigcup M'L \in M'v\},$$

for $v \in \mathit{Vars}(\varphi')$.

Let $\bigcup L = \bigcup R$ in φ'. Then $\bigcup M'L = \bigcup M'R$. Let $s \in \bigcup ML$. Then either $s \in \bigcup M'L \setminus Mx$ or $s = \bigcup ML'$ for some L'. In the first case, we readily have $s \in \bigcup M'R \setminus Mx \subseteq \bigcup MR$, whereas in the latter case we have that $\bigcup M'L' \in \bigcup M'L = \bigcup M'R'$ so that $\bigcup ML' \in \bigcup MR$. Hence, $\bigcup ML \subseteq \bigcup MR$. Analogously we can prove $\bigcup MR \subseteq \bigcup MR$, and therefore $M \models \bigcup L = \bigcup R$ holds.

Let $\bigcup L \neq \bigcup R$ be in φ'. By definition of M', there exists a set s of rank α that belongs to either $\bigcup M'L \setminus \bigcup M'R$ or to $\bigcup M'R \setminus \bigcup M'L$, and since $M' \models \Xi^x_\varphi$ we have $Mx \subseteq \bigcup ML$ and $Mx \subseteq \bigcup MR$. Thus $s \notin Mx$ and therefore either s is in $\bigcup Ml \setminus \bigcup MR$ or is in $\bigcup MR \setminus \bigcup ML$. In either cases $M \models \bigcup L \neq \bigcup R$ holds.

We can prove that M also models the membership literals of φ' by the same arguments used in the proof of Theorem 5.7.

Finally notice that since $\mathsf{rk}(M'x) = \alpha + 1$, then $Mx = \emptyset$ so that $M \models \varphi'$. Therefore $M \models \varphi$, and also $M \models x = \varnothing$, proving (b).

Concerning (c) it is enough to notice that $\emptyset \subseteq s$, for all sets s. Thus, any model for $\varphi \wedge x = \varnothing$ models also $\varphi \wedge \Xi^x_\varphi$. □

Analogously, we can also prove that $x = \varnothing$ is $\mathcal{O}(n)$-expressible in $\mathsf{MLS}(\cap,=,\neq,\in,\notin)$.

We can merge the above results with others already known:

(a) the literal $x \subseteq y$ is existentially expressible in $\mathsf{BST}(\cup,=)$ and in $\mathsf{BST}(\cap,=)$;

(b) the literal $x \nsubseteq y$ is existentially expressible in $\mathsf{BST}(\cup,\neq)$ and in $\mathsf{BST}(\cap,\neq)$;

(c) the literal $\mathsf{Disj}(x,y)$ is existentially expressible in $\mathsf{BST}(\cap,=\varnothing)$ and therefore, by (a), it is also $\mathcal{O}(n)$-expressible in $\mathsf{BST}(\cap,=)$;

(d) the literal $\neg\mathsf{Disj}(x,y)$ is existentially expressible in $\mathsf{BST}(\subseteq,\neq)$; therefore, by (a), it is also existentially expressible in both of $\mathsf{BST}(\cup,=,\neq)$ and $\mathsf{BST}(\cap,=,\neq)$.

Therefore we can reduce in linear time any formula of the two languages $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq, \in, \notin)$ and $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq, \in, \notin)$ into $\mathsf{MLS}(\cup, =, \neq, \in, \notin)$ and $\mathsf{MLS}(\cap, =, \neq, \in, \notin)$ respectively. Thus, we have:

Lemma 5.13. *The satisfiability decision problem for both* $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq, \in, \notin)$ *and* $\mathsf{BST}(\cap, =\varnothing, \neq\varnothing, \mathsf{Disj}, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq, \in, \notin)$ *can be solved in polynomial time.*

5.6 A complete complexity taxonomy for MLS

In this chapter we explored the fragments of Multi-Level Syllogistic. Following the same strategies used for BST and MST, we spotted the minimal NP-complete and the maximal polynomial fragments of MLS. Table 5.2 lists all such fragments and can be used to establish, given any fragment of MLS, whether or not its satisfiability problem can be solved in polynomial time.

$\cup$	$\cap$	$\setminus$	$=\varnothing$	$\neq\varnothing$	Disj	$\neg$Disj	$\subseteq$	$\not\subseteq$	$=$	$\neq$	$\in$	$\notin$	Complexity
		★									★		NP-complete
		★								★			NP-complete
		★						★					NP-complete
		★				★							NP-complete
		★		★									NP-complete
★	★										★		NP-complete
★	★									★			NP-complete
★	★							★					NP-complete
★	★		★	★									NP-complete
★	★				★	★							NP-complete
★	★		★			★							NP-complete
★	★			★	★								NP-complete
★					★						★		NP-complete
★					★	★			★				NP-complete
★				★	★				★				NP-complete
★					★	★	★						NP-complete
★					★				★	★			NP-complete
★					★			★	★				NP-complete
★				★	★		★						NP-complete
★					★		★			★			NP-complete
★					★		★	★					NP-complete
★			★	★	★	★		★		★		★	$\mathcal{O}(n^3)$
★			★	★		★	★	★	★	★	★	★	$\mathcal{O}(n^5)$
	★		★	★	★	★	★	★	★	★	★	★	$\mathcal{O}(n^3)$
★	★	★	★		★		★		★			★	$\mathcal{O}(1)$
★	★			★		★	★		★			★	$\mathcal{O}(1)$

TABLE 5.2: Complete taxonomy of theory MLS

Chapter 6

Combining MLS with non-set theoretic theories

In the previous chapters we proved that the satisfiability problem for MLS belongs to the class of NP-complete problems, and we presented several fragments of MLS imbued with a deterministic polynomial-time decision procedures.

Here we will show that MLS and its fragments also enjoy a useful property that in favorable cases allow us to combine efficiently (i.e., in deterministic polynomial time) their decision procedures, with those of other theories outside Set Theory. This result was first published in [CDDM21].

6.1 Combining decision procedures and the Nelson-Oppen method

In the process of developing reliable and provably correct software, it is often necessary to express and then subsequently verify properties that belong to different logical languages. Thus, the correctness of a software system depends on being able to prove these conditions, expressed in distinct first-order signatures with equality. The search for a satisfying assignment of a given formula with respect to some background first-order theory is known as the SMT (Satisfiability Modulo Theories) problem.

The Satisfiability Modulo Theories (SMT) problem concerns the satisfiability of formulae from multiple background theories, usually expressed in the language of first-order predicate logic with equality. SMT solvers are often based on variants of the Nelson-Oppen combination method, a solver for the quantifier-free fragment of the combination of theories with disjoint signatures, via cooperation among their decision procedures.

SMT solvers [BST10] are particularly useful tools for the automated verification of properties expressed with quantifier-free first-order formulae. Some theories usually integrated with common SMT solvers are the theory of arrays, of bit-vectors, of linear arithmetic, and the theory of uninterpreted functions. Every background theory used in some SMT solver comes along with its own satisfiability procedure. The problem of modularly combining such special-purpose

algorithms is highly non-trivial, since without the appropriate restrictions it is not even decidable [BGN+06].

We will now briefly introduce some definitions to understand how to tackle the SMT problem and under which assumptions one can do it effectively.

Definition 6.1 (Stably Infinitess)**.** *A first-order quantifier-free theory T, identified with the set of its theorems, is* stably infinite *if every formula φ satisfiable in T is satisfiable in an infinite model of T.*

It is not difficult to prove that MLS is stably infinite.

Definition 6.2. *Let Σ_1 and Σ_2 be signatures for a first-order language. A $(\Sigma_1 \cup \Sigma_2)$-formula φ is* pure *if every literal in φ is either a Σ_1-literal or a Σ_2-literal.*

It is easy to see that every quantifier-free $(\Sigma_1 \cup \Sigma_2)$-formula φ can be *purified*, yet maintaining satisfiability, by

(i) substituting every *impure* subterm of the form $f(t)$ with $f(x)$, where x is a new variable,

(ii) adding to φ the conjunct $x = t$, and

(iii) recursively purifying the term t, if needed.

Definition 6.3. *We say that two theories T_1 and T_2 over the signatures Σ_1 and Σ_2, respectively, are* disjoint *when Σ_1 and Σ_2 do not share any non-logical symbols.*

The Nelson-Oppen [NO79] procedure provides a method for combining decision procedures for disjoint, stably infinite theories T_1 and T_2 into one for $T_1 \oplus T_2$, namely the $(\Sigma_1 \cup \Sigma_2)$-theory defined as the deductive closure of the union of the theories T_1 and T_2.

Given two disjoint stable infinite theories T_1 and T_2, the Nelson-Oppen combination technique establishes the satisfiability of a conjunction of pure formulae $\varphi_1 \wedge \varphi_2$ (where φ_i has signature Σ_i) in $T_1 \oplus T_2$ from the decision procedures for φ_1 and φ_2. The key idea is to propagate equalities $x = y$ to φ_2 whenever $T_1 \cup \varphi_1$ implies $x = y$, and conversely. Usually this iterative process is nondeterministic, however it can be performed quickly in deterministic polynomial time, when the theories involved are *convex*.

Definition 6.4. *A theory T is* convex *if for all conjunctions of literals φ in T and, for all nonempty disjunctions $\bigvee_{i=1}^{n} x_i = y_i$ of equalities, φ implies $\bigvee_{i=1}^{n} x_i = y_i$ in T if and only if φ implies $x_i = y_i$ in T for some $i \in \{1, ..., n\}$.*

Examples of convex theories are the theory of Linear Rational Arithmetic $\mathsf{T}_{\mathsf{LRA}}$ and the theory of list structure T_{L}.

The non-logical symbols of the theory of $\mathsf{T}_{\mathsf{LRA}}$ are $+$, $-$, $\leqslant$, 0, 1; following [BM07, Chapter 3.4.2], its axioms (universally quantified) are:

$$\begin{array}{ll} x+0=x, & x+(-x)=0, \\ (x+y)+z=x+(y+z), & x+y=y+x, \\ x\leqslant y\wedge y\leqslant x\to x=y, & x\leqslant y\vee y\leqslant x, \\ x\leqslant y\to x+z\leqslant y+z, & x\leqslant y\wedge y\leqslant z\to x\leqslant z, \\ nx=0\to x=0, & (\exists y)\ x=ny \qquad \text{(for each positive integer } n\text{)}, \end{array}$$

where nx stands for $\underbrace{x+\cdots+x}_{n \text{ times}}$.

After [NO79], the non-logical symbols of the theory of list structure T_{L} are car, cdr, cons, and atom, and its axioms are:

$$\begin{array}{l} \mathsf{car}(\mathsf{cons}(x,y))=x, \\ \mathsf{cdr}(\mathsf{cons}(x,y))=y, \\ \neg\mathsf{atom}(x)\to\mathsf{cons}(\mathsf{car}(x),\mathsf{cdr}(x))=x, \\ \neg\mathsf{atom}(\mathsf{cons}(x,y)), \end{array}$$

where (i) cons is a binary function, with $\mathsf{cons}(x,y)$ representing the list constructed by prepending the object x to the list y, (ii) car and cdr are unary functions, the left and right projections, respectively, and (iii) atom is true if and only if x is a single-element list.

In what follows we will prove that when it is suitably represented conjuntively also the theory MLS is convex. Specifically, we restrict to the case in which to MLS is represented as the collection of all the conjunctions of atoms of the following types:

$$x=y\setminus z, \quad x\in y. \tag{6.1}$$

6.2 Convexity of MLS

Our main goal is to prove that the theory MLS is convex, namely that, for any MLS-conjunction φ and any given finite *nonempty* set $\mathcal{E}$ of equalities among variables, we have:

$$\models \varphi\longrightarrow\bigvee\mathcal{E} \quad\Longrightarrow\quad \models\varphi\longrightarrow x=y, \text{ for some equality } x=y \text{ in } \mathcal{E}.$$

To prove that the theory MLS is convex, we will proceed by way of contradiction.

Thus, let us suppose that there exists an MLS-conjunction φ (namely a conjunction of literals of type (6.1)) and a finite, nonempty set $\mathcal{E}$ of equalities among variables such that:

(C1) $\models\varphi\longrightarrow\bigvee\mathcal{E}$;

(C2) $\not\models \varphi \longrightarrow x = y$, for every $x = y$ in $\mathcal{E}$
(that is, for every $x = y$ in $\mathcal{E}$ there exists some set assignment $M_{x,y}$ such that $M_{x,y} \models \varphi \wedge x \neq y$).

It is not restrictive to additionally assume that $\mathit{Vars}(\mathcal{E}) \subseteq \mathit{Vars}(\varphi)$.[1]

In view of condition (C2), our formula φ is satisfiable. Among all the models for φ, we select one, say M, that satisfies as few as possible equalities in $\mathcal{E}$, namely such that the cardinality of $\mathcal{E}^+_M := \{\ell \in \mathcal{E} \mid M \models \ell\}$ is *minimal*. We also set $\mathcal{E}^-_M := \{\neg\ell \mid \ell \in \mathcal{E} \setminus \mathcal{E}^+_M\}$, so $\mathcal{E}^-_M$ is the collection of the inequalities $x \neq y$ such that $x = y$ is in $\mathcal{E}$ and $M \not\models x = y$ (hence, $M \models x \neq y$).

Plainly, we have $M \models \varphi \ \wedge \ \bigwedge \mathcal{E}^+_M \ \wedge \ \bigwedge \mathcal{E}^-_M$. Notice that, while $\bigwedge \mathcal{E}^-_M$ may be empty, the conjunction $\bigwedge \mathcal{E}^+_M$ is guaranteed to contain at least one literal by condition (C1).

Let $\overline{\ell}$ be any equality $\overline{x} = \overline{y}$ in $\bigwedge \mathcal{E}^+_M$, which will be referred to in the rest of our proof as the *designated equality of* $\mathcal{E}$. We will prove that the conjunction

$$\varphi^* \ := \varphi \ \wedge \ \bigwedge \left(\mathcal{E}^+_M \setminus \{\overline{\ell}\}\right) \ \wedge \ \bigwedge \mathcal{E}^-_M \ \wedge \ \overline{x} \neq \overline{y}$$

is satisfiable, thereby contradicting the assumed minimality of M, since for every model M^* for φ^* we would have $\mathcal{E}^+_{M^*} = \mathcal{E}^+_M \setminus \{\overline{\ell}\}$, and therefore $|\mathcal{E}^+_{M^*}| < |\mathcal{E}^+_M|$.

Before diving into the details of the proof, we provide an overview of how the set assignment M can be suitably enlarged into another set assignment M^* that satisfies all the conjuncts of $\varphi \wedge \bigwedge \mathcal{E}^+_M \wedge \bigwedge \mathcal{E}^-_M$ but the designated equality $\overline{\ell}$, thus proving that φ^* is satisfiable.

Proof overview

The construction of M^* consists in two phases: the first phase, the BOOLEAN PHASE, takes care of the satisfiability of the Boolean literals of φ^*, namely the literals in φ^* of type $x = y \setminus z$, $x = y$, and $x \neq y$, whereas the second phase, the MEMBERSHIP PHASE, takes care of the satisfiability of the membership literals of φ^*, namely those of the form $x \in y$.

In order to model $\overline{x} \neq \overline{y}$, we add to exactly one between $M\overline{x}$ and $M\overline{y}$ a new member $\mathfrak{s}$ not already occurring in $\bigcup_{x \in \mathit{Vars}(\varphi)} Mx$. The set $\mathfrak{s}$ must be chosen with care to prevent that it may be accidentally formed during the subsequent membership phase. In addition, the set $\mathfrak{s}$ must be added to the right sets Mx in order that the resulting assignment keeps satisfying all the Boolean literals in $\varphi \wedge \bigwedge \mathcal{E}^+_M \wedge \bigwedge \mathcal{E}^-_M$ other than the designated equality $\overline{x} = \overline{y}$. The first problem is solved by selecting as $\mathfrak{s}$ any set of rank strictly greater than that of M. As for the second condition, recalling that, by (C2), the conjunction $\varphi \wedge \overline{x} \neq \overline{y}$ is satisfiable, we can select a model $\overline{M}$ for it. Therefore $\overline{M}\overline{x} \neq \overline{M}\overline{y}$, and so we can pick some element $\mathfrak{t}$ belonging to exactly one of the sets $\overline{M}\overline{x}$ and $\overline{M}\overline{y}$. By adding our special set $\mathfrak{s}$ as an element to all and only those sets Mx such that $\mathfrak{t} \in \overline{M}x$,

[1] Indeed, without disrupting conditions (C1) and (C2), for any variable $x \in \mathit{Vars}(\mathcal{E})$ one may add to φ the literal $x \in w$, where w stands for some fresh variable.

for $x \in \mathit{Vars}(\varphi)$, we obtain a new assignment, which will be denoted M_0. It turns out that M_0 correctly models all the conjuncts in φ^*, but the membership literals $x \in y$ for which $M_0 x \neq Mx$. We denote by V_0 the collection of variables x in φ such that $M_0 x \neq Mx$.

Example 6.1. *We illustrate the Boolean phase of our enlargement process with the following* MLS*-conjunction*

$$\varphi \;\coloneqq\; x = \overline{y} \setminus z \;\wedge\; x = \overline{x} \setminus w \;\wedge\; x \neq \overline{y} \;\wedge\; \overline{y} \in w \;\wedge\; w \in v \;\wedge\; z \in v$$

and with the equality $\overline{x} = \overline{y}$.

Let M *and* $\overline{M}$ *be the set assignments over* $\mathit{Vars}(\varphi) = \{v, w, x, \overline{x}, \overline{y}, z\}$ *so defined, where to enhance readability we use the shorthand* $\{\emptyset\}^2 \coloneqq \{\{\emptyset\}\}$*—likewise,* $\{\emptyset\}^4$ *will denote the set* $\{\{\{\{\emptyset\}\}\}\}$*:*

$$\begin{array}{lll} Mx = \emptyset, & M\overline{x} = M\overline{y} = \{\emptyset\}, & Mz = Mw = \{\emptyset, \{\emptyset\}\}, \quad Mv = \{\{\emptyset, \{\emptyset\}\}\}, \\ \overline{M}x = \emptyset, & \overline{M}\overline{y} = \overline{M}z = \{\emptyset\}, & \overline{M}\overline{x} = \overline{M}w = \{\emptyset\}^2, \qquad \overline{M}v = \{\{\emptyset\}, \{\emptyset\}^2\}. \end{array}$$

It can easily be checked that $M \models \varphi \wedge \overline{x} = \overline{y}$ *and* $\overline{M} \models \varphi \wedge \overline{x} \neq \overline{y}$ *hold.*

Let $\mathfrak{s} \coloneqq \{\emptyset\}^4$*, so that* $\mathsf{rk}(\mathfrak{s}) = 4 > 3 = \mathsf{rk}(Mv) = \mathsf{rk}(M)$*. Since* $\emptyset \in \overline{M}\overline{y} \setminus \overline{M}\overline{x}$*, we can put* $\mathsf{t} \coloneqq \emptyset$*, and so we have:*

$$M_0 u = \begin{cases} \{\emptyset, \mathfrak{s}\} & \text{if } u = \overline{y} \\ \{\emptyset, \{\emptyset\}, \mathfrak{s}\} & \text{if } u = z \\ Mu & \text{otherwise} \end{cases}$$

and $\mathsf{V}_0 = \{\overline{y}, z\}$.

Plainly, M_0 *satisfies all literals in* $\varphi \wedge \overline{x} \neq \overline{y}$ *but the literals* $\overline{y} \in w$ *and* $z \in v$. □

The subsequent membership phase performs the following enlargement step, for $k = 0, 1, 2, \ldots$, until needed:

> extend the assignment M_k by putting, for each $u \in \mathsf{V}_k$,
>
> $$M_{k+1} u \;\coloneqq\; M_k u \;\cup\; \{M_k v \mid v \in \mathsf{V}_k \text{ and } Mv \in Mu\},$$
>
> while setting $M_{k+1} u \coloneqq M_k u$ for the remaining variables u in $\mathit{Vars}(\varphi)$, and define V_{k+1} as the collection of variables u in $\mathit{Vars}(\varphi)$ such that $M_{k+1} u \neq M_k u$.

For $k = 0, 1, 2, \ldots$, it turns out that each M_k correctly models all the Boolean literals in φ^* and all the membership literals in φ^* but those of the form $x \in y$ with $x \in \mathsf{V}_{k+1}$. Hence, as soon as some V_k is empty, the assignment M_k is plainly a model for φ^*, and so the membership phase can stop. By the well-foundedness of the membership relation, such a situation occurs in at most $\overline{n} \coloneqq |\mathit{Vars}(\varphi)|$ steps, and therefore $M_{\overline{n}}$ is a model for φ^*, proving that φ^* is satisfiable.

Example 6.2 (cntd)**.** *We continue our example by illustrating the membership phase of our enlargement process. We recall that* $\mathsf{V}_0 = \{\overline{y}, z\}$. *Since* $M\overline{y} \in Mw = Mz$ *and* $Mz \in Mv$, *we have* $\mathsf{V}_1 = \{z, w, v\}$, $M_1 z = \{\emptyset, \{\emptyset\}, \mathfrak{s}, \{\emptyset, \mathfrak{s}\}\}$, $M_1 w = \{\emptyset, \{\emptyset\}, \{\emptyset, \mathfrak{s}\}\}$, $M_1 v = \{\{\emptyset, \{\emptyset\}\}, \{\emptyset, \{\emptyset\}, \mathfrak{s}\}\}$, *and* $M_1 u = M_0 u$ *for all* $u \neq z, w, v$. *Next, since* $Mw \in Mv$ *and* $Mz \in Mv$, *we have* $\mathsf{V}_2 = \{v\}$, $M_2 u = M_1 u$ *for all* $u \neq v$, *and*

$$M_2 v = \{\{\emptyset, \{\emptyset\}\}, \{\emptyset, \{\emptyset\}, \mathfrak{s}\}, \{\emptyset, \{\emptyset\}, \mathfrak{s}, \{\emptyset, \mathfrak{s}\}\}, \{\emptyset, \{\emptyset\}, \{\emptyset, \mathfrak{s}\}\}\}.$$

Finally, since $Mv \notin \bigcup_{u \in Vars(\varphi)} Mu$, *we can actually stop. In fact, at this point we have* $M_2 = M_3 = M_4 = \cdots$. *Plainly,* $M_2 \models \varphi \wedge \overline{x} \neq \overline{y}$. □

Proof details

For any $V \subseteq Vars(\varphi)$, we will use the notation MV to denote the set $\{Mv \mid v \in V\}$. Let $\mathfrak{s}$ be any fixed set whose rank is larger than the rank of M, namely such that $\mathsf{rk}(\mathfrak{s}) > \mathsf{rk}(M)$.

We define by recursion two sequences $\{\mathsf{V}_n\}_{n\in\mathbb{N}}$ and $\{M_n\}_{n\in\mathbb{N}}$, respectively of subsets of $Vars(\varphi)$ and of set assignments over $Vars(\varphi)$, by putting:

$$\mathsf{V}_0 \coloneqq \{u \in Vars(\varphi) \mid \mathfrak{t} \in \overline{M}u\}, \tag{6.2}$$

$$\mathsf{V}_n \coloneqq \{u \in Vars(\varphi) \mid Mu \cap M\mathsf{V}_{n-1} \neq \emptyset\}, \quad \text{for } n \geqslant 1, \tag{6.3}$$

and

$$M_0 v \coloneqq \begin{cases} Mv \cup \{\mathfrak{s}\} & \text{if } v \in \mathsf{V}_0 \\ Mv & \text{if } v \in Vars(\varphi) \setminus \mathsf{V}_0, \end{cases} \tag{6.4}$$

$$M_n v \coloneqq \begin{cases} M_{n-1}v \cup M_{n-1}\{u \in \mathsf{V}_{n-1} \mid Mu \in Mv\} & \text{if } v \in \mathsf{V}_n \\ M_{n-1}v & \text{if } v \in Vars(\varphi) \setminus \mathsf{V}_n, \end{cases} \tag{6.5}$$

for $n \geqslant 1$ and $v \in Vars(\varphi)$.

As a direct consequence of (6.4) and (6.5), the following results can be easily proved by induction:

Lemma 6.1. *(a) For every* $v \in Vars(\varphi)$, *we have*

$$Mv \subseteq M_0 v \subseteq \cdots \subseteq M_n v \subseteq \cdots.$$

(b) For all $v \in Vars(\varphi)$ *and* $n \in \mathbb{N}$, *we have:*

$$M_n v \subseteq Mv \cup \{\mathfrak{s}\} \cup \bigcup_{k=0}^{n-1} M_k\{u \in \mathsf{V}_k \mid Mu \in Mv\}.$$

Lemma 6.1(a) implies that the sequence of assignments $\{M_n\}_{n\in\mathbb{N}}$ is plainly pointwise convergent. As a consequence of the next lemma and corollary, it will follow in fact that $\{M_n\}_{n\in\mathbb{N}}$ converges "uniformly", and it does so in at most $|Vars(\varphi)|$ steps.

Lemma 6.2. *Let* $k \in \mathbb{N}$. *We have:*

(a) *if* $\mathsf{V}_k = \emptyset$ *then, for all* $n \geqslant k$,

(a_1) $\mathsf{V}_n = \emptyset$,

(a_2) $M_n = M_k$;

(b) *if* $\mathsf{V}_k \neq \emptyset$, *then*

$$k \;\leqslant\; \min\big(|\mathit{Vars}(\varphi)| - 1, \mathsf{rk}\,(M)\big). \tag{6.6}$$

Proof. If $\mathsf{V}_k = \emptyset$, then $\mathsf{V}_{k+1} = \emptyset$ and $M_{k+1} = M_k$ by (6.3) and (6.5), respectively. By iterating the same argument, one can easily prove that $\mathsf{V}_n = \emptyset$ and $M_n = M_k$, for all $n \in \mathbb{N}$, proving (a).

As for (b), we preliminarily observe that, by (6.3), for all $v \in \mathit{Vars}(\varphi)$ and $n \geqslant 1$ we have

$$v \in \mathsf{V}_n \implies (\exists u \in \mathsf{V}_{n-1}) Mu \in Mv. \tag{6.7}$$

Thus, if $\mathsf{V}_k \neq \emptyset$, by picking any $v_k \in \mathsf{V}_k$ and by repeatedly applying (6.7), it follows that there exist $v_0, v_1, \ldots, v_{k-1} \in \mathit{Vars}(\varphi)$ such that

$$Mv_0 \;\in\; Mv_1 \;\in\; \cdots \;\in\; Mv_{k-1} \;\in\; Mv_k. \tag{6.8}$$

By the well-foundedness of $\in$, the variables $v_0, v_1, \ldots, v_{k-1}, v_k$ must be pairwise distinct. Hence, $k+1 \leqslant |\mathit{Vars}(\varphi)|$. In addition, (6.8) also yields $k \leqslant \mathsf{rk}\,(Mv_k) \leqslant \mathsf{rk}\,(M)$. Thus, (6.6) follows, proving (b). □

The preceding lemma yields immediately the following result.

Corollary 6.3. *For all* $h, k > \min\big(|\mathit{Vars}(\varphi)| - 1, \mathsf{rk}\,(M)\big)$, *we have* $M_h = M_k$.

Letting $\overline{n} \coloneqq |\mathit{Vars}(\varphi)|$, Corollary 6.3 implies that $M_n = M_{\overline{n}}$, for all $n \geqslant \overline{n}$.

Next we prove a number of technical lemmas that will culminate in the proof that

$$M_{\overline{n}} \;\models\; \varphi \;\wedge\; \bigwedge \mathcal{E}^-_M \;\wedge\; \overline{x} \neq \overline{y}$$

holds, where we recall that $\overline{x} = \overline{y}$ is the designated equality of $\mathcal{E}$. Thus, we will have that

$$\mathcal{E}^-_M \subsetneq \mathcal{E}^-_{M_{\overline{n}}} \quad \text{and} \quad |\mathcal{E}^+_{M_{\overline{n}}}| < |\mathcal{E}^+_M|,$$

contradicting the minimality of $|\mathcal{E}^+_M|$. Hence, the convexity of MLS will follow, since our initial assumption on φ and $\mathcal{E}$ that conditions (C1) and (C2) hold will be proved to be untenable.

The following lemma provides some useful bounds on the rank of $M_n v$, for $n \in \mathbb{N}$ and $v \in \mathit{Vars}(\varphi)$.

Lemma 6.4. *For all* $n \in \mathbb{N}$ *and* $v \in \mathit{Vars}(\varphi)$, *we have*

- $\mathsf{rk}\,(M_n v) = \mathsf{rk}\,(\mathfrak{s}) + n + 1$, *if* $v \in \mathsf{V}_n$,
- $\mathsf{rk}\,(M_n v) \leqslant \mathsf{rk}\,(\mathfrak{s}) + n$, *if* $v \notin \mathsf{V}_n$.

Proof. We proceed by induction on n. For $n = 0$ and $v \in \mathsf{V}_0$, from (6.4) we have $M_0 v = Mv \cup \{\mathfrak{s}\}$. Hence, $\mathsf{rk}(M_0 v) = \max\{\mathsf{rk}(Mv), \mathsf{rk}(\{\mathfrak{s}\})\} = \mathsf{rk}(\mathfrak{s}) + 1$, since $\mathsf{rk}(\mathfrak{s}) > \mathsf{rk}(Mv)$. On the other hand, if $v \notin \mathsf{V}_0$, then $\mathsf{rk}(M_0 v) = \mathsf{rk}(Mv) < \mathsf{rk}(\mathfrak{s})$.

Next, let $n > 0$ and $v \in \mathsf{V}_n$. By (6.5), we have:

$$\mathsf{rk}(M_n v) = \max\big(\mathsf{rk}(M_{n-1} v), \mathsf{rk}(M_{n-1}\{z \in \mathsf{V}_{n-1} \mid Mz \in Mv\})\big). \tag{6.9}$$

By inductive hypothesis, we readily have

- $\mathsf{rk}(M_{n-1} v) \leqslant \mathsf{rk}(\mathfrak{s}) + n$, and
- $\mathsf{rk}(M_{n-1}\{z \in \mathsf{V}_{n-1} \mid Mz \in Mv\}) \leqslant \mathsf{rk}(\mathfrak{s}) + n + 1$.

In addition, since $v \in \mathsf{V}_n$, then by (6.3), $Mu \in Mv$ for some $u \in \mathsf{V}_{n-1}$. Hence, again by inductive hypothesis, $\mathsf{rk}(Mu) = \mathsf{rk}(\mathfrak{s}) + n$, and since $u \in \{z \in \mathsf{V}_{n-1} \mid Mz \in Mv\}$, we have

$$\mathsf{rk}(M_{n-1}\{z \in \mathsf{V}_{n-1} \mid Mz \in Mv\}) = \mathsf{rk}(\mathfrak{s}) + n + 1.$$

Thus, by (6.9), we get $\mathsf{rk}(M_n v) = \mathsf{rk}(\mathfrak{s}) + n + 1$.

On the other hand, if $v \notin \mathsf{V}_n$, then by (6.5) and by the inductive hypothesis we have $\mathsf{rk}(M_n v) = \mathsf{rk}(M_{n-1} v) \leqslant \mathsf{rk}(\mathfrak{s}) + n$. □

Next we prove that the set $\mathfrak{s}$ can enter M_n only when $n = 0$.

Lemma 6.5. *For all $n \in \mathbb{N}$ and $v \in Vars(\varphi)$, we have:*

(a) $M_n v \neq \mathfrak{s}$;

(b) $\mathfrak{s} \in M_n x \iff \mathfrak{s} \in M_0 x$.

Proof. Concerning (a), we proceed by induction on n.

For $n = 0$, by (6.4) we have:

- $\mathsf{rk}(M_0 v) = \mathsf{rk}(\mathfrak{s}) + 1$, if $v \in \mathsf{V}_0$ (by Lemma 6.4);
- $\mathsf{rk}(M_0 v) = \mathsf{rk}(Mv) < \mathsf{rk}(\mathfrak{s})$, if $v \notin \mathsf{V}_0$.

In both cases, it follows that $M_0 v \neq \mathfrak{s}$.

For the inductive step, let $n > 1$. If $v \in \mathsf{V}_n$, then by Lemma 6.4 we have $\mathsf{rk}(M_n v) = \mathsf{rk}(\mathfrak{s}) + n + 1$, and therefore $M_n v \neq \mathfrak{s}$. On the other hand, if $v \notin \mathsf{V}_n$, then $M_n v = M_{n-1} v \neq \mathfrak{s}$, by (6.5) and by the inductive hypothesis.

Next we prove (b) by induction on n.

The base case $n = 0$ is trivial.

For the inductive step, let $n > 0$. If $\mathfrak{s} \in M_0 x$, then Lemma 6.1(a) yields readily $\mathfrak{s} \in M_n x$. Conversely, let $\mathfrak{s} \in M_n x$. If $x \notin \mathsf{V}_n$, then by (6.5) we have

$\mathfrak{s} \in M_n x = M_{n-1} x$, and therefore by inductive hypothesis $s \in M_0 x$. On the other hand, if $x \in \mathsf{V}_n$, then again by (6.5) we have

$$\mathfrak{s} \in M_n x = M_{n-1} x \cup M_{n-1}\{y \in \mathsf{V}_{n-1} \mid My \in Mx\}.$$

In view of (a), the latter formula yields $\mathfrak{s} \in M_{n-1} x$, and therefore $s \in M_0 x$ follows again by inductive hypothesis, completing the proof of (b), and in turn of the lemma. □

The following lemma proves that, at each construction step of the assignments M_n's, only elements of rank at least $\mathsf{rk}(\mathfrak{s})$ can enter into play.

Lemma 6.6. *For every set $\mathfrak{q} \in M_n x$, for some $x \in Vars(\varphi)$ and $n \in \mathbb{N}$, if $\mathsf{rk}(\mathfrak{q}) < \mathsf{rk}(\mathfrak{s})$ then $\mathfrak{q} \in Mx$.*

Proof. Let $x \in Vars(\varphi)$, $n \in \mathbb{N}$, and $\mathfrak{q} \in M_n x$, with $\mathsf{rk}(\mathfrak{q}) < \mathsf{rk}(\mathfrak{s})$. From Lemma 6.1(b), we have

$$M_n x \subseteq Mx \cup \{\mathfrak{s}\} \cup \bigcup_{k=0}^{n-1} M_k\{y \in \mathsf{V}_k \mid My \in Mx\}.$$

Since, by Lemma 6.4, the rank of each member of $\{\mathfrak{s}\} \cup \bigcup_{k=0}^{n-1} M_k\{y \in \mathsf{V}_k \mid My \in Mx\}$ is greater than or equal to $\mathsf{rk}(\mathfrak{s})$, then necessarily $\mathfrak{q} \in Mx$ □

All the inequalities $x \neq y$ satisfied by M are satisfied by every M_n, as proved in the following corollary.

Corollary 6.7. *If $Mx \neq My$, for some $x, y \in Vars(\varphi)$, then $M_n x \neq M_n y$, for every $n \in \mathbb{N}$.*

Proof. W.l.o.g., let us assume that $Mx \not\subseteq My$, and let $\mathfrak{q} \in Mx \setminus My$. Also, let $n \in \mathbb{N}$. By Lemma 6.1(a), $\mathfrak{q} \in M_n x$. Plainly, $\mathsf{rk}(\mathfrak{q}) < \mathsf{rk}(Mx) < \mathsf{rk}(\mathfrak{s})$ and $\mathfrak{q} \notin My$. Thus, Lemma 6.6 yields $\mathfrak{q} \notin M_n y$, proving that $M_n x \neq M_n y$. □

To show that every membership $x \in y$ satisfied by M is correctly modeled by $M_{\overline{n}}$, we will need the following result.

Lemma 6.8. *For all $n \in \mathbb{N}$ and $x, y \in Vars(\varphi)$, if $x \in \mathsf{V}_n$ and $Mx \in My$, then $y \in \mathsf{V}_{n+1}$ and $M_n x \in M_{n+1} y$.*

Proof. Let $n \in \mathbb{N}$ and assume that $Mx \in My$, for some $x, y \in Vars(\varphi)$, and that $x \in \mathsf{V}_n$. Then, by (6.3), $y \in \mathsf{V}_{n+1}$. In addition, from (6.5), the latter membership relation yields immediately that $M_n x \in M_{n+1} y$. □

We are now ready to prove our main lemma.

Lemma 6.9. *The assignment $M_{\overline{n}}$ satisfies φ.*

Proof. We prove the lemma, by showing that $M_{\overline{n}}$ correctly models all the conjuncts in φ. We recall that, in view of the reduction process outlined in Corollary 2.2, our formula φ contains conjuncts of two types only, namely $x \in y$ and $x = y \setminus z$.

Conjuncts of type $x \in y$. Let $x \in y$ occur in φ, so that $Mx \in My$ holds. If $x \notin \mathsf{V}_n$ for all $n \in \mathbb{N}$, then $M_{\overline{n}}x = Mx \in My \subseteq M_{\overline{n}}y$ (by Lemma 6.1(a)), from which $M_{\overline{n}}x \in M_{\overline{n}}y$ follows.
Conversely, if $x \in \mathsf{V}_n$, for some $n \in \mathbb{N}$, we set $\overline{m} := \max\{n \in \mathbb{N} \mid x \in \mathsf{V}_n\}$. In addition, since $Mx \in My$ and $x \in \mathsf{V}_{\overline{m}}$, Lemma 6.8 implies $y \in \mathsf{V}_{\overline{m}+1}$ and therefore, by Lemma 6.2(b), $\overline{m} + 1 \leqslant |Vars(\varphi)| - 1 \leqslant \overline{n}$. Thus, Lemma 6.8 again together with Lemma 6.1(a) yields $M_{\overline{n}}x = M_{\overline{m}}x \in M_{\overline{m}+1}y \subseteq M_{\overline{n}}y$, from which $M_{\overline{n}}x \in M_{\overline{n}}y$ follows.

Conjuncts of type $x = y \setminus z$. Let $x = y \setminus z$ occur in φ, so that $Mx = My \setminus Mz$ holds. We will prove that $M_{\overline{n}} \models x = y \setminus z$, by proving that $M_{\overline{n}}x \subseteq M_{\overline{n}}y \setminus M_{\overline{n}}z$ and $M_{\overline{n}}y \setminus M_{\overline{n}}z \subseteq M_{\overline{n}}x$ hold.

Proof of $M_{\overline{n}}x \subseteq M_{\overline{n}}y \setminus M_{\overline{n}}z$. From Lemma 6.1(b), we have:

$$M_{\overline{n}}x \;\subseteq\; Mx \;\cup\; \{\mathfrak{s}\} \;\cup\; \bigcup_{k=0}^{\overline{n}-1} M_k\{u \in \mathsf{V}_k \mid Mu \in Mx\}.$$

Let $\mathfrak{q} \in M_{\overline{n}}x$. We first consider that case in which $\mathfrak{q} \in Mx$. Then $\mathfrak{q} \in My \setminus Mz$. Hence, by Lemma 6.1(a), $\mathfrak{q} \in M_{\overline{n}}y$. In addition, since $\mathfrak{q} \notin Mz$ and $\mathsf{rk}(\mathfrak{q}) < \mathsf{rk}(\mathfrak{s})$, Lemma 6.6 yields $\mathfrak{q} \notin M_{\overline{n}}z$. Thus, $\mathfrak{q} \in M_{\overline{n}}y \setminus M_{\overline{n}}z$.

Next, let $\mathfrak{q} = \mathfrak{s}$. Hence, $\mathfrak{s} \in M_0x$ (by Lemma 6.5(b)), so that $x \in \mathsf{V}_0$ (by (6.4) and (6.2)), and therefore $\mathfrak{t} \in \overline{M}x$. Since $\overline{M}x = \overline{M}y \setminus \overline{M}z$, then $\overline{M}x \subseteq \overline{M}y$, and so $\mathfrak{t} \in \overline{M}y$ and $\mathfrak{t} \notin \overline{M}z$. From $\mathfrak{t} \in \overline{M}y$, it follows that $y \in \mathsf{V}_0$ and $z \notin \mathsf{V}_0$. Therefore (by (6.4) and Lemma 6.1(b)) $\mathfrak{s} \in M_0y \subseteq M_{\overline{n}}y$ and also (by (6.2)) $\mathfrak{s} \notin M_0z$. Thus, by Lemma 6.5(b), $\mathfrak{s} \notin M_{\overline{n}}z$. In conclusion, if $\mathfrak{q} = \mathfrak{s}$ then $\mathfrak{q} \in M_{\overline{n}}y \setminus M_{\overline{n}}z$, as in the preceding case.

Finally, let $\mathfrak{q} = M_ku$, for some $0 \leqslant k < \overline{n}$ and $u \in \mathsf{V}_k$ such that $Mu \in Mx$. Recalling that $Mx = My \setminus Mz$, then $Mu \in My$, so that $M_ku \in M_{k+1}y \subseteq M_{\overline{n}}y$ (by Lemma 6.8). In addition, $Mu \notin Mz$. By Lemma 6.4, $M_ku \notin Mz \cup \{\mathfrak{s}\}$. Since, by Lemma 6.1(b),

$$M_{\overline{n}}z \;\subseteq\; Mz \;\cup\; \{\mathfrak{s}\} \;\cup\; \bigcup_{k=0}^{\overline{n}-1} M_k\{v \in \mathsf{V}_k \mid Mv \in Mz\}, \tag{6.10}$$

to prove that $M_ku \notin M_{\overline{n}}z$, it is sufficient to show that $M_ku \notin \bigcup_{h=0}^{\overline{n}-1} M_h\{v \in \mathsf{V}_h \mid Mv \in Mz\}$. By way of contradiction, let us assume that $M_ku = M_hv$, for some $0 \leqslant h < \overline{n}$ and $v \in \mathsf{V}_h$ such that $Mv \in Mz$. By Lemma 6.4, and since $u \in \mathsf{V}_k$, we must have $h = k$. Since $Mu \notin Mz$ while $Mv \in Mz$, we plainly have $Mu \neq Mv$. Hence, by Corollary 6.7, $M_hu \neq M_hv = M_ku$, which contradicts our preceding assumption $M_ku = M_hu$. Thus, $M_ku \notin \bigcup_{h=0}^{\overline{n}-1} M_h\{v \in \mathsf{V}_h \mid Mv \in Mz\}$ holds. In view of $M_ku \notin Mz \cup \{\mathfrak{s}\}$ and (6.10), the latter

equation implies $M_k u \notin M_{\overline{n}} z$, proving that $\mathfrak{q} \in M_{\overline{n}} y \setminus M_{\overline{n}} z$ even in the case in which $\mathfrak{q} \in \bigcup_{k=0}^{\overline{n}-1} M_k\{u \in \mathsf{V}_k \mid Mu \in Mz\}$.

From the arbitrariness of $\mathfrak{q} \in M_{\overline{n}} x$, we conclude that $M_{\overline{n}} x \subseteq M_{\overline{n}} y \setminus M_{\overline{n}} z$ holds.

Proof of $M_{\overline{n}} y \setminus M_{\overline{n}} z \subseteq M_{\overline{n}} x$. Let us assume now that $\mathfrak{q} \in M_{\overline{n}} y \setminus M_{\overline{n}} z$, so that $\mathfrak{q} \in M_{\overline{n}} y$. Again from Lemma 6.1(b), we have:

$$M_{\overline{n}} y \;\subseteq\; My \;\cup\; \{\mathfrak{s}\} \;\cup\; \bigcup_{k=0}^{\overline{n}-1} M_k\{v \in \mathsf{V}_k \mid Mv \in My\}.$$

First we consider the case in which $\mathfrak{q} \in My$, so that $\mathsf{rk}(\mathfrak{q}) < \mathsf{rk}(\mathfrak{s})$. Since $\mathfrak{q} \notin M_{\overline{n}} z$, then by Lemma 6.1(a) $\mathfrak{q} \notin Mz$, and therefore $\mathfrak{q} \in My \setminus Mz = Mx \subseteq M_{\overline{n}} x$. Thus, $\mathfrak{q} \in M_{\overline{n}} x$.

Next, if $\mathfrak{q} = \mathfrak{s}$, then $\mathfrak{s} \in M_{\overline{n}} y \setminus M_{\overline{n}} z$. Thus, $\mathfrak{s} \in M_0 y$ and $\mathfrak{s} \notin M_0 z$ by Lemmas 6.5(b) and 6.1(a), respectively. Hence, by (6.4), $y \in \mathsf{V}_0$ and $z \notin \mathsf{V}_0$, so that $\mathfrak{t} \in \overline{M}y \setminus \overline{M}z = \overline{M}x$ (since $\overline{M} \models \varphi$). In view of (6.2), the latter membership relation yields $x \in \mathsf{V}_0$. Thus $\mathfrak{q} = \mathfrak{s} \in M_0 x \subseteq M_{\overline{n}} x$, which readily implies $\mathfrak{q} \in M_{\overline{n}} x$.

Finally, let us assume that $\mathfrak{q} = M_k v$, for some $0 \leqslant k < \overline{n}$ such that $v \in \mathsf{V}_k$ and $Mv \in My$. Plainly, $Mv \notin Mz$, otherwise by Lemma 6.8 we should have $\mathfrak{q} = M_k v \in M_{k+1} z \subseteq M_{\overline{n}} z$, contradicting $\mathfrak{q} \in M_{\overline{n}} y \setminus M_{\overline{n}} z$. Thus, $Mv \in My \setminus Mz = Mx$, so that $Mv \in Mx$. But then, by Lemma 6.8 again, we get $\mathfrak{q} = M_k v \in M_{k+1} x \subseteq M_{\overline{n}} x$, from which $\mathfrak{q} \in M_{\overline{n}} x$ follows even in the last case.

Thus, in all cases we have $\mathfrak{q} \in M_{\overline{n}} x$. By the arbitrariness if $\mathfrak{q}$ in $M_{\overline{n}} y \setminus M_{\overline{n}} z$, we therefore obtain $M_{\overline{n}} y \setminus M_{\overline{n}} z \subseteq M_{\overline{n}} x$.

In view of the reverse inclusion $M_{\overline{n}} x \subseteq M_{\overline{n}} y \setminus M_{\overline{n}} z$ established earlier, the latter inclusion yields $M_{\overline{n}} x = M_{\overline{n}} y \setminus M_{\overline{n}} z$, namely $M_{\overline{n}} \models x = y \setminus z$.

Summing up, we have proved that the assignment $M_{\overline{n}}$ satisfies all the conjuncts of φ, and therefore $M_{\overline{n}}$ satisfies φ. □

Together with Corollary 6.7, the preceding lemma implies

$$M_{\overline{n}} \;\models\; \varphi \;\wedge\; \bigwedge \mathcal{E}_M^-.$$

To find a contradiction, it only remains to prove that $M_{\overline{n}} \models \overline{x} \neq \overline{y}$, where $\overline{x} = \overline{y}$ is the designated equality of $\mathcal{E}_M^+$, which we do next.

Lemma 6.10. *The assignment $M_{\overline{n}}$ models the inequality $\overline{x} \neq \overline{y}$ correctly.*

Proof. We know that $\mathfrak{t} \in \overline{M}\overline{x} \setminus \overline{M}\overline{y}$, therefore $\overline{x} \in \mathsf{V}_0$ and $\overline{y} \notin \mathsf{V}_0$, $\mathfrak{s} \in M_0\overline{x} \setminus M_0\overline{y}$. Hence, from Lemma 6.5(b) it follows that $\mathfrak{s} \in M_{\overline{n}}\overline{x} \setminus M_{\overline{n}}\overline{y}$, proving that $M_{\overline{n}} \models \overline{x} \neq \overline{y}$. □

From Lemmas 6.9 and 6.10 and Corollary 6.7, we have:

$$M_{\overline{n}} \models \varphi \wedge \bigwedge \mathcal{E}^-_M \wedge \overline{x} \neq \overline{y}.$$

Setting $\mathcal{E}^+_{M_{\overline{n}}} := \{\ell \in \mathcal{E} \mid M_{\overline{n}} \models \ell\}$ and $\mathcal{E}^-_{M_{\overline{n}}} := \{\neg\ell \mid \ell \in \mathcal{E} \setminus \mathcal{E}^+_{M_{\overline{n}}}\}$, we have $\mathcal{E}^-_M \subsetneq \mathcal{E}^-_{M_{\overline{n}}}$ and so $|\mathcal{E}^+_{M_{\overline{n}}}| < |\mathcal{E}^+_M|$, contradicting the minimality of $|\mathcal{E}^+_M|$ among all the set assignments that satisfy φ. Thus, our initial hypothesis that MLS were not convex is inadmissible, and therefore we can conclude that:

Theorem 6.11. *The theory* MLS *is convex.*

We expect that the proof of convexity of MLS can be suitably generalized to show that also two extensions of MLS, MLS$\bigcap$ that also includes literals of the form $x = \bigcap y$, and MLSC$\bigcap$ that also include cardinality comparisons $|x| = |y|$ and $|x| < |y|$, are convex too.

Several fragments of MLS admit polynomial-time decision procedures, so they are very appealing in the context of combination of decision procedures *à la* Nelson-Oppen. We briefly review them next.

6.2.1 Polynomial fragments of MLS

Although MLS is convex, we can not use the Nelson-Oppen strategy to obtain efficient, polynomial, decision procedure for those theories comprising MLS, simply because the decision procedure of MLS is itself NP-complete.

However convexity of MLS is plainly inherited by all of its fragments.

Therefore all the polynomial satisfiability procedures for fragments of MLS, that we have highlighted in the previous chapters, can be efficiently combined using the Nelson-Oppen strategy with other polynomial decision procedures of other convex theories disjoint from set theory to obtain SMT solvers that runs in deterministic polynomial time.

6.3 Non-convex extensions of MLS

Some of the extensions of MLS are non-convex. To prove such results we rely on the following property.

Lemma 6.12. *Let* T *be any extension of* MLS *containing a conjunction φ with a designated variable $\overline{x}$ such that, for some integer $k \geqslant 2$, we have:*

- $\models \varphi \longrightarrow |\overline{x}| \leqslant k$,
- $\varphi \wedge |\overline{x}| = k$ *is satisfiable,*

where $|\overline{x}|$ stands for the cardinality of $\overline{x}$. Then T *is not convex.*

Proof sketch. Given φ, $\overline{x}$, and k as in the hypotheses, it is enough to set $\Phi := \varphi \wedge \bigwedge_{i=1}^{k+1} x_i \in \overline{x}$, where $x_1, \ldots, x_{k+1}$ are pairwise distinct variables

not occurring in φ. Then, we have:

$$\models \Phi \;\longrightarrow \bigvee_{1 \leqslant i < j \leqslant k+1} x_i = x_j \,.$$

In addition, each conjunction $\Phi \wedge x_i \neq x_j$, with $1 \leqslant i < j \leqslant k+1$, is satisfiable. Hence, none of the statements

$$\models \Phi \;\longrightarrow\; x_i = x_j$$

can hold, for $1 \leqslant i < j \leqslant k+1$. Thus, the theory T is not convex. □

Using Lemma 6.12, we argue next that the following extensions of MLS are non-convex:

- MLSS = MLS + '$\{\cdot\}$': MLS extended with the singleton operator $x = \{y\}$ (see [FOS80]),
- MLSP = MLS + '$\mathscr{P}(\cdot)$': MLS extended with the powerset operator (see [CSF85]),
- MLS$\bigcup$ = MLS + '$\bigcup\cdot$': MLS extended with the general union operator (see [CFS87]),
- MLS$\times$ = MLS + '$\times$': MLS extended with the Cartesian product operator.[2]

Concerning the theory MLSS, let us consider the conjunction

$$\varphi \coloneqq \;\; x = \{y\} \;\wedge\; x' = \{y'\} \;\wedge\; \overline{x} = x \cup x' \,.$$

Then,

- for every model M for φ, we have $M\overline{x} = \{My, My'\}$, so that $|M\overline{x}| \leqslant 2$ holds;
- letting $\overline{M}$ be the set assignment for φ such that

  $$\overline{M}y = \emptyset, \qquad \overline{M}y' = \overline{M}x = \{\emptyset\}, \qquad \overline{M}x' = \{\{\emptyset\}\}, \qquad \overline{M}\overline{x} = \{\emptyset, \{\emptyset\}\},$$

  then $\overline{M}$ satisfies φ and $|\overline{M}\overline{x}| = 2$.

Thus, by Lemma 6.12, MLSS is non-convex.

Next, as for the theory MLSP, let us consider the conjunction

$$\varphi \coloneqq \;\; x = \varnothing \;\wedge\; y = \mathscr{P}(x) \;\wedge\; \overline{x} = \mathscr{P}(y) \,.$$

Then, φ is plainly satisfiable and, for every set assignment M satisfying φ, we have $M\overline{x} = \{\emptyset, \{\emptyset\}\}$, so that $|M\overline{x}| = 2$. Thus, by Lemma 6.12, MLSP is non-convex.

[2] The decision problem for MLS$\times$ is still open.

Concerning the fragment $\mathsf{MLS}\bigcup$, let us consider the conjunction

$$\varphi \coloneqq \quad x = \varnothing \ \wedge \ \bigcup y = x \ \wedge \ \bigcup \overline{x} = y \, .$$

Then,

- for every set assignment M satisfying φ, we have $M\overline{x} \subseteq \{\emptyset, \{\emptyset\}\}$ so that $|M\overline{x}| \leqslant 2$ holds;
- letting $\overline{M}$ be the set assignment over $\mathit{Vars}(\varphi)$ such that

$$\overline{M}x = \emptyset, \qquad My = \{\emptyset\}, \qquad \overline{M}\overline{x} = \{\emptyset, \{\emptyset\}\},$$

  we readily have that $\overline{M}$ satisfies φ and $|\overline{M}\overline{x}| = 2$.

Hence, by Lemma 6.12, $\mathsf{MLS}\bigcup$ is non-convex.

Since MLSSP is an extension of non-convex theories, namely MLSS and MLSP, it follows immediately that MLSSP is non-convex as well.

Regarding the extension $\mathsf{MLS}\times$ of MLS with the Cartesian product, we have

$$\models x \times y = \varnothing \ \longrightarrow \ (x = \varnothing \ \vee \ y = \varnothing). \tag{6.11}$$

Since the two conjunctions $x \times y = \varnothing \ \wedge x \neq \varnothing$ and $x \times y = \varnothing \ \wedge y \neq \varnothing$ are clearly satisfiable, then

$$\not\models x \times y = \varnothing \ \longrightarrow \ x = \varnothing \qquad \text{and} \qquad \not\models x \times y = \varnothing \ \longrightarrow \ y = \varnothing.$$

Together with (6.11), the latter statements imply that $\mathsf{MLS}\times$ is non-convex.

By replacing in the above proof the Cartesian product $\times$ by the unordered Cartesian product $\otimes$, one can readily show that the extension $\mathsf{MLS}\otimes$ of MLS with the unordered Cartesian product $\otimes$ is non-convex too.

Finally, notice that the membership relator did not play any role in the above proof of non-convexity of $\mathsf{MLS}\times$ and $\mathsf{MLS}\otimes$. Therefore, by exactly the same argument as the above one, we can show that the extensions $\mathsf{BST}\times$ and $\mathsf{BST}\otimes$ of BST with the Cartesian product and the unordered Cartesian product are non-convex.

Summarizing, we have proved:

Lemma 6.13. *The theories* MLSS, MLSP, MLSSP, $\mathsf{MLS}\bigcup$, $\mathsf{MLS}\times$, $\mathsf{MLS}\otimes$, $\mathsf{BST}\times$, *and* $\mathsf{BST}\otimes$ *are all non-convex.*

Conclusions

In this dissertation we presented an in-depth study of the theory Multi-Level Syllogistic, whose satisfiability problem is decidable and belongs to the class of NP-complete problems as first seen in [COP90]. Specifically, our study has mainly focused on spotting the fragments of MLS satisfiability problem can be solved in deterministic polynomial time.

The main rationale behind such study is twofold: solving the foundational problem of finding the boundaries between NP-completeness and polynomiality between fragments of MLS, and finding new more efficient algorithms to solve the decision problem for MLS under the suitable constraints.

The task of studying all the fragments of MLS, that are more than two thousand, seemed to be very lengthy at first. However, thanks to existential expressibility and its new extension $\mathcal{O}(f)$-expressibility, we managed to shrink considerably the number of fragments to analyze. In fact, it turned out that we could focus only on very small, therefore more easy to approach, fragments, such as $\mathsf{BST}(\cup, =, \neq)$, then using the expressibility tools to expand those fragments to much larger ones, such as $\mathsf{BST}(\cup, =\varnothing, \neq\varnothing, \neg\mathsf{Disj}, \subseteq, \not\subseteq, =, \neq)$.

Another result obtained using the expressibility tools is the reduction of each MLS-formula into a conjunction of just two kinds of literals: $x \in y$ and $x = y \setminus z$, simplifying most of the proofs involving MLS. For example each time we need to prove that a set assignment models an MLS-formula we can just focus on literals of those two forms.

At the end of our study we obtained Table 5.2, which lists all the minimal NP-complete and maximal polynomial fragments of MLS. Table 5.2 is enough to discern given any fragment of the theory whether it is polynomial or NP-complete. In fact, if a given fragment is a sub-language of a maximal polynomial fragment then it is polynomial otherwise it is NP-complete.

Table 5.2 is also very useful in practical situation to enhance the performance of any tool that needs to establish the satisfiability of MLS-formulae, such as the proof verifier ÆtnaNova. In fact, instead of simply running the satisfiability test for MLS over each MLS-formula the tool may select in favorable cases a polynomial satisfiability test in place of the nondeterministic polynomial test for full MLS.

Finally we proved that the theory MLS is convex, meaning that we can use the Nelson-Oppens approach to combine the decision procedure for the polynomial fragments of MLS with the decision procedures of other convex theories, outside Set theory, in deterministic polynomial time.

Although having provided a complete complexity taxonomy of MLS may seems to end our study, that is not case. When we find a minimal NP-complete fragment we can not say more about that fragment, while when we found a maximal polynomial fragment, it can be useful to study some of its sub-languages since, as we seen for instance in the case of $\mathsf{BST}(\cup, \neq, \mathsf{Disj}, \neg\mathsf{Disj})$ and $\mathsf{BST}(\cup, \mathsf{Disj}, \neq)$, it may be possible to find more efficient satisfiability test specialized for such sub-languages. Moreover, we have not proved that the satisfiability tests we have found for the polynomial fragments are the most efficient possible, for instance we found a cubic time satisfiability test for $\mathsf{BST}(\cup, =, \neq)$, but we do not now if this fragments admits some satisfiability test strictly less than cubic.

The most natural way to continue this research for polynomial fragments is that of analyzing the fragments of the extensions of MLS. We notice, during the study of the taxonomy of MLS, that starting from small languages and slowly adding new operators and predicates is a winning strategy. The next step then could be to add the singleton atom, $x = \{y\}$, to our taxonomy, yielding a taxonomy for MLSS. We must notice, however, that most of the results in the taxonomy of MLS were possible thanks to the fact that the satisfiability problem of MLS is decidable and furthermore is NP-complete: for example all our NP-completeness proofs are based on the fact that the decision problem of MLS is in the class NP. The decision problem for several extensions of MLS, such as MLS$\times$, is still open, hence another task for the future is to solve such problems.

In conclusion, the complete complexity taxonomy might be a significant addition to the field of Computable Set Theory, nevertheless the field still presents a large set of further direction to study.

Bibliography

[AHU74] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

[BGN+06] M.P. Bonacina, S. Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli. Decidability and undecidability results for Nelson-Oppen and rewrite-based decision procedures. In U. Furbach and N. Shankar, editors, *Automated Reasoning. IJCAR 2006*, volume 4130 of *Lecture Notes in Computer Science*, pages 513–527, 2006.

[BM07] Aaron R. Bradley and Zohar Manna. *The calculus of computation - decision procedures with applications to verification*. Springer, 2007.

[BST10] C. Barrett, A. Stump, and C. Tinelli. The satisfiability modulo theories library, 2010.

[Can91] Domenico Cantone. Decision procedures for elementary sublanguages of set theory. X. Multilevel syllogistic extended by the singleton and powerset operators. *J. Autom. Reasoning*, 7(2):193–230, June 1991.

[Can13] Domenico Cantone. *Decision Procedures for Elementary Sublanguages of Set Theory. XVII. Commonly Occurring Decidable Extensions of Multi-level Syllogistic*, pages 47–85. Springer London, London, 2013.

[CC89] Domenico Cantone and Vincenzo Cutello. Decision procedures for elementary sublanguages of set theory. XVI. Multilevel syllogistic extended by singleton, rank comparison and unary intersection. *Bulletin of EATCS*, 39:139–148, 1989.

[CDDM21] Domenico Cantone, Andrea De Domenico, and Pietro Maugeri. On the convexity of a fragment of pure set theory with applications within a nelson-oppen framework. In Pierre Ganty and Davide Bresolin, editors, Proceedings 12th International Symposium on *Games, Automata, Logics, and Formal Verification,* Padua, Italy, 20-22 September 2021, volume 346 of *Electronic Proceedings in Theoretical Computer Science*, pages 195–210. Open Publishing Association, 2021.

[CDDMO20] Domenico Cantone, Andrea De Domenico, Pietro Maugeri, and Eugenio G. Omodeo. A quadratic reduction of constraints over

nested sets to purely Boolean formulae in CNF. In Francesco Calimeri, Simona Perri, and Ester Zumpano, editors, *Proceedings of the 35th Italian Conference on Computational Logic, Rende, Italy, October 13-15, 2020*, volume 2710 of *CEUR Workshop Proceedings*, pages 214–230. CEUR-WS.org, 2020. ISSN 1613-0073.

[CDMO21] Domenico Cantone, Andrea De Domenico, Pietro Maugeri, and Eugenio G. Omodeo. Complexity assessments for decidable fragments of set theory. I: A taxonomy for the Boolean case. *Fundamenta Informaticae*, 181:37–69, 2021.

[CFO89] Domenico Cantone, Alfredo Ferro, and Eugenio G. Omodeo. *Computable set theory.* Number 6 in International Series of Monographs on Computer Science, Oxford Science Publications. Clarendon Press, Oxford, UK, 1989.

[CFS87] D. Cantone, A. Ferro, and J.T. Schwartz. Decision procedures for elementary sublanguages of set theory. V: Multilevel syllogistic extended by the general union operator. *Journal of Computer and System Sciences*, 34(1):1–18, 1987.

[CMO20] Domenico Cantone, Pietro Maugeri, and Eugenio G. Omodeo. Complexity assessments for decidable fragments of set theory. II: A taxonomy for 'small' languages involving membership. *Theoretical Computer Science*, 848:28–46, 2020.

[CMO21] Domenico Cantone, Pietro Maugeri, and Eugenio G. Omodeo. Two Crucial Cubic-Time Components of Polynomial-Maximal Decidable Boolean Languages. In Claudio Sacerdoti Coen and Ivano Salvo, editors, *Proceedings of the 22nd Italian Conference on Theoretical Computer Science, Bologna, Italy, September 13-15, 2021.*, volume 3072 of *CEUR Workshop Proceedings*, pages 164–178. CEUR-WS.org, 2021. ISSN 1613-0073.

[COP90] Domenico Cantone, Eugenio G. Omodeo, and Alberto Policriti. The automation of syllogistic. II: Optimization and complexity issues. *J. Autom. Reasoning*, 6(2):173–187, June 1990.

[COP01] Domenico Cantone, Eugenio G. Omodeo, and Alberto Policriti. *Set theory for computing - From decision procedures to declarative programming with sets.* Monographs in Computer Science. Springer-Verlag, New York, 2001.

[CSF85] D. Cantone, J. T. Schwartz, and A. Ferro. Decision procedures for elementary sublanguages of set theory. VI. multi-level syllogistic extended by the powerset operator. *Communications on Pure and Applied Mathematics*, 38(5):549–571, 1985.

[CU14] Domenico Cantone and Pietro Ursino. Formative processes with applications to the decision problem in set theory: II. powerset and singleton operators, finiteness predicate. *Information and Computation*, 237:215–242, 2014.

[CU18] Domenico Cantone and Pietro Ursino. *An Introduction to the Technique of Formative Processes in Set Theory*. Springer International Publishing, 2018.

[CUO02] Domenico Cantone, Pietro Ursino, and Eugenio G. Omodeo. Formative processes with applications to the decision problem in set theory. I: Powerset and singleton operators. *Information and Computation*, 172(2):165–201, 2002.

[FOS80] Alfredo Ferro, Eugenio G. Omodeo, and Jacob T. Schwartz. Decision procedures for elementary sublanguages of set theory. I: Multilevel syllogistic and some extensions. *Comm. Pure Appl. Math.*, 33:599–608, 1980.

[NO79] G. Nelson and D.C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979.

[SCO11] Jacob T. Schwartz, Domenico Cantone, and Eugenio G. Omodeo. *Computational logic and set theory: Applying formalized logic to analysis*. Springer-Verlag, 2011. Foreword by M. Davis.

[SDDS86] Jacob T Schwartz, Robert BK Dewar, Edward Dubinsky, and Edith Schonberg. *Programming with sets: An introduction to SETL*. Springer Science & Business Media, 1986.