A Decision Procedure for a Fragment of Set Theory Involving Monotone, Additive, and Multiplicative Functions

Calogero G. Zarba, Domenico Cantone, Jacob T. Schwartz

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Decision Procedure for a Fragment of Set Theory Involving Monotone, Additive, and Multiplicative Functions

Calogero G. Zarba — Domenico Cantone — Jacob T. Schwartz

N° 5267

July 2004

Thème SYM

ISSN 0249-6399 ISRN INRIA/RR--5267--FR+ENG

A Decision Procedure for a Fragment of Set Theory Involving Monotone, Additive, and Multiplicative Functions

Calogero G. Zarba* , Domenico Cantone† , Jacob T. Schwartz‡

Thème SYM — Systèmes symboliques
Projet CASSIS

Rapport de recherche n° 5267 — July 2004 — 22 pages

Abstract: **2LS** is a decidable many-sorted set-theoretic language involving one sort for elements and one sort for sets of elements. In this report we extend **2LS** with constructs for expressing monotonicity, additivity, and multiplicativity properties of set-to-set functions. We call the resulting language **2LSmf**. We prove that **2LSmf** is decidable by reducing the problem of determining the satisfiability of its sentences to the problem of determining the satisfiability of sentences of **2LS**. Furthermore, we prove that the language **2LSmf** is stably infinite with respect to the sort of elements. Therefore, using a many-sorted version of the Nelson-Oppen combination method, **2LSmf** can be combined with other languages modeling the sort of elements.

Key-words: Automated deduction, Decision procedures, Computable set theory.

* LORIA and INRIA-Lorraine, 615, rue du Jardin Botanique, BP 101, 54602 Villers-lès-Nancy Cedex, France, Email: `calogero.zarba@loria.fr`.

† Dipartimento di Matematica e Informatica, Università di Catania, Viale Andrea Doria 6, 95125 Catania, Italy, Email: `cantone@dmi.unict.it`.

‡ Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA, Email: `schwartz@cs.nyu.edu`.

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,

Une procédure de décision pour un fragment de la théories des ensembles qui considère fonctions monotones, additives, et multiplicatives

Résumé : **2LS** est un sous-langage décidable de la théorie des ensembles avec une sorte pour les éléments et une sorte pour les ensembles d'éléments. Dans ce rapport nous étendrons **2LS** avec symboles pour exprimer propriétés de monotonicité, d'additivité, et de multiplicativité des fonctions sur les ensembles. Nous appelons le langage résultant **2LSmf**. Nous prouvons que **2LSmf** est décidable par une réduction du problème de la satisfiabilité de ses mots a le problème de la satisfiabilité des mots de **2LS**. En plus, nous prouvons que le langage **2LSmf** est stable-infini en ce qui concerne la sorte des éléments. Donc, en employant une version de la méthode de combinaison de Nelson-Oppen avec sortes, **2LSmf** peut être combiné avec autres langages qui modèlent la sorte des éléments.

Mots-clés : Déduction automatique, Procédures de décision, Théorie calculable des ensembles.

Contents

1 Introduction

Sets are ubiquitous in automated reasoning and program verification. For instance, set theory forms the central kernel of the high-level programming language SETL [SDDS86], as well as of the specification languages Z [Spi88] and B [Abr96]. Even for programming and specification languages that are not entirely based on set theory, sets still play a crucial role because they can be used either at the meta-level or as a data structure.

Because of the importance of sets in automated reasoning, it would be desirable to have an efficient proof system *completely* based on set theory. However, this "dream" is not realizable because:

- the full language of set theory is undecidable;
- when reasoning about sets over elements of a given nature (such as integer, reals, or lists), it is more efficient to target the elements by employing a specialized decision procedure for them.

Therefore, a more realistic approach consists of *designing and implementing a proof system based on the combination of decision procedures for fragments of set theory with decision procedures for elements.*

This approach can be conveniently achieved by using the many-sorted language **2LS** (*two-level syllogistic*), which contains a sort elem for the elements, a sort set for sets of elements, a symbol $=$ for equality, and the set-theoretic constructs membership, inclusion, singleton, union, intersection, and difference. The decidability of **2LS** was proved for the first time by Ferro and Omodeo [FO78].

In this report we introduce the fragment of set theory **2LSmf** (*two-level syllogistic with monotone functions*), which extends **2LS** with free set-to-set function symbols and the following predicates for expressing monotonicity, additivity, and multiplicativity properties of set-to-set functions:

- $inc(f)$, which holds iff f is *increasing*, that is, $a \subseteq b \rightarrow f(a) \subseteq f(b)$, for all sets a, b;
- $dec(f)$, which holds iff f is *decreasing*, that is, $a \subseteq b \rightarrow f(b) \subseteq f(a)$, for all sets a, b;
- $add(f)$, which holds iff f is *additive*, that is, $f(a \cup b) = f(a) \cup f(b)$, for all sets a, b;
- $mul(f)$, which holds iff f is *multiplicative*, that is, $f(a \cap b) = f(a) \cap f(b)$, for all sets a, b;
- $f \preceq g$, which holds iff $f(a) \subseteq g(a)$, for every set a.

We prove that **2LSmf** is decidable by providing a reduction algorithm which maps each sentence of **2LSmf** into an equisatisfiable sentence of **2LS**. Then the decidability of **2LSmf** follows from the decidability of **2LS**.

Furthermore, we prove that the language **2LSmf** is stably infinite[1] with respect to the sort of elements. Therefore, using a many-sorted version of the Nelson-Oppen combination method [TZ04], it is possible to combine **2LSmf** with other languages modeling the sort of elements.

1.1 Related work

Computable set theory [CFO89, COP01] is that area of mathematics and computer science which studies the decidability properties of fragments of set theory. This field was initiated in 1980 by the seminal paper of Ferro, Omodeo, and Schwartz [FOS80a], who proved the decidability of the pure fragment of set-theory **MLSS** (*multi-level syllogistic with singleton*). Intuitively, **MLSS** is a one-sorted version of **2LS** in which all variables are of sort set.

The literature abounds with decidability results for set-theoretic languages involving free function symbols. Ferro, Omodeo, and Schwartz [FOS80b] and Beckert and Hartmer [BH98] proved the decidability of an extension of **MLSS** with free function symbols, but with no monotonicity, additivity, and multiplicativity constructs. Cantone and Zarba [CZ00] proved the decidability of an extension of **2LS** involving monotonicity constructs, but no additivity and multiplicativity constructs.

Our reduction algorithm is an *augmentation* method, that is, a method that uses as a black box a decision procedure for a language L in order to obtain a decision procedure for a nontrivial extension L' of L. Other augmentation methods for set-theoretic languages can be found in [Zar02a, Zar02b].

This report is inspired by the results in [CSZ03], where we proved the decidability of the language **MLSSmf**. Intuitively, **MLSSmf** is the one-sorted version of **2LSmf** in which all variables are of sort set.

Our decision to shift our focus from **MLSSmf** to **2LSmf** is due to the following reasons:

1. As explained above, a one-sorted set-theoretic language is not directly suitable for practical applications in program verification.

2. The correctness proof of **MLSSmf** is extremely complicated and difficult to read. Due to feedback from the research community, we commited ourself to simplify the correctness proof. Indeed, it turned out that a considerable simplification of the technical details could be achieved by considering instead the language **2LSmf**.

1.2 Organization of the report

The report is organized as follows. In Section 2 we formally define the syntax and semantics of the languages **2LS** and **2LSmf**, and we give other useful notions which will be needed in what follows. In Section 3 we present our reduction algorithm for mapping sentences of **2LSmf** into equisatisfiable sentences of **2LS**. In Section 4 we prove that our reduction algorithm is correct, and we assess its complexity. In Section 5 we show how **2LSmf** can

[1] See Definition 22.

be combined with other languages modeling the sort of elements. Finally, in Section 6 we draw conclusions from our work.

2 Preliminaries

2.1 Two-level syllogistic

The language **2LS** (*two-level syllogistic*) is a quantifier-free many-sorted language with two sorts, elem and set, plus the following symbols:

- arbitrarily many variables of sort τ, for each $\tau \in \{\mathsf{elem}, \mathsf{set}\}$;
- the constant $\emptyset$, of sort set (*empty set*);
- the operators
  - $\{\cdot\}$, of sort $\mathsf{elem} \to \mathsf{set}$ (*singleton set*);
  - $\cup, \cap, \setminus$, of sort $\mathsf{set} \times \mathsf{set} \to \mathsf{set}$ (*union*, *intersection*, and *set difference*);
- the predicates
  - $\in$, of sort $\mathsf{elem} \times \mathsf{set}$ (*membership*);
  - $\subseteq$, of sort $\mathsf{set} \times \mathsf{set}$ (*set inclusion*);
- an equality symbol $=_\tau$, for each $\tau \in \{\mathsf{elem}, \mathsf{set}\}$;[2]
- the propositional connectives $\neg$, $\wedge$, $\vee$, $\to$, and $\leftrightarrow$.

2LS-*terms* (respectively, **2LS**-*formulae*) are well-sorted terms (respectively, formulae) constructed using the symbols of the language **2LS**.

Example 1. Let u be an elem-variable, and let x, y be set-variables. Then the expression

$$\neg\,[(x \setminus y = \emptyset \ \wedge\ u \in x) \ \to\ y \neq \emptyset] \tag{1}$$

is an example of a **2LS**-formula. □

An *interpretation* $\mathcal{A}$ is a mapping of the sorts, variables and symbols of the language **2LS** satisfying the following conditions:

- each sort $\tau \in \{\mathsf{elem}, \mathsf{set}\}$ is mapped to a set A_τ such that:
  - $A_{\mathsf{elem}} \neq \emptyset$;
  - $A_{\mathsf{set}} = \mathcal{P}(A_{\mathsf{elem}})$;

[2] We will write $=$ in place of $=_\tau$ when τ is clear from the context.

- each variable x of sort τ is mapped to an element $x^{\mathcal{A}}$ in A_τ;
- the symbols $\emptyset$, $\{\cdot\}$, $\cap$, $\cup$, $\setminus$, $\in$, $\subseteq$ are interpreted according to their standard interpretation;
- $=_\tau$ is interpreted as the identity in A_τ.

Unless otherwise specified, we follow the convention that calligraphic letters $\mathcal{A}$, $\mathcal{B}$, ... denote interpretations, and that the corresponding Roman letters, opportunely subscripted, denote the domains of the interpretations.

For a term t, we denote with $t^{\mathcal{A}}$ the evaluation of t under the interpretation $\mathcal{A}$. Likewise, for a formula φ, we denote with $\varphi^{\mathcal{A}}$ the truth-value of φ under the interpretation $\mathcal{A}$. If T is a set of terms and $\mathcal{A}$ is an interpretation, we denote with $T^{\mathcal{A}}$ the set $\{t^{\mathcal{A}} : t \in T\}$.

A *model* of a **2LS**-formula φ is an interpretation $\mathcal{A}$ such that $\varphi^{\mathcal{A}}$ is true. A **2LS**-formula is *satisfiable* if it has a model.

The *satisfiability problem* of **2LS** is the problem of determining whether or not a given **2LS**-formula is satisfiable. This problem is decidable [FO78].

Example 2. The **2LS**-formula (1) in Example 1 is not satisfiable. In fact, for every interpretation $\mathcal{A}$, if $x^{\mathcal{A}} \setminus y^{\mathcal{A}} = \emptyset$ and $u^{\mathcal{A}} \in x^{\mathcal{A}}$, it must be the case that $u^{\mathcal{A}} \in x^{\mathcal{A}} \cap y^{\mathcal{A}}$, which implies that $y^{\mathcal{A}} \neq \emptyset$. □

Lemma 3 ([Zar04]). *Let Γ be a satisfiable conjuction of* **2LS***-literals, and let U be the collection of* elem*-variables occurring in Γ. Assume that Γ does not contain any literal of the form $s \neq_{\mathsf{set}} t$ or $s \not\subseteq t$.*

Then Γ has a model $\mathcal{A}$ such that $A_{\mathsf{elem}} = U^{\mathcal{A}}$. □

2.2 Two-level syllogistic with monotone functions

In this report we are primarily interested in the satisfiability problem of the language **2LSmf** (*two-level syllogistic with monotone functions*), which extends **2LS** with arbitrarily many free function symbols of sort $\mathsf{set} \to \mathsf{set}$, and the special symbols *inc*, *dec*, *add*, *mul*, and $\preceq$.

The semantics of **2LSmf** is defined on top of the semantics of **2LS**, by requiring that if $\mathcal{A}$ is an interpretation and f is a free function symbol of sort $\mathsf{set} \to \mathsf{set}$, then $f^{\mathcal{A}}$ is a function from A_{set} into A_{set}. Moreover, for any interpretation $\mathcal{A}$, we agree that:

- $inc(f)$ holds in $\mathcal{A}$ if and only if $f^{\mathcal{A}}$ is *increasing*, that is, $a \subseteq b$ implies $f^{\mathcal{A}}(a) \subseteq f^{\mathcal{A}}(b)$, for all sets a, b in A_{set};
- $dec(f)$ holds in $\mathcal{A}$ if and only if $f^{\mathcal{A}}$ is *decreasing*, that is, $a \subseteq b$ implies $f^{\mathcal{A}}(b) \subseteq f^{\mathcal{A}}(a)$, for all sets a, b in A_{set};
- $add(f)$ holds in $\mathcal{A}$ if and only if $f^{\mathcal{A}}$ is *additive*, that is, $f^{\mathcal{A}}(a \cup b) = f^{\mathcal{A}}(a) \cup f^{\mathcal{A}}(b)$, for all sets a, b in A_{set};

- $mul(f)$ holds in $\mathcal{A}$ if and only if $f^{\mathcal{A}}$ is *multiplicative*, that is, $f^{\mathcal{A}}(a \cap b) = f^{\mathcal{A}}(a) \cap f^{\mathcal{A}}(b)$, for all sets a, b in A_{set};
- $f \preceq g$ holds in $\mathcal{A}$ if and only if $f^{\mathcal{A}}(a) \subseteq g^{\mathcal{A}}(a)$, for every set a in A_{set}.

Example 4. Let f, g be two free function symbols of sort $\mathsf{set} \to \mathsf{set}$, and let u be an elem-variable. Then the **2LSmf**-formula

$$\neg\,[(add(f) \ \wedge \ f \preceq g) \ \rightarrow \ f(\emptyset) \subseteq g(\{u\})]$$

is not satisfiable. Intuitively, this is due to the fact that every additive function is also increasing. □

Example 5. Let A be a nonempty set. Then, for every function $f : A \to A$ and every subset B of A, the function $F : \mathcal{P}(A) \to \mathcal{P}(A)$ defined by

$$F(X) = \{f(x) \ \mid \ x \in X \cap B\}$$

is additive and multiplicative. □

Example 6. Let $\Sigma = \{a_1, \ldots, a_n\}$ be an alphabet, and let $A = \Sigma^*$ be the set of words over Σ. Convene that $\Sigma \subseteq \Sigma^*$. In other words, we identify the symbols in Σ with the words in Σ^* of length 1.

Then the function $F : \mathcal{P}(A) \to \mathcal{P}(A)$ defined by

$$F(X) = \{a \in \Sigma \ \mid \ a \text{ occurs in some word } w \in X\}$$

is additive, whereas the function $G : \mathcal{P}(A) \to \mathcal{P}(A)$ defined by

$$G(X) = [F(X)]^* = \{w = x_1 \cdots x_n \ \mid \ x_i \in \Sigma \cap X\,, \text{ for } i = 1 \ldots n\}$$

is multiplicative. □

2.3 Normalized literals

In order to simplify the presentation, we will often consider conjunctions of *normalized* **2LSmf**-literals of the form:

$$\begin{array}{llll} x \subseteq y\,, & x = \{u\}\,, & x = y \cup z\,, & x = y \setminus z\,, \\ x = f(y)\,, & inc(f)\,, & dec(f)\,, & add(f)\,, \\ mul(f)\,, & f \preceq g\,, & & \end{array} \tag{2}$$

where u is an elem-variable, x, y, z are set-variables, and f, g are free function symbols of sort $\mathsf{set} \to \mathsf{set}$.

Let φ be a **2LSmf**-formula. By suitably introducing new variables, it is possible to convert φ into an equisatisfiable formula $\psi_1 \vee \ldots \vee \psi_k$ in disjunctive normal form, where each ψ_i is a conjunction of normalized **2LSmf**-literals of the form (2). Thus, we have the following result.

Lemma 7. *The satisfiability problem of* **2LSmf** *is equivalent to the satisfiability problem of conjunctions of normalized* **2LSmf***-literals of the form* (2). □

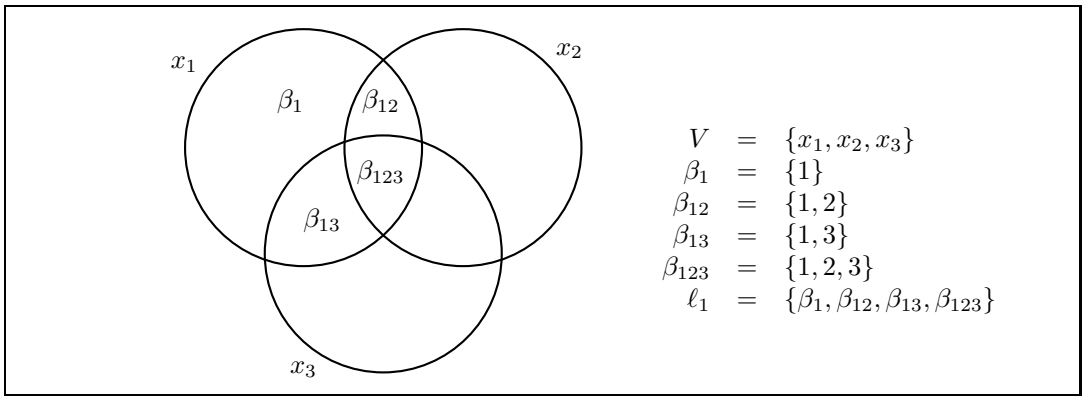

Figure 1: Intuition behind the notation ℓ_j.

3 The reduction algorithm

Let Γ be a conjunction of normalized **2LSmf**-literals, and denote with $V = \{x_1, \ldots, x_n\}$ and F the collections of set-variables and free function symbols occurring in Γ, respectively. In this section we describe a reduction algorithm for converting Γ into an equisatisfiable conjunction Γ^* of **2LS**-formulae.

We will use the following notation. Given a set a, $\mathcal{P}^+(a)$ denotes the set $\mathcal{P}(a) \setminus \{\emptyset\}$. Moreover, we denote with ℓ_j the set

$$\ell_j = \left\{\alpha \in \mathcal{P}^+(\{1, \ldots, n\}) \;\mid\; j \in \alpha\right\}, \qquad \text{for } 1 \leq j \leq n\,.$$

Remark 8. The intuition behind the notation $\mathcal{P}^+$ is as follows. Given a collection $V = \{x_1, \ldots, x_n\}$ of set-variables, each $\alpha \in \mathcal{P}^+(\{1, \ldots, n\})$ can be thought as representing a Venn region. For instance, in Figure 1, the Venn region $(x_1 \cap x_2) \setminus x_3$ is represented by $\beta_{12} = \{1, 2\}$. In general, a set $\alpha \in \mathcal{P}^+(\{1, \ldots, n\})$ represents the Venn region $\bigcap_{i \in \alpha} x_i \setminus \bigcup_{j \notin \alpha} x_j$.

The intuition behind the notation ℓ_j is to represent the union of all Venn regions that are contained in the set x_j. For instance, in Figure 1, the set x_1 is represented by $\ell_1 = \{\beta_1, \beta_{12}, \beta_{13}, \beta_{123}\}$. □

The reduction algorithm is shown in Figure 2, and consists of three steps. In the **first step**, we generate new variables whose intuitive meaning is as follows:

- for each $\alpha \in \mathcal{P}^+(\{1, \ldots, n\})$, the new variable v_α is intented to represent the Venn region $\bigcap_{i \in \alpha} x_i \setminus \bigcup_{j \notin \alpha} x_j$;
- for each $\ell \subseteq \mathcal{P}^+(\{1, \ldots, n\})$, the new variable $w_{f,\ell}$ is intended to represent the value of the function f over the set $\bigcup_{\alpha \in \ell} v_\alpha$.

Reduction algorithm

Input: a conjunction Γ of normalized **2LSmf**-literals.

Output: a conjunction Γ^* of **2LS**-formulae.

Notation:

- $V = \{x_1, \ldots, x_n\}$ is the collection of variables occurring in Γ;
- F is the collection of function symbols occurring in Γ;
- $\mathcal{P}^+(a) = \mathcal{P}(a) \setminus \{\emptyset\}$, for each set a;
- ℓ_j stands for the set $\{\alpha \in \mathcal{P}^+(\{1, \ldots, n\}) \mid j \in \alpha\}$, for $1 \leq j \leq n$.

Step 1. Generate the following new variables:

$$v_\alpha\,, \qquad \text{for each } \alpha \in \mathcal{P}^+(\{1, \ldots, n\})\,,$$
$$w_{f,\ell}\,, \qquad \text{for each } f \in F \text{ and } \ell \subseteq \mathcal{P}^+(\{1, \ldots, n\})\,.$$

Step2. Add to Γ the following **2LS**-formulae:

$$v_\alpha = \bigcap_{i \in \alpha} x_i \setminus \bigcup_{j \notin \alpha} x_j\,, \qquad \text{for each } \alpha \in \mathcal{P}^+(\{1, \ldots, n\})\,,$$

and

$$\bigcup_{\alpha \in \ell} v_\alpha = \bigcup_{\beta \in m} v_\beta \rightarrow w_{f,\ell} = w_{f,m}\,, \qquad \text{for each } f \in F \text{ and } \ell, m \subseteq \mathcal{P}^+(\{1, \ldots, n\})\,.$$

Step 3. Replace the **2LSmf**-literals in Γ containing free function symbols with **2LS**-formulae as follows:

$$\begin{array}{lcl}
x_i = f(x_j) & \Longrightarrow & x_i = w_{f,\ell_j} \\
inc(f) & \Longrightarrow & \bigwedge_{\ell \subseteq m} (w_{f,\ell} \subseteq w_{f,m}) \\
dec(f) & \Longrightarrow & \bigwedge_{\ell \subseteq m} (w_{f,m} \subseteq w_{f,\ell}) \\
add(f) & \Longrightarrow & \bigwedge_{\ell,m} (w_{f,\ell \cup m} = w_{f,\ell} \cup w_{f,m}) \\
mul(f) & \Longrightarrow & \bigwedge_{\ell,m} (w_{f,\ell \cap m} = w_{f,\ell} \cap w_{f,m}) \\
f \preceq g & \Longrightarrow & \bigwedge_{\ell} (w_{f,\ell} \subseteq w_{g,\ell})
\end{array}$$

Figure 2: The reduction algorithm.

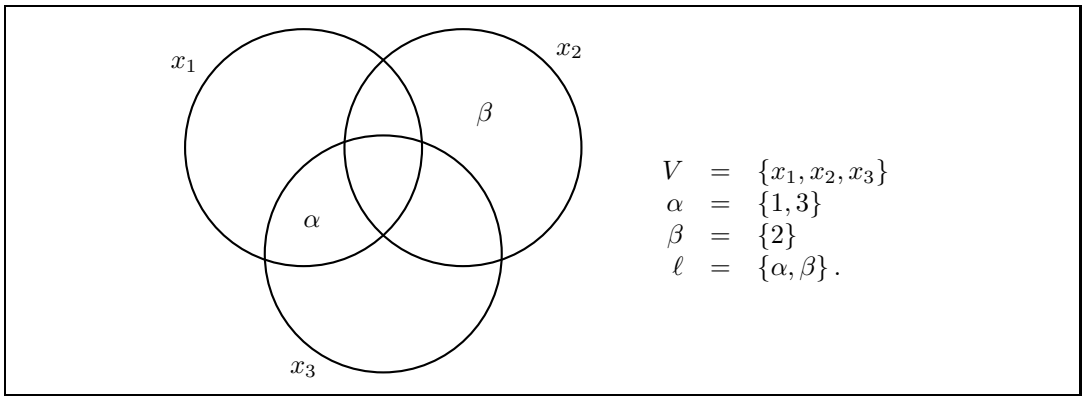

Figure 3: Intuition behind the generated variables v_α and $w_{f,\ell}$.

Example 9. Let $V = \{x_1, x_2, x_3\}$, $\alpha = \{1, 3\}$, and $\beta = \{2\}$, as graphically represented in Figure 3. Then v_α represents $(x_1 \cap x_3) \setminus x_2$, whereas v_β represents $x_2 \setminus (x_1 \cup x_3)$.

Next, if $\ell = \{\alpha, \beta\}$ then $w_{f,\ell}$ represents $f(v_\alpha \cup v_\beta)$. □

In the **second step** of the reduction algorithm, we add to Γ appropriate **2LS**-formulae whose purpose is to model the variables v_α and $w_{f,\ell}$ according to their intuitive meaning. In particular, the variables $w_{f,\ell}$ are modeled by noticing that for each $\ell, m \subseteq \mathcal{P}^+(\{1, \ldots, n\})$, if $\bigcup_{\alpha \in \ell} v_\alpha = \bigcup_{\beta \in m} v_\beta$ then $f\left(\bigcup_{\alpha \in \ell} v_\alpha\right) = f\left(\bigcup_{\beta \in m} v_\beta\right)$.

Finally, in the **third step** of the reduction algorithm, we remove from Γ all **2LSmf**-literals involving free function symbols. This is done by replacing all **2LSmf**-literals of the form $x_i = f(x_j)$, $inc(f)$, $dec(f)$, $add(f)$, $mul(f)$, and $f \preceq g$ with **2LS**-literals involving only the variables x_i and the new variables $w_{f,\ell}$.

We claim that our reduction algorithm is correct. More specifically, we claim that if Γ^* is the result of applying to Γ our reduction algorithm then:

- the reduction is *sound*, namely, if Γ is satisfiable, so is Γ^*;
- the reduction is *complete*, namely, if Γ^* is satisfiable, so is Γ.

The next section proves that our reduction algorithm is sound and complete, and therefore it yields a decision procedure for **2LSmf**.

Example 10. Let us use our reduction algorithm to prove that the conjunction of **2LSmf**-literals

$$\Delta = \left\{ \begin{array}{l} inc(f)\,, \\ x_1 \subseteq x_2\,, \\ f(x_1) \not\subseteq f(x_2) \end{array} \right\}$$

is not satisfiable.

To do so, we first convert Δ into the following equisatisfiable conjunction of **2LSmf**-literals:

$$\Gamma = \left\{ \begin{array}{l} inc(f)\,, \\ x_1 \subseteq x_2\,, \\ x_3 = f(x_1)\,, \\ x_4 = f(x_2)\,, \\ x_5 = \{u\}\,, \\ x_6 = x_3 \setminus x_4\,, \\ x_5 \subseteq x_6 \end{array} \right\}.$$

Then, let Γ^* be the result of applying our reduction algorithm to Γ. Clearly, we do not have neither the space nor the patience to write here all formulae in Γ^*. However, we can explain succinctly why Γ^* is not satisfiable.

For $j = 1, \ldots, 6$, denote with ℓ_j the set

$$\ell_j = \left\{ \alpha \in \mathcal{P}^+(\{1, \ldots, 6\}) \;\mid\; j \in \alpha \right\},$$

and note that Γ^* entails the formulae

$$x_j = \bigcup_{\alpha \in \ell_j} v_\alpha\,, \qquad\qquad \text{for } j = 1, \ldots, 6\,.$$

Next, let

$$\ell_{12} = \left\{ \alpha \in \mathcal{P}^+(\{1, \ldots, 6\}) \;\mid\; \{1, 2\} \subseteq \alpha \right\},$$

and note that Γ^* entails the formula

$$x_1 \cap x_2 = \bigcup_{\alpha \in \ell_{12}} v_\alpha\,.$$

Since Γ^* contains the formula $x_1 \subseteq x_2$, Γ^* entails $x_1 = x_1 \cap x_2$. It follows that Γ^* also entails

$$\bigcup_{\alpha \in \ell_1} v_\alpha = \bigcup_{\alpha \in \ell_{12}} v_\alpha\,,$$

as well as

$$w_{f,\ell_1} = w_{f,\ell_{12}}\,.$$

Since $inc(f)$ is in Γ and $\ell_{12} \subseteq \ell_2$, Γ^* contains the literal $w_{f,\ell_{12}} \subseteq w_{f,\ell_2}$. Thus, Γ^* entails $w_{f,\ell_1} \subseteq w_{f,\ell_2}$.

Since Γ^* contains the literals $x_3 = w_{f,\ell_1}$ and $x_4 = w_{f,\ell_2}$, it follows that Γ^* entails $x_3 \subseteq x_4$. But since Γ^* contains the literals $x_5 = \{u\}, x_6 = x_3 \setminus x_4$, and $x_5 \subseteq x_6$, it follows that Γ^* also entails $x_3 \not\subseteq x_4$. Thus, Γ^* is not satisfiable. □

4 Correctness

4.1 Soundness

Let Γ be a satisfiable conjunction of normalized **2LSmf**-literals, and let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2. The key idea of the soundness proof is that, given a model $\mathcal{A}$ of Γ, a model $\mathcal{B}$ of Γ^* can be constructed in the most natural way if we remember the intuitive meaning of the variables v_α and $w_{f,\ell}$.

Lemma 11 (Soundness). *Let Γ be a conjunction of normalized* **2LSmf***-literals, and let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2. Then if Γ is satisfiable, so is Γ^*.* □

Proof. Let $\mathcal{A}$ be a model of Γ, and denote with $V = \{x_1, \ldots, x_n\}$ and F the collections of set-variables and free function symbols occurring in Γ, respectively. We claim that the interpretation $\mathcal{B}$ defined by letting

$$\begin{aligned} B_{\mathsf{elem}} &= A_{\mathsf{elem}}\,, \\ B_{\mathsf{set}} &= A_{\mathsf{set}} = \mathcal{P}(A_{\mathsf{elem}})\,, \end{aligned}$$

and

$$\begin{aligned} u^{\mathcal{B}} &= u^{\mathcal{A}}\,, && \text{for each elem-variable } u\,, \\ x_i^{\mathcal{B}} &= x_i^{\mathcal{A}}\,, && \text{for each } i \in \{1, \ldots, n\}\,, \\ v_\alpha^{\mathcal{B}} &= \bigcap_{i \in \alpha} x_i^{\mathcal{A}} \setminus \bigcup_{j \notin \alpha} x_j^{\mathcal{A}}\,, && \text{for each } \alpha \in \mathcal{P}^+(\{1, \ldots, n\})\,, \\ w_{f,\ell}^{\mathcal{B}} &= f^{\mathcal{A}}\left(\bigcup_{\alpha \in \ell} v_\alpha^{\mathcal{B}}\right)\,, && \text{for each } f \in F \text{ and } \ell \subseteq \mathcal{P}^+(\{1, \ldots, n\}) \end{aligned}$$

is a model of Γ^*.

To see this, note that formulae in Γ^* can be partitioned into the following categories:

(I) literals of the form $x \subseteq y$, $x = \{u\}$, $x = y \cup z$, and $x = y \setminus z$ originally present in Γ;

(II) literals of the form $v_\alpha = \bigcup_{i \in \alpha} x_i \setminus \bigcap_{j \notin \alpha} x_j$;

(III) formulae of the form $\bigcup_{\alpha \in \ell} v_\alpha = \bigcup_{\beta \in m} v_\beta \rightarrow w_{f,\ell} = w_{f,m}$;

(IV) literals of the form $x_i = w_{f,\ell_j}$, which replace literals of the form $x_i = f(x_j)$ in Γ;

(V) conjunctions replacing literals of the form $inc(f)$, $dec(f)$, $add(f)$, $mul(f)$, and $f \preceq g$ in Γ.

Literals of the form (I) are true in $\mathcal{B}$ because they contain only variables already occurring in Γ, and by construction $\mathcal{B}$ agrees with $\mathcal{A}$ on all such variables.

Concerning literals of the form (II), we have $v_\alpha^{\mathcal{B}} = \bigcap_{i\in\alpha} x_i^{\mathcal{A}} \setminus \bigcup_{j\notin\alpha} x_j^{\mathcal{A}} = \bigcap_{i\in\alpha} x_i^{\mathcal{B}} \setminus \bigcup_{j\notin\alpha} x_j^{\mathcal{B}}$.

Concerning literals of the form (III), let $\bigcup_{\alpha\in\ell} v_\alpha^{\mathcal{B}} = \bigcup_{\beta\in m} v_\beta^{\mathcal{B}}$. Then $f^{\mathcal{A}}\left(\bigcup_{\alpha\in\ell} v_\alpha^{\mathcal{B}}\right) = f^{\mathcal{A}}\left(\bigcup_{\beta\in m} v_\beta^{\mathcal{B}}\right)$, for every $f \in F$. Thus, $w_{f,\ell}^{\mathcal{B}} = f^{\mathcal{A}}\left(\bigcup_{\alpha\in\ell} v_\alpha^{\mathcal{B}}\right) = f^{\mathcal{A}}\left(\bigcup_{\beta\in m} v_\beta^{\mathcal{B}}\right) = w_{f,m}^{\mathcal{B}}$, for every $f \in F$.

Concerning literals of the form (IV), let $x_i = f(x_j)$ be in Γ. Then we have $x_i^{\mathcal{B}} = x_i^{\mathcal{A}} = f^{\mathcal{A}}\left(x_j^{\mathcal{A}}\right) = f^{\mathcal{A}}\left(\bigcup_{\alpha\in\ell_j} v_\alpha^{\mathcal{B}}\right) = w_{f,\ell_j}^{\mathcal{B}}$, where $\ell_j = \{\alpha \in \mathcal{P}^+(\{1,\ldots,n\}) \mid j \in \alpha\}$.

Concerning literals of the form (V), consider a literal of the form $inc(f)$ in Γ and suppose that $\ell \subseteq m$, with $\ell, m \subseteq \mathcal{P}^+(\{1,\ldots,n\})$. We want to show that the literal $w_{f,\ell} \subseteq w_{f,m}$ occurring in Γ^* is true in $\mathcal{B}$. To do so, observe that, since $\ell \subseteq m$, we have $\bigcup_{\alpha\in\ell} v_\alpha^{\mathcal{B}} \subseteq \bigcup_{\beta\in m} v_\beta^{\mathcal{B}}$. Since $f^{\mathcal{A}}$ is increasing, we also have $f^{\mathcal{A}}\left(\bigcup_{\alpha\in\ell} v_\alpha^{\mathcal{B}}\right) \subseteq f^{\mathcal{A}}\left(\bigcup_{\beta\in m} v_\beta^{\mathcal{B}}\right)$, and therefore $w_{f,\ell}^{\mathcal{B}} \subseteq w_{f,m}^{\mathcal{B}}$. The cases regarding conjunctions introduced in Γ^* to replace literals of the form $dec(f)$, $add(f)$, $mul(f)$ and $add(f)$ can be treated similarly. ∎

4.2 Completeness

Let Γ be a conjunction of normalized **2LSmf**-literals. As before, let us denote with $V = \{x_1,\ldots,x_n\}$ and F the collections of set-variables and free function symbols occurring in Γ, respectively. Also, let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2. To show the completeness of our reduction algorithm, we need to prove that if Γ^* is satisfiable, so is Γ.

To do so, let $\mathcal{B}$ be a model of Γ^*, and let us start to define an interpretation $\mathcal{A}$ by letting

$$A_{\mathsf{elem}} = B_{\mathsf{elem}}\,,$$
$$A_{\mathsf{set}} = B_{\mathsf{set}} = \mathcal{P}(B_{\mathsf{elem}})\,,$$

and

$$u^{\mathcal{A}} = u^{\mathcal{B}}\,, \qquad \text{for each elem-variable } u\,,$$
$$x_i^{\mathcal{A}} = x_i^{\mathcal{B}}\,, \qquad \text{for each } i = 1,\ldots,n\,.$$

In order to define $\mathcal{A}$ over the free function symbols in F, let us recall that the intuitive meaning of a variable of the form $w_{f,\ell}$ is to represent the expression $f(\bigcup_{\alpha\in\ell} v_\alpha)$. Thus, our definition of $f^{\mathcal{A}}$ should satisfy the property that $f^{\mathcal{A}}(\bigcup_{\alpha\in\ell} v_\alpha^{\mathcal{B}}) = w_{f,\ell}^{\mathcal{B}}$, for every $\ell \subseteq \mathcal{P}^+(\{1,\ldots,n\})$. But how do we define $f^{\mathcal{A}}(a)$ in the more general case in which a is not the union of sets of the form $v_\alpha^{\mathcal{B}}$? The idea is to define opportunely a *discretization function* $\lambda : B_{\mathsf{set}} \to \mathcal{P}(\mathcal{P}^+(\{1,\ldots,n\}))$, and then let

$$f^{\mathcal{A}}(a) = w_{f,\lambda(a)}^{\mathcal{B}}\,, \qquad \text{for each } f \in F \text{ and each set } a \in A_{\mathsf{set}} = B_{\mathsf{set}}\,.$$

To achieve completeness, we need a *good* discretization function.

Definition 12. Let Γ be a conjunction of normalized **2LSmf**-literals and let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2. A discretization function $\lambda : B_{\mathsf{set}} \to \mathcal{P}(\mathcal{P}^+(\{1,\ldots,n\}))$ is GOOD with respect to a model $\mathcal{B}$ of Γ^* if the following conditions hold:

(A) λ is increasing;

(B) λ is additive;

(C) λ is multiplicative;

(D) if $a = \bigcup_{\alpha\in\ell} v^{\mathcal{B}}_{\alpha}$ then $a = \bigcup_{\alpha\in\lambda(a)} v^{\mathcal{B}}_{\alpha}$, for each $\ell \subseteq \mathcal{P}^+(\{1,\ldots,n\})$. □

Lemma 13. *Let Γ be a conjunction of normalized* **2LSmf***-literals, let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2, and let $\mathcal{B}$ be a model of Γ^*. Assume that there exists a discretization function $\lambda : B_{\mathsf{set}} \to \mathcal{P}(\mathcal{P}^+(\{1,\ldots,n\}))$ which is good with respect to $\mathcal{B}$.*

Then Γ is satisfiable. □

PROOF. Let $V = \{x_1,\ldots,x_n\}$ (respectively, F) be the collection of variables (respectively, free function symbols) occurring in Γ. We now prove that the interpretation $\mathcal{A}$ defined by letting

$$\begin{aligned} A_{\mathsf{elem}} &= B_{\mathsf{elem}}\,, \\ A_{\mathsf{set}} &= B_{\mathsf{set}} = \mathcal{P}(B_{\mathsf{elem}})\,, \end{aligned}$$

and

$$\begin{aligned} u^{\mathcal{A}} &= u^{\mathcal{B}}\,, && \text{for each } \mathsf{elem}\text{-variable } u\,, \\ x_i^{\mathcal{A}} &= x_i^{\mathcal{B}}\,, && \text{for each } i = 1,\ldots,n\,, \\ f^{\mathcal{A}}(a) &= w^{\mathcal{B}}_{f,\lambda(a)}\,, && \text{for each } f \in F \text{ and each set } a \in A_{\mathsf{set}}\,, \end{aligned}$$

is a model of Γ by showing that $\mathcal{A}$ satisfies all literals occurring in Γ.

Literals of the form $x \subseteq y$, $x = \{u\}$, $x = y \cup z$, $x = y \setminus z$. These literals are true in $\mathcal{A}$ since $\mathcal{A}$ agrees with $\mathcal{B}$ on all variables occurring in Γ.

Literals of the form $x_i = f(x_j)$. Let $a = \bigcup_{\alpha\in\ell_j} v^{\mathcal{B}}_{\alpha}$, where $\ell_j = \{\alpha \in \mathcal{P}^+(\{1,\ldots,n\}) \mid j \in \alpha\}$. Then, by property (D) of Definition 12, we have $a = \bigcup_{\alpha\in\lambda(a)} v^{\mathcal{B}}_{\alpha}$, and therefore $w^{\mathcal{B}}_{f,\ell_j} = w^{\mathcal{B}}_{f,\lambda(a)}$. Since the literal $x_i = w_{f,\ell_j}$ is in Γ^*, we have $x_i^{\mathcal{A}} = x_i^{\mathcal{B}} = w^{\mathcal{B}}_{f,\ell_j} = w^{\mathcal{B}}_{f,\lambda(a)} = f^{\mathcal{A}}(a) = f^{\mathcal{A}}\left(\bigcup_{\alpha\in\ell_j} v^{\mathcal{B}}_{\alpha}\right) = f^{\mathcal{A}}\left(x_j^{\mathcal{B}}\right) = f^{\mathcal{A}}\left(x_j^{\mathcal{A}}\right)$.

Literals of the form $inc(f)$, $dec(f)$, $add(f)$, $mul(f)$ **and** $f \preceq g$. Let the literal $inc(f)$ be in Γ, and let $a \subseteq b$. Since λ is increasing, $\lambda(a) \subseteq \lambda(b)$, so that the literal $w_{f,\lambda(a)} \subseteq w_{f,\lambda(b)}$ must be in Γ^*. We have $f^{\mathcal{A}}(a) = w^{\mathcal{B}}_{f,\lambda(a)} \subseteq w^{\mathcal{B}}_{f,\lambda(b)} = f^{\mathcal{A}}(b)$, implying that $f^{\mathcal{A}}$ is increasing. Similarly, one can prove that also all literals of the form $dec(f)$, $add(f)$, $mul(f)$ and $f \preceq g$ in Γ are satisfied by $\mathcal{A}$. ■

Remark 14. By a careful analysis of the above proof, it turns out that the monotonicity of the discretization function λ is needed only to show the satisfiability of literals of the form $inc(f)$ and $dec(f)$. Likewise, the additivity of λ is needed only for literals of the form $add(f)$, whereas the multiplicativity of λ is needed only for literals of the form $mul(f)$. Finally, property (D) of λ (cf. Definition 12) is only needed to prove the satisfiability of literals of the form $x_i = f(x_j)$. □

Lemma 13 shows that the existence of good discretization functions is enough to ensure the completeness of our reduction algorithm. But how do we define good discretization functions?

As a first attempt, given an arbitrary model $\mathcal{B}$ of Γ^*, let us define

$$\lambda^+_{\mathcal{B}}(a) = \{\alpha \in \mathcal{P}^+(\{1,\ldots,n\}) \mid v^{\mathcal{B}}_\alpha \cap a \neq \emptyset\}, \qquad \text{for each set } a \in B_{\mathsf{set}}.$$

It is easy to see that $\lambda^+_{\mathcal{B}}$ satisfies properties (A), (B), and (D) of Definition 12. However, in general $\lambda^+_{\mathcal{B}}$ is not multiplicative. As a counter-example, assume that there exist two disjoint sets a, b in B_{set} and some $\alpha \subseteq \mathcal{P}^+(\{1,\ldots,n\})$ such that $a \cap v^{\mathcal{B}}_\alpha \neq \emptyset$ and $b \cap v^{\mathcal{B}}_\alpha \neq \emptyset$. Then $\alpha \in \lambda^+_{\mathcal{B}}(a) \cap \lambda^+_{\mathcal{B}}(b)$ but $\lambda^+_{\mathcal{B}}(a \cap b) = \emptyset$.

By Remark 14, in the proof of Lemma 13 the hypothesis that $\lambda^+_{\mathcal{B}}$ is multiplicative is used only to show that the literals of the form $mul(f)$ in Γ are satisfied by $\mathcal{A}$. Therefore, if we define $\mathbf{2LSmf}^+$ to be the language obtained from **2LSmf** by removing the symbol mul, we get the following partial result.

Lemma 15. *Let Γ be a conjunction of normalized $\mathbf{2LSmf}^+$-literals and let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2. Then if Γ^* is satisfiable, so is Γ.* □

Combining Lemma 11 and Lemma 15 we obtain the decidability of $\mathbf{2LSmf}^+$.

Theorem 16. *The satisfiability problem for $\mathbf{2LSmf}^+$ is decidable.* □

As a second attempt to find a good discretization function, let us define

$$\lambda^\times_{\mathcal{B}}(a) = \{\alpha \in \mathcal{P}^+(\{1,\ldots,n\} \mid \emptyset \neq v^{\mathcal{B}}_\alpha \subseteq a\}, \qquad \text{for each set } a \in B_{\mathsf{set}}.$$

It is easy to see that $\lambda^\times_{\mathcal{B}}$ satisfies properties (A), (C), and (D) of Definition 12. However, in general $\lambda^\times_{\mathcal{B}}$ is not additive. As a counter-example, assume that there exist two sets a, b and some $\alpha \subseteq \mathcal{P}^+(\{1,\ldots,n\})$ such that $v^{\mathcal{B}}_\alpha \subseteq a \cup b$, $v^{\mathcal{B}}_\alpha \not\subseteq a$, and $v^{\mathcal{B}}_\alpha \not\subseteq b$. Then $\alpha \in \lambda^\times_{\mathcal{B}}(a \cup b)$ but $\alpha \notin \lambda^\times_{\mathcal{B}}(a) \cup \lambda^\times_{\mathcal{B}}(b)$.

By Remark 14, in the proof of Lemma 13 the hypothesis that $\lambda^\times_{\mathcal{B}}$ is additive is used only to show that the literals of the form $add(f)$ in Γ are satisfied by $\mathcal{A}$. Therefore, if we define

2LSmf$^\times$ to be the language obtained from **2LSmf** by removing the symbol *add*, we get the following partial result.

Lemma 17. *Let* Γ *be a conjunction of normalized* **2LSmf**$^\times$*-literals and let* Γ^* *be the result of applying to* Γ *the reduction algorithm in Figure 2. Then if* Γ^* *is satisfiable, so is* Γ. □

Combining Lemma 11 and Lemma 17 we obtain the decidability of **2LSmf**$^\times$.

Theorem 18. *The satisfiability problem for* **2LSmf**$^\times$ *is decidable.* □

So far, it appears as neither $\lambda^+_{\mathcal{B}}$ nor $\lambda^\times_{\mathcal{B}}$ are good discretization functions. However, assume that we have a model $\mathcal{B}$ of Γ^* such that:

$$|v^{\mathcal{B}}_\alpha| \leq 1\,, \qquad \text{for each } \alpha \in \mathcal{P}^+(\{1,\ldots,n\})\,. \tag{3}$$

Then it is easy to see that in this case $\lambda^+_{\mathcal{B}}$ and $\lambda^\times_{\mathcal{B}}$ coincide, and therefore they are both additive and multiplicative. Thus, both $\lambda^+_{\mathcal{B}}$ and $\lambda^\times_{\mathcal{B}}$ are good discretization functions with respect to any model $\mathcal{B}$ of Γ^* satisfying (3).

But do models of Γ^* satisfying (3) exist? The following lemma gives an affirmative answer to this question.

Lemma 19. *Let* Γ *be a conjunction of normalized* **2LSmf***-literals, and let* Γ^* *be the result of applying to* Γ *the reduction algorithm in Figure 2. Assume also that* Γ^* *is satisfiable. Then there exists a model* $\mathcal{B}$ *of* Γ^* *such that* $|v^{\mathcal{B}}_\alpha| \leq 1$*, for each* $\alpha \in \mathcal{P}^+(\{1,\ldots,n\})$. □

Proof. Let U be the collection of elem-variables occurring in Γ, and let $V = \{x_1,\ldots,x_n\}$ be the collection of set-variables occurring in Γ.

Note that U is also the collection of elem-variables occurring in Γ^*. Furthermore, Γ^* is a conjunction of **2LS**-literals that does not contain any literal of the form $s \neq_{\mathsf{set}} t$ or $s \not\subseteq t$. Therefore, by Lemma 3, there exists a model $\mathcal{B}$ of Γ such that $B_{\mathsf{elem}} = U^{\mathcal{B}}$.

Let $\alpha \in \mathcal{P}^+(\{1,\ldots,n\})$, and assume that $v^{\mathcal{B}}_\alpha \neq \emptyset$. Then there exists an element $a \in v^{\mathcal{B}}_\alpha$ such that:

- $a = u^{\mathcal{B}}$, for some elem-variable u in U;
- a literal of the form $x_i = \{u\}$ is in Γ^*.

But then $v^{\mathcal{B}}_\alpha \subseteq x^{\mathcal{B}}_i$, which implies $|v^{\mathcal{B}}_\alpha| = 1$. ■

Combining Lemma 19 with Lemma 13, we can finally obtain the completeness of our reduction algorithm, using either $\lambda^+_{\mathcal{B}}$ or $\lambda^\times_{\mathcal{B}}$ as a good discretization function with respect to a model $\mathcal{B}$ of Γ^* satisfying (3).

Lemma 20 (Completeness). *Let* Γ *be a conjunction of normalized* **2LSmf***-literals and let* Γ^* *be the result of applying to* Γ *the reduction algorithm in Figure 2. Then if* Γ^* *is satisfiable, so is* Γ. □

Combining Lemma 11 and Lemma 20, we obtain the decidability of **2LSmf**.

Theorem 21 (Decidability). *The satisfiability problem for* **2LSmf** *is decidable.* □

4.3 Complexity

Let Γ be a conjunction of normalized **2LSmf**-literals containing n variables and m free function symbols, and let Γ^* be the result of applying to Γ the reduction algorithm in Figure 2.

Then Γ^* involves $\mathcal{O}(2^n)$ variables of type v_α and $\mathcal{O}(m \cdot 2^{2^n})$ variables of type $w_{f,\ell}$. Moreover, the collective size of all formulae generated in Step 2 is bounded by

$$\mathcal{O}(n \cdot 2^n) \ + \ \mathcal{O}(m \cdot 2^n \cdot 2^{2^{n+1}}) ,$$

whereas the collective size of all formulae generated in Step 3 is bounded by

$$\mathcal{O}(p \cdot 2^{2^{n+1}}) ,$$

where p is the number of literals in Γ of the form

$$\begin{array}{lll} x = f(y) , & inc(f) , & dec(f) , \\ add(f) , & mul(f) , & f \preceq g . \end{array}$$

Thus, if we denote with K the size of Γ, since $m, n, p \leq K$, we have the following upper bound on the size of Γ^*:

$$\mathcal{O}(K \cdot 2^K \cdot 2^{2^{K+1}}) .$$

Finally, to estimate the complexity of our decision procedure, we need to take into account that the formula Γ^* must then be tested for satisfiability, and it is known that the satisfiability problem for **2LS** is *NP*-complete [Zar04]. Though the satisfiability test for **2LS** is quite efficient in practice, it becomes very expensive when it runs on such large formulae as Γ^*.

5 Combination results

In program verification, one often needs to reason about logical formulae that involve sets over elements of a given nature such as integers, reals, or lists. In order to reason about these elements, the language **2LSmf** is not expressive enough. In fact, the only **2LSmf**-formulae expressing pure facts over the elements are just boolean combinations of atoms of the form $u =_{\mathsf{elem}} v$.

Thus, for instance, when one wants to reason about sets of integers, it is better to use a many-sorted language L that properly extends **2LSmf** with symbols that are specific to the domain of integers. This language L can be seen as the union of two languages, that is, $L = \mathbf{2LSmf} \cup L_1$, where L_1 is a language that models the sort elem as the set of integers.

Then, in order to decide the satisfiability problem of $L = \mathbf{2LSmf} \cup L_1$, one needs to *combine* our decision procedure for the satisfiability problem of **2LSmf** with a decision procedure for the satisfiability problem of L_1.

At first sight, one might think that, in order to combine **2LSmf** with L_1, it suffices to employ the classic combination method due to Nelson and Oppen [NO79]. Unfortunately, this is not the case since, technically speaking, the original Nelson-Oppen method is restricted to first-order logic, and does not work for many-sorted logic.

Nevertheless, the Nelson-Oppen method has recently been extended to order-sorted logic—a logic that subsumes many-sorted logic—by Tinelli and Zarba [TZ04]. In particular, the following combination result holds for many-sorted logic.

Definition 22. A many-sorted language L is STABLY INFINITE with respect to a set of sorts S if for every satisfiable quantifier-free formula φ in the language L, there exists a model $\mathcal{A}$ of φ such that A_s is infinite, for each sort s in S. □

Theorem 23 ([TZ04]). *Let L_1 and L_2 be two many-sorted languages, and let S be the set of sorts shared by L_1 and L_2. Assume that:*

(i) for any quantifier-free formula φ in L_i, it is possible to decide whether or not φ is satisfiable in L_i, for $i = 1, 2$;

(ii) L_1 and L_2 do not share logical symbols besides equality and the sorts in S;

(iii) L_1 and L_2 are stably infinite with respect to S.

Then it is possible to decide the satisfiability of any quantifier-free formula φ in $L_1 \cup L_2$. □

In view of Theorem 23, if we want to combine the language **2LSmf** with another language L_1 modelling the sort elem of elements, it becomes essential that **2LSmf** be stably infinite with respect to the sort elem. The next lemma shows that this is indeed the case.

Lemma 24. *Let φ be a satisfiable* **2LSmf***-formula. Then there exists a model $\mathcal{A}$ of φ such that A_{elem} is infinite.* □

PROOF. Let $\mathcal{A}$ be a model of φ, and assume that A_{elem} is finite. Then we can construct a model $\mathcal{B}$ of φ such that B_{elem} is infinite as follows.

Fix an infinite set X of elements disjoint from A_{elem}, and let

$$B_{\mathsf{elem}} = A_{\mathsf{elem}} \cup X\,,$$
$$B_{\mathsf{set}} = \mathcal{P}(B_{\mathsf{elem}})\,,$$

and

$$x^{\mathcal{B}} = x^{\mathcal{A}}\,, \qquad \text{for each variable } x\,,$$
$$f^{\mathcal{B}}(a) = f^{\mathcal{A}}(a \setminus X)\,, \qquad \text{for each free function symbol } f \text{ and } a \in B_{\mathsf{set}}\,.$$

Note that, by construction, $t^{\mathcal{B}} = t^{\mathcal{A}}$, for each term t. Furthermore, we claim that $\psi^{\mathcal{A}} = \psi^{\mathcal{B}}$, for every atomic formula ψ. This is trivial for atomic formulae of the form $u =_{\mathsf{elem}} v$, $s =_{\mathsf{set}} t$, $u \in t$, $s \subseteq t$.

Concerning an atomic formula of the form $inc(f)$, we need to show that $f^{\mathcal{A}}$ is increasing if and only if $f^{\mathcal{B}}$ is increasing. Assume first that $f^{\mathcal{A}}$ is increasing, and let $a, b \in B_{\mathsf{set}}$ such that $a \subseteq b$. Then $a \setminus X \subseteq b \setminus X$. Thus, $f^{\mathcal{B}}(a) = f^{\mathcal{A}}(a \setminus X) \subseteq f^{\mathcal{A}}(b \setminus X) = f^{\mathcal{B}}(b)$. Vice versa, assume that $f^{\mathcal{B}}$ is increasing, and let $a, b \in A_{\mathsf{set}}$ such that $a \subseteq b$. Then $f^{\mathcal{A}}(a) = f^{\mathcal{B}}(a) \subseteq f^{\mathcal{B}}(b) = f^{\mathcal{A}}(b)$.

Similarly, one can show that $\mathcal{A}$ and $\mathcal{B}$ also agree on all atomic formulae of the form $dec(f)$, $add(f)$, $mul(f)$, and $f \preceq g$.

Finally, by structural induction, one can show that $\psi^{\mathcal{A}} = \psi^{\mathcal{B}}$, for each formula ψ. It follows that $\varphi^{\mathcal{A}} = \varphi^{\mathcal{B}}$. ■

Combining Lemma 24 with Theorem 23, we obtain the following combination result.

Theorem 25 (Combination). *Let L be any many-sorted language satisfying the following properties:*

(i) for each quantifier-free formula φ in L, it is possible to decide whether or not φ is satisfiable in L;

(ii) L does not contain the sort set*;*

(iii) L is stably-infinite with respect to the sort elem*.*

Then the satisfiability problem for **2LSmf** $\cup\, L$ *is decidable.* □

6 Conclusion

We presented a decision procedure for the many-sorted fragment of set theory **2LSmf** extending **2LS** with constructs for expressing monotonicity, additivity, and multiplicativity properties of set-to-set functions. Our decision procedure consists of a reduction algorithm which maps each sentence of **2LSmf** into an equisatisfiable sentence of **2LS**. Then the decidability of **2LSmf** follows from the decidability of **2LS**.

Furthermore, we showed that by using a many-sorted version of the Nelson-Oppen combination method, it is possible to combine **2LSmf** with other languages modeling the sort of elements.

Our work can have applications in an interactive proof environment in which the user helps the system by telling which expressions are monotonic, while our decision procedure performs the tedious combinatoric steps. As an example, consider the formula

$$\{f(u) \mid u \in x \setminus z\} \subseteq \{f(u) \mid u \in (x \cup y) \setminus z\}, \tag{4}$$

where u is an elem-variable, x, y, z are set-variables, and f is a function symbol of sort elem $\to$ elem. When proving the validity of (4), the user can instruct the system with the insight that the function

$$F(w) = \{f(u) \mid u \in w \setminus z\}$$

is increasing in the set-variable w. Then, the system would conclude that to prove that (4) is valid, it suffices to prove that

$$inc(F) \ \rightarrow \ F(x) \subseteq F(x \cup y) \tag{5}$$

is valid. Since (5) is a **2LSmf**-formula, its validity can be automatically proven by our decision procedure.

Future directions of research may involve extensions of our decision procedure to handle other constructs related to set-to-set functions, such as injectivity and surjectivity of functions, as well as a fixed-point operator on monotone functions. Also, further work is needed to make our reduction algorithm more efficient.

Acknowledgements

This work was in part supported by the project GECCOO in the context of the French national research program *ACI Sécurité Informatique*, and by MURST grant prot. 2001017741 under the Italian project "Ragionamento su aggregati e numeri a supporto della programmazione e relative verifiche".

References

[Abr96] Jean-Raymond Abrial. *The B-Book: Assigning Programs to Meanings.* Cambridge University Press, 1996.

[BH98] Bernhard Beckert and Ulrike Hartmer. A tableau calculus for quantifier-free set theoretic formulae. In Harrie C. M. de Swart, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 1397 of *Lecture Notes in Computer Science*, pages 93–107. Springer, 1998.

[CFO89] Domenico Cantone, Alfredo Ferro, and Eugenio G. Omodeo. *Computable Set Theory*, volume 6 of *International Series of Monographs on Computer Science.* Clarendon Press, 1989.

[COP01] Domenico Cantone, Eugenio G. Omodeo, and Alberto Policriti. *Set Theory for Computing. From Decision Procedures to Logic Programming with Sets.* Monographs in Computer Science. Springer, 2001.

[CSZ03] Domenico Cantone, Jacob T. Schwartz, and Calogero G. Zarba. A decision procedure for a sublanguage of set theory involving monotone, additive, and multiplicative functions. In Ingo Dahn and Laurent Vigneron, editors, *First-Order Theorem Proving*, volume 86.1 of *Electronic Notes in Theoretical Computer Science.* Elsevier, 2003.

[CZ00] Domenico Cantone and Calogero G. Zarba. A tableau calculus for integrating first-order reasoning with elementary set theory reasoning. In Roy Dyckhoff, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 1847 of *Lecture Notes in Computer Science*, pages 143–159. Springer, 2000.

[FO78] Alfredo Ferro and Eugenio G. Omodeo. An efficient validity test for formulae in extensional two-level syllogistic. *Le Matematiche*, 33:130–137, 1978.

[FOS80a] Alfredo Ferro, Eugenio G. Omodeo, and Jacob T. Schwartz. Decision procedures for elementary sublanguages of set theory. I. Multi-level syllogistic and some extensions. *Communications on Pure and Applied Mathematics*, 33(5):599–608, 1980.

[FOS80b] Alfredo Ferro, Eugenio G. Omodeo, and Jacob T. Schwartz. Decision procedures for some fragments of set theory. In Wolfgang Bibel and Robert A. Kowalski, editors, *5th Conference on Automated Deduction*, volume 87 of *Lecture Notes in Computer Science*, pages 88–96. Springer, 1980.

[NO79] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979.

[SDDS86] Jacob T. Schwartz, Robert B. K. Dewar, Ed Dubinsky, and Edmond Schonberg. *Programming with Sets: An Introduction to SETL*. Springer, 1986.

[Spi88] J. Michael Spivey. *Understanding Z: A Specification Language and its Formal Semantics*, volume 3 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1988.

[TZ04] Cesare Tinelli and Calogero G. Zarba. Combining decision procedures for theories in sorted logics. Technical Report 04-01, The University of Iowa, 2004.

[Zar02a] Calogero G. Zarba. Combining multisets with integers. In Andrei Voronkov, editor, *Automated Deduction – CADE-18*, volume 2392 of *Lecture Notes in Computer Science*, pages 363–376. Springer, 2002.

[Zar02b] Calogero G. Zarba. Combining sets with integers. In Alessandro Armando, editor, *Frontiers of Combining Systems*, volume 2309 of *Lecture Notes in Computer Science*, pages 103–116. Springer, 2002.

[Zar04] Calogero G. Zarba. Combining sets with elements. In Nachum Dershowitz, editor, *Verification: Theory and Practice*, volume 2772 of *Lecture Notes in Computer Science*, pages 762–782. Springer, 2004.

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399