# A Decision Procedure for Monotone Functions over Bounded and Complete Lattices

Domenico Cantone[1] and Calogero G. Zarba[2]

[1] Università degli Studi di Catania, Italy
cantone@dmi.unict.it
[2] Universität des Saarlandes, Germany
zarba@alan.cs.uni-sb.de

**Abstract.** We present a decision procedure for the quantifier-free satisfiability problem of the language **BLmf** of bounded lattices with monotone unary functions. The language contains the predicates $=$ and $\leq$, as well as the operators $\sqcap$ and $\sqcup$ over terms which may involve uninterpreted unary function symbols. The language also contains predicates for expressing increasing and decreasing monotonicity of functions, as well as a predicate for pointwise function comparison.

Our decision procedure runs in polynomial time $\mathcal{O}(m^4)$ for normalized conjunctions of $m$ literals, thus entailing that the quantifier-free satisfiability problem for **BLmf** is $\mathcal{NP}$-complete. Furthermore, our decision procedure can be used to decide the quantifier-free satisfiability problem for the language **CLmf** of complete lattices with monotone functions. This allows us to conclude that the languages **BLmf** and **CLmf** are equivalent for quantifier-free formulae.

## 1 Introduction

Lattices are partial orders in which every pair of elements has a least upper bound and a greatest lower bound. They have several applications in mathematics and computer science, including model checking [7], knowledge representation [11], partial order programming [12], denotational semantics [10], rewrite systems [4], relational methods in computer science [5], computer security [9], and so on.

In this paper we introduce the language **BLmf** (*Bounded Lattices with monotone functions*) for expressing constraints over lattices and monotone functions. The language **BLmf** contains the equality predicate $=$, the ordering predicate $\leq$, and the operators $\sqcap$ (meet) and $\sqcup$ (join). The language also allows for uninterpreted unary function symbols, and has the following predicates for expressing monotonicity properties of functions:

- the predicate symbol $inc(f)$, stating that the function $f$ is increasing;
- the predicate symbol $dec(f)$, stating that the function $f$ is decreasing;
- the predicate symbol $const(f)$, stating that the function $f$ is constant;
- the predicate symbol $leq(f, g)$, stating that $f(a) \leq g(a)$, for each $a$.

H. de Swart et al. (Eds.): TARSKI II, LNAI 4342, pp. 318–333, 2006.

We prove that the quantifier-free satisfiability problem of **BLmf** is decidable. In particular, we present a decision procedure that allows one to decide the **BLmf**-satisfiability of normalized conjunctions[1] of $m$ literals in polynomial time $\mathcal{O}(m^4)$. Such result entails at once that the quantifier-free satisfiability problem of **BLmf** is $\mathcal{NP}$-complete.

We also study the language **CLmf** of complete lattices with monotone functions. The syntax of **CLmf** is the same of that of **BLmf**. Semantically, **CLmf** differs from **BLmf** in that a model of the language **CLmf** involves a complete lattice rather than just a bounded lattice.

We show that our decision procedure for the quantifier-free satisfiability problem of **BLmf** is also a decision procedure for the quantifier-free satisfiability problem of **CLmf**. Therefore it follows that a quantifier-free formula is **BLmf**-satisfiable if and only if it is **CLmf**-satisfiable, so that the language **BLmf** and **CLmf** are equivalent for quantifier-free formulae.

### 1.1 Related Work

Cantone, Ferro, Omodeo, and Schwartz [1] provide a decision procedure for the quantifier-free language **POSMF** of lattices extended with unary function symbols and the predicates $inc(f)$ and $dec(f)$. The language **POSMF** does not contain the operators $\sqcap$ and $\sqcup$, and it does not contain the predicates $const(f)$ and $leq(f, g)$. The decision procedure for **POSMF** is based on a nondeterministic quadratic reduction to the quantifier-free fragment of set theory **MLS** (cf. [6]).

Sofronie-Stokkermans [13] proved that the quantifier-free languages of (a) partially ordered sets, (b) totally ordered sets, (c) dense totally ordered sets, (d) semilattices, (e) lattices, (f) distributive lattices, (g) boolean algebras, and (h) real numbers can be extended, while still preserving decidability, with one or more monotone increasing unary functions.

In a preliminary version of this paper [2], which dealt only with the quantifier-free satisfiability problem of the language **CLmf**, the authors give a flawed proof of the decidability of **CLmf**. The bug is as follows: In [2, page 9], the partial order $\langle \mathcal{A}, \leq^{\mathcal{A}} \rangle$ is not necessarily a lattice, and therefore the functions $\sqcap^{\mathcal{A}}$ and $\sqcup^{\mathcal{A}}$ are not well-defined in general. Here we fix such problem by taking the Dedekind-MacNeille completion of the partial order $\langle \mathcal{A}, \leq^{\mathcal{A}} \rangle$ (see Section 4, Proposition 28, for details).

Tarski [14] proved that the fully quantified language of lattices is undecidable.

### 1.2 Organization of the Paper

In Section 2 we introduce some basic notions of lattice theory, and we define the syntax and semantics of the language **BLmf**. In Section 3 we present our decision procedure for the quantifier-free satisfiability problem of **BLmf**, and we give an example of our decision procedure in action. In Section 4 we prove that

[1] The notion of normalized set of literals will be defined in Definition 20.

our decision procedure is correct, and we analyze its complexity. In Section 5 we discuss the language **CLmf**. In Section 6 we draw some final conclusions.

## 2 Preliminaries

### 2.1 Partial Orders

**Definition 1.** A PARTIAL ORDER is a pair $(A, \leq)$ where $A$ is a nonempty set and $\leq$ is a reflexive, antisymmetric, and transitive binary relation of $A$. □

**Definition 2.** Let $(A, \leq)$ be a partial order and let $\emptyset \neq X \subseteq A$. We say that $y$ is a MAXIMUM of $X$ with respect to $(A, \leq)$ if the following conditions hold:

- $y \in X$;
- $x \leq y$, for each $x \in X$. □

When it exists, the maximum of $X$ with respect to $(A, \leq)$ is unique. Consequently, we use the notation $max(X, A, \leq)$ to denote the unique maximum of $X$ with respect to $(A, \leq)$ when it exists; otherwise, we let $max(X, A, \leq) = undef$.

**Definition 3.** Let $(A, \leq)$ be a partial order and let $\emptyset \neq X \subseteq A$. We say that $y$ is a MINIMUM of $X$ with respect to $(A, \leq)$ if the following conditions hold:

- $y \in X$;
- $y \leq x$, for each $x \in X$. □

When it exists, the minimum of $X$ with respect to $(A, \leq)$ is unique. Consequently, we use the notation $min(X, A, \leq)$ to denote the unique minimum of $X$ with respect to $(A, \leq)$ when it exists; otherwise, we let $min(X, A, \leq) = undef$.

**Definition 4.** Let $(A, \leq)$ be a partial order and let $\emptyset \neq X \subseteq A$. We say that $y$ is a LEAST UPPER BOUND of $X$ with respect to $(A, \leq)$ if the following conditions hold:

- $x \leq y$, for each $x \in X$;
- if $x \leq z$, for each $x \in X$, then $y \leq z$. □

When it exists, the least upper bound of $X$ with respect to $(A, \leq)$ is unique. Consequently, we use the notation $lub(X, A, \leq)$ to denote the unique least upper bound of $X$ with respect to $(A, \leq)$ when it exists; otherwise, we let $lub(X, A, \leq) = undef$.

**Proposition 5.** *Let $(A, \leq)$ be a partial order and let $\emptyset \neq X \subseteq A$. Then, $max(X, A, \leq) \neq undef$ implies $max(X, A, \leq) = lub(X, A, \leq)$.* □

**Definition 6.** Let $(A, \leq)$ be a partial order and let $\emptyset \neq X \subseteq A$. We say that $y$ is a GREATEST LOWER BOUND of $X$ with respect to $(A, \leq)$ if the following conditions hold:

- $y \leq x$, for each $x \in X$;
- if $z \leq x$, for each $x \in X$, then $z \leq y$. □

When it exists, the greatest lower bound of $X$ with respect to $(A, \leq)$ is unique. Consequently, we use the notation $glb(X, A, \leq)$ to denote the unique greatest lower bound of $X$ with respect to $(A, \leq)$ when it exists; otherwise, we let $glb(X, A, \leq) = undef$.

**Proposition 7.** *Let $(A, \leq)$ be a partial order and let $\emptyset \neq X \subseteq A$. Then, $min(X, A, \leq) \neq undef$ implies $min(X, A, \leq) = glb(X, A, \leq)$.* □

### 2.2 Lattices

**Definition 8.** A LATTICE is a tuple $(A, \leq, \sqcup, \sqcap)$ where:

- $(A, \leq)$ is a partial order;
- $glb(\{a, b\}, A, \leq) \neq undef$ and $lub(\{a, b\}, A, \leq) \neq undef$, for all $a, b \in A$;
- $a \sqcup b = lub(\{a, b\}, A, \leq)$;
- $a \sqcap b = glb(\{a, b\}, A, \leq)$. □

**Definition 9.** A BOUNDED LATTICE is a tuple $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ where:

- $(A, \leq, \sqcup, \sqcap)$ is a lattice;
- $\mathbf{1} = max(A, A, \leq)$;
- $\mathbf{0} = min(A, A, \leq)$. □

**Definition 10.** A COMPLETE LATTICE is a tuple $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ where:

- $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ is a bounded lattice;
- $glb(X, A, \leq) \neq undef$ and $lub(X, A, \leq) \neq undef$, for each $\emptyset \neq X \subseteq A$. □

**Remark 11.** If $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ is a complete lattice, we let $lub(\emptyset, A, \leq) = \mathbf{0}$ and $glb(\emptyset, A, \leq) = \mathbf{1}$. □

**Proposition 12.** *Let $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ be a complete lattice. Then, the following properties hold:*

$$\begin{aligned} a \sqcup b &= b \sqcup a\,, & a \sqcap b &= b \sqcap a\,, \\ (a \sqcup b) \sqcup c &= a \sqcup (b \sqcup c)\,, & (a \sqcap b) \sqcap c &= a \sqcap (b \sqcap c)\,, \\ a \sqcup a &= a\,, & a \sqcap a &= a\,, \\ a \sqcup (a \sqcap b) &= a\,, & a \sqcap (a \sqcup b) &= a\,. \end{aligned}$$

*Moreover, we have:*

$$a \leq b \quad \leftrightarrow \quad a \sqcup b = b \quad \leftrightarrow \quad a \sqcap b = a\,.$$ □

**Proposition 13.** *Let $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ be a complete lattice, and let $X, Y \subseteq A$. Then, $X \subseteq Y$ implies $lub(X) \leq lub(Y)$.* □

**Proposition 14.** *Let $(A, \leq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ be a complete lattice, and let $X, Y \subseteq A$. Then, $X \subseteq Y$ implies $glb(Y) \leq glb(X)$.* □

### 2.3 Dedekind-MacNeille Completion

The Dedekind-MacNeille completion allows one to extend a partial order $(A, \leq)$ into a complete lattice. It was introduced by MacNeille [8], who generalized the Dedekind completion [3] for constructing the set $\mathbb{R}$ of real numbers from the set $\mathbb{Q}$ of rational numbers.

**Proposition 15 ([8]).** *Let $(A, \leq)$ be a partial order. Then there exists a unique*[2] *minimal complete lattice $(B, \sqsubseteq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ such that:*

*(a)* $A \subseteq B$*;*
*(b)* $a \leq b$ *iff* $a \sqsubseteq b$*, for each* $a, b \in A$*;*
*(c) If* $lub(X, A, \leq) \neq undef$*, then* $lub(X, B, \sqsubseteq) = lub(X, A, \leq)$*, for each* $\emptyset \neq X \subseteq A$*;*
*(d) If* $glb(X, A, \leq) \neq undef$*, then* $glb(X, B, \sqsubseteq) = glb(X, A, \leq)$*, for each* $\emptyset \neq X \subseteq A$*.* □

**Definition 16.** Let $(A, \leq)$ be a partial order. The Dedekind-MacNeille completion of $(A, \leq)$ is the unique complete lattice $(B, \sqsubseteq, \sqcup, \sqcap, \mathbf{1}, \mathbf{0})$ satisfying properties (a)–(d) of Proposition 15. □

### 2.4 Syntax of BLmf

The language **BLmf** (*Bounded Lattices with monotone functions*) is a quantifier-free language containing the following symbols:

- arbitrarily many variables $x, y, z, \ldots$;
- the constant symbols $\mathbf{1}$ and $\mathbf{0}$;
- the function symbols $\sqcup$ and $\sqcap$;
- the binary predicate symbols $\leq$ and $=$;
- arbitrarily many unary function symbols $f$, $g$, ...
- the predicate symbol $inc(f)$;
- the predicate symbol $dec(f)$;
- the predicate symbol $const(f)$;
- the predicate symbol $leq(f, g)$.

**Definition 17.** The set of **BLmf**-terms is the smallest set satisfying the following conditions:

- Every variable is a **BLmf**-term;
- $\mathbf{1}$ and $\mathbf{0}$ are **BLmf**-terms;
- If $s$ and $t$ are **BLmf**-terms, so are $s \sqcup t$ and $s \sqcap t$;
- If $s$ is a **BLmf**-term and $f$ is a function symbol, then $f(s)$ is a **BLmf**-term.

[2] Up to an isomorphism.

**BLmf**-ATOMS are of the form:

$$s = t\,, \qquad s \leq t\,, \qquad \mathit{inc}(f)\,,$$
$$\mathit{dec}(f)\,, \qquad \mathit{const}(f)\,, \qquad \mathit{leq}(f,g)\,,$$

where $s, t$ are **BLmf**-terms and $f, g$ are unary function symbol.

**BLmf**-FORMULAE are constructed from **BLmf**-atoms using the propositional connectives $\neg$, $\vee$, $\wedge$, $\rightarrow$, and $\leftrightarrow$. **BLmf**-LITERALS are **BLmf** atoms or their negations. □

If $\varphi$ is a **BLmf**-formula, we denote with $\mathit{vars}(\varphi)$ the set of variables occurring in $\varphi$. If $\Phi$ is a set of **BLmf**-formulae, we let $\mathit{vars}(\Phi) = \bigcup_{\varphi \in \Phi} \mathit{vars}(\varphi)$.

### 2.5 Semantics of BLmf

**Definition 18.** A **BLmf**-INTERPRETATION $\mathcal{A}$ is a pair $\left(A, (\cdot)^{\mathcal{A}}\right)$ where $A \neq \emptyset$ and $(\cdot)^{\mathcal{A}}$ interprets the symbols of the language **BLmf** as follows:

- $\left(A, \leq^{\mathcal{A}}, \sqcup^{\mathcal{A}}, \sqcap^{\mathcal{A}}, \mathbf{1}^{\mathcal{A}}, \mathbf{0}^{\mathcal{A}}\right)$ is a bounded lattice;
- $=^{\mathcal{A}}$ is interpreted as the identity in $A$;
- each variable $x$ is mapped to an element $x^{\mathcal{A}} \in A$;
- each unary function symbol $f$ is mapped to a function $f^{\mathcal{A}} : A \rightarrow A$;
- $[\mathit{inc}(f)]^{\mathcal{A}} = \mathit{true}$ iff $a \leq^{\mathcal{A}} b$ implies $f^{\mathcal{A}}(a) \leq^{\mathcal{A}} f^{\mathcal{A}}(b)$, for each $a, b \in A$;
- $[\mathit{dec}(f)]^{\mathcal{A}} = \mathit{true}$ iff $a \leq^{\mathcal{A}} b$ implies $f^{\mathcal{A}}(b) \leq^{\mathcal{A}} f^{\mathcal{A}}(a)$, for each $a, b \in A$.
- $[\mathit{const}(f)]^{\mathcal{A}} = \mathit{true}$ iff $f^{\mathcal{A}}(a) = f^{\mathcal{A}}(b)$, for each $a, b \in A$.
- $[\mathit{leq}(f,g)]^{\mathcal{A}} = \mathit{true}$ iff $f^{\mathcal{A}}(a) \leq^{\mathcal{A}} g^{\mathcal{A}}(a)$, for all $a \in A$. □

Let $\varphi$ be either a **BLmf**-formula or a **BLmf**-term, and let $\mathcal{A}$ be a **BLmf**-interpretation. We denote with $\varphi^{\mathcal{A}}$ the evaluation of $\varphi$ under $\mathcal{A}$.

**Definition 19.** A **BLmf**-formula $\mathcal{A}$ is **BLmf**-SATISFIABLE if there exists a **BLmf**-interpretation $\mathcal{A}$ such that $\varphi^{\mathcal{A}} = \mathit{true}$. A set $\Phi$ of **BLmf**-formulae is **BLmf**-SATISFIABLE if there exists a **BLmf**-interpretation $\mathcal{A}$ such that $\varphi^{\mathcal{A}} = \mathit{true}$, for each $\varphi \in \Phi$. □

## 3 A Decision Procedure for BLmf

In this section we present a decision procedure for the quantifier-free satisfiability problem for the language **BLmf**. Without loss of generality, we restrict ourselves to normalized sets of **BLmf**-literals.

**Definition 20.** A set $\Gamma$ of **BLmf**-literals is NORMALIZED if it satisfies the following conditions:

1. Each **BLmf**-literal in $\Gamma$ is of the form

$$x = y\,, \qquad x \neq y\,, \qquad x \leq y\,, \qquad x \not\leq y\,,$$
$$x = y \sqcup z\,, \qquad x = y \sqcap z\,, \qquad x = f(y)\,,$$
$$\mathit{inc}(f)\,, \qquad \mathit{dec}(f)\,, \qquad \mathit{const}(f)\,, \qquad \mathit{leq}(f,g)\,,$$

where:

  - $x, y, z$ can be either variables or the constant symbols $\mathbf{1}$ and $\mathbf{0}$;
  - $f, g$ are unary function symbols.
2. For each unary function symbol $f$, no more than one of the following **BLmf**-literals is in $\Gamma$:

$$\mathit{inc}(f)\,, \qquad\qquad \mathit{dec}(f)\,, \qquad\qquad \mathit{const}(f)\,. \qquad \square$$

**Proposition 21.** *Every finite set of* **BLmf***-literals can be converted in polynomial time into a* **BLmf***-equisatisfiable normalized set of* **BLmf***-literals.* □

Proof. Let $\Gamma$ be a finite set of **BLmf**-literals. By opportunely introducing fresh variables, we can convert all literals in $\Gamma$—while still preserving **BLmf**-satisfiability—to literals conforming condition 1 of Definition 20. In particular, literals of the form $\neg \mathit{inc}(f)$ can be replaced by a conjunction $x \leq y \wedge u = f(x) \wedge v = f(y) \wedge u \not\leq v$, where $x$, $y$, $u$, and $v$ are fresh variables. Similar replacements can be performed for the literals of the form $\neg \mathit{dec}(f)$, $\neg \mathit{const}(f)$, and $\neg \mathit{leq}(f, g)$. Finally, condition 2 of Definition 20 can be enforced by exploiting the following equivalences:

$$\begin{aligned} \mathit{inc}(f) \wedge \mathit{dec}(f) &\equiv \mathit{const}(f)\,, \\ \mathit{inc}(f) \wedge \mathit{const}(f) &\equiv \mathit{const}(f)\,, \\ \mathit{dec}(f) \wedge \mathit{const}(f) &\equiv \mathit{const}(f)\,, \\ \mathit{inc}(f) \wedge \mathit{dec}(f) \wedge \mathit{const}(f) &\equiv \mathit{const}(f)\,. \end{aligned} \qquad \blacksquare$$

Given a normalized set $\Gamma$ of **BLmf**-literals, we define the following four pairwise disjoint sets:

- $\mathit{INC}(\Gamma)$ contains all unary function symbols $f$ such that the literal $\mathit{inc}(f)$ is in $\Gamma$.
- $\mathit{DEC}(\Gamma)$ contains all unary function symbols $f$ such that the literal $\mathit{dec}(f)$ is in $\Gamma$.
- $\mathit{CONST}(\Gamma)$ contains all unary function symbols $f$ such that the literal $\mathit{const}(f)$ is in $\Gamma$.
- $\mathit{NORM}(\Gamma)$ contains all unary function symbols that do not belong to $\mathit{INC}(\Gamma) \cup \mathit{DEC}(\Gamma) \cup \mathit{CONST}(\Gamma)$.

Our decision procedure is based on the inference rules shown in Figure 1. In order to ensure termination we require the following:

- If $\mathcal{R}$ is not a fresh-variable rule, then $\mathcal{R}$ cannot be applied to a conjunction of normalized literals $\Gamma$ if the conclusion of $\mathcal{R}$ is already in $\Gamma$.
- If $\mathcal{R}$ is a fresh-variable rule whose conclusion is a literal $\ell$, then $\mathcal{R}$ cannot be applied to $\Gamma$ if the literal $\ell\{\mathsf{fresh}/w\}$ is already in $\Gamma$, for some variable $w$.

**Definition 22.** A normalized set $\Gamma$ of **BLmf**-literals is saturated if no inference rule in Figure 1 can be applied to $\Gamma$. □

*=-rules*

$$\frac{}{x = x} \qquad \frac{\begin{array}{c} x = y \\ \ell \end{array}}{\ell\{x/y\}}$$

*≤-rules*

$$\frac{}{x \leq x} \qquad \frac{\begin{array}{c} x \leq y \\ y \leq x \end{array}}{x = y} \qquad \frac{\begin{array}{c} x \leq y \\ y \leq z \end{array}}{x \leq z} \qquad \frac{}{x \leq \mathbf{1}} \qquad \frac{}{\mathbf{0} \leq x}$$

*⊓-rules*

$$\frac{x = y \sqcap z}{\begin{array}{c} x \leq y \\ x \leq z \end{array}} \qquad \frac{\begin{array}{c} x = y \sqcap z \\ w \leq y \\ w \leq z \end{array}}{w \leq x}$$

*⊔-rules*

$$\frac{x = y \sqcup z}{\begin{array}{c} y \leq x \\ z \leq x \end{array}} \qquad \frac{\begin{array}{c} x = y \sqcup z \\ y \leq w \\ z \leq w \end{array}}{x \leq w}$$

*Functions rules*

$$\frac{\begin{array}{c} x = x' \\ y = f(x) \\ y' = f(x') \end{array}}{y = y'} \qquad \frac{\begin{array}{c} inc(f) \\ x \leq x' \\ y = f(x) \\ y' = f(x') \end{array}}{y \leq y'} \qquad \frac{\begin{array}{c} dec(f) \\ x \leq x' \\ y = f(x) \\ y' = f(x') \end{array}}{y' \leq y} \qquad \frac{\begin{array}{c} const(f) \\ y = f(x) \\ y' = f(x') \end{array}}{y = y'} \qquad \frac{\begin{array}{c} leq(f, g) \\ y = f(x) \\ y' = g(x) \end{array}}{y \leq y'}$$

*Fresh-variables rules*

$$\frac{\begin{array}{c} leq(f, g) \\ y = f(x) \end{array}}{\mathsf{fresh} = g(x)} \qquad \frac{\begin{array}{c} leq(f, g) \\ y = g(x) \end{array}}{\mathsf{fresh} = f(x)} \qquad \frac{}{\mathsf{fresh} = f(\mathbf{1})} \qquad \frac{}{\mathsf{fresh} = f(\mathbf{0})}$$

*Notes*

- In the second =-rule, the literal $\ell$ does not contain any function symbol. Additionally, by $\ell\{x/y\}$ we mean the literal obtained by replacing *any* occurrence of $x$ in $\ell$ by $y$.
- In the first = rule, first ≤-rule, and last two ≤-rules, the variables $x$ already occurs in $\Gamma$.
- In the fresh-variables rules, fresh stands for a newly introduced variable.

**Fig. 1.** Inference rules for computing $closure(\Gamma)$

If $\Gamma$ is a normalized set of **BLmf**-literals, we denote with $closure(\Gamma)$ the smallest saturated set of **BLmf**-literals containing $\Gamma$. Note that the set $closure(\Gamma)$ is normalized.

The above two constraints on the applicability of the inference rules in Figure 1 imply that $closure(\Gamma)$ has at most $\mathcal{O}(m^4)$ literals, for any normalized set $\Gamma$ of **BLmf**-literals with $m$ literals.

**Proposition 23.** *Let $\Gamma$ be a normalized set of* **BLmf***-literals with $m$ literals. Then $closure(\Gamma)$ has at most $\mathcal{O}(m^4)$ literals.* □

Proof. Clearly, the first two fresh-variables rules introduce at most $\mathcal{O}(m^2)$-variables. The second two fresh-variables rules introduce at most $\mathcal{O}(k)$-variables, where $k$ is the number of unary function symbols in $\Gamma$. Since $k = \mathcal{O}(m)$, it follows that all the fresh-variables rules introduce at most $\mathcal{O}(m^2)$-variables, and therefore $closure(\Gamma)$ contains at most $\mathcal{O}(m^2)$-variables. Therefore, the remaining rules can introduce at most $\mathcal{O}((m^2)^2)$ literals, which implies that $closure(\Gamma)$ contains at most $\mathcal{O}(m^4)$-literals. ■

**Definition 24.** A normalized set $\Gamma$ of **BLmf**-literals is consistent if it does not contain any two complementary literals $\ell$,$\neg\ell$; otherwise it is inconsistent.□

Given a finite normalized set $\Gamma$ of **BLmf**-literals, our decision procedure consists of the following two steps:

**Step 1.** Compute $\Delta = closure(\Gamma)$.
**Step 2.** Output `satisfiable` if $\Delta$ is consistent; otherwise output `unsatisfiable`.

**Example 25.** Let $\Gamma$ be the following set of **BLmf**-literals

$$\Gamma = \left\{ \begin{array}{l} inc(f)\,, \\ dec(g)\,, \\ leq(f,g)\,, \\ f(\mathbf{0}) = g(\mathbf{0})\,, \\ f(x) \neq g(x) \end{array} \right\}.$$

We claim that $\Gamma$ is **BLmf**-unsatisfiable. In fact, the first four literals imply that $f = g$, which contradicts the last literal.

We use our decision procedure in order to automatically check that $\Gamma$ is **BLmf**-unsatisfiable. First, note that $\Gamma$ is **BLmf**-equisatisfiable with the following normalized set $\Gamma'$ of **BLmf**-literals:

$$\Gamma' = \left\{ \begin{array}{l} inc(f)\,, \\ dec(g)\,, \\ leq(f,g)\,, \\ y_1 = f(\mathbf{0})\,, \\ y_2 = g(\mathbf{0})\,, \\ y_1 = y_2\,, \\ z_1 = f(x)\,, \\ z_2 = g(x)\,, \\ z_1 \neq z_2 \end{array} \right\}$$

Then, note that $closure(\Gamma')$ must contain, among others, the following literals:

$$\begin{aligned}
&\mathbf{0} \leq x\,, && \text{by the fifth } \leq\text{-rule}\,,\\
&y_1 \leq z_1\,, && \text{by the second functions rule}\,,\\
&z_1 \leq z_2\,, && \text{by the fifth functions rule}\,,\\
&z_2 \leq y_2\,, && \text{by the third functions rule}\,,\\
&z_2 \leq y_1\,, && \text{by the second } =\text{-rule}\,,\\
&y_1 \leq z_2\,, && \text{by the third } \leq\text{-rule}\,,\\
&z_2 = y_1\,, && \text{by the second } \leq\text{-rule}\,,\\
&z_1 \leq y_1\,, && \text{by the second } =\text{-rule}\,,\\
&z_1 = y_1\,, && \text{by the second } \leq\text{-rule}\,,\\
&z_1 = z_2\,, && \text{by the second } =\text{-rule}\,.
\end{aligned}$$

Since $closure(\Gamma')$ contains the complementary literals $z_1 = z_2$ and $z_1 \neq z_2$, our decision procedure outputs `unsatisfiable`, as desired. □

## 4 Correctness and Complexity

**Proposition 26.** *Let $\Gamma$ be a* **BLmf***-satisfiable normalized set of* **BLmf***-literals, and let $\Gamma'$ be the result of extending $\Gamma$ by means of an application of one of the inference rules in Figure 1. Then $\Gamma'$ is* **BLmf***-satisfiable.* □

Proof. Let $\mathcal{A}$ be a **BLmf**-interpretation satisfying $\Gamma$. If $\Gamma'$ involves the same variables of $\Gamma$, then it is routine to verify that $\mathcal{A}$ satisfies $\Gamma'$ too. Otherwise, if $\Gamma'$ is obtained from $\Gamma$ by applying a fresh-variables rule, and therefore it involves a variable $\mathsf{fresh}$ not present in $\Gamma$, then it can easily be argued that $\Gamma'$ is satisfied by a suitable variant $\mathcal{A}'$ of the **BLmf**-interpretation $\mathcal{A}$, which assigns the same values to every symbol of the language, except possibly the variable $\mathsf{fresh}$. ■

**Proposition 27 (Soundness).** *Let $\Gamma$ be a* **BLmf***-satisfiable finite normalized set of* **BLmf***-literals. Then $closure(\Gamma)$ is* **BLmf***-satisfiable.* □

Proof. By Propositions 23 and 26. ■

**Proposition 28.** *Any saturated and consistent normalized set of* **BLmf***-literals is* **BLmf***-satisfiable.* □

Proof. Let $\Gamma$ be a saturated and consistent set of **BLmf**-literals.

Let $X = vars(\Gamma) \cup \{\mathbf{1}, \mathbf{0}\}$, and let $\sim$ be the binary relation of $X$ induced by the literals of the form $x = y$ in $\Gamma$. By saturation with respect to the $=$-rules, $\sim$ is an equivalence relation. Consequently, we can form the quotient set $A = X/\!\sim$.

Let $\preceq$ be the binary relation of $A$ defined as follows:

$$[x]_\sim \preceq [y]_\sim \qquad \Longleftrightarrow \qquad x \leq y \text{ is in } \Gamma\,.$$

Since $\sim$ is an equivalence relation, $\preceq$ is well-defined. Moreover, by saturation with respect to the $\leq$-rules, $(A, \preceq)$ is a partial order with maximum $[\mathbf{1}]_\sim$ and minimum $[\mathbf{0}]_\sim$.[3]

Let $(B, \sqsubseteq, +, \cdot, \top, \bot)$ be the Dedekind-MacNeille completion of $(A, \preceq)$. Note that we have $\top = [\mathbf{1}]_\sim$ and $\bot = [\mathbf{0}]_\sim$.

We define a **BLmf**-interpretation $\mathcal{B} = \big(B, (\cdot)^\mathcal{B}\big)$ by letting:

- $a =^\mathcal{B} b$ iff $a = b$.
- $a \leq^\mathcal{B} b$ iff $a \sqsubseteq b$;
- $a \sqcup^\mathcal{B} b = a + b$;
- $a \sqcap^\mathcal{B} b = a \cdot b$;
- $\mathbf{1}^\mathcal{B} = \top = [\mathbf{1}]_\sim$;
- $\mathbf{0}^\mathcal{B} = \bot = [\mathbf{0}]_\sim$;
- $x^\mathcal{B} = [x]_\sim$;
- $f^\mathcal{B}(a) = \mathit{lub}(Z_{f,a}, B, \sqsubseteq)$ where

$$Z_{f,a} = X_{f,a} \cup \bigcup_{\mathit{leq}(h,f) \in \Gamma} X_{h,a} \,,$$

and

$$X_{f,a} = \begin{cases} \{[y]_\sim \mid y = f(x) \text{ is in } \Gamma \text{ and } a = [x]_\sim\}\,, & \text{if } f \in \mathit{NORM}(\Gamma)\,, \\ \{[y]_\sim \mid y = f(x) \text{ is in } \Gamma \text{ and } [x]_\sim \sqsubseteq a\}\,, & \text{if } f \in \mathit{INC}(\Gamma)\,, \\ \{[y]_\sim \mid y = f(x) \text{ is in } \Gamma \text{ and } a \sqsubseteq [x]_\sim\}\,, & \text{if } f \in \mathit{DEC}(\Gamma)\,, \\ \{[y]_\sim \mid y = f(x) \text{ is in } \Gamma\}\,, & \text{if } f \in \mathit{CONST}(\Gamma)\,. \end{cases}$$

By construction, $\mathcal{B}$ is a **BLmf**-interpretation. Next, we show that $\mathcal{B}$ satisfies all **BLmf**-literals in $\Gamma$.

**Literals of the form $x = y$.** We have $x \sim y$, which implies $x^\mathcal{B} = [x]_\sim = [y]_\sim = y^\mathcal{B}$.

**Literals of the form $x \neq y$.** If it were $x^\mathcal{B} = y^\mathcal{B}$, we would have $x \sim y$, which implies that the literal $x = y$ is in $\Gamma$, a contradiction.

**Literals of the form $x \leq y$.** We have $[x]_\sim \preceq [y]_\sim$, so that $[x]_\sim \sqsubseteq [y]_\sim$. Therefore $[x]_\sim \leq^\mathcal{B} [y]_\sim$, which in turn implies $[x \leq y]^\mathcal{B} = \mathit{true}$.

**Literals of the form $\neg(x \leq y)$.** We have $[x]_\sim \not\preceq [y]_\sim$. Hence $[x]_\sim \not\sqsubseteq [y]_\sim$, which implies $[x \leq y]^\mathcal{B} = \mathit{false}$.

**Literals of the form $x = y \sqcap z$.** By saturation with respect to the $\sqcap$-rules, we have that $[x]_\sim = \mathit{glb}(\{[y],[z]\}, A, \preceq)$. It follows that $[x]_\sim = \mathit{glb}(\{[y],[z]\}, B, \sqsubseteq)$.

[3] In general, $(A, \preceq)$ is not a lattice. As an example, consider $\Gamma = \mathit{closure}(\{u \leq x, v \leq x, u \leq y, v \leq y\})$. This is the flaw in [2], which we correct in this paper by taking the Dedekind-MacNeille completion of $(A, \preceq)$.

**Literals of the form $\boldsymbol{x = y \sqcup z}$.** By saturation with respect to the $\sqcup$-rules, we have that $[x]_\sim = lub(\{[y],[z]\}, A, \preceq)$. It follows that $[x]_\sim = lub(\{[y],[z]\}, B, \sqsubseteq)$.

**Literals of the form $\boldsymbol{y = f(x)}$.** Let the literal $y = f(x)$ be in $\Gamma$. We need to show that $y^{\mathcal{B}} = f^{\mathcal{B}}\left(x^{\mathcal{B}}\right)$. This amounts to verify that $[y]_\sim = lub(Z_{f,[x]_\sim}, B, \sqsubseteq)$. Since the literal $y = f(x)$ is in $\Gamma$, we have immediately $[y]_\sim \in X_{f,[x]_\sim} \subseteq Z_{f,[x]_\sim}$. Therefore, it is enough to show that $[y']_\sim \sqsubseteq [y]_\sim$ holds, for each $[y']_\sim \in Z_{f,[x]_\sim}$.

Thus, let $[y']_\sim \in Z_{f,[x]_\sim}$. It is convenient to distinguish the following two cases.

**Case 1:** $[y']_\sim \in X_{f,[x]_\sim}$. We consider the following four subcases.

(1a) Let $f \in NORM(\Gamma)$. Then a literal of the form $y' = f(x')$ is in $\Gamma$, and $[x]_\sim = [x']_\sim$. Hence the literal $x = x'$ is in $\Gamma$. By saturation with respect to the rules of Figure 1, it follows that also the literal $y = y'$ is in $\Gamma$, and therefore $[y]_\sim = [y']_\sim$.

(1b) Let $f \in INC(\Gamma)$. Then a literal of the form $y' = f(x')$ is in $\Gamma$. Moreover, $[x']_\sim \sqsubseteq [x]_\sim$, which implies $[x']_\sim \preceq [x]_\sim$, so that the literal $x' \leq x$ is in $\Gamma$. By saturation, it follows that the literal $y' \leq y$ is in $\Gamma$ too, which implies $[y']_\sim \preceq [y]_\sim$, and therefore $[y']_\sim \sqsubseteq [y]_\sim$.

(1c) Let $f \in DEC(\Gamma)$. Then a literal of the form $y' = f(x')$ is in $\Gamma$. Moreover, $[x]_\sim \sqsubseteq [x']_\sim$, which implies $[x]_\sim \preceq [x']_\sim$, so that the literal $x \leq x'$ is in $\Gamma$. By saturation, it follows that also the literal $y' \leq y$ must be in $\Gamma$, which implies $[y']_\sim \preceq [y]_\sim$, and therefore $[y']_\sim \sqsubseteq [y]_\sim$.

(1d) Let $f \in CONST(\Gamma)$. Then a literal of the form $y' = f(x')$ is in $\Gamma$. By saturation, it follows that the literal $y = y'$ is in $\Gamma$, and therefore $[y]_\sim = [y']_\sim$.

**Case 2:** $[y']_\sim \in X_{h,[x]_\sim}$, where the literal $leq(h, f)$ is in $\Gamma$, and $h$ is a function symbol distinct from $f$. We consider the following four subcases.

(2a) Let $h \in NORM(\Gamma)$. Then a literal of the form $y' = h(x')$ is in $\Gamma$, such that $[x]_\sim = [x']_\sim$. By saturation, it follows that the literal $y = f(x')$ is in $\Gamma$. But then, again by saturation, the literal $y' \leq y$ must be in $\Gamma$. Therefore, $[y']_\sim \preceq [y]_\sim$, which implies $[y']_\sim \sqsubseteq [y]_\sim$.

(2b) Let $h \in INC(\Gamma)$. Then a literal of the form $y' = h(x')$ is in $\Gamma$, such that $[x']_\sim \sqsubseteq [x]_\sim$. Therefore, $[x']_\sim \preceq [x]_\sim$, so that the literal $x' \leq x$ is in $\Gamma$. By saturation, a literal of the form $y'' = h(x)$ must be in $\Gamma$. Therefore, again by saturation, the literals $y' \leq y''$ and $y'' \leq y$ are in $\Gamma$, so that also the literal $y' \leq y$ is in $\Gamma$. Hence, $[y']_\sim \preceq [y]_\sim$, which implies $[y']_\sim \sqsubseteq [y]_\sim$.

(2c) Let $h \in DEC(\Gamma)$. Then a literal of the form $y' = h(x')$ is in $\Gamma$, such that $[x]_\sim \sqsubseteq [x']_\sim$. Therefore, $[x]_\sim \preceq [x']_\sim$, so that the literal $x \leq x'$ is in $\Gamma$. By saturation, a literal of the form $y'' = h(x)$ must be in $\Gamma$. Therefore, again by saturation, the literals $y' \leq y''$ and $y'' \leq y$ are in $\Gamma$, so that also the literal $y' \leq y$ is in $\Gamma$. Hence, $[y']_\sim \preceq [y]_\sim$, which implies $[y']_\sim \sqsubseteq [y]_\sim$.

(2d) Let $h \in CONST(\Gamma)$. Then a literal of the form $y' = h(x')$ is in $\Gamma$. By saturation, a literal of the form $y'' = h(x)$ is in $\Gamma$. Therefore, again by saturation, the literals $y' = y''$ and $y'' \leq y$ are in $\Gamma$, so that also the literal $y' \leq y$ must be in $\Gamma$. Therefore, $[y']_\sim \preceq [y]_\sim$, which implies $[y']_\sim \sqsubseteq [y]_\sim$.

**Literals of the form *inc*$(f)$.** Let the literal $inc(f)$ be in $\Gamma$. We need to show that the function $f^{\mathcal{B}}$ is increasing in the lattice $(B, \sqsubseteq, +, \cdot, \top, \bot)$. Thus, let $a, b \in B$ such that $a \sqsubseteq b$. To prove that $f^{\mathcal{B}}(a) \sqsubseteq f^{\mathcal{B}}(b)$, or equivalently that $lub(Z_{f,a}, B \sqsubseteq) \sqsubseteq lub(Z_{f,b}, B, \sqsubseteq)$, it is enough to show that for each $[y]_\sim \in Z_{f,a}$ there exists $[y']_\sim \in Z_{f,b}$ such that $[y]_\sim \sqsubseteq [y']_\sim$.

Thus, let $[y]_\sim \in Z_{f,a}$. We distinguish two cases.

**Case 1:** $[y]_\sim \in X_{f,a}$. Then the literal $y = f(x)$ is in $\Gamma$ and $[x]_\sim \sqsubseteq a \sqsubseteq b$, which implies $[y]_\sim \in X_{f,b} \subseteq Z_{f,b}$.

**Case 2:** $[y] \in X_{h,a}$, where the literal $leq(h, f)$ is in $\Gamma$, and $h$ is a function symbol distinct from $f$. We consider the following four subcases.

(2a) $h \in NORM(\Gamma)$. Then a literal of the form $y = h(x)$ is in $\Gamma$, and $a = [x]_\sim$. It follows that the literal $y' = f(x)$ is in $\Gamma$. But then, by saturation, the literal $y \leq y'$ is in $\Gamma$. Therefore, $[y]_\sim \preceq [y']_\sim$, which implies $[y]_\sim \sqsubseteq [y']_\sim$. Moreover, $[y']_\sim \in X_{h,b} \subseteq Z_{f,b}$.

(2b) $h \in INC(\Gamma)$. Then a literal of the form $y = h(x)$ is in $\Gamma$, and $[x]_\sim \sqsubseteq a \sqsubseteq b$. Therefore, $[y] \in X_{h,b} \subseteq Z_{f,b}$.

(2c) $h \in DEC(\Gamma)$. Then a literal of the form $y = h(x)$ is in $\Gamma$, and $a \sqsubseteq [x]_\sim$. By saturation, the following literals are in $\Gamma$: $y' = h(\mathbf{0})$, $y'' = f(\mathbf{0})$, $y \leq y'$, $y' \leq y''$, and $y \leq y''$. It follows that $[y]_\sim \preceq [y'']_\sim$, which implies $[y]_\sim \sqsubseteq [y'']_\sim$. Moreover, since $[\mathbf{0}]_\sim = \mathbf{0}^{\mathcal{B}} \sqsubseteq b$, we have $[y'']_\sim \in X_{f,b} \subseteq Z_{f,b}$.

(2d) $h \in CONST(\Gamma)$. Then a literal of the form $y = h(x)$ is in $\Gamma$. By saturation, the following literals are in $\Gamma$: $y' = h(\mathbf{0})$, $y'' = f(\mathbf{0})$, $y = y'$, $y' \leq y''$, and $y \leq y''$. It follows that $[y]_\sim \preceq [y'']_\sim$, which implies $[y]_\sim \sqsubseteq [y'']_\sim$. Moreover, since $[\mathbf{0}]_\sim = \mathbf{0}^{\mathcal{B}} \sqsubseteq b$, we have $[y'']_\sim \in X_{f,b} \subseteq Z_{f,b}$.

**Literals of the form *dec*$(f)$.** This case is similar to the case of literals of the form $inc(f)$.

**Literals of the form *const*$(f)$.** This case is similar to the case of literals of the form $inc(f)$.

**Literals of the form *leq*$(f, g)$.** We have $Z_{f,a} \subseteq Z_{g,a}$. Therefore, $f^{\mathcal{B}}(a) = lub(Z_{f,a}, B, \sqsubseteq) \sqsubseteq lub(Z_{g,a}, B, \sqsubseteq) = g^{\mathcal{B}}(a)$. ■

**Proposition 29 (Completeness).** *Let $\Gamma$ be a normalized set of* **BLmf***-literals, and assume that $closure(\Gamma)$ is consistent. Then $\Gamma$ is* **BLmf***-satisfiable.* □

Proof. By Proposition 28, $closure(\Gamma)$ is **BLmf**-satisfiable. It follows that $\Gamma$ is **BLmf**-satisfiable, since $\Gamma \subseteq closure(\Gamma)$. ■

**Proposition 30.** *The satisfiability problem for finite sets of* **BLmf***-literals is decidable in polynomial time.* □

Proof. By Propositions 21 and 23. ■

**Proposition 31.** *The satisfiability problem for* **BLmf**-*formulae is* $\mathcal{NP}$-*complete.* □

Proof. The satisfiability problem for **BLmf**-formulae is clearly $\mathcal{NP}$-hard, In order to show membership to $\mathcal{NP}$, it suffices to note that one can check whether a **BLmf**-formula is **BLmf**-satisfiable by:

1. guessing a disjunct $\Gamma$ of a DNF of $\varphi$;
2. converting $\Gamma$ to a conjunction of normalized literals $\Gamma'$;
3. computing $closure(\Gamma')$;
4. checking whether $closure(\Gamma')$ is consistent. ■

## 5 The Language CLmf

In this section we define the language **CLmf** (*complete lattices with monotone functions*) and we prove that it is equivalent to the language **BLmf** for quantifier-free formulae.

Syntactically, the language **CLmf** coincides with **BLmf**. Semantically, we have the following definition.

**Definition 32.** A **CLmf**-interpretation $\mathcal{A}$ is **BLmf**-interpretation in which the lattice $\left(A, \leq^{\mathcal{A}}, \sqcup^{\mathcal{A}}, \sqcap^{\mathcal{A}}, \mathbf{1}^{\mathcal{A}}, \mathbf{0}^{\mathcal{A}}\right)$ is complete. □

**Proposition 33.** *Let* $\varphi$ *be a quantifier-free* **BLmf**- *or* **CLmf**-*formula. Then* $\varphi$ *is* **BLmf**-*satisfiable if and only if* $\varphi$ *is* **CLmf**-*satisfiable.* □

Proof. Assume first that $\varphi$ is **CLmf**-satisfiable. Since every **CLmf**-interpretation is also a **BLmf**-interpretation, it follows that $\varphi$ is **BLmf**-satisfiable.

Conversely, assume that $\varphi$ is **BLmf**-satisfiable. Without loss of generality, we can assume that $\varphi$ is a normalized set of **BLmf**-literals. Let $\psi = closure(\varphi)$. By Proposition 26, $\psi$ is **BLmf**-satisfiable. It follows that $\psi$ is consistent. By Proposition 28, $\psi$ is **CLmf**-satisfiable. Since $\varphi \subseteq \psi$, it follows that $\varphi$ is **CLmf**-satisfiable. ■

## 6 Conclusion

We presented a decision procedure for the quantifier-free satisfiability problem of the language **BLmf** (*Bounded Lattices with monotone functions*). The language contains the predicates $=$ and $\leq$, the operators $\sqcap$ and $\sqcup$ over terms which may involve uninterpreted unary function symbols, predicates for expressing increasing and decreasing monotonicity of functions, and a predicate for pointwise function comparison.

We proved that our decision procedure runs in polynomial time $\mathcal{O}(m^4)$ for conjunctions of literals, thus entailing that the quantifier-free satisfiability problem for **BLmf** is $\mathcal{NP}$-complete.

Finally, we defined the language **CLmf** (*Complete Lattices with monotone functions*), and we proved that the languages **CLmf** and **BLmf** are equivalent for quantifier-free formulae.

In our proofs, we used the hypothesis that lattices are bounded (see, for instance, Proposition 28, case of literals of the form *inc*($f$), subcases 2c and 2d). Thus, a possible direction of future research would be to relax this hypothesis, and study the language **Lmf** (*Lattices with monotone functions*) in which the semantics does not require lattices to be bounded. We conjecture that **Lmf** is decidable. A promising result in this direction can be found in [13], where decidability is proved after removing from **Lmf** the predicates *dec*($f$), *const*($f$), and *leq*($f, g$).

## Acknowledgments

We are grateful to Viorica Sofronie-Stokkermans for fruitful discussions on lattice theory.

## References

1. D. Cantone, A. Ferro, E. G. Omodeo, and J. T. Schwartz. Decision algorithms for some fragment of analysis and related areas. *Communications on Pure and Applied Mathematics*, 40(3):281–300, 1987.
2. D. Cantone and C. G. Zarba. A decision procedure for monotone functions over lattices. In F. Buccafurri, editor, *Joint Conference on Declarative Programming APPIA-GULP-PRODE*, pages 1–12, 2003.
3. R. Dedekind. *Stetigkeit und Irrationale Zahlen.* Braunschweig: F. Vieweg, 1872.
4. N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. V. Leeuwen, editor, *Handbook of Theoretical Computer Science (vol. B): Formal Models and Semantics*, pages 243–320. MIT Press, Cambridge, MA, USA, 1990.
5. J. Desharnais, B. Möller, and G. Struth. Modal Kleene algebra and applications — A survey —. *Journal on Relational Methods in Computer Science*, 1:93–131, 2004.
6. A. Ferro, E. G. Omodeo, and J. T. Scwhartz. 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.
7. S. Hazelhurst and C.-J. H. Seger. Model checking lattices: Using and reasoning about information orders for abstraction. *Logic Journal of the IGPL*, 7(3):375–411, 1999.
8. H. M. MacNeille. Partially ordered sets. *Transactions of the American Mathematical Society*, 42:416–460, 1937.
9. D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science.* Kluwer Academic Publishers, Boston, Massachusetts, 2002.
10. P. D. Mosses. Denotational semantics. In J. V. Leeuwen, editor, *Handbook of Theoretical Computer Science (vol. B): Formal Models and Semantics*, pages 575–631. MIT Press, Cambridge, MA, USA, 1990.

11. F. J. Oles. An application of lattice theory to knowledge representation. *Theoretical Computer Science*, 249(1):163–196, 2000.
12. M. Osorio, B. Jayaraman, and D. A. Plaisted. Theory of partial-order programming. *Science of Computer Programming*, 34(3):207–238, 1999.
13. V. Sofronie-Stokkermans. Hierarchic reasoning in local theory extensions. In R. Nieuwenhuis, editor, *Automated Deduction – CADE-20*, volume 3632 of *Lecture Notes in Computer Science*, pages 219–234. Springer, 2005.
14. A. Tarski. Undecidability of the theory of lattices and projective geometries. *Journal of Symbolic Logic*, 14(1):77–78, 1949.