Reachability Analysis of the Equivalence of Two Terms in Free Orthomodular Lattices

Tsubasa Takagi[1], Canh Minh Do[2] and Kazuhiro Ogata[2]

[1] *Tokyo Institute of Technology, Tokyo, Japan*
[2] *Japan Advanced Institute of Science and Technology, Nomi, Japan*

Abstract
Since 1936, the algebraic structure of quantum mechanics called orthomodular lattices have attracted many logicians' attention. Over its long history, various complex theorems have been manually proven. Some tools have been developed to automatically verify these theorems by means of checking the equivalence of two terms (the word problem). However, existing tools cannot deal with terms that consist of three or more free variables. Although the number of normal forms in free orthomodular lattices with three or more generators is infinite, that of two generators is only 96. To overcome this limitation, we transform the word problem into a reachability problem. Using this idea, we implement a support tool in Maude. It consists of a formal specification of free orthomodular lattices and an implementation of (1) the 96 normal forms of two generators and (2) a theorem describing when the distributive laws can be applied. The reachability analysis is performed using the `search` command in Maude to deal with terms that consist of even three or more variables. We conduct some case studies with the support tool to verify various complex theorems that existing tools cannot verify.

Keywords
Orthomodular Lattice, Word Problem, Reachability Analysis, Maude

1. Introduction

The algebraic structure of quantum mechanics has been discussed by many logicians since 1936 [1]. Among various algebras, orthomodular lattices (sometimes called quantum logic) [2] are particularly significant because the set of all closed subspaces of a Hilbert space forms an orthomodular lattice, which is called a Hilbert lattice [3].

So far, some theorems such as the Foulis-Holland theorem [4, 5] and a theorem [2, Corollary VII.7.7] about skew join and skew meet are verified [6] with the aid of computer programs [7, 8]. Furthermore, several new theorems regarding the verification of the equivalence of two terms have been obtained with the aid of these programs. For example, all associative operators [9], monotonic operators [10], and weak associative operators [11] in orthomodular lattices have been enumerated with their results, which could be hardly obtained without computer support.

The word problem for free orthomodular lattices remains an open problem in general [12]. To make matters worse, even if the word problem is solvable, it may not be implemented by finite convergent AC (associative and commutative) term rewriting systems. For example, it is

International Workshop on Formal Analysis and Verification of Post-Quantum Cryptographic Protocols 2023
takagi.t.ah@m.titech.ac.jp (T. Takagi); canhdo@jaist.ac.jp (C. M. Do); ogata@jaist.ac.jp (K. Ogata)
0000-0001-9890-1015 (T. Takagi); 0000-0002-1601-4584 (C. M. Do); 0000-0002-4441-3259 (K. Ogata)

known that there is no finite convergent AC term rewriting system for the equational theory of lattices [13]. However, although the word problem may not be solvable, it is still worth implementing a tool that can solve the word problem, not for any terms but for some specific terms of free orthomodular lattices, which consist of three or more free variables.

There are two existing programs [14, 7] for checking the equivalence of two terms in free orthomodular lattices. However, these programs cannot deal with terms that consist of three or more free variables. This is because there are infinite normal forms in three or more free generators, even though there are only 96 normal forms in the case of two free generators (Theorem 3.3). This limitation to some two free generators is fatal when proving theorems expressed by three or more generators in orthomodular lattices.

In this paper, we overcome this limitation by incorporating the idea of reachability analysis into a neither confluent nor terminating term rewriting system for free orthomodular lattices. The word problem is transformed into a reachability problem in the term rewriting system by searching for the reachable state space from an initial state. It is done by the breadth-first search (BFS) to find a solution in which the two terms in the word problem are rewritten into the same term after a finite number of rewrite steps, where each rewrite step can be regarded as a state transition. The reachability problem is conducted through a breadth-first search in an incremental way, which does not strictly require the reachable state space to be finite. However, the reachability analysis may not terminate in general.

Based on this idea, we implement a support tool in Maude, a rewriting logic-based specification/programming language that can deal with terms that consist of three or more free variables. The reachability analysis is performed using the `search` command, a reachability analyzer, in Maude. The support tool consists of a formal specification of free orthomodular lattices and an implementation of (1) the 96 normal forms of two generators (see Theorem 3.3) and (2) a theorem (see Theorem 3.4) describing when the distributive laws can be applied to check the word problem for free orthomodular lattices. Theorem 3.3 has been used in previous work [14, 7], while Theorem 3.4 is a new theorem proposed in this paper. To demonstrate the effectiveness of our approach, we verify the validity of some axioms with three free variables in several implication algebras [15, 16, 17, 18].

The rest of the paper is organized as follows. Section 2 reviews some kinds of lattices. Section 3 describes the theoretical background for the word problem for various free lattices and two significant theorems in orthomodular lattices. Section 5 presents the implementation of our tool in Maude. Section 6 shows some case studies. Section 7 compares our tool and existing tools. Finally, Section 8 concludes the paper together with some future directions.

2. Preliminaries

This section presents some kinds of lattices, such as distributive lattices, modular lattices, ortholattices, Boolean lattices, modular ortholattices, and orthomodular lattices. All of these lattices are defined using only equations. This makes it possible to implement them in algebraic specification languages (e.g., Maude [19]).

2.1. Distributive Lattice and Modular Lattice

Definition 2.1. A lattice is a triple $(L, \wedge, \vee)$ that consists of a non-empty set L and functions $\wedge : L \times L \to L$ and $\vee : L \times L \to L$ satisfying

1. (Associativity) $p \wedge (q \wedge r) = (p \wedge q) \wedge r$ and $p \vee (q \vee r) = (p \vee q) \vee r$,
2. (Commutativity) $p \wedge q = q \wedge p$ and $p \vee q = q \vee p$,
3. (Idempotency) $p \wedge p = p$ and $p \vee p = p$,
4. (Absorption) $p \wedge (p \vee q) = p$ and $p \vee (p \wedge q) = p$.

Given a lattice $(L, \wedge, \vee)$, a partial order (a binary relation that is reflexive, transitive, and antisymmetric) $\leq$ on L is defined by

$$\leq \, = \{(p, q) : p \wedge q = p\} = \{(p, q) : p \vee q = q\}.$$

Then, $p \wedge q$ is identical to the infimum (the greatest lower bound) of $\{p, q\}$, and $p \vee q$ is identical to the supremum (the least upper bound) of $\{p, q\}$.

The following properties of lattices are of particular significance.

Definition 2.2. A lattice $(L, \wedge, \vee)$ is said to be

- bounded if it has the least element (denoted by $\curlywedge$) and the greatest element (denoted by $\curlyvee$) under the partial order $\leq$.
- distributive if the distributive law holds:

  $$p \wedge (q \vee r) = (p \wedge q) \vee (p \wedge r)$$

  for any $p, q, r \in L$.
- modular if the modular law holds:

  $$p \wedge (q \vee (p \wedge r)) = (p \wedge q) \vee (p \wedge r)$$

  for any $p, q, r \in L$.

Example 2.3 (Powerset Lattice)**.** Let $\wp(X)$ be the powerset of a set X. Then, $(\wp(X), \cap, \cup)$ called the powerset lattice on X is a distributive lattice.

Example 2.4. Let $\mathcal{G} = (G, +)$ be an additive group, and $L(\mathcal{G})$ be the set of all subgroups of $\mathcal{G}$. Then, $(L(\mathcal{G}), \cap, +)$ is a modular lattice, where $+$ on $L(\mathcal{G})$ is defined by

$$G_1 + G_2 = \{g_1 + g_2 : g_1 \in G_1, g_2 \in G_2\}.$$

Note that $G_1 + G_2$ is the smallest subgroup of $\mathcal{G}$ containing $G_1 \cup G_2$.

By using the partial order $\leq$, the modular law can be rephrased as follows [20]:

$$p \leq r \text{ implies } p \vee (q \wedge r) = (p \vee q) \wedge r.$$

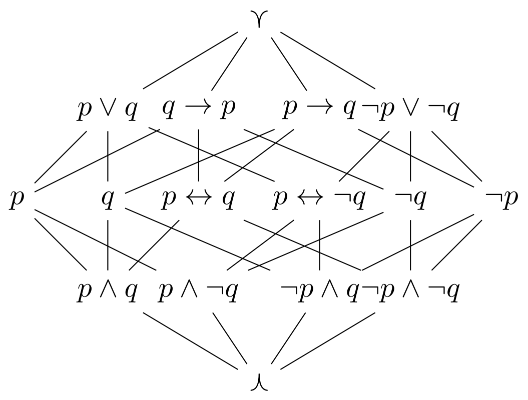
2.2. Ortholattice and Orthomodular Lattice

Definition 2.5. An ortholattice (also called an orthocomplemented lattice) is a bounded lattice equipped with an orthocomplementation. An orthocomplementation on a bounded lattice $(L, \wedge, \vee)$ is a function $\neg : L \to L$ such that

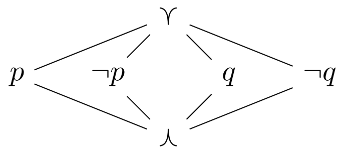
1. $p \wedge \neg p = \curlywedge$ and $p \vee \neg p = \curlyvee$,
2. $\neg\neg p = p$,
3. $\neg(p \wedge q) = \neg p \vee \neg q$ and $\neg(p \vee q) = \neg p \wedge \neg q$,

for any $p, q \in L$. In particular, distributive ortholattices are called Boolean lattices.

Example 2.6 (Four-dimensional Hypercube)**.** The lattice depicted in the following Hasse diagram is a Boolean lattice, where $p \to q$ and $p \leftrightarrow q$ are abbreviations for $\neg p \vee q$ and $(p \wedge q) \vee (\neg p \wedge \neg q)$, respectively. This lattice is called the four-dimensional hypercube and is denoted by $\mathbf{2}^4$.



Example 2.7 (Chinese Lantern)**.** The lattice depicted in the following Hasse diagram is a (non-distributive) modular ortholattice. This lattice is called the Chinese lantern and is denoted by MO_2.



Example 2.8 (Boolean Powerset Lattice)**.** Let ^{c} be the set complementation in a set X. Then, $(\wp(X), \subseteq, ^c)$ is a Boolean lattice and is called the Boolean powerset lattice on X.

Example 2.9 ([3, Proposition 4.3])**.** Let $\mathcal{H} = (H^{\text{fin}}, +, \cdot)$ be a finite-dimensional Hilbert space over the complex numbers and $\mathcal{C}(\mathcal{H})$ be the set of all (closed) subspaces of $\mathcal{H}$. Then, $(\mathcal{C}(\mathcal{H}), \cap, +, ^\perp)$ is a modular ortholattice, where $+$ on $\mathcal{H}$ is defined by

$$V_1 + V_2 = \{v_1 + v_2 : v_1 \in V_1, v_2 \in V_2\},$$

and $v' \in V^\perp$ if and only if the inner product of v' and v is 0 for any $v \in V$.

Note that $\mathcal{C}(\mathcal{H})$ is not modular if the dimension of $\mathcal{H}$ is infinite [3, Proposition 4.4]. However, quantum mechanics are formulated by an infinite dimensional Hilbert space in general. This motivates us to weaken the modular law to the orthomodular law below.

Definition 2.10. An orthomodular lattice is an ortholattice $(L, \wedge, \vee, \neg)$ satisfying the orthomodular law [3, Proposition 3.5]:

$$p \leq q \text{ implies } q = p \vee (\neg p \wedge q),$$

for any $p, q \in L$

For any Hilbert space $\mathcal{H}$, regardless of its dimension, $(\mathcal{C}(\mathcal{H}), \cap, +, ^{\perp})$ is an orthomodular lattice [3, Proposition 4.5]. Thus, modular ortholattices and orthomodular lattices are different.

By the following theorem, the class of orthomodular lattices is characterized only by equations.

Theorem 2.11. *Let $(L, \wedge, \vee, \neg)$ be an ortholattice. $(L, \wedge, \vee, \neg)$ is an orthomodular lattice if and only if*

$$p \vee q = p \vee (\neg p \wedge (p \vee q)),$$

for any $p, q \in L$.

Proof. See [2, Theorem II.5.1], and use the absorption law. □

Definition 2.12. Let $\mathcal{L}_1 = (L_1, \wedge_1, \vee_1, \neg_1)$ and $\mathcal{L}_2 = (L_2, \wedge_2, \vee_2, \neg_2)$ be ortholatices. Then,

$$\mathcal{L}_1 \times \mathcal{L}_2 = (L_1 \times L_2, \wedge, \vee, \neg)$$

defined by

$$\begin{aligned}(p_1, p_2) \wedge (q_1, q_2) &= (p_1 \wedge_1 q_1, q_1 \wedge_2 q_2),\\ (p_1, p_2) \vee (q_1, q_2) &= (p_1 \vee_1 q_1, q_1 \vee_2 q_2),\\ \neg(p_1, p_2) &= (\neg_1 p_1, \neg_2 p_2)\end{aligned}$$

is an ortholattice and is called the direct product of $\mathcal{L}_1$ and $\mathcal{L}_2$.

3. Theoretical Background

The main contribution of this paper is to implement a support tool to verify the equivalence of two terms in free orthomodular lattices. Its principle is based on two fundamental theorems explained in this section. Although the first theorem (Theorem 3.3) has been employed for the implementation in the previous work [14, 7], the second theorem (Theorem 3.4) is a new theorem to be proved in this paper and has not been used in previous work. The second theorem allows verifying the equivalence between a broader class of terms than existing work.

3.1. Free Algebra

Definition 3.1. Let $\mathcal{C}$ be a non-empty class of algebras. An algebra $F_X \in \mathcal{C}$ is called a free algebra in $\mathcal{C}$ generated by X, if F_X is generated by $X \subseteq F_X$ and every function $V : X \to A \in \mathcal{C}$ can be uniquely extended to a homomorphism $\widehat{V} : F_X \to A$.

Example 3.2. The Boolean lattice $\mathbf{2}^4$ in Example 2.6 is a free Boolean lattice with two generators p and q.

In general, the free Boolean lattice with n generators is the Boolean lattice $\mathbf{2}^{2^n}$ that consists of 2^{2^n} elements [20, Theorem Ⅲ.7].

3.2. The Word Problem for Various Free Lattices

The problem of deciding whether or not two given terms (words) of algebras are equivalent is called the word problem for the algebras. One possible way to check it is to try to reduce the two given terms to the same term in a finite number of steps.

The word problem for various kinds of lattices is one of the central topics in lattice theory. It has been shown that the word problem for free lattices is solvable [21] (for more details, see [22]). The following results have been obtained for various free lattices with some restrictions.

- The word problem for free distributive lattices is solvable [23].
- The word problem for free modular lattices with $n \leq 3$ generators is solvable (because the free modular lattice with three generators has only 28 terms [24]), and that for free modular lattices with $n \geq 4$ generators is unsolvable [25].
- The word problem for free ortholattices is solvable [26].
- The word problem for the free modular ortholattice with $n \leq 2$ generators is solvable (because the free modular ortholattice with two generators has only 96 terms [27]), and that for general free modular ortholattices remains an open problem [28].
- The word problem for the free orthomodular lattice with $n \leq 2$ generators is solvable (because the free orthomodular lattice with two generators has only 96 terms [2, Theorem Ⅲ.2.8]), and that for general free modular ortholattices remains an open problem [12].

From the perspective of quantum mechanics, the word problem for free orthomodular lattices is the most significant among the above-listed word problems. This is because the set of all closed subspaces (experimental propositions [1]) of a Hilbert space is an orthomodular lattice [3, Proposition 4.5].

3.3. Two Fundamental Theorems for Orthomodular Lattices

To solve the word problem for some specific free orthomodular lattices, we use two fundamental theorems for orthomodular lattices.

The first fundamental theorem states that there are only 96 normal forms in the free orthomodular lattice with two generators. This theorem has been used in previous work [14, 7].

Theorem 3.3. *$MO_2 \times \mathbf{2}^4$ is isomorphic to the free orthomodular lattice with two generators.*

Proof. See [2, Theorem III.2.8]. $\square$

Because the number of normal forms in MO_2 is 6 (recall Example 2.7) and that in $\mathbf{2}^4$ is 16 (recall Example 2.6), that of $MO_2 \times \mathbf{2}^4$ is $6 \times 16 = 96$.

Also, Theorem 3.3 clarifies the relationship between any two normal forms. Let a be an element of MO_2 distinct from $\curlywedge$ and $\curlyvee$, and $\mathbf{2} = (\{\curlywedge, \curlyvee\}, \wedge, \vee, \neg)$ be the two-element Boolean lattice. Then, for example, p (referred to as $\mathfrak{p}_{22}$) and $\neg p \wedge (p \vee q)$ (referred to as $\mathfrak{p}_{68}$) are normal forms corresponding to $(a, \curlyvee, \curlyvee, \curlywedge, \curlywedge)$ and $(\neg a, \curlywedge, \curlywedge, \curlyvee, \curlywedge)$ in $MO_2 \times \mathbf{2} \times \mathbf{2} \times \mathbf{2} \times \mathbf{2}$, which is isomorphic to $MO_2 \times \mathbf{2}^4$, respectively [2, Fig. 18]. Note that a be an element of MO_2. By definitions of MO_2 and $\mathbf{2}$,

$$(a, \curlyvee, \curlyvee, \curlywedge, \curlywedge) \vee (\neg a, \curlywedge, \curlywedge, \curlyvee, \curlywedge) = (\curlyvee, \curlyvee, \curlyvee, \curlyvee, \curlywedge).$$

Here, $(\curlyvee, \curlyvee, \curlyvee, \curlyvee, \curlywedge)$ corresponds to $p \vee q$ [2, Fig. 18] (referred to as $\mathfrak{p}_{92} = \mathfrak{p}_{22} \vee \mathfrak{p}_{68}$), which is also one of the 96 normal forms. Therefore, we obtain the result that

$$p \vee (\neg p \wedge (p \vee q)) = p \vee q$$

holds in orthomodular lattices. In fact, this equation is the orthomodular law (recall Theorem 2.11). A list describing the relationship between any two normal forms is in [2, Fig. 18].

The second fundamental theorem tells us when the distributive law can be applied in orthomodular lattice. This is a new theorem proposed in this paper.

Theorem 3.4. *Let $(L, \wedge, \vee, \neg)$ be an orthomodular lattice. Then, the following are equivalent:*

1. $p \wedge (q \vee r) = (p \wedge q) \vee (p \wedge r)$;
2. $p \wedge (\neg p \vee q) = p \wedge q$.

Dually, the following are also equivalent:

1. $p \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$;
2. $p \vee (\neg p \wedge q) = p \vee q$.

Proof. We only prove the first part of the theorem. The dual part is shown similarly.

Proof of (1) $\Rightarrow$ (2). By (1),

$$p \wedge (\neg p \vee q) = (p \wedge \neg p) \vee (p \wedge q) = p \wedge q.$$

Proof of (2) $\Rightarrow$ (1). Recall that the distributive inequality

$$(p \wedge q) \vee (p \wedge r) \leq p \wedge (r \vee q)$$

always holds even in non-distributive lattices (see [29, Exercise 2.9 (ii)]). Thus, $p \wedge (q \vee r)$ is an upper bound of $\{p \wedge q, p \wedge r\}$. Thus, it remains to show that $p \wedge (q \vee r)$ is the least upper bound of $\{p \wedge q, p \wedge r\}$: $p \wedge (q \vee r) \leq s$ for any upper bound s of $\{p \wedge q, p \wedge r\}$. Clearly,

$$(p \wedge (q \vee r)) \wedge s \leq p \wedge (q \vee r).$$

By applying the orthomodular law (Definition 2.10),

$$p \wedge (q \vee r) = ((p \wedge (q \vee r)) \wedge s) \vee ((\neg((p \wedge (q \vee r)) \wedge s)) \wedge (p \wedge (q \vee r))).$$

Hence, it suffices to show

$$(\neg((p \wedge (q \vee r)) \wedge s)) \wedge (p \wedge (q \vee r)) = \curlywedge.$$

Because s is an upper bound of $\{p \wedge q, p \wedge r\}$, we obtain $p \wedge q \leq s$ and $p \wedge r \leq s$. Therefore,

$$p \wedge q = (p \wedge q) \wedge s \leq (p \wedge (q \vee r)) \wedge s$$

and

$$p \wedge r = (p \wedge r) \wedge s \leq (p \wedge (q \vee r)) \wedge s.$$

That is,

$$\neg((p \wedge (q \vee r)) \wedge s) \leq \neg(p \wedge q) = \neg p \vee \neg q$$

and

$$\neg((p \wedge (q \vee r)) \wedge s) \leq \neg(p \wedge r) = \neg p \vee \neg r.$$

It follows from the assumption (2) that

$$\begin{aligned} &\neg((p \wedge (q \vee r)) \wedge s) \wedge (p \wedge (q \vee r)) \leq (\neg p \vee \neg q) \wedge (p \wedge (q \vee r)) \\ &= ((\neg p \vee \neg q) \wedge p) \wedge (q \vee r) = (p \wedge \neg q) \wedge (q \vee r). \end{aligned}$$

Similarly, we have

$$\neg((p \wedge (q \vee r)) \wedge s) \wedge (p \wedge (q \vee r)) \leq (p \wedge \neg r) \wedge (q \vee r).$$

Consequently,

$$\begin{aligned} &\neg((p \wedge (q \vee r)) \wedge s) \wedge (p \wedge (q \vee r)) \leq ((p \wedge \neg q) \wedge (q \vee r)) \wedge ((p \wedge \neg r) \wedge (q \vee r)) \\ &= (p \wedge (\neg q \wedge \neg r)) \wedge (q \vee r) = p \wedge (\neg(q \vee r) \wedge (q \vee r)) = p \wedge \curlywedge = \curlywedge. \end{aligned}$$

□

In the next section, we combine these two fundamental theorems to implement a support tool.

4. Implementation

We develop a support tool in Maude [19] to check the word problem for free orthomodular lattices. Maude is a high-performance specification/programming based on rewriting logic [30]. The language can directly specify order-sorted equational logic and rewriting logic, and it provides several formal analysis methods, such as reachability analysis through a breadth-first search of the reachable state space from a given state. The reachability analysis is performed using the `search` command in Maude. In the support tool, we formally specify free orthomodular lattices, and develop an implementation of Theorem 3.3 and Theorem 3.4 to check the word problem using the `search` command. This section gives the syntax of the Maude language in a nutshell (see [19] for more details) and briefly describes how the tool is developed.

4.1. Overview of Maude

Functional modules

A functional module $\mathcal{M}$ specifies an order-sorted equational logic theory (Σ, E) with the syntax: **`fmod`** $\mathcal{M}$ **`is`** (Σ, E) **`endfm`**, where Σ is an order-sorted signature and E is the collection of equations in the functional module. (Σ, E) may contain a set of declarations as follows:

- importations of previously defined modules (**`protecting`** ... or **`extending`** ... or **`including`** ...)
- declarations of sorts (**`sort`** s **`.`** or **`sorts`** $s\ s'$ **`.`**)
- subsort declarations (**`subsort`** $s < s'$ **`.`**)
- declarations of function symbols (**`op`** f **`:`** $s_1\ \ldots\ s_n \rightarrow s\ [att_1 \ldots\ att_k]$ **`.`**)
- declarations of variables (**`vars`** $v\ v'$ **`.`**)
- declarations of unconditional equations (**`eq`** $t = t'$ **`.`**)
- declarations of conditional equations (**`ceq`** $t = t'$ **`if`** $cond$ **`.`**)

where $s, s_1, \ldots, s_n$ are sort names, v, v' are variable names, t, t' are terms, $cond$ is a conjunction of equations (i.e., $t = t'$ and/or $t \Rightarrow t'$), and $att_1, \ldots att_k$ are equational attributes. Equations are used as *equational rules* to perform the simplification in which instances of the left-hand side pattern are subterms of a subject term replaced by the corresponding instances of the right-hand side. The process is called *term rewriting*, and the result of simplifying a term is called its *normal form*.

System modules

A system module $\mathcal{R}$ specifies a rewrite theory (Σ, E, R) with the syntax: **`mod`** $\mathcal{R}$ **`is`** (Σ, E, R) **`endm`**, where Σ and E are the same as those in an equational theory, and R is the collection of rewrite rules in the system module. (Σ, E, R) may contain all possible declarations in (Σ, E) and rewrite rules in R as follows:

- declarations of unconditional rewrite rules (**`rl`** `[`lbl`]` **`:`** $u \Rightarrow v$ **`.`**)
- declarations of conditional rewrite rules (**`crl`** `[`lbl`]` **`:`** $u \Rightarrow v$ **`if`** $cond$ **`.`**)

where lbl is the name of a rewrite rule, u, v are terms, and $cond$ is a conjunction of equations and/or rewrites (e.g., $t \Rightarrow t'$). rewrite rules are executed as rewriting from left to right modulo the equations in the system module and are regarded as *local transition rules*, making possible many state transitions from a given state as concurrent changes of a system.

Reachability analysis

The word problem of two terms t_1 and t_2 can be transformed into a reachability problem. Suppose that t `=?=` t `=` `true` for any term t, then the word problem is to check whether `true` is reachable from t_1 `=?=` t_2 referred to as t_1 `=?=` t_2 $\rightarrow^*_{R,E}$ `true`, where R, E are the rewrite rules and equations in the specification of a system concerned. If that is the case, the two terms t_1 and t_2 are reduced to the same term in a finite number of rewrite steps, meaning that the word problem is solved for t_1 and t_2.

For the sake of simplicity, we use a Boolean lattice $\mathcal{L} = (L, \wedge, \vee, \neg)$ as an example to demonstrate how to use the `search` command in Maude to tackle the reachability problem to solve the word problem for $\mathcal{L}$. The following system module formally specifies $\mathcal{L}$.

```
mod BOOLEAN-LATTICE is
  sort Lattice .
  ops bot top : -> Lattice [ctor] .
  op not_ : Lattice -> Lattice [prec 31] .
  op _and_ : Lattice Lattice -> Lattice [assoc comm prec 32] .
  op _or_ : Lattice Lattice -> Lattice [assoc comm prec 33] .
  op _|->_ : Lattice Lattice -> Lattice [prec 34] .
  vars P Q R : Lattice .
  rl [imply] : P |-> Q => not P or Q .
  --- Idempotent Laws
  eq P and P = P .    eq P or P = P .
  --- Absorption Laws
  eq P and (P or Q) = P .    eq P or (P and Q) = P .
  --- Orthocomplementation
  eq P and not P = bot .    eq P or not P = top .
  eq not(not P) = P .
  eq not (P and Q) = not P or not Q .
  eq not (P or Q) = not P and not Q .
  --- Distributive Laws
  eq P or (Q and R) = (P or Q) and (P or R) .
  eq P and (Q or R) = (P and Q) or (P and R) .
  --- Additional Identities
  eq not top = bot .    eq not bot = top .
  eq P and top = P .    eq P and bot = bot .
  eq P or bot = P .     eq P or top = top .
  --- Checking Equivalence of Two Lattices
  op true : -> Lattice [ctor] .
  op _=?=_ : Lattice Lattice -> Lattice .
  eq P =?= P = true .
endm
```

where the `_and_`, `_or_`, `not_`, `bot`, `top` operators represent $\wedge, \vee, \neg, \curlywedge, \curlyvee$ in the Boolean lattice $\mathcal{L}$, respectively. `P`, `Q`, and `R` are Maude variables of the sort `Lattice` representing elements of L. The `assoc` and `comm` attributes denote associative and commutative properties. The `prec_` attribute, where _ is a natural number, denotes the precedences of operators for parsing in Maude (e.g., `not P and Q or R` is parsed as `((not P)and Q)or R` without using parentheses). The idempotent laws, absorption laws, distributive laws, involution, complements, identities, and De Morgan's laws are defined in the form of equations, while the implication is defined in the form of a rewrite rule (see the rule `imply`). Although we should define the implication using an equation, we intentionally use the rewrite rule `imply` intending to make concurrent changes in this example. For example, if a term consists of at least two sub-terms of in the form of implication, either one of the two sub-terms can be rewritten using the rewrite rule `imply`, making concurrent changes for the subject term. This is because rewrite rules can be used instead of equations for equational reasoning. The `_=?=_` operator is used for constructing the initial term of the word problem in reachability analysis such that whenever two terms at both sides of the operator are equivalent, it becomes `true`, which is a special element of P denoting the validity of the word problem.

We define some arbitrary elements of P in the form of some operators (or fresh constants) in the following module:

```
mod BOOLEAN-LATTICE-TEST is
  pr BOOLEAN-LATTICE .
  ops p q r : -> Lattice .
endm
```

We can verify the axiom (I3) of Abbott's implication algebra [15] using the following command in Maude.

```
search [1] p |-> (q |-> r) =?= q |-> (p |-> r) =>* true .
```

The command says that we would like to search on the reachable state space of $\mathcal{L}$ from the initial state `p |-> (q |-> r)=?= q |-> (p |-> r)` to check whether there is one solution in which `true` is reachable from the initial state after zero or more rewrite steps. Note that given the initial state as a term, the rewrite rule `imply` can be applied at four positions of the term, making four possible successor states from the initial state as follows:

```
(1) not p or (q |-> r) =?= q |-> p |-> r
(2) p |-> (not q or r) =?= q |-> p |-> r
(3) p |-> q |-> r =?= not q or (p |-> r)
(4) p |-> q |-> r =?= q |-> (not p or r)
```

In this way, the reachable state space of $\mathcal{L}$ from the initial state is constructed. The result of the command returns a solution and then we can conclude that the axiom (I3) is verified using the `search` command in Maude.

5. A Support Tool

We develop a support tool in Maude and use the `search` command to check the word problem for free orthomodular lattices. The tool is publicly available at https://github.com/canhminhdo/FOM. The basic idea of developing the tool is similar to what we have described above on how to use the `search` command to verify the axiom (I3) in Maude. Moreover, we develop an implementation of Theorem 3.3 and Theorem 3.4 in Maude to check the word problem for free orthomodular lattices.

First, we prepare a formal specification of free orthomodular lattices in Maude, where some properties are specified, such as associativity, commutativity, idempotency of $\wedge$ and $\vee$, the absorption laws, De Morgan's laws, and so on. All properties are described by means of equations in Maude. Let E_{FOM} be the sets of equations from specifying free orthomodular lattices.

Second, we formalize Theorem 3.4 with the compatibility relation using rewrite rules. If the subject term can be applied by the rewrite rules of Theorem 3.4 at multiple positions, the order of applying the rewrite rules may produce different results in the end. Therefore, we specify Theorem 3.4 using rewrite rules instead of equations to make concurrent changes in our system so that all possible cases can be taken into account using reachability analysis. Moreover, we may need to apply distributive laws from not only left to right but also right to left. If we use equations to do so, it will make an infinite computation because of a loop caused by constantly applying the equations from left to right and vice versa for a subject term if distributive laws are applicable. Therefore, we use rewrite rules to specify Theorem 3.4. Let R_{DIST} be the set of rewrite rules obtained from specifying Theorem 3.4.

Third, we formalize the 96 normal forms (Theorem 3.3), and then automatically generate all equations for each $\mathfrak{p}_i \wedge \mathfrak{p}_j$, $\mathfrak{p}_i \vee \mathfrak{p}_j$, and $\neg\mathfrak{p}_i$ that is again an element $\mathfrak{p}_k$, where $\mathfrak{p}_i, \mathfrak{p}_j, \mathfrak{p}_m$, and $\mathfrak{p}_k$ are elements of the 96 normal forms in [2, Fig. 18]. Because $\neg\mathfrak{p}_i$ can be obtained through applying De Morgan's laws specified in E_{FOM}, then we do not need to construct equations for $\neg\mathfrak{p}_i$ of 96 normal forms. Hence, we only focus on generating equations for $\mathfrak{p}_i \wedge \mathfrak{p}_j$ and $\mathfrak{p}_i \vee \mathfrak{p}_j$. For example, $\mathfrak{p}_{92} = \mathfrak{p}_{22} \vee \mathfrak{p}_{68}$ (see Subsection 3.3 for more details), and thus we can obtain the following equation:

```
eq P or (not P and (P or Q)) = P or Q .
```

where `P` and `Q` are variables representing elements of free orthomodular lattices. Let E_{NF} be the set of equations obtained from generating all equations of the 96 normal forms. Note that the left-hand side of each generated equation is simplified by using E_{FOM} beforehand and unnecessary equations (e.g., **eq** `P = P .`) are removed. Hence, there are 5,520 distinct equations in E_{NF} generated automatically by the support tool.

Now the word problem is rephrased as the reachability problem t_1 `=?=` $t_2 \rightarrow^*_{R,E}$ `true` with the relation $\rightarrow^*_{R,E}$ that is $\rightarrow^*_{R_{DIST},E_{FOM}\cup E_{NF}}$.

6. Case Studies

We apply the support tool to show that some axioms with three free variables in several implication algebras [16, 18, 15, 17] are valid. In these experiments, we used one node in computing servers available at our institute that carries a 2.8 GHz microprocessor with 64 cores and 1.5 TB memory of RAM.

Before that, we briefly explain what implication algebras are. The operators $\wedge$, $\vee$, and $\neg$ in orthomodular lattices (also in Boolean lattices) correspond to conjunction, disjunction, and negation, respectively. In Boolean lattices $(L, \wedge, \vee, \neg)$, the operator $\rightarrow$ (material implication) on P is defined by $p \rightarrow q = \neg p \vee q$. This operator $\rightarrow$ satisfies all the minimal requirements for implication called the minimal implicative criteria:

1. $p \leq q$ implies $p \rightarrow q = \curlyvee$;
2. (Modus Ponens) $p \wedge (p \rightarrow q) \leq q$;

3. (Modus Tollens) $\neg q \wedge (p \to q) \le \neg p$.

On the other hand, in orthomodular lattices, there are three implication operators $\rightsquigarrow$ (Sasaki implication [31] or quasi implication [15]), $\rightarrowtail$ (Dishkant implication [18] or ortho-implication [16]), and $\twoheadrightarrow$ (relevance implication [18]) that satisfy the minimal implicative criteria [3, Chapter 8] defined by

$$p \rightsquigarrow q = \neg p \vee (p \wedge q), \quad p \rightarrowtail q = (\neg p \wedge \neg q) \vee q,$$
$$p \twoheadrightarrow q = ((p \wedge q) \vee (\neg p \wedge q)) \vee (\neg p \wedge \neg q).$$

Note that all these three implication operators coincide with $p \to q$ in Boolean lattices (using the distributive law).

For each implication satisfying the minimal implicative criteria, algebras that the only operator is the implication (and $\curlywedge$) have been proposed and are called implication algebras. All the axioms of these implication algebras are satisfied in orthomodular lattices by translating an implication into a term in orthomodular lattices (for example, $p \rightsquigarrow q$ is translated into $\neg p \vee (p \wedge q)$).

Because any axioms that consist of two variables can be verified by existing tool [14, 7], we confine our attention to axioms that consist of three variables in several implication algebras:

- The axiom (Q2) of quasi-implication algebra [15]:

$$(p \rightsquigarrow q) \rightsquigarrow (p \rightsquigarrow r) = (q \rightsquigarrow p) \rightsquigarrow (q \rightsquigarrow r).$$

- The axiom (O2) of ortho-implication algebra [16]:

$$p \rightarrowtail ((q \rightarrowtail p) \rightarrowtail r) = p \rightarrowtail r.$$

- The axiom (O5) of orthomodular implication algebra [17]:

$$(((p \rightarrowtail q) \rightarrowtail q) \rightarrowtail r) \rightarrowtail (p \rightarrowtail r) = \curlyvee$$

- The axiom (O6) of orthomodular implication algebra [17]:

$$\begin{aligned}&((((((((p \rightarrowtail q) \rightarrowtail q) \rightarrowtail r) \rightarrowtail r) \rightarrowtail r) \rightarrowtail p) \rightarrowtail p) \rightarrowtail r) \rightarrowtail p) \rightarrowtail p \\ &= (((p \rightarrowtail q) \rightarrowtail q) \rightarrowtail r) \rightarrowtail r\end{aligned}$$

- The axiom (J4) of Sasaki implication algebra [18]:

$$\begin{aligned}&p \rightsquigarrow ((p \rightsquigarrow ((q \rightsquigarrow ((q \rightsquigarrow r) \rightsquigarrow \curlywedge)) \rightsquigarrow \curlywedge)) \rightsquigarrow \curlywedge) \\ &= r \rightsquigarrow ((r \rightsquigarrow ((p \rightsquigarrow ((p \rightsquigarrow q) \rightsquigarrow \curlywedge)) \rightsquigarrow \curlywedge)) \rightsquigarrow \curlywedge).\end{aligned}$$

- The axiom (K5) of Dishkant implication algebra [18]:

$$(((p \rightarrowtail q) \rightarrowtail q) \rightarrowtail r) \rightarrowtail r = (p \rightarrowtail ((q \rightarrowtail r) \rightarrowtail r)) \rightarrowtail ((q \rightarrowtail r) \rightarrowtail r).$$

- The axiom (L6) of relevance implication algebra [18]:

$$(((p \twoheadrightarrow q) \twoheadrightarrow q) \twoheadrightarrow r) \twoheadrightarrow r = (p \twoheadrightarrow ((q \twoheadrightarrow r) \twoheadrightarrow r)) \twoheadrightarrow ((q \twoheadrightarrow r) \twoheadrightarrow r).$$

Table 1
Experimental results for validating some axioms with our support tool

Target Axiom	Time
The axiom (Q2) in [15]	1,213ms
The axiom (O2) in [16]	736ms
The axiom (O5) in [17]	705ms
The axiom (O6) in [17]	716ms
The axiom (J4) in [18]	715ms
The axiom (K5) in [18]	723ms
The axiom (L6) in [18]	6d:19h:40m

The experimental data for checking these axioms are shown in Table 1. The first and second columns denote the name of each axiom and the time taken to validate each axiom using our support tool. For the first six axioms, our tool can quickly verify their validity for a few moments, which is almost impossible to do so without the aid of computer programs, especially the axioms (O5), (J4), and (K5). For the last axiom (L6), our tool needs to spend six days, 19 hours, and 40 minutes to verify its validity because of its complexity, making a huge state space in the reachability analysis. In summary, we can conclude the validity of axioms (Q2), (O2), (O5), (J4), and (K5) using our support tool. Transforming the word problem into a reachability problem and the use of not only the 96 normal forms (Theorem 3.3) but also Theorem 3.4 for the applicability of distributive laws make our tool extremely useful and distinguishable from existing tools for checking the word problem in free orthomodular lattices.

7. Related Work

In [14], a program is implemented to reduce a term in the free orthomodular lattice $F(p, q)$ with two generators p and q to the normal (canonical) form. It relies on the fact that $F(p, q)$ consists of the 96 normal forms (Theorem 3.3) [2].

After that, in [7], a program that can reduce a term in the free orthomodular lattice $F(p, q, r_1, \dots, r_n)$ generated by $n + 2$ terms $p, q, r_1, \dots, r_n$, where $1 \leq n \leq 9$, and each r_i commutes with all other generators. Note that p is said to commute with q, denoted by pCq, if

$$p = (p \wedge q) \vee (p \wedge \neg q).$$

The implementation in [7] relies on the fact that $F(p, q, r_1, \dots, r_n)$ is isomorphic to the direct product of 2^n copies of $F(p, q)$ [32]. Different from the program in [14], that in [7] makes it possible to verify a much broader class of theorems in orthomodular lattice theory [6], such as the Foulis-Holland theorem [4, 5] and a theorem [2, Corollary VII.7.7] about skew join and skew meet. In addition, some problems, such as verification of associative operators [9], monotonic operators [10], and weak associative operators [11] of operations on orthomodular lattices, which can be reduced to verification of terms in $F(p, q, r_1, \dots, r_n)$ are solved using the program in [7].

Although both implementations in [14] and [7] can help to solve various problems, they still have limitations: that in [14] can only deal with terms consisting of two generators, and that in [7] can only deal with terms in a free orthomodular lattice with restricted generators.

Our tool can overcome these limitations when three or more generators are allowed to be used in terms. Based on Theorem 3.3 and 3.4, our tool effectively checks the word problem in free orthomodular lattices without any restrictions and can cover a broader class of the word problems compared to existing tools.

8. Conclusion

Using a reachability analysis, we have described how to develop the support tool for checking the word problem with three or more generators for free orthomodular lattices. The tool implemented in Maude consists of the formal specification of free orthomodular lattices and the implementation of two fundamental theorems (Theorem 3.3 and Theorem 3.4). We have used the `search` command in Maude to conduct the reachability analysis to conclude the validity of some complicated theorems, demonstrating the power of our support tool.

Acknowledgments

The research of the first author was supported by Grant-in-Aid for JSPS Research Fellow Grant Number JP22KJ1483. The research of the second and the third authors was supported by JST SICORP Grant Number JPMJSC20C2 and JSPS KAKENHI Grant Number JP24H03370.

References

[1] G. Birkhoff, J. von Neumann, The logic of quantum mechanics, Annals of mathematics 57 (1936) 823–843.

[2] L. Beran, Orthomodular lattices: algebraic approach, volume 18 of *Mathematics and its Applications*, 1985.

[3] M. Rédei, Quantum logic in algebraic approach, volume 91 of *Fundamental Theories of Physics*, Springer, 1998.

[4] D. J. Foulis, A note on orthomodular lattices, Portugaliae mathematica 21 (1962) 65–72.

[5] S. S. Holland, A radon-nikodym theorem in dimension lattices, Transactions of the American Mathematical Society 108 (1963) 66–87.

[6] M. Hyčko, M. Navara, Decidability in orthomodular lattices, International Journal of Theoretical Physics 44 (2005) 2239–2248.

[7] M. Hyčko, Implications and equivalences in orthomodular lattices, Demonstratio Mathematica 38 (2005) 777–792.

[8] N. D. Megill, M. Pavičić, Quantum implication algebras, International Journal of Theoretical Physics 42 (2003) 2807–2822.

[9] J. Gabriëls, M. Navara, Associativity of operations on orthomodular lattices, Mathematica Slovaca 62 (2012) 1069–1078.

[10] J. J. Gabriëls, M. Navara, Computer proof of monotonicity of operations on orthomodular lattices, Information Sciences 236 (2013) 205–217.

[11] S. M. Gagola, J. J. Gabriëls, M. Navara, Weaker forms of associativity in orthomodular lattices, Algebra universalis 73 (2015) 249–266.

[12] G. Bruns, J. Harding, Algebraic aspects of orthomodular lattices, in: Current research in operational quantum logic, volume 111 of *Fundamental Theories of Physics*, 2000, pp. 37–65.

[13] R. Freese, J. Ježek, J. B. Nation, Term rewrite systems for lattice theory, Journal of Symbolic Computation 16 (1993) 279–288.

[14] N. D. Megill, M. Pavičić, Orthomodular lattices and a quantum algebra, International Journal of Theoretical Physics 40 (2001) 1387–1410.

[15] G. M. Hardegree, Quasi-implication algebras, part I: Elementary theory, Algebra Universalis 12 (1981) 30–47.

[16] J. C. Abbott, Orthoimplication algebras, Studia Logica 35 (1976) 173–177.

[17] I. Chajda, R. Halaš, H. Länger, Orthomodular implication algebras, International Journal of Theoretical Physics 40 (2001) 1875–1884.

[18] G. N. Georgacarakos, Equationally definable implication algebras for orthomodular lattices, Studia Logica 39 (1980) 5–18.

[19] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, C. L. Talcott (Eds.), All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic, volume 4350 of *Lecture Notes in Computer Science*, 2007.

[20] G. Birkhoff, Lattice theory, volume 25, third ed., American Mathematical Society, 1940.

[21] P. M. Whitman, Free lattices, Annals of Mathematics 42 (1941) 325–330.

[22] R. Freese, J. Ježek, J. B. Nation, Free lattices, volume 42 of *Mathematical Surveys and Monographs*, American Mathematical Society, 1995.

[23] K. Takeuchi, The word problem for free distributive lattices, Journal of the Mathematical Society of Japan 21 (1969) 330–333.

[24] R. Dedekind, Ueber die von drei moduln erzeugte dualgruppe, Mathematische Annalen 53 (1900) 371–403.

[25] C. Herrmann, On the word problem for the modular lattice with four free generators, Mathematische Annalen 265 (1983) 513–527.

[26] G. Bruns, Free ortholattices, Canadian Journal of Mathematics 28 (1976) 977–985.

[27] J. Kotas, An axiom system for the modular logic, Studia Logica 21 (1967) 17–38.

[28] M. S. Roddy, On the word problem for orthocomplemented modular lattices, Canadian Journal of Mathematics 41 (1989) 961–1004.

[29] B. A. Davey, H. A. Priestley, Introduction to lattices and order, 2nd ed., Cambridge University Press, 2002.

[30] J. Meseguer, Twenty years of rewriting logic, J. Log. Algebraic Methods Program. 81 (2012) 721–781.

[31] L. Herman, E. Marsden, R. Piziak, Implication connectives in orthomodular lattices., Notre Dame Journal of Formal Logic 16 (1975) 305–328.

[32] M. Navara, On generating finite orthomodular sublattices, Tatra Mountains Mathematical Publications 10 (1997) 109–117.