Orthologic with Axioms

SIMON GUILLOUD and VIKTOR KUNČAK, EPFL, IC, INR 318, Station 14, Switzerland

We study the proof theory and algorithms for orthologic, a logical system based on ortholattices, which have shown practical relevance in simplification and normalization of verification conditions. Ortholattices weaken Boolean algebras while having polynomial-time equivalence checking that is sound with respect to Boolean algebra semantics. We generalize ortholattice reasoning and obtain an algorithm for proving a larger class of classically valid formulas.

As the key result, we analyze a proof system for orthologic augmented with axioms. An important feature of the system is that it limits the number of formulas in a sequent to at most two, which makes the extension with axioms non-trivial. We show a generalized form of cut elimination for this system, which implies a sub-formula property. From there we derive a cubic-time algorithm for provability from axioms, or equivalently, for validity in finitely presented ortholattices. We further show that propositional resolution of width 5 proves all formulas provable in orthologic with axioms. We show that orthologic system subsumes resolution of width 2 and arbitrarily wide unit resolution and is complete for reasoning about generalizations of propositional Horn clauses.

Moving beyond ground axioms, we introduce effectively propositional orthologic (by analogy with EPR for classical logic), presenting its semantics as well as a sound and complete proof system. Our proof system implies the decidability of effectively propositional orthologic, as well as its fixed-parameter tractability for a bounded maximal number of variables in each axiom. As a special case, we obtain a generalization of Datalog with negation and disjunction.

1 INTRODUCTION

Our goal is to construct efficient building blocks for theorem proving and program verification. coNP-hardness of propositional logic already presents a barrier to large-scale reasoning, such as simplification of large formulas and using intermediate assertions to help software verification. We aim to improve the worst-case efficiency of reasoning while preserving the spirit of a specification language with conjunction, disjunction, and negation. We therefore investigate the concepts of *ortholattices* and *orthologics* as a basis of predictable reasoning.

Non-distributive generalizations of classical logic, including orthologic, were introduced as *quantum logic* to describe experiments in quantum mechanics, where it was realized that distributivity fails [Bell 1983; Birkhoff and Von Neumann 1936]. The term *orthologic* was used in [Goldblatt 1974] for the logic corresponding to the algebraic class of ortholattices, a generalization of Boolean algebras. In particular, the class of closed subsets of a Hilbert space is an ortholattice, but not a Boolean algebra [Rawling and Selesnick 2000]. In theoretical physics, ortholattices are an intermediate step towards the study of *orthomodular lattices* and *modular ortholattices* among others, [Hardegree 1981; Hyčko 2005; Kalmbach 1983; Sherif 1997]. Ortholattices have also found application in modelling of epistemic modal logic [Holliday 2023; Holliday and Mandelkern 2022]. Recently, researchers have proposed to use ortholattices as an efficient approximation to classical logic in automated reasoning. This approach has been applied to design kernels of proof assistants [Guilloud et al. 2023b], as well as in software verification tools [Guilloud et al. 2023a]. These results suggest that ortholattices can be used to simplify large formulas using polynomial-time algorithms, while providing soundness, as well as a clear mental model of the degree of its incompleteness. In the domain of proof assistants and automated provers, it is a necessary property of proofs that

Authors' address: Simon Guilloud; Viktor Kunčak, EPFL, IC, INR 318, Station 14, CH-1015 Lausanne, Switzerland, firstname.lastname@epfl.ch.

. XXXX-XXXX/2023/10-ART
https://doi.org/

they can be efficiently and reliably checked. However, tools and users building said proof may enjoy the additional flexibility offered by a partial yet predictable and efficient automation over either a heuristic approach or strictly syntactic approach. This was the motivation behind the use of *OL*-reasonning in the logical Kernel of the LISA proof assistant [Guilloud et al. 2023b].

In [Guilloud et al. 2023a], an efficient normal form algorithm for ortholattices is used in a software verifier to improve the cache hit ratio of verification conditions. It also serves to simplify and shorten formulas before sending them to a general but potentially costly SMT solver. This approach generally improved the performance of the verifier.

We believe there are other applications in the domain of programming languages, and possibly more broadly in computer science, for which long runtime and non-determinism leading to difficulties in reproducing results are major drawbacks, and hence where, predictability and efficiency at the expense of classical completeness can be a desirable tradeoff. For example, type checking procedures for programming language with variants of liquid and refinement types [Freeman and Pfenning 1991; Vazou et al. 2014] or union/intersection types may benefit from a similar approach. Logic programming, in particular extensions of Datalog with a negation operator [Clark 1978; Kunen 1987] is also a suitable candidate. This last point is supported by our results in Section 6.

As an example, an ortholattice algorithm in [Guilloud et al. 2023a] may reduce $x \wedge z \wedge \neg(u \wedge \neg x)$ to the normal form $x \wedge z$. This normal form is based on the laws that hold in *all* ortholattices (equivalently, in the free ortholattice). This makes the technique widely applicable, but it also makes it weak in terms of classical and domain-specific tautologies it can prove. This paper explores making orthologic-based reasoning more precise and more usable, asking the following questions:

(1) Can we formally extend orthologic with non-logical axioms?
(2) Can we find a complete and efficient algorithm for it?
(3) What are classes of formulas in classical logic for which orthologic proofs always exist?
(4) Can orthologic be used effectively beyond propositional logic, for classes of predicate logic?

Our approach to these questions is to use a sound and complete proof system for orthologic that we extend to support arbitrary non-logical axioms. Algebraically, our proof system is complete for establishing inequalities in the class of ortholattices specified by a given *presentation*, a set of ortholattice inequalities. From the practical point of view, using axioms to represent part of the input formula gives a sound and strictly stronger approximation of classical logic than using ortholattices without axioms.

1.1 Ortholattices

Ortholattices are a weaker structure than Boolean algebras, where distributivity does not necessarily hold. Table 1 shows their axiomatization. All Boolean algebras are ortholattices; they are precisely those ortholattices that are distributive. Figure 1 shows two characteristic finite non-distributive ortholattices; keeping these structures in mind may provide intuition for reasoning inside the class of all ortholattices. *Orthologic* is the logical system that corresponds to ortholattices, analogously to how classical logic corresponds to Boolean algebras (and intuitionistic logic to Heyting algebras).

1.2 Example of Using Axioms

Using axioms in orthologic inference allows us to prove more classical implications than by encoding the entire problem into one formula, increasing the power of reasoning. To understand why, note that proving validity of an implication $L \longrightarrow R$ in all ortholattices can be phrased as proving $L \leq R$ in all ortholattices, for all values to which L and R can evaluate in those ortholattices. Such an inequality needs to hold in the ortholattice O_6 in Figure 1 when L evaluates to, for example, b. On the other hand, using axioms, we can encode an implication problem as follows: prove that, in

V1:	$x \vee y = y \vee x$	V1':	$x \wedge y = y \wedge x$
V2:	$x \vee (y \vee z) = (x \vee y) \vee z$	V2':	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$
V3:	$x \vee x = x$	V3':	$x \wedge x = x$
V4:	$x \vee 1 = 1$	V4':	$x \wedge 0 = 0$
V5:	$x \vee 0 = x$	V5':	$x \wedge 1 = x$
V6:	$\neg\neg x = x$		
V7:	$x \vee \neg x = 1$	V7':	$x \wedge \neg x = 0$
V8:	$\neg(x \vee y) = \neg x \wedge \neg y$	V8':	$\neg(x \wedge y) = \neg x \vee \neg y$
V9:	$x \vee (x \wedge y) = x$	V9':	$x \wedge (x \vee y) = x$

Table 1. Laws of ortholattices, algebraic varieties with signature $(S, \wedge, \vee, 0, 1, \neg)$ from [Guilloud et al. 2023a]. These non-minimal laws illustrate the duality between $\wedge$ and $\vee$ and have explicit bottom and top element. See, e.g., [McCune 1998] for an equivalent smaller axiomatization.

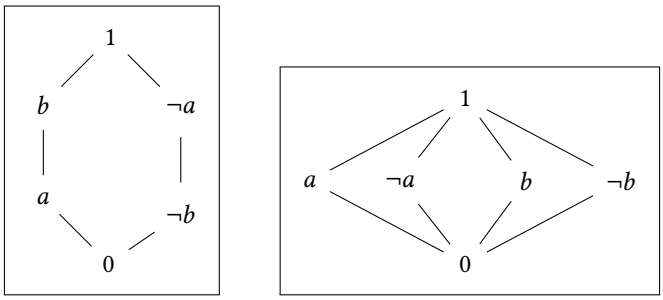

Fig. 1. Ortholattices O_6 and M_4. An ortholattice is distributive iff if it contains neither as a sub-ortholattice.

every ortholattice, if $L = 1$, then also $R = 1$. Because L is restricted to be 1, what remains to prove is a weaker statement, provable for more formulas. The conclusion remains sound with respect to the $\{0, 1\}$ lattice of classical logic, where $L = 1$ is the only non-trivial case to check for inequality.

For example, $x \wedge (\neg x \vee u) \leq u$ does not hold in O_6 of Figure 1 (take $x \mapsto b$, $u \mapsto a$ as a counterexample). On the other hand, in any ortholattice, if $x \wedge (\neg x \vee u) = 1$ then $u = 1$. Indeed, consider any ortholattice, and suppose $x \wedge (\neg x \vee u) = 1$. Recall that, in any bounded lattice with 1 as a top element, if $p \wedge q = 1$ then $p = 1$ and $q = 1$ because $1 \leq p \wedge q \leq p$. In our example, we conclude $x = 1$ and $(\neg x \vee u) = 1$. Now, substituting $x = 1$ and using $\neg 1 = 0$ gives us $u = 1$.

Such algebraic reasoning has a counterpart in proof-theoretic derivations. We present in Section 3 a system for derivation of formulas from axioms. Our system is complete for algebraic reasoning in ortholattices, allowing us to derive u if we allow $x \wedge (\neg x \vee u)$ as an axiom. Importantly, proof search in our system remains polynomial time, a result that we establish by showing a generalized notion of cut elimination.

The use of axioms (equivalently, ortholattice presentations) cannot emulate *all* instances of classical propositional logic axioms (indeed, proof search in our system remains in polynomial time instead of coNP). However, the above example hints that we can indeed use axioms to prove a larger set of classically valid problems than by using one monolithic formula in orthologic. Indeed, we show a number of practically important classes of problems for which reasoning in orthologic from axioms is complete, pointing to scenarios where orthologic may find useful applications.

1.3 Contributions

This paper shows how to use ortholattice reasoning with axioms as a sound polynomial-time deductive approach. We make the following contributions:

- We first show that a proof system for orthologic with axioms satisfies a form of the Cut Elimination property, where Cut rules are restricted to eliminate only axioms and can only appear near leaves in the proof. From this, we deduce a subformula property.
- We show that, in the presence of axioms, there is an orthologic backward proof search procedure with worst case asymptotic time $O(n^2(|A|+1))$, where n is the size of the problem and $|A|$ the total number of axioms. Without axioms, the algorithm is quadratic.
- We study how orthologic can help solve some classes of classical problems and show that special case of satisfiability instances (2CNF, Horn clauses, renamed Horn, q-Horn, extended Horn) admit orthologic proofs, i.e. these problems are satisfiable in orthologic if and only if they are satisfiable in classical logic.
- We show that orthologic decision problems can be flattened, similarly to the Tseitin transform [Tseitin 1983], and we use this to give an upper bound on the proving power of orthologic in terms of the width of classical resolution proofs.
- We show how orthologic reasoning can be extended to fragments of *predicate* logic. We show that such quantified orthologic agrees with classical logic on the semantic of Datalog, and hence that Datalog programs admit *OL* proofs, making *OL* another possible generalization of Datalog to logic programming with negation and disjunction.

2 PRELIMINARIES

We briefly present key concepts and notation which will be used in the present article. Ortholattices are the algebraic variety whose equational laws are presented in Table 1. As in any lattice, we can define an order relation $\leq$ by:

$$a \leq b \iff (a = (a \wedge b))$$

which is also equivalent to $(b = (a \vee b))$. This yields a partially ordered set (poset) whose corresponding axiomatization is shown in Table 2 [Kalmbach 1983; Meinander 2010]. Note also that for any terms x, y we have $x = y$ if and only if both $x \leq y$ and $y \leq x$. The relation $=$ defined this way is a congruence relation for $\leq$, $\wedge$, $\vee$ and $\neg$ and thus becomes equality in the quotient structure. From the point of view of first-order logic with equality, each model of Table 1 axioms can be extended to a model of Table 2 by defining the inequality $a \leq b$ as the truth value of the atomic formula $a = (a \wedge b)$. Moreover, we have the converse: each model of Table 2 axioms induces a quotient structure with respect to the $x \leq y \;\&\; y \leq x$ relation; this structure is a model of Table 1 axioms.

By definition, Boolean algebras are precisely ortholattices that are distributive. Figure 1 shows ortholattices O_6 and M_4 that are *not* Boolean algebras. In fact, an ortholattice is a Boolean algebra if and only if it does not contain O_6 nor M_4 as a sub-ortholattice [Davey and Priestley 2002; Kalmbach 1983]. Despite being strictly weaker than laws of Boolean algebra, properties in Table 1 and Table 2 allow us to prove a number of desirable facts: all laws of bounded lattices (absorption, reordering and de-duplicating conjuncts and disjuncts), and laws relating complement to lattice operators (including the laws needed to transform formulas to negation-normal form). Laws of Boolean algebra that do not necessarily hold include distributivity, modularity, and properties such as $(\neg x \vee y) = 1$ implying $x \leq y$.

Similarly to how Boolean Algebra is the algebraic structure corresponding to classical logic, we find it natural that ortholattices form a class of structures that corresponds to a logic, *orthologic*, for which we study proof-theoretic and algorithmic properties in the following sections. We denote classical logic by *CL* and orthologic by *OL*.

P1:	$x \leq x$		
P2:	$x \leq y \ \& \ y \leq z \implies x \leq z$		
P3:	$0 \leq x$	P3':	$x \leq 1$
P4:	$x \wedge y \leq x$	P4':	$x \leq x \vee y$
P5:	$x \wedge y \leq y$	P5':	$y \leq x \vee y$
P6:	$x \leq y \ \& \ x \leq z \implies x \leq y \wedge z$	P6':	$x \leq z \ \& \ y \leq z \implies x \vee y \leq z$
P7:	$x \leq \neg\neg x$	P7':	$\neg\neg x \leq x$
P8:	$x \leq y \implies \neg y \leq \neg x$		
P9:	$x \wedge \neg x \leq 0$	P9':	$1 \leq x \vee \neg x$

Table 2. Axiomatization of ortholattices in the signature $(S, \leq, \wedge, \vee, 0, 1, \neg)$ as partially ordered sets. We use & for conjunction between atomic formulas of axioms, to differentiate it from the term-level lattice operation $\wedge$. This axiomatization corresponds to one in Table 1 defining $x \leq y$ to be $x \wedge y = x$ in one direction and $x = y$ to be $(x \leq y)$ & $(y \leq x)$ in the other.

Definition 2.1 (Terms). $\mathcal{T}_{OL}$ denotes the term algebra over the signature of ortholattices over a fixed countably infinite set of variables, that is, the set of all terms which can be built from variables and $(\wedge, \vee, 0, 1, \neg)$. We typically represent terms with lowercase Greek letters and variables with x, y, z, possibly with indices. Terms are constructed inductively as trees. Leaves are labeled with 0, 1, or variables. Nodes are labeled with logical symbols. Since $\vee$ and $\wedge$ are commutative, the children of a node form a set (non-ordered). $\mathcal{T}_{CL} = \mathcal{T}_{OL}$, and is also the set of formulas for both classical logic and orthologic.

Note that the laws of both *OL* and *CL* imply that 0 can always be represented by $x \wedge \neg x$ and 1 as $x \vee \neg x$. To simplify proofs, we thus may omit the cases corresponding to 0 and 1.

The *word problem* for an algebra consists in, given two terms in the language of the algebra, deciding if they are always equal by the laws of the algebra or not. For ortholattices, we can relax the definition to allow inequality queries, as they can be expressed as equivalent equalities. A (finite) *presentation* for an algebra is a (finite) set of equalities (which we relax to inequalities in ortholattices) $\{\phi_1 \leq \psi_1, ..., \phi_n \leq \psi_n\}$. The *uniform word problem* for presented ortholattices is the task consisting in, given a presentation A and two terms ϕ and ψ, deciding if $\phi \leq \psi$ follows from the laws of ortholattices and the axioms in A.

In the terminology of classical first-order logic, the laws of ortholattice in Table 2 are a finite set of *universally quantified* formulas, $\mathcal{T}$, and they define a first-order theory. The presentation (axioms) A is a set of *quantifier-free* formulas, whereas $\phi \leq \psi$ is also a quantifier-free formula, with variables possibly in common with those of A. We can then view the uniform word problem as a special case of the question of semantic consequence in first-order logic: $\mathcal{T} \cup A \models \phi \leq \psi$.

3 COMPLETE PROOF SYSTEM AND CUT ELIMINATION

We formulate our proof system for orthologic as a sequent calculus. We represent sequents by decorating the formulas with superscript ^{L} or ^{R}, depending on whether they appear on the left or right side. For example, ϕ^L, ϕ^R stands for $\phi \vdash \psi$ in more conventional notation.

Definition 3.1. If ϕ is a formula, we call ϕ^L and ϕ^R annotated formulas. A *sequent* is a set of at most two annotated formulas. We use Γ and Δ to represent sets that are either empty or contain exactly one annotated formula ($|\Gamma| \leq 1, |\Delta| \leq 1$).

Figure 2 shows our sequent calculus for orthologic, parameterised by a set of sequents A called axioms. In the present article, *orthologic*, or *OL* denotes this specific proof system. Without the

support for arbitrary axioms, an equivalent system, with a different presentation, was introduced in [Schulte Mönting 1981].

Note that the axioms we consider in this section are not universally quantified: they refer to arbitrary but fixed propositions.

$$\frac{}{\phi^L, \phi^R}\ \text{Hyp}$$

$$\frac{\Gamma, \psi^R \qquad \psi^L, \Delta}{\Gamma, \Delta}\ \text{Cut}$$

$$\frac{\Gamma}{\Gamma, \Delta}\ \text{Weaken}$$

$$\frac{\Gamma, \phi^L}{\Gamma, (\phi \wedge \psi)^L}\ \text{LeftAnd} \qquad \frac{\Gamma, \phi^R \qquad \Gamma, \psi^R}{\Gamma, (\phi \wedge \psi)^R}\ \text{RightAnd}$$

$$\frac{\Gamma, \phi^L \qquad \Gamma, \psi^L}{\Gamma, (\phi \vee \psi)^L}\ \text{LeftOr} \qquad \frac{\Gamma, \phi^R}{\Gamma, (\phi \vee \psi)^R}\ \text{RightOr}$$

$$\frac{\Gamma, \phi^R}{\Gamma, (\neg\phi)^L}\ \text{LeftNot} \qquad \frac{\Gamma, \phi^L}{\Gamma, (\neg\phi)^R}\ \text{RightNot}$$

(a) Deduction rules of Orthologic. Each holds for arbitrary Γ, Δ, ϕ, ψ

$$\frac{}{\Gamma, \Delta}\ \text{Ax}(\Gamma,\Delta) \qquad \text{If } \Gamma \cup \Delta \in A$$

(b) Rule for additional non-logical axioms.

Fig. 2. Sequent-calculus rules for Orthologic derivations from a set of axioms A. Sets Γ and Δ are either empty or a single formula on any side.

One can think of this proof system as Gentzen's sequent calculus for classical logic [Gentzen 1935] restricted to ensure the following syntactic invariant:

At any given place in a proof, a sequent never has more than two formulas on both sides combined.

This restriction on the proof system bears resemblance to the syntactic restriction of intuitionistic sequent calculus, where a sequent can never have more than one formula on the right side, a restriction lifted in the classical logic sequent calculus system. Compared to intuitionistic logic, orthologic allows us to prove $\vdash \phi, \neg\phi$, represented as $\phi^R, (\neg\phi)^R$, using the following steps.

$$\frac{\dfrac{}{\phi^L, \phi^R}\ \text{Hyp}}{\phi^R, (\neg\phi)^R}\ \text{RightNot}$$

On the other hand, orthologic restricts the number of assumptions on the left side of the sequent. This strong restriction will be rewarded by the existence of a polynomial-time proof search procedure.

Definition 3.2. We say that a deduction rule is *admissible* if any sequent that can be proven with the rule can be proven without.

3.1 Ortholattice Semantics for Orthologic

We interpret a sequent ϕ^L, ψ^R as $\phi \leq \psi$ in an ortholattice. More generally, we have the following definition.

Definition 3.3. The *interpretation* of a sequent is given by the following table, where $\emptyset$ denotes the empty sequent:

sequent S	ortholattice inequality $\overline{S}$ (up to equivalence)
ϕ^L, ψ^R	$\phi \leq \psi$
ϕ^L, ψ^L	$\phi \leq \neg\psi$
ϕ^R, ψ^R	$\neg\phi \leq \psi$
ϕ^L	$\phi \leq 0$
ϕ^R	$1 \leq \phi$
$\emptyset$	$1 \leq 0$

The intended reading of the table above is a mapping of sequents (which are sets) to ortholattice atomic formulas *up to logical equivalence*. The set $\{\phi^L, \psi^L\}$ can be mapped to either $\phi \leq \neg\psi$ or to $\neg\psi \leq \phi$, but these are equivalent in an ortholattice (analogously for mapping $\{\phi^R, \psi^R\}$).

The interpretation of a deduction rule in Figure 2 with k premises $P_1, \ldots, P_k$ and a conclusion C, is the universally quantified first-order logic formula $\overline{P}_1 \;\&\; \ldots \;\&\; \overline{P}_n \implies \overline{C}$.

Given an axiom set A we talk about ortholattice with presentation A by taking the interpretation of all axioms in A. Our proof system can prove every axiom of ortholattices (Table 2) and, conversely.

Lemma 3.4 (Soundness and Completeness of *OL* Deduction Rules)**.** Let A be an arbitrary (possibly infinite) set of axioms. A sequent has a derivation from A using the rules of orthologic (Figure 2) iff its interpretation is in the first-order theory of ortholattices (Table 2) with presentation A.

Proof. Sketch. For every Table 2 law of the form $\overline{P}_1 \wedge \ldots \wedge \overline{P}_n \rightarrow \overline{C}$, a matching deduction rule

$$\frac{P_1 \qquad \ldots \qquad P_n}{C}$$

is easily seen to be admissible. Conversely, for every deduction rule of Figure 2, the corresponding law is a consequence (in first order logic) of the axioms of Table 2. □

For any axiom set A, this makes our system (with the Cut rule) sound and complete for the class of all ortholattices satisfying axioms in A. Note that this interpretation is compatible with the interpretation of sequents in classical logic. We can use the soundness and completeness to obtain simple model-theoretic proofs for orthologic. We can show, for example, that substitution of equivalent formulas is admissible.

Lemma 3.5. Let Γ and Δ denote sets with at most one labelled formula each. Let $\Gamma[\chi := \phi]$ denote the substitution inside Γ of χ (a placeholder formula symbol) by ψ. The following rule for substitution of equivalent formulas is admissible in orthologic:

$$\frac{\Gamma[\chi := \phi], \Delta[\chi := \phi] \qquad \phi^L, \psi^R \qquad \psi^L, \phi^R}{\Gamma[\chi := \psi], \Delta[\chi := \psi]}\ \text{Subst}$$

Said otherwise, if both ϕ^L, ψ^R and ψ^L, ϕ^R can be shown then arbitrary occurrences of ϕ in a proven sequent can be replaced by ψ.

Proof. The argument is semantic. Fix any ortholattice O satisfying the axioms. Since both ϕ^L, ψ^R and ψ^L, ϕ^R are provable, it follows that $\phi = \psi$ in O. Hence, $\Gamma[\chi := \phi] = \Gamma[\chi := \psi]$ and $\Delta[\chi := \phi] = \Delta[\chi := \psi]$. By completeness, the sequent $\Gamma[\chi := \psi], \Delta[\chi := \psi]$ is provable. □

3.2 Partial Cut Elimination

As a sequent calculus, our system has structural rules, introduction rules for each logical symbol and a Cut rule, but no elimination rule. Consequently, by inspecting all rules, we conclude that *Cut is the only rule whose premises contain formulas that are not subformulas of the concluding sequent.*

Definition 3.6. In an instance of a Cut rule in Figure 2, we call the formula ψ the *cut formula*. In an instance of a left or right rule, the newly constructed formula is called the *principal formula*. In the Weaken rule, if Δ contains a formula, it is also called the principal formula.

[Schulte Mönting 1981] showed that orthologic, without arbitrary non-logical axioms, admits cut elimination. The crucial challenge is that, in contrast to classical or intuitionistic calculus, we cannot simply add additional assumptions to the left-hand side of sequents in orthologic derivations. The reason is the restriction on the number of formulas in sequents. The following example illustrates this phenomenon.

Example 3.1. We saw in Section 1.2 that $(x \wedge (\neg x \vee u)) \leq u$ is not always valid. In particular, the sequent $(x \wedge (\neg x \vee u))^L, u^R$ is not provable in orthologic without axioms. However, with axiom $(x \wedge (\neg x \vee u))^R$, the sequent u^R is provable as follows. First, $(\neg x \vee u)^R$ is provable:

$$\dfrac{\dfrac{}{(x \wedge (\neg x \vee u))^R}\,\text{Ax} \qquad \dfrac{\dfrac{}{(\neg x \vee u)^L, (\neg x \vee u)^R}\,\text{Hyp}}{(x \wedge (\neg x \vee u))^L, (\neg x \vee u)^R}\,\text{L.And}}{(\neg x \vee u)^R}\,\text{Cut}$$

Then,

$$\dfrac{\dfrac{\dfrac{}{(x \wedge (\neg x \vee u))^R}\,\text{Ax} \qquad \dfrac{\dfrac{}{x^L, x^R}\,\text{Hyp}}{(x \wedge (\neg x \vee u))^L, x^R}\,\text{L.And}}{\dfrac{x^R}{(\neg x)^L}\,\text{L.Not}}\,\text{Cut}}{\dfrac{(\neg x \vee u)^R \qquad \dfrac{(\neg x)^L, u^R}{(\neg x \vee u)^L, u^R}\,\text{Weaken} \qquad \dfrac{}{u^L, u^R}\,\text{Hyp}}{u^R}\,\text{L.Or}\,\text{Cut}}$$

In a *classical* sequent calculus system, the above derivation using the axiom $x \wedge (\neg x \vee u)$ could be transformed into a new derivation where each sequent has an additional assumption $x \wedge (\neg x \vee u)$ and where the use of axiom rule is replaced with the use of the Hyp rule. This transformation does not apply to *OL*: it would create sequents with more than two formulas, which, by the definition in Figure 2 cannot appear in *OL* proofs.

For this reason, the ability to add non-logical axioms is crucial in orthologic. We aim to extend the cut elimination property to proofs containing arbitrary axioms. This will allow us to devise an efficient decision procedure for orthologic with axioms, and, by extension, the word problem for finitely presented ortholattices. Moreover, the proof we present is constructive, in the sense that it shows an algorithmic way to eliminate instances of the Cut rule from a proof. Furthermore, we need not worry about the size of the transformed proof, because our Cut elimination property will enable us to derive a subformula property and a bound on the size of the proof of any given formula.

However, the system does *not* allow for complete cut elimination in the presence of axioms, as the following short example shows.

Example 3.2. Let x_1, x_2, y be distrinct variables and let the sequent $(x_1 \vee x_2)^L, y^R$ be the only axiom. The sequent x_1^L, y^R is then provable:

$$\dfrac{\dfrac{\dfrac{}{x_1^L, x_1^R}\,\text{Hyp}}{x_1^L, (x_1 \vee x_2)^R}\,\text{RightOr} \qquad \dfrac{}{(x_1 \vee x_2)^L, y^R}\,\text{Ax}}{x_1^L, y^R}$$

but it cannot be proven without using the Cut rule. To see why, note that Hyp, LeftAnd, RightAnd, LeftOr, RightOr, LeftNot, RightNot do not yield sequents whose syntactic form can be x_1^L, y^R. Furthermore, Ax does not produce the desired sequent as it is not the axiom. Finally, Weaken does not help because its premise would not be provable: neither x_1^L nor y^R are individually provable from the axiom, as can be seen by a semantic argument: it could be, for example, that both x_1 and y have value 0 or both have value 1 in an ortholattice that satisfies the axiom.

Thus, cut rule is in general necessary when reasoning from axioms, and we need to formulate a suitable generalization of the concept of cut elimination. For this purpose, we define the *rank* of an instance of the cut rule.

Definition 3.7. An instance of the Cut rule has *rank 1* if either of its premises is an axiom. It has *rank 2* if either of its premises is the conclusion of a rank 1 Cut rule.

The following theorem is our main result. It implies that the Cut rule can be eliminated or restricted to only cut with respect to axioms. Part (1) has immediate consequences for the subformula property. Part (2) gives further insight into normalized proofs, further restricts our proof procedure in the next section, and it helps with the inductive argument in the proof of the theorem.

Theorem 3.8 (Cut Elimination for Orthologic)**.** If a sequent is provable in the system of Figure 2 with axioms

$$\dfrac{}{a_i^\circ, b_i^\square}\,\text{Ax}(a_i^\circ, b_i^\square)$$

for all $(a_i^\circ, b_i^\square) \in A$ (a_i and b_i are formulas, $^\circ$ and $^\square$ are side annotation), then there is a proof of that sequent from the same axioms such that:

(1) All instances of the Cut rule use only formulas among $a_1, \ldots a_n, b_1, \ldots, b_n$ as cut formulas.
(2) All instances of the Cut rule are rank 1 or 2.

Proof. If a proof does not satisfy the properties (1) and (2) of the theorem statement, then there is a Cut rule for which the condition in (1) or (2) does not hold. Consider a derivation using such a Cut rule as the last step; when Cut is not bottommost, the properties follow trivially by induction. Let the proof be of the form:

$$\dfrac{\dfrac{\mathcal{A}}{\Gamma, \psi^R} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Gamma, \Delta}\,\text{Cut}$$

where $\mathcal{A}$ and $\mathcal{B}$ are the proof trees whose conclusions are respectively the left and right premises and ψ is the Cut formula. We show that there exists a proof of Γ, Δ that satisfies the two properties of the theorem.

We proceed by induction on the length of the proof. Hence, we can assume by induction that $\mathcal{A}$ and $\mathcal{B}$ satisfy properties 1 and 2. By induction hypothesis, it suffices to show how to transform the proof of Γ, Δ into one where Cut is used only in subproofs strictly smaller than the proof we

started with. We do case analysis on $\mathcal{A}$ and $\mathcal{B}$, showing for each case how to transform the proof in this way. $\hookrightarrow$ denotes this transformation.

Case 1. Suppose $\mathcal{A}$ ends (and hence starts) with a Hypothesis rule. Then, $\Gamma = \psi^L$ and ψ^L, Δ can be reached using only $\mathcal{B}$. The case where $\mathcal{B}$ is a Hypothesis rule is symmetric.

Case 2. ($\mathcal{A}$ ends with Weaken)
Case 2.a Suppose $\mathcal{A}$ ends with a Weaken rule and ψ^R is not the *principal formula* (see Definition 3.6).

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\psi^R}}{\Gamma, \psi^R}\,\text{Weaken} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Gamma, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\mathcal{A}'}{\psi^R} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Delta}\,\text{Cut}}{\Gamma, \Delta}\,\text{Weaken}$$

In the transformed proof, $\mathcal{A}'$ is part of $\mathcal{A}$, so Cut applies to a smaller subproof and can be transformed to satisfy the properties (1) and (2) by inductive hypothesis.
Case 2.b Suppose $\mathcal{A}$ ends with a Weaken rule and ψ^R is the principal formula.

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\Gamma}}{\Gamma, \psi^R}\,\text{Weaken} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Gamma, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \dfrac{\dfrac{\mathcal{A}'}{\Gamma}}{\Gamma, \Delta}\,\text{Weaken}$$

Case 3. ($\mathcal{A}$ ends with a Left rule where ψ is not principal)
Case 3.a Suppose $\mathcal{A}$ ends with a LeftAnd rule where $\Gamma = (\alpha \wedge \beta)^L$

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^L, \psi^R}}{(\alpha \wedge \beta)^L, \psi^R}\,\text{LeftAnd} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{(\alpha \wedge \beta)^L, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^L, \psi^R} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\alpha^L, \Delta}\,\text{Cut}}{(\alpha \wedge \beta)^L, \Delta}\,\text{LeftAnd}$$

Case 3.b Suppose $\mathcal{A}$ ends with a LeftOr rule where $\phi = \alpha \vee \beta$

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^L, \psi^R} \qquad \dfrac{\mathcal{A}''}{\beta^L, \psi^R}}{(\alpha \vee \beta)^L, \psi^R}\,\text{LeftOr} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{(\alpha \vee \beta)^L, \Delta}\,\text{Cut}$$

$$\hookrightarrow$$

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^L, \psi^R} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\alpha^L, \Delta}\,\text{Cut} \qquad \dfrac{\dfrac{\mathcal{A}''}{\beta^L, \psi^R} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\beta^L, \Delta}\,\text{Cut}}{(\alpha \vee \beta)^L, \Delta}\,\text{LeftOr}$$

Case 3.c Suppose $\mathcal{A}$ ends with a LeftNot rule, i.e. $\Gamma = \neg\alpha$

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^R, \psi^R}}{(\neg\alpha)^L, \psi^R}\,\text{LeftNot} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\alpha^L, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \text{in} \quad \dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^R, \psi^R} \qquad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\alpha^R, \Delta}\,\text{Cut}}{(\neg\alpha)^L, \Delta}\,\text{LeftNot}$$

The cases where $\mathcal{B}$ ends with a Right rule are symmetric.

Case 4. If $\mathcal{A}$ ends with Right rule where ψ^R is not the principal formula, the transformation is symmetric to the Left rule case. Similarly if $\mathcal{B}$ ends with a Left rule where ψ^L is not the principal formula.

Case 5. ($\mathcal{A}$ ends with Right rule and $\mathcal{B}$ with a Left rule, ψ is principal in both)
Case 5.a Suppose $\mathcal{A}$ ends with a RightOr rule, i.e. $\psi = (\alpha \vee \beta)$. In this case, $\mathcal{B}$ has to end with a LeftOr rule, as it is the only Left rule that can produce $(\alpha \vee \beta)^L$

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\Gamma, \alpha^R}}{\Gamma, (\alpha \vee \beta)^R}\,\text{RightOr} \quad \dfrac{\dfrac{\mathcal{B}'}{\alpha^L, \Delta} \quad \dfrac{\mathcal{B}''}{\beta^L, \Delta}}{(\alpha \vee \beta)^L, \Delta}\,\text{LeftOr}}{\Gamma, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \dfrac{\dfrac{\mathcal{A}'}{\Gamma, \alpha^R} \quad \dfrac{\mathcal{B}'}{\alpha^L, \Delta}}{\Gamma, \Delta}\,\text{Cut}$$

Case 5.b If $\mathcal{A}$ ends with a RightAnd rule and $\mathcal{B}$ with a LeftAnd, the transformation is symmetric to case 5.a
Case 5.c If $\mathcal{A}$ ends with a RightNot rule and $\mathcal{B}$ a LeftNot rule, and $\phi = \neg\alpha$ is principal in both:

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\Gamma, \alpha^L}}{\Gamma, (\neg\alpha)^R}\,\text{RightNot} \quad \dfrac{\dfrac{\mathcal{B}'}{\alpha^R, \Delta}}{(\neg\alpha)^L, \Delta}\,\text{RightNot}}{\Gamma, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \dfrac{\dfrac{\mathcal{A}'}{\Gamma, \alpha^L} \quad \dfrac{\mathcal{B}'}{\alpha^R, \Delta}}{\Gamma, \Delta}\,\text{Cut}$$

Case 6. Suppose either $\mathcal{A}$ or $\mathcal{B}$ end with an Axiom rule. Then, properties 1 and 2 are immediate.

Case 7. Suppose $\mathcal{A}$ ends with a Cut rule, which by induction we can assume is rank 1 or 2.
Case 7.a If $\mathcal{A}$ is rank 1, the following transformation works, since if $\mathcal{A}''$ is an axiom both properties immediately hold for both Cuts, and if $\mathcal{A}'$ is an axiom, both properties hold for the last cut and the other cut has smaller size.

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\Gamma, \phi^R} \quad \dfrac{\mathcal{A}''}{\phi^L, \psi^R}}{\Gamma, \psi^R}\,\text{Cut} \quad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Gamma, \Delta}\,\text{Cut} \quad \hookrightarrow \quad \dfrac{\dfrac{\mathcal{A}'}{\Gamma, \phi^R} \quad \dfrac{\dfrac{\mathcal{A}''}{\phi^L, \psi^R} \quad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\phi^L, \Delta}\,\text{Cut}}{\Gamma, \Delta}\,\text{Cut}$$

Case 7.b Suppose $\mathcal{A}$ is rank 2 and it is $\mathcal{A}'$ that ends with a rank 1 Cut:

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{A}'_1}{\Gamma, \chi^R} \quad \dfrac{\mathcal{A}'_2}{\chi^L, \phi^R}}{\Gamma, \phi^R}\,\text{Cut}^1 \quad \dfrac{\mathcal{A}''}{\phi^L, \psi^R}}{\Gamma, \psi^R}\,\text{Cut}^2 \quad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Gamma, \Delta}\,\text{Cut}^3$$

Either $\mathcal{A}'_1$ or $\mathcal{A}'_2$ is an axiom. If $\mathcal{A}'_2$ is, the same transformation as above works because the last cut again has the axiom ϕ as a cut formula. If $\mathcal{A}'_1$ is an Axiom, we transform to the following:

$$\hookrightarrow \quad \dfrac{\dfrac{\mathcal{A}'_1}{\Gamma, \chi^R} \quad \dfrac{\dfrac{\dfrac{\mathcal{A}'_2}{\chi^L, \phi^R} \quad \dfrac{\mathcal{A}''}{\phi^L, \psi^R}}{\chi^L, \psi^R}\,\text{Cut}^1 \quad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\chi^R, \Delta}\,\text{Cut}^2}{\Gamma, \Delta}\,\text{Cut}^3$$

Then, the new Cut^3 is of rank 1 and its cut formula is part of an axiom. Cut^2 is of strictly smaller size than the proof we started from, so by induction its conclusion can be obtained with a proof satisfying properties 1 and 2.

Case 7.c Suppose now that $\mathcal{A}''$ is a rank 1 Cut. We transform the proof as follows:

$$\dfrac{\dfrac{\mathcal{A}'}{\Gamma, \phi^R} \quad \dfrac{\dfrac{\mathcal{A}_1''}{\phi^L, \chi^R} \quad \dfrac{\mathcal{A}_2''}{\chi^L, \psi^R}}{\phi^L, \psi^R}\,\text{Cut}}{\Gamma, \psi^R} \quad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\Gamma, \Delta}\,\text{Cut}$$

$$\hookrightarrow$$

$$\dfrac{\dfrac{\mathcal{A}'}{\Gamma, \phi^R} \quad \dfrac{\mathcal{A}_1''}{\phi^L, \chi^R}}{\Gamma, \chi^R}\,\text{Cut}^1 \quad \dfrac{\dfrac{\mathcal{A}_2''}{\chi^L, \psi^R} \quad \dfrac{\mathcal{B}}{\psi^L, \Delta}}{\chi^L, \Delta}\,\text{Cut}^2}{\Gamma, \Delta}\,\text{Cut}^3$$

Either $\mathcal{A}_1''$ or $\mathcal{A}_2''$ is an axiom. In both cases, Cut^3 is of rank 2 and its cut formula is part of an axiom. The proofs ending with Cut^1 and Cut^2 are of strictly smaller size than the original proof, so by induction they can be made to satisfy the desired properties 1 and 2 (one of them is already rank 1).

The above cases cover all possibilities for how the premises of the topmost cut in the proof are constructed, concluding the proof. □

Since sequent calculus for orthologic has no elimination rule, traversing a sequent S backwards in its proof obtained by Theorem 3.8 we obtain the following subformula property.

Corollary 3.9 (Subformula Property for Orthologic)**.** If a sequent S has a proof in the proof system of Figure 2 with axioms, then it has such a proof where each formula in each sequent ocurring in the proof is a subformula of S or a subformula of an axiom.

Proof. Let S be a sequent that has a proof. By Theorem 3.8, consider a proof of S with properties (1) and (2). We view the proof as a directed tree with root S, whose nodes are the sequents occurring in the proof. We show that each formula in the node is a subformula of S or of an axiom, by induction on the distance of the tree node to the root S. The property trivially holds for the root S. Suppose now that the property holds for a node S' in the tree; we show that it also holds for its children. We consider all applicable rules in Figure 2. Hyp rule has no children, so there is nothing to show. Consider a Cut rule and one of its children, Γ, ψ^R. Then Γ is a subformula because it is a part of S', whereas ψ is a part of the axiom by property (1). The case for the other child ψ, Δ of the Cut rule is analogous. The case of the Left and Right rules are easy: we observe that their premises are sequents whose formulas are subformulas of the conclusion S', so they are a subformula of S' or of axioms, by inductive hypothesis and the transitivity of the "subformula" relation. Finally, if S' is obtained using the axiom rule, then its formulas are subformulas of the axioms by definition. □

4 CUBIC-TIME PROOF SEARCH

In this section we show that the subformula property (Corollary 3.9) not only implies a quadratic bound on the size of proofs, but also allows us to define an $O(n^3)$ proof search procedure. Such deterministic polynomial-time search is in contrast to co-NP completeness of validity in classical propositional logic [Cook 1971] and to PSPACE completeness of validity in intuitionistic logic [Statman 1979; Urzyczyn 1997]. We assume that the set A of axioms is finite throughout this section.

Algorithm 1: Cubic-Time Proof Search for *OL* with Axioms

```
type Formula*                                  // an annotated formula or a special None value
A: Set[(Formula*, Formula*)] ← Input                                          // set of axioms
AxFormulas: Set[Formula] ← ⋃{{a, b} | {a□, b°} ∈ A}        // formulas from axiom sequents
proven: Set[(Formula*, Formula*)] ← Set.empty                    // formulas already proven
visited: Set[(Formula*, Formula*)] ← Set.empty // formulas whose proof is being attempted
def prove(Γ: Formula*, Δ: Formula*)
    if proven.contains((Γ, Δ)) then return True
    else if visited.contains((Γ, Δ)) then return False
    else
        visited.add((Γ, Δ))                                // necessary to avoid infinite loops
        success ← {
            ((Γ==φ^L && Δ == φ^R) || (Γ==φ^R && Δ == φ^L)) ||                                  // Hyp
            ((Γ, Δ) ∈ A) ||                                                                // Ax(Γ, Δ)
            (Γ!=None && Δ!=None && (prove(Γ, None) || prove(None, Δ))) ||                    // Weaken
            (Γ==(¬φ)^L && prove(φ^R, Δ)) ||                                                  // LeftNot
            (Γ==(φ ∧ ψ)^L && (prove(φ^L, Δ) || prove(ψ^L, Δ)) ||                             // LeftAnd
            (Γ==(φ ∨ ψ)^L && prove(φ^L, Δ) && prove(ψ^L, Δ) ||                               // LeftOr
            ... ||                                          // analogous Right cases for Γ
            ... ||                                  // analogous Left and Right cases for Δ
            AxFormulas.exists((x:Formula) →
               prove(Γ, x^R) && prove(x^L, Δ) || prove(Δ, x^R) && prove(x^L, Γ))      // Cut with axiom
        }
        if success then
            proven.add((Γ, Δ))
        return success
```

Our approach is to eliminate from the proof system in Figure 2 deduction steps which do not satisfy the conditions of Theorem 3.8. We assume that no axiom is a trivial one $a \leq a$, as that one does not help prove anything.

For a formula, sequent or set of formulas or sequents o, let $\|o\|$ denote the number of subformulas in o. This is asymptotically equal to the number of symbols needed to represent o. In particular for the set of axioms, $\|A\|$ is the number of subformulas in all axioms of A whereas $|A|$ is the number of axioms.

Lemma 4.1 (Bound on Intermediate Sequents)**.** There is a function that when given S computes a set relevant(S) containing at most $4(\|S\| + \|A\|)^2$ intermediate sequents such that if S is provable then it is provable with a proof whose all sequents appear in relevant(S).

Proof. By Corollary 3.9, there is a proof where each formula is a subformula of S or of A. Let relevant(S) be the set of all sequents built from these formulas. There are at most $2(\|S\| + \|A\|)$ of labelled subformulas. A sequent has at most two labelled subformulas, and the number of sequents is bounded by the number of ordered pairs of labelled formulas, which is $4(\|S\| + \|A\|)^2$. □

Lemma 4.2 (Bound on Branching)**.** Let A be a set of axioms, and let S be a sequent. Then, if S has a proof in the system of Figure 2 then there are at most $7 + 4|A|$ valid instances of rules whose conclusion is S. We call the sequents above S in these steps the *possible parents* of S.

Proof. Let S be a sequent of the form Γ, Δ. We state in parentheses the maximal number of valid instances for each of the rules in Figure 2:

- (1) S can be deduced by an application of the hypothesis rule or from the axiom rule (not both as we assume the axioms are non-trivial)
- (2) Γ, Δ can follow using weakening, from either Γ or Δ
- (2) Γ, Δ can follow using a Left or a Right rule on Γ in at most two ways, depending on the structure of Γ:
  - If $\Gamma = (\neg\phi)^L$ then S can be deduced from ϕ^R, Δ with LeftNot
  - If $\Gamma = (\phi \wedge \psi)^L$ then S can be deduced in two ways using LeftAnd: from ϕ^R, Δ and from ψ^R, Δ
  - If $\Gamma = (\phi \vee \psi)^L$ then S can be deduced in exactly one way using LeftOr: from both ϕ^R, Δ and ψ^R, Δ.
  - Right rules are symmetrical and either Left or Right rules apply to Γ, not both
- (2) Γ, Δ can follow using a Left or Right rule on Δ in also at most two ways, entirely analogously to the previous case.
- (4$|A|$) Γ, Δ can be deduced using the Cut rule in $4|A|$ different ways: the cut formula ϕ can be any formula among the left or right sides of axioms (thanks to Theorem 3.8 property (1)), for at most $2|A|$ different formulas, and for each the Cut instance can be either of

$$\frac{\Gamma, \phi^R \qquad \phi^L, \Delta}{\Gamma, \Delta}\ \text{Cut} \qquad \text{or} \qquad \frac{\Delta, \phi^R \qquad \phi^L, \Gamma}{\Gamma, \Delta}\ \text{Cut}$$

We thus obtain the desired bound $1 + 2 + 2 + 2 + 4|A| = 7 + 4|A|$. □

Theorem 4.3. There is a proof search procedure for *OL* running in time $O((1 + |A|)n^2)$, where $|A|$ is the number of given axioms and n the total size of the problem.

Proof. In Algorithm 1 we present pseudocode for backward search. Each line from 8 to 16 corresponds to trying a specific deduction rule.

For an input sequent S, the proof strategy consists in recursively computing the possible parents of S. By Lemma 4.1, proof of a sequent S need only involve at most $4(\|S\| + \|A\|)^2$ intermediate sequents, which is $O(n^2)$. Moreover, note that to compute the possible parents of a sequent, we need not observe the formulas entirely but only their roots, so computing one possible parent is constant time. Moreover, by Lemma 4.2, a sequent can only have at most $C + 4|A|$ possible parents, so reducing a sequent to all of its possible parents has complexity $O(1 + |A|)$. The final complexity of the proof search procedure is bounded by generating parents for all possible sequents we can encounter, which is $O((C + |A|)n^2) = O((1 + |A|)n^2)$. □

Note on the complexity of memoization. The complexity argument of the previous proof requires memoization to be efficient. However, a naive implementation of the "visited" and "proven" sets can increase the total runtime if the sets are implemented as lists or rely on structural equality of formulas, which has itself complexity linear in the size of the formulas. In practice, the following is most efficient. Assign to every annotated formula a unique integer (of size $O(\log(n))$. Then assign a field to every annotated formula, storing a hash table m with Integers as keys and Booleans as values. When a sequent of the form a, b is proven, set a.m(b.id) and b.m(a.id) to True. If the proof search fails, set them to False. With this representation, with this efficient representation, the checks of lines 7-10 reduce to one access in a hash table with a key of $O(\log(n))$ bits (which takes constant time in the usual word RAM model).

4.1 Merging Axioms for Quadratic Complexity

In the particular case where $A = \emptyset$, Algorithm 1 is quadratic, which is also the best known result for the word problem and normalization problem in both ortholattices and lattices [Guilloud et al.

2023a; Whitman 1941]. In general, it can be beneficial to keep the number of axioms as small as possible. For this purpose, we can combine axioms with the same left-hand side into one. Given two axioms representing $a \leq b_1$ and $a \leq b_2$ we can merge them into an equivalent one $a \leq b_1 \wedge b_2$. Indeed, given an axiom sequent $\{a^L, (b_1 \wedge b_2)^R\}$ we can derive $\{a^L, b_1^R\}$ as follows:

$$\dfrac{\dfrac{}{a^L, (b_1 \wedge b_2)^R}\,\text{Ax} \qquad \dfrac{\dfrac{}{b_1^L, b_1^R}\,\text{Hyp}}{(b_1 \wedge b_2)^L, b_1^R}\,\text{LeftAnd}}{a^L, b_1^R}\,\text{Cut}$$

Dually, we can merge axioms with the same right-hand side, $a_1 \leq b$ and $a_2 \leq b$ into $a_1 \vee a_2 \leq b$. Finally, $a \leq \neg b$ can be rewritten into $b \leq \neg a$ and vice versa. We can repeat this process until all left-hand sides and all right-hand sides of axioms are distinct, and no left side is a complement of a right side (we can even use normal forms for ortholattices to make such checks more general [Guilloud et al. 2023a]). Such axiom pre-processing transformations do not change the set of provable formulas. They can be done in time $O(n^2)$ and they reduce $|A|$ while not increasing n. Using such transformations can thus improve the cubic bound for certain kinds of axiom sets. As a very special case, if all axioms have the form $1 \leq b_i$ and $a_i \leq 0$ (corresponding to singleton sequents), we can combine them into a single axiom, obtaining $O(n^2)$ complexity.

Corollary 4.4. There is an $O(n^2)$ algorithm for checking provability from axiom sets A in which all axioms are singleton sequents.

5 PROOF STRENGTH OF ORTHOLOGIC WITH AXIOMS

The previous section established a cubic time algorithm (Algorithm 1) for deriving all consequences of axioms that hold in orthologic. This generalization is sound for classical logic while still being efficient. A key question then is: how precise is it as an approximation of classical logic?

To help answer this question, we present several classes of classical problems that our algorithm solves *exactly*: it is not only sound for them (as it is for all problems), but also *complete*: it always finds a proof if, e.g., a SAT solver would find it. Furthermore, we partly characterize our *OL* proof system in terms of restricted forms of resolution for propositional logic.

We are interested in traditional classes of deduction problems that are solvable by *OL* proofs. Formally, we define the deduction problem in orthologic, and, respectively, classical logic.

Definition 5.1. An instance of the deduction problem is characterized by a pair (A, S) where A is a set of axioms and S the goal, all of which are sequents whose interpretation as inequality is given by Section 3. The deduction problem in *OL* (resp. *CL*) consists in deciding if the goal S can be derived from axioms A in orthologic (resp. classical logic). If this is the case, then the instance is called *valid*.

Definition 5.2. An instance of the deduction problem is *OL*-solvable if and only if it has the same validity in *OL* and *CL*. A class of instances of the deduction problem is *OL*-solvable if and only if all its members are *OL*-solvable.

As *OL* is sound relative to *CL*, the following are equivalent formulations of *OL*-solvability:

- The instance has a proof in *OL* if and only if it has a proof in *CL*.
- The goal of the instance is true in all ortholattice interpretations satisfying the axioms if and only if it is true in all $\{0, 1\}$ interpretations satisfying the axioms.

In particular, if the goal of the instance is the empty sequent (hence, we talk about the consistency of axioms), *OL*-solvability of $(A, \emptyset)$ is equivalent to each of the following statements:

- The axioms of the instance are either unsatisfiable in *OL* or satisfiable in *CL*.

- The axioms of the instance either have a non-trivial Boolean model, or admit only the trivial one-element structure as a model among all ortholattices.

In particular, Theorem 4.3 gives a polynomial-time decision procedure with respect to classical logic for any class of deduction problems instances that are *OL*-solvable.

In the sequel, we look at the satisfiability of propositional logic formulas in conjunctive normal form (CNF), which are conjunctions of clauses, as their analysis plays an important role in proof theory of *CL* and the practice of SAT solving. Among the simplest and most studied refutationally complete systems for *CL* is resolution on clauses, shown in Figure 3.

$$\frac{C, x \qquad C', \neg x}{C, C'}\ \text{Resolution} \qquad \frac{}{C, x, \neg x}\ \text{Hypothesis} \qquad \frac{C}{C, C'}\ \text{Weaken}$$

Fig. 3. The Resolution proof system with hypothesis and weakening rules. C and C' represent arbitrary sets of literals. This system is complete for deriving contradictions in *CL*, with formulas expressed in conjunctive normal form, even without the Weaken and Hypothesis rules [Robinson and Voronkov 2001, Chapter 2].

5.1 Completeness for 2SAT

We start with the simplest example of a CNF, the 2CNF class. A 2CNF formula is a finite set of clauses $C_1, ..., C_m$, where each clause is a disjunction of two literals or a single literal. For example, $(x \vee \neg y), (\neg x \vee z), (\neg z)$ is a 2CNF formula. 2SAT if the problem of deciding if a 2CNF formula is satisfiable, i.e. if it has a model in the two-element Boolean algebra. Conversely, the instance is unsatisfiable if and only if the conjunction of the clauses implies falsity.

We next show how to encode a 2SAT instance into an *OL* deduction problem. The idea is to view a 2SAT instance as a deduction problem $(A, \emptyset)$ where each axiom in A is a sequent containing at most two labelled *variables* as formulas. We create an axiom sequent for each clause, where a negative literal $\neg p$ becomes labelled formula p^L and a positive literal p becomes p^R. For example, $\{\neg x, y\}$ becomes x^L, y^R. Similarly, $\{x, y\}$ becomes x^R, y^R, whereas $\{x\}$ becomes x^R. This encoding is equivalent (in *CL*) to the 2SAT instance, with the interpretation of sequents given in Definition 3.3.

Consider the Resolution prof system shown in Figure 3. For 2SAT instances, the outcome of resolution can be simulated by orthologic using the Cut rule, which allows us to prove the following.

Theorem 5.3. 2SAT is *OL*-solvable.

Proof. Consider a 2CNF and its representation as a set of sequents. The instance is unsatisfiable in *CL* if and only if there exist a derivation of the empty clause in Resolution. We proceed by induction on the Resolution derivation to show that if a clause is derived, then there is an *OL*-proof of the corresponding sequent.

Consider a Resolution step between two clauses of (at most) two elements. The sequent corresponding to its conclusion can be deduced from the sequent corresponding to its premises by a single application of the Cut rule in orthologic.

$$\frac{\{\gamma, y\} \qquad \{\neg y, \delta\}}{\gamma, \delta}\ \text{Resolution} \quad \hookrightarrow \quad \frac{\{\Gamma, y^R\} \qquad \{y^L, \Delta\}}{\Gamma, \Delta}\ \text{Cut}$$

Where γ and δ are one arbitrary literals (or no literal) and Γ and Δ represent the corresponding annotated formula (or absence thereof). Weaken and Hypothesis steps are similarly simulated by the eponymous steps in *OL*.

Conversely, if the empty sequent is derivable in orthologic, then no non-trivial ortholattice satisfy the assumptions, and hence no Boolean algebra. □

5.2 Orthologic Emulates Unit Resolution

For an arbitrary clause $N \cup P$, let it be encoded as the sequent $(\bigwedge N)^L, (\bigvee P)^R$, where N (resp. P) is the set of negative (resp. positive) variables in the clause. A *Unit Resolution* step is a Resolution step (Figure 3) where $C = \emptyset$ or $C' = \emptyset$. Interpreted over sequents, the application of a Unit Resolution step on two clauses $C_1 = \{x_i\}$ and $C_2 = (\{\neg x_1, \ldots, \neg x_n\} \cup P)$ corresponds to the deduction rule UnitResolutionR. Dually, for $C_1 = \{\neg x_i\}$ we obtain UnitResolutionL:

$$\frac{(x_i)^R \qquad (x_1 \wedge \ldots \wedge x_{i-1} \wedge x_i \wedge x_{i+1} \wedge \ldots \wedge x_n)^L, (\bigvee P)^R}{(x_1 \wedge \ldots \wedge x_{i-1} \wedge x_{i+1} \wedge \ldots \wedge x_n)^L, (\bigvee P)^R}\ \text{UnitResolutionR}$$

$$\frac{(x_i)^L \qquad (\bigwedge N)^L, (x_1 \vee \ldots \vee x_{i-1} \vee x_i \vee x_{i+1} \vee \ldots \vee x_n)^R}{(\bigwedge N)^L, (x_1 \vee \ldots \vee x_{i-1} \vee x_{i+1} \vee \ldots \vee x_n)^R}\ \text{UnitResolutionL}$$

Lemma 5.4 (*OL* simulates Unit Resolution)**.** UnitResolutionL and UnitResolutionR are admissible rules (Definition 3.2) for *OL* proof system of Figure 2.

Proof. We aim to show that this step is admissible in *OL* proofs. Instead of giving a syntactic transformation, we can see that the steps are sound in *OL* with a short semantic argument. For UnitResolutionL, consider any ortholattice and assume that the premises of the rules are true. The meaning of x_i^R is $1 \leq x^R$, which implies $x_i = 1$. This implies $(x_1 \wedge \ldots \wedge x_i \wedge \ldots \wedge x_n) = (x_1 \wedge \ldots \wedge 1 \wedge \ldots x_n)$, so the value of the non-unit premise clause reduces to the truth of the conclusion of the rule. By completeness (Lemma 3.4), there exists a proof of the conclusion from the premises. Thus, UnitResolutionR is admissible. The argument for UnitResolutionL is dual. □

5.3 Completeness for Horn Clauses

A Horn clause is a disjunction of literals such that at most one literal is positive. We encode a Horn clause into a sequent as $(a_1 \wedge \ldots \wedge a_n)^L, b^R$ where a_i are the negated variables of the clause (if any), and b the positive literal of the clause (if it exists). A Horn instance is a conjunction of Horn clauses, and is unsatisfiable if and only if the empty clause can be deduced from it using Resolution.

We encode a Horn instance into a deduction problem by adding an axiom for each Horn clause, and the empty sequent as the goal.

Horn instances can be solved using Unit Resolution only [Minoux 1988]. By Lemma 5.4, *OL* is complete for Horn instances. Other classes solvable using only Unit Resolution include *q-Horn*, *extended Horn* and *renamed Horn* instances [Čepek and Kučera 2005].

Corollary 5.5 (Horn Clause Completeness). Horn Clause instances, q-Horn instances, extended Horn instances, and renamed Horn instances are *OL*-solvable classes.

Note that, despite our use of semantic techniques to show completeness for resolution, the results of Theorem 4.3 apply, provide polynomial-time guarantees for solving these instances.

Renamed Horn instances are an interesting extension of Horn instances. A conjunction of clauses is renamed Horn if and only if there exists a set of variables V such that complementing variables of V in the instance yields a Horn instance. In particular, a clause in a renamed Horn instance can contain multiple positive and negative literals. Unit Resolution is stable under such renaming, meaning that a unit resolution derivation of the empty clause remains a valid unit resolution derivation of the empty clause after renaming. This is the reason that Unit Resolution is also complete for renamed Horn clauses.

5.4 Renaming Deduction Problems

Motivated by renamed Horn instances, we now consider renaming of general deduction problems. We show that renamings of *OL*-solvable instances are *OL*-solvable.

Definition 5.6. For an arbitrary variable x, the complement of x is $\neg x$ and the complement of $\neg x$ is x. Two deduction problems I_1 and I_2 are *renamings* of each other if there exists a set of variables V such that complementing variables of V in the axioms and goal of I_1 yields I_2.

Lemma 5.7. Let I be a deduction instance such that I is valid in *OL* if and only if it is valid in *CL*. Then all renamed versions of I are valid in *OL* if and only if they are valid in *CL*.

Proof. In *OL*, if I_1 and I_2 are renamed versions of each other by a set of variables V, I_1 and I_2 have the same validity. Indeed, suppose there is an ortholattice O and assignement $s_1 : V \to O$ that is a counter model of I (meaning it satisfies the axioms but not the goal, so I_1 is invalid). Then define s_2 such that $s_2(x) = s_1(x)$ if $x \notin V$ and $s_2(x) = \neg s_1(x)$ if $x \in V$. It is then easy to check that since $\neg\neg x = x$ in *OL*, s_2 is a counter model for I_2. Hence if I_1 is invalid then I_2 is invalid. Conversely, if I_2 admits a counter model, then I_1 admits a counter model as well.

Similarly in *CL*, I_1 and I_2 have the same validity by the same argument, considering assignments $V \to \{0, 1\}$ as counter models. Hence, since I_1 has the same validity in *OL* and *CL*, so does I_2.

□

5.5 Tseitin's Transformation for Orthologic Axioms

Tseitin's transformation for classical logic transforms a formula with arbitrary alternations of conjunctions and disjunctions into one in Conjunctive Normal Form (CNF) that is equisatisfiable. The transformation works by introducing a linear number of new variables, and runs in near linear time. This justifies the focus of SAT solvers on solving formulas in CNF.

The essence of Tseitin's transformation in classical logic is to introduce variables, such as x, that serve as names for subformulas, such as F, with the interpretation of x bound to be equal the interpretation of F. Unfortunately, we cannot express such transformation as a single *OL* formula, because knowing that $\neg x \vee F$ equals 1 inside a formula does not imply $x \leq F$. On the other hand, once we make use of the power of axioms, the usual Tseitin's transformation again becomes possible. This shows the importance of adding axioms to orthologic reasoning.

Definition 5.8 (*OL*-Tseitin transformation). Given a deduction problem instance with axioms A and goal S (where we assume without loss of generality that all formulas are in negation normal form), pick an arbitrary strict subexpression e of a formula in A or S of the form $x \wedge y$ or $x \vee y$, for some literals x and y. Pick a fresh variable c, introduce the axioms c^L, e^R and e^L, c^R and replace e by c in A and S. Repeat until all formulas have height at most 2. We say that a problem (A, S) is in Tseitin normal form if it is obtained from another problem using this transformation.

The transformation does not alter the validity of a deduction problem instance in *OL*, since we merely defined aliases for subexpressions. Indeed, having both (c^L, e^R) and (e^L, c^R) as axioms is equivalent to forcing $e = c$ in all models.

5.6 Resolution Width for Orthologic Proofs in CNF

Consider an *OL* proof for a deduction problem to which we applied *OL*-Tseitin's transformation. If $\circ$ and $\square$ denote arbitrary $\{\_^L, \_^R\}$ annotations, the resulting problem will only contain sequents of the form: $\{a^\circ, (b \wedge c)^\square\}$, $\{a^\circ, (b \vee c)^\square\}$, $\{a^\circ, b^\square\}$, $\{a^\circ\}$, and $\emptyset$, for some literals a, b and c. Moreover, remember that by the subformula property (Corollary 3.9), if the problem admits a proof, then it has a proof that only uses formulas among a, $a \wedge b$ and $a \vee b$ (for any literals a and b appearing in

the problem). Hence, we can constrain every proof of S to involve only sequents of at most 4 literals. In classical logic, every sequent appearing in the proof would then be equivalent to a conjunction of disjunctions of literals (i.e. a conjunction of clauses). In the simplest case:

$$(w \wedge x)^L, (y \vee z)^R \quad \rightsquigarrow \quad (\neg w \vee \neg x \vee y \vee z)$$

For sequents involving a left disjunction or right conjunction, using greatest lower bound and lowest upper bound properties of $\wedge$ and $\vee$:

$$\begin{aligned}
(w \vee x)^L, (y \vee z)^R &\quad \rightsquigarrow \quad (\neg w \vee y \vee z) \wedge (\neg x \vee y \vee z) \\
(w \wedge x)^L, (y \wedge z)^R &\quad \rightsquigarrow \quad (\neg w \vee \neg x \vee y) \wedge (\neg w \vee \neg x \vee z) \\
(w \vee x)^L, (y \wedge z)^R &\quad \rightsquigarrow \quad (\neg w \vee y) \wedge (\neg w \vee z) \wedge (\neg x \vee y) \wedge (\neg x \vee z)
\end{aligned}$$

And similarly with all combinations of ^{L} and ^{R}, where, for example, a conjunction with L polarity behaves much like a disjunction with R polarity. We say that such a set of clauses *represents* the corresponding sequent. Crucially, each of these clauses contains at most 4 literals.

We now consider again the Resolution proof system of Figure 3. Note that we work with plain resolution and *not* extended resolution that introduces fresh variables on the fly [Tseitin 1983].

Definition 5.9. The **width** of a resolution proof is the number of literals in the largest clause appearing in the proof.

The next theorem characterizes the width of those classical logic resolution proofs that suffice to establish all formulas provable by *OL* derivations.

Theorem 5.10. An *OL* proof of a problem in *OL*-Tseitin normal form can be simulated by Figure 3 proofs of width 5.

Proof. Consider an *OL* proof that satisfies properties of Theorem 3.8. We proceed by induction on the structure of such proof. The cases of non-cut rules are immediate. Namely, Hypothesis and Weakening in *OL* are directly simulated by the corresponding steps in Resolution. In LeftNot, the principal formula must be a literal, so that the clause representation of the conclusion is the same as the interpretation of the premise. In LeftOr, similarly, the representation of the conclusion is the conjunction of the representations of the premises. LeftAnd is simulated in a Resolution proof by Weakening. Right- steps are symmetrical to Left-steps. The only non-trivial case is the Cut rule. The cut formula can have different shape

(1) The cut formula is a literal

$$\frac{\Gamma, x^R \qquad x^L, \Delta}{\Gamma, \Delta}\ \text{Cut}$$

We then have technically 36 different cases to describe depending on whether Γ and Δ are conjunctions, disjunctions or literals and their polarity. We show the two extreme cases, and all other can be deduced by symmetry.
If Γ and Δ are left conjunctions, right disjunctions or literals, the Cut rule is simulated with a single Resolution instance:

$$\frac{(a \wedge b)^L, x^R \qquad x^L, (c \vee d)^R}{\Gamma, \Delta}\ \text{Cut} \quad \hookrightarrow \quad \frac{\{\neg a, \neg b, x\} \qquad \{\neg x, c, d\}}{\{\neg a, \neg b, c, d\}}\ \text{Resolution}$$

If Γ and Δ are left disjunctions or right conjunctions, 2 applications of Resolution are necessary to obtain each of the two clauses in the conclusion (4 if both):

$$\dfrac{(a \vee b)^L, x^R \qquad x^L, (c \wedge d)^R}{\Gamma, \Delta}\ \text{Cut}$$

$$\hookrightarrow$$

$$\dfrac{\{\neg a, x\}, \{\neg b, x\} \qquad \{\neg x, c\}, \{\neg x, d\}}{\{\neg a, c\}, \{\neg a, d\}, \{\neg b, c\}, \{\neg b, d\}}\ 4\times \text{Resolution}$$

Where each of the clause in the conclusion can be reached by using Resolution on two of the clauses in the premises.

(2) Consider now the case where the Cut formula is a conjunction (the disjunction case is symmetrical).

$$\dfrac{\Gamma, (x \wedge y)^R \qquad (x \wedge y)^L, \Delta}{\Gamma, \Delta}\ \text{Cut}$$

We again present the two extreme cases. If Γ and Δ are both left conjunctions or right disjunctions:

$$\dfrac{(a \wedge b)^L, (x \wedge y)^R \qquad (x \wedge y)^L, (c \vee d)^R}{(a \wedge b)^L, (c \vee d)^R}\ \text{Cut}$$

$$\hookrightarrow$$

$$\dfrac{\dfrac{\{\neg a, \neg b, x\}, \{\neg a, \neg b, y\} \qquad \{c, d, \neg x, \neg y\}}{\{\neg a, \neg b, c, d, \neg y\}}\ \text{Resolution on } x}{\{\neg a, \neg b, c, d\}}\ \text{Resolution on } y$$

The conclusion can be reached by applying Resolution twice successively, but here the intermediate clause $\{\neg a, \neg b, c, d, \neg y\}$ reaches width 5.

If Γ and Δ are left disjunctions or right conjunctions:

$$\dfrac{(a \vee b)^L, (x \wedge y)^R \qquad (x \wedge y)^L, (c \wedge d)^R}{(a \vee b)^L, (c \wedge d)^R}\ \text{Cut}$$

$$\hookrightarrow$$

$$\dfrac{\dfrac{\{\neg a, x\}, \{\neg a, y\}, \{\neg b, x\}, \{\neg b, y\} \qquad \{\neg x, \neg y, c\}, \{\neg x, \neg y, d\}}{\{\neg y, \neg a, c\}, \{\neg y, \neg a, d\}, \{\neg y, \neg b, c\}, \{\neg y, \neg b, d\}}\ 4\times \text{Resolution on } x}{\{\neg a, c\}, \{\neg a, d\}, \{\neg b, c\}, \{\neg b, d\}}\ 4\times \text{Resolution on } y$$

the Cut can be simulated by first resolving 4 times on x and then 4 times on y.

Hence, Resolution of width 5 can simulate all *OL* proofs. □

6 EFFECTIVELY PROPOSITIONAL ORTHOLOGIC

So far we have studied propositional orthologic. In this section we introduce decidable classes of *predicate* orthologic. Our inspiration is the Bernays–Schönfinkel-Ramsey (BSR) class of classical first-order logic formulas [Börger et al. 1997, Section 6.2.2], which consists of formulas of first order logic that contain predicates and term variables but no function symbols, and whose prenex normal form is of the form $\exists x_1, ...x_n.\forall y_1, ..., y_n.\phi$ where ϕ is quantifier free. Syntactically, a formula in the BSR class can be represented as a quantifier-free formula with constants and variables symbols, where the variables are implicitly universally quantified. It is also called *Effectively Propositional Logic* (EPR) [Piskac et al. 2010], because deciding the validity of such formula can be reduced to deciding the validity of a formula in propositional logic by a grounding process. This is possible because formulas in the BSR class have finite Herbrand universe [Robinson and Voronkov 2001, p.1798]. BSR class (with its multi-sorted logic generalization) has found applications in verification [Padon et al. 2017].

We show that orthologic also admits a similar grounding process, allowing us to define the class of *effectively propositional orthologic*. The class is EXPTIME in the worst case, in contrast to co-NEXPTIME for BSR. Moreover, we show that it becomes polynomial if we restrict the maximal number of variables in axioms, which is in contrast to the corresponding restriction yielding an NP-hard class for classical logic [Börger et al. 1997, Section 6.2.2].

In this section, *variables* denote variable symbols at the term level, and not propositional variables. We fix two disjoint countably infinite sets of symbols: constants C, and variables V. A predicate signature Σ specifies a finite set of predicate symbols $\{p_1, \ldots, p_n\}$ with their non-negative arities s_i, and a finite non-empty set of constants $\bar{C} = \{c_1, \ldots, c_n\}$. Define the set of atomic formulas over Σ as

$$P_\Sigma = \bigcup_{i=1}^{n} \{p_i(\vec{x}) \mid \vec{x} \in (V \cup \bar{C})^{s_i}\}$$

An EPR formula (over Σ) is a formula constructed from P_Σ using $\wedge, \vee, \neg$, corresponding to $\mathcal{T}_{P_\Sigma}(OL)$. An annotated EPR formula is a^L or a^R where a is an EPR formula. An EPR sequent is a set of at most two annotated EPR formulas. The degree of a formula or sequent is the number of distinct free variables in it. The degree of a finite set A of sequents, $d(A)$, is the maximum of degrees of its sequents. An atomic formula, formula, or a sequent is *ground* if it contains no variables (only constants), that is, it has degree zero.

Definition 6.1. An EPR deduction instance is a set (A, S) where A (the axioms) is a set of EPR sequents and S (the goal) is an EPR sequent.

Definition 6.2. An instance of a formula (respectively, sequent) is a formula (or sequent) obtained by replacing all occurrences of some variables by other variables or constants. The expansion of a formula s (respectively, sequent), denoted s^*, is the set of all of its instances. s^* is countable, and infinite if s contains at least one variable. If A is a set of sequents then $A^* = \bigcup\{s^* \mid s \in A\}$. For an EPR deduction instance (A, S) its expansion is (A^*, S).

Definition 6.3. EPR-OL-D is the problem, given a signature Σ and EPR deduction instance (A, S) over Σ, decide whether its expansion (A^*, S) is a valid OL deduction instance (in the sense of Definition 5.1).

We first show as an intermediate lemma that we only need to look at instances of A using variables appearing in S and constants in S and A.

Lemma 6.4. Suppose S has a proof $\mathcal{P}$ involving axioms in A^*. Then it has a proof containing only variables that appear in S and constants in S and A.

Proof. If a variable z appears somewhere in $\mathcal{P}$ but not in S, then it has to be eliminated by a Cut rule at some point.

$$\dfrac{\dfrac{\mathcal{A}}{\Gamma, \phi(z)^R} \qquad \dfrac{\mathcal{B}}{\phi(z)^L, \Delta}}{\Gamma, \Delta}\ \text{Cut}$$

Let $c \in \bar{C}$ be any constant symbol of Σ. Let $\mathcal{A}[z := c]$ be the proof $\mathcal{A}$ with every instance of z replaced by c. For any given axiom $a \in A^*$, $a[z := c]$ is also an axiom of A^*, so that all axioms steps occurring in $\mathcal{A}[z := c]$ are correct. It is easy to see that all non-axioms steps in $\mathcal{A}[z := c]$ remain correct under the substitution. Because by assumption the Cut rules eliminates z from the formula, Γ does not contain z and hence the conclusion of $\mathcal{A}[z := c]$ is exactly $\Gamma, \phi(c)^R$. The same can be done to $\mathcal{B}$, so that we obtain a proof where z does not appear:

$$\dfrac{\dfrac{\mathcal{A}[z := c]}{\Gamma, \phi(c)^R} \quad \dfrac{\mathcal{B}[z := c]}{\phi(c)^L, \Delta}}{\Gamma, \Delta}\ \text{Cut}$$

To eliminate constant, we use the same argument except that since axioms are not stable under renaming of constants (but all other rules and in particular the hypothesis rule are), we cannot eliminate constant symbols appearing in A. □

Theorem 6.5. The EPR-OL-D problem (A, S) of size n and degree $d(A)$ is solvable in $PTIME(n^{d(A)})$.

Proof. We will reduce EPR-OL-D to propositional OL deduction problems, which Algorithm 1 can then solve. By completeness of OL (Lemma 3.4), whether S holds in all ortholattices where A holds is equivalent to the following question: Does there exist a finite subset A' of A^* such that S has an orthologic proof with axioms among A'?

Lemma 6.4 implies that for any sequent S, we only need to consider a finite number of axioms, namely the axioms involving variables in S and constants in A and S (minimum one). Each axiom a has at most $(|S| + ||A||)^{d(A)}$ such instances, so the total number of axiom we need to consider is $O(|A| \cdot (|S| + ||A||)^{d(A)}) = O(n^{d(A)+1})$. Combining this result with Theorem 4.3 gives $PTIME(n^{d(A)})$, or, more precisely, $O(n^{3(d(A)+1)})$. □

6.1 Instantiation as a Rule

Instead of starting by grounding all axioms, we can delay instantiation until later in the proof, yielding shorter proofs in some cases. Formally, we add an instantiation step to the proof calculus of Figure 2 over $\mathcal{T}_{OL}(P)$:

$$\dfrac{\Gamma, \Delta}{\Gamma[\vec{x} := \vec{t}], \Delta[\vec{x} := \vec{t}]}\ \text{Inst}$$

Holding for arbitrary sets of term variable $\vec{x}$ and terms $\vec{t}$. We note the resulting proof system OL^I.

Lemma 6.6. For a sequent S and set of axioms A over quantifier-free predicate logic, S has has an OL proof from A^* if and only if S has an OL^I proof from A.

Proof.

→ : Given an OL proof with axioms in A^*, said axioms can be obtained from A in OL^I by an application of the instantiation rule:

$$\dfrac{\dfrac{}{\dfrac{\Gamma^*, \Delta^*}{\dfrac{\cdots}{S}}}\ \text{Ax}}{} \quad \hookrightarrow \quad \dfrac{\dfrac{}{\dfrac{\dfrac{\Gamma, \Delta}{\dfrac{\Gamma^*, \Delta^*}{\dfrac{\cdots}{S}}}\ \text{Inst}}{}}\ \text{Ax}}{}$$

Where $(\Gamma, \Delta) \in A$ and $(\Gamma^*, \Delta^*) \in A^*$.

← : Note that if the Inst rule are only uses right after axioms, then we can reverse the transformation above. We show that given a proof in OL^I using Inst, the instances of Inst can be swapped with other rules and be pushed to axioms. For example

$$\dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^L, \Delta} \quad \dfrac{\mathcal{A}''}{\beta^L, \Delta}}{(\alpha \vee \beta)^L, \Delta}\ \text{LeftOr}}{(\alpha^* \vee \beta^*)^L, \Delta^*}\ \text{Inst} \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\mathcal{A}'}{\alpha^L, \Delta}}{(\alpha^*)^L, \Delta^*}\ \text{Inst} \quad \dfrac{\dfrac{\mathcal{A}''}{\beta^L, \Delta}}{(\beta^*)^L, \Delta^*}\ \text{Inst}}{(\alpha^* \vee \beta^*)^L, \Delta^*}\ \text{LeftOr}$$

The cases for all other rules are similar. Then, any conclusion of an Inst rule is a member of $\mathcal{A}^*$ and can be replaced by an Axiom rule to obtain an OL proof from A^*. □

6.2 Proof Search with Unification

While searching for a proof, we usually want to delay decision-making (such as which variable to instantiate) for as long as possible. In backward proof search, this means we want to delay it until the sequent is an axiom of A^*. In forward proof search, however, being able to use the Inst rule allows delaying instantiation as much as possible, as in resolution for first-order logic classical [Robinson 1965], [Robinson and Voronkov 2001, Chapter 2].

We thus adopt unification to decide when and how to instantiate a variable, whenever we use a rule with two premises. The corresponding directed rules are shown in Figure 4. Without function symbols, the most general unifier of ϕ and ψ is the substitution θ of smallest support such that $\theta(\phi) = \theta(\psi)$ [Robinson and Voronkov 2001, Chapter 8].

$$\dfrac{\dfrac{\Gamma, \phi^R}{\theta(\Gamma), \theta(\phi)^R}\,\text{Inst} \qquad \dfrac{\psi^L, \Delta}{\theta(\psi)^L, \theta(\Delta)}\,\text{Inst}}{\theta(\Gamma), \theta(\Delta)}\,\text{Cut}$$

where θ is the most general unifier of ϕ and ψ

$$\dfrac{\dfrac{\Gamma_1, \phi^R}{\theta(\Gamma_1), \theta(\phi)^R}\,\text{Inst} \qquad \dfrac{\Gamma_2, \psi^R}{\theta(\Gamma_2), \theta(\psi)^R}\,\text{Inst}}{\theta(\Gamma_1), \theta(\phi \wedge \psi)^R}\,\text{RightAnd} \qquad \dfrac{\dfrac{\Gamma_1, \phi^L}{\theta(\Gamma_1), \theta(\phi)^R}\,\text{Inst} \qquad \dfrac{\Gamma_2, \psi^L}{\theta(\Gamma_2), \theta(\psi)^R}\,\text{Inst}}{\theta(\Gamma_1), \theta(\phi \vee \psi)^L}\,\text{LeftOr}$$

where θ is the most general unifier of Γ_1 and Γ_2

Fig. 4. Sequent-calculus style deduction rules with unification for Effectively Propositional Orthologic.

Theorem 6.7. A sequent S over $\mathcal{T}_{OL}(P)$ has a proof in OL^I if and only if there exists a sequent S' such that S is a particular instantiation of S' and S' has a proof where the Inst rule is only used in the specific cases of Figure 4 or to rename variables.

Proof. (Sketch). The proof is once again by induction and case analysis on the proof $\mathcal{P}$ of S, except we move the instantiation step toward the conclusion of the proof. Any instance of the Inst rule in $\mathcal{P}$ can be swaped with the next rule, unless it is a Cut, LeftOr or RightAnd rule. Consider the proof rule that follows the instantiation:

- Hypothesis is a leaf rule, so it can never follow a step.
- Weaken is immediate, as long as the variables in Δ are properly renamed

$$\dfrac{\dfrac{\dfrac{\mathcal{A}}{\Gamma}}{\sigma(\Gamma)}\,\text{Inst}}{\sigma(\Gamma), \Delta}\,\text{Weaken} \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\mathcal{A}}{\Gamma}}{\sigma(\Gamma), \pi(\Delta)}\,\text{Weaken}}{\sigma(\Gamma), \Delta}\,\text{Inst}$$

  Where π is a renaming of variables in Δ to names that are fresh. In particular, it is invertible.
- LeftNot and RightNot are immediate.
- For RightOr and LeftAnd, the transformation is the same as Weaken.
- In the Cut case, assume that the two premises (Γ, ϕ^R) and (ψ^L, Δ) have no shared variables, by renaming them if necessary:

$$\dfrac{\dfrac{\dfrac{\Gamma, \phi^R}{\sigma_1(\Gamma), \sigma_1(\phi)^R}\ \text{Inst} \qquad \dfrac{\psi^L, \Delta}{\sigma_2(\psi)^L, \sigma_2(\Delta)}\ \text{Inst}}{\sigma_1(\Gamma), \sigma_2(\Delta)}\ \text{Cut}}{}$$

$$\hookrightarrow$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, \phi^R}{\theta(\Gamma), \theta(\phi)^R}\ \text{Inst} \qquad \dfrac{\psi^L, \Delta}{\theta(\psi)^L, \theta(\Delta)}\ \text{Inst}}{\theta(\Gamma), \theta(\Delta)}\ \text{Cut}}{\sigma_1(\psi)^L, \sigma_2(\Delta)}\ \text{Inst}}{}$$

Note that since θ is the most general unifier for ϕ and ψ, and $\sigma_1(\phi) = \sigma_2(\psi)$, θ factors in both σ_1 and σ_2, so that the last Inst step is correct.

- LeftOr and RightAnd are similar to the Cut step.

□

6.3 Solving and Extending Datalog Programs with Orthologic

Datalog is a logical and declarative programming language admitting formulas in a further restriction of the BSR class where ϕ is forced to be a Horn clause (over predicates). A Datalog program is then a conjunction of such formulas (or clauses) [Dantsin et al. 2001; Ullman 1988, 1989]. While the validity problem for the BSR class is coNEXPTIME complete [Piskac et al. 2010; Ramsey 1930], solving a Datalog program is only EXPTIME-complete. This makes Datalog a suitable language for logic programming and database queries. Typically, a Datalog query asks if a certain fact (an atom without variable) is a consequence of the clauses in the program. This naturally corresponds to solving a deduction problem, with the axioms corresponding to the program and the goal to the query.

Lemma 6.8. Datalog program can be evaluated using orthologic.

Proof. Datalog programs and queries form a subset of the BSR class. Theorem 6.5 shows that such problems can be reduced via grounding to purely propositional *OL*. Moreover, as the resulting set of axioms contains only Horn clauses, Corollary 5.5 implies that the Datalog program has the same semantic in *OL* and in *CL*. □

This means that for any Datalog program, the *OL* semantic agrees with the classical semantic. Algorithm 1 hence provides a decision procedure for Datalog with complexity $PTIME(n^{d(A)})$ (as Theorem 6.5 shows). This matches known complexity classes of Datalog, which has exponential query complexity (corresponding here to axioms) and polynomial data complexity (corresponding to the goal and axioms with no variables, or *facts*) [Dantsin et al. 2001].

6.4 Axiomatizing Congruence and Equality Relations

We next show that, when equality is axiomatized in effectively propositional orthologic, a substitution rule becomes admissible. Let X be a countably infinite set of term variables whose elements are noted $x, y, z, ...$ Let be given a presentation with predicate symbols $p_1, ..., p_n$ each of arity s_i and $\sim$ a predicate symbol representing a congruence relation on X. Consider the set of axiom $A_\sim$ containing the following sequents that axiomatize the equivalence property:

$$(x \sim x)^R$$
$$(x \sim y)^L, (y \sim x)^R$$
$$(x \sim y \wedge y \sim z)^L, (x \sim z)^R$$

and for each symbol p_i and each $1 \leq j \leq s_i$, the congruence property for p_i:

$$(x \sim y \wedge p_i(z_1, ..., z_{j-1}, x, z_{j+1}, ..., z_{s_i}))^L, p_i(z_1, ..., z_{j-1}, y, z_{j+1}, ..., z_{s_i})^R$$

Again, this does not constitute a finite presentation of an ortholattice, as there are infinitely many possible instances of axioms. However, by Theorem 6.5, if a sequent S over $X_\sim$ has a proof involving axioms in $A_\sim$, then it has a proof with only variables that appear in S. Moreover, the degree of $A_\sim$ is $d(A_\sim) = \max(3, \max_i(s_i) + 1)$ axioms, so that the complexity of the proof search is exponential in the arity of the predicates in the language and polynomial in the size of the problem, for a fixed language. The following lemma shows that in any decision problem whose axioms contain A, we can add a substitution rule for equality.

Lemma 6.9. Fix a set of predicate symbols and constants. Let A be a set of axiom such that $\mathcal{A}_\sim \subset A$. The following rule for substitution of equal terms is admissible in OL with axioms in A:

$$\frac{\Gamma[x := s], \Delta[x := s] \qquad s \sim t}{\Gamma[x := t], \Delta[x := t]}\ \text{Subst}_\sim$$

Proof. Suppose x occurs only once in Γ, Δ (if it appears multiple time, we repeat the argument). Suppose without loss of generality that this unique occurrence is in Γ. Let $a(x) \equiv p_i(u_1, ..., x, ...u_{s_i})$ be the atomic formula containing this occurrence of x, i.e. $\Gamma = \Gamma'[\chi := a(x)]$, for a propositional variable χ. Axioms in $A_\sim$ allow the following proof, where we first derive $a(s)^L, a(t)^R$:

$$\dfrac{\dfrac{\dfrac{\dfrac{s \sim t^R}{a(s)^L, s \sim t^R}\ \text{Weak.} \qquad \dfrac{}{a(s)^L, a(s)^R}\ \text{Hyp.}}{a(s)^L, (s \sim t \wedge a(s))^R}\ \text{R.And} \qquad \dfrac{}{(s \sim t \wedge a(s))^L, a(t)^R}\ \text{Ax.}}{a(s)^L, a(t)^R}\ \text{Cut}$$

and then conclude, using an analogous derivation of $a(t)^L, a(s)^R$

$$\dfrac{\dfrac{\Gamma[x := s], \Delta}{\Gamma'[\chi := a(s)], \Delta} \qquad a(s)^L, a(t)^R \qquad \dfrac{\dfrac{s \sim t^R \qquad \dfrac{}{s \sim t^L, t \sim s^R}\ \text{Ax.}}{t \sim s^R}\ \text{Cut}}{\dfrac{\ldots(\text{analogous})\ldots}{a(t)^L, a(s)^R}}}{\dfrac{\Gamma'[\chi := a(t)], \Delta}{\Gamma[x := t], \Delta}}\ \text{Subst}$$

The dashed lines denote syntactic equality. Note that we have shown the Subst rule for propositions to be admissible in orthologic in Lemma 3.5. Hence, $\text{Subst}_\sim$ is admissible in orthologic with any axiomatization containing $A_\sim$. □

7 FURTHER RELATED WORK

Bruns [1976] first solved the word problem for free ortholattices in 1976 using algebraic techniques extending the work of Whitman [1941], who first solved the word problem for free lattices.

The observation that we can obtain a proof system for Orthologic by restricting Gentzen's sequent calculus to sequents with at most two formulas was already made by Schulte Mönting [1981]. They also showed that the system without axioms admits Cut Elimination. Egly and Tompits [2003] used the same system to describe a backward proof-search procedure exponential both in time and proof size, and a polynomial ($\Omega(n^7)$) forward procedure. We improved this result to a $O(n^2)$ (time and size) backward proof search procedure. Other proof systems have been considered. Meinander [2010] used a different set of inference rule to show that the word problem for finitely

presented ortholattices is decidable in polynomial time. Their solution involves exhaustive forward deduction, and they give no precise exponent of the polynomial.

Laurent [2016] introduces two other sequent-based proof systems for orthologic using concepts from linear logic, and in particular *focusing*, to constrain proof search. Their procedure is forward-driven. While no complexity analysis is provided, their algorithm is clearly polynomial, and experimental benchmark shows improvement over the algorithm of [Egly and Tompits 2003].

Kawano [2018] describes a proof system of a different flavour for orthologic without axioms. It is based on label sequents and designed to allow for an implication symbol. However, no complexity of proof search is given and the system does not have a limit on the number of formulas per sequent, preventing a bound on the total number of sequents as in Lemma 4.2.

Guilloud et al. [2023a] use ortholattices in the context of software verification as an approximation of Boolean algebra. They present an algorithm able to normalize any formula into an equivalent one (by ortholattices laws) of smallest size, with the goal of improving caching efficiency and reducing formula size for SMT solving. Their approach does not involve a proof system and does not support axioms in general.

Researchers have explored extensions of Datalog with negation and disjunction, with various computable semantics. Most of these models do not correspond to classical models of predicate logic. Datalog is declarative, but most of its extensions rely on procedural semantics. In a line of work starting with [Clark 1978], negation has been introduced with a failure meaning, where if a cannot be verified, then $\neg a$ is taken to hold. [Kunen 1987] introduces a formal (but not decidable) semantic for negation as failure. [Apt et al. 1988] introduces the notion of stratified programs. This consists in specifying layers of clauses evaluated increasingly, so that if a is not shown in a level, $\neg a$ is assumed in the next, which differs from orthologic semantics. [Gelfond and Lifschitz 1991] introduce a notion of classical negation but the logic is not classical, as $\neg a \longrightarrow b$ and $\neg b \longrightarrow a$ are not equivalent in the proposed semantics. The use of Datalog in program analysis inspired researchers to define Datalog with lattice semantics [Madsen et al. 2016], which explicitly incorporates the concept of fixed point of particular lattices into the language semantics. Our approach is instead to view Datalog in the broader context of validity in all ortholattices, with orthologic as a convervative approximation of validity for classical logic that is always sound and, in several cases we identified, complete.

8 CONCLUSIONS

We have studied algorithmic and proof-theoretic properties of orthologic, a sound generalization of classical logic based on ortholattices. We have shown a form of generalized cut elimination for propositional orthologic in the presence of axiom, implying a subformula property. We have used this result to design a cubic-time proof search procedure for orthologic with axioms (quadratic with bounded cardinality of axiom sets). Furthermore, we have shown that some classes of classical decision problems including 2CNF, propositional Horn clause generalizations and Datalog always admit *OL* proofs. This provides sound and complete polynomial-time reasoning for a number of theorem proving tasks in classical logic. We anticipate applications of orthologic with axioms in predictable proof automation inside proof checkers, program verifiers, and expressive type systems.

REFERENCES

Krzysztof R. Apt, Howard A. Blair, and Adrian Walker. 1988. Towards a Theory of Declarative Knowledge. In *Foundations of Deductive Databases and Logic Programming*, Jack Minker (Ed.). Morgan Kaufmann, San Francisco, CA, United States, 89–148. https://doi.org/10.1016/b978-0-934613-40-8.50006-3

J. L. Bell. 1983. Orthologic, Forcing, and The Manifestation of Attributes. In *Studies in Logic and the Foundations of Mathematics (Studies in Logic and the Foundations of Mathematics, Vol. 111)*, C. T. Chong and M. J. Wicks (Eds.). Elsevier, Singapore, 13–36. https://doi.org/10.1016/S0049-237X(08)70953-4

Garrett Birkhoff and John Von Neumann. 1936. The Logic of Quantum Mechanics. *Annals of Mathematics* 37, 4 (1936), 823–843. https://doi.org/10.2307/1968621 jstor:1968621

Egon Börger, Erich Grädel, and Yuri Gurevich. 1997. *The Classical Decision Problem.* Springer, Berlin, Heidelberg.

Günter Bruns. 1976. Free Ortholattices. *Canadian Journal of Mathematics* 28, 5 (Oct. 1976), 977–985. https://doi.org/10.4153/CJM-1976-095-6

Ondřej Čepek and Petr Kučera. 2005. Known and New Classes of Generalized Horn Formulae with Polynomial Recognition and SAT Testing. *Discrete Applied Mathematics* 149, 1 (Aug. 2005), 14–52. https://doi.org/10.1016/j.dam.2003.12.011

Keith L. Clark. 1978. Negation as Failure. In *Logic and Data Bases*, Hervé Gallaire and Jack Minker (Eds.). Springer US, Boston, MA, 293–322. https://doi.org/10.1007/978-1-4684-3384-5_11

Stephen A. Cook. 1971. The Complexity of Theorem-Proving Procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (Shaker Heights, Ohio, USA) *(STOC '71)*. Association for Computing Machinery, New York, NY, USA, 151–158. https://doi.org/10.1145/800157.805047

Evgeny Dantsin, Thomas Eiter, Georg Gottlob, and Andrei Voronkov. 2001. Complexity and Expressive Power of Logic Programming. *Comput. Surveys* 33, 3 (Sept. 2001), 374–425. https://doi.org/10.1145/502807.502810

B. A. Davey and H. A. Priestley. 2002. *Introduction to Lattices and Order* (2 ed.). Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9780511809088

Uwe Egly and Hans Tompits. 2003. On Different Proof-Search Strategies for Orthologic. *Studia Logica* 73, 1 (Feb. 2003), 131–152. https://doi.org/10.1023/A:1022993408070

Tim Freeman and Frank Pfenning. 1991. Refinement Types for ML. *ACM SIGPLAN Notices* 26, 6 (May 1991), 268–277. https://doi.org/10.1145/113446.113468

Michael Gelfond and Vladimir Lifschitz. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing* 9, 3 (Aug. 1991), 365–385. https://doi.org/10.1007/BF03037169

G. Gentzen. 1935. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift* 39 (1935), 176–210.

R. I. Goldblatt. 1974. Semantic Analysis of Orthologic. *Journal of Philosophical Logic* 3, 1 (March 1974), 19–35. https://doi.org/10.1007/BF00652069

Simon Guilloud, Mario Bucev, Dragana Milovancevic, and Viktor Kuncak. 2023a. Formula Normalizations in Verification. In *35th International Conference on Computer Aided Verification (Lecture Notes in Computer Science)*. Springer, Paris, –.

Simon Guilloud, Sankalp Gambhir, and Viktor Kuncak. 2023b. LISA – A Modern Proof System. In *14th Conference on Interactive Theorem Proving (Leibniz International Proceedings in Informatics)*. Daghstuhl, Bialystok, 0.

Gary M. Hardegree. 1981. Material Implication in Orthomodular (and Boolean) Lattices. *Notre Dame Journal of Formal Logic* 22, 2 (April 1981), 163–182. https://doi.org/10.1305/ndjfl/1093883401

Wesley H. Holliday. 2023. A Fundamental Non-Classical Logic. *Logics* 1, 1 (March 2023), 36–79. https://doi.org/10.3390/logics1010004

Wesley H. Holliday and Matthew Mandelkern. 2022. The Orthologic of Epistemic Modals. https://doi.org/10.48550/ARXIV.2203.02872

Marek Hyčko. 2005. Implications and Equivalences in Orthomodular Lattices. *Demonstratio Mathematica [electronic only]* 38 (Oct. 2005), III–792. https://doi.org/10.1515/dema-2005-0402

Gudrun Kalmbach. 1983. *Orthomodular Lattices.* Academic Press Inc, London ; New York.

Tomoaki Kawano. 2018. Labeled Sequent Calculus for Orthologic. *Bulletin of the Section of Logic* 47, 4 (Dec. 2018), 217–232. https://doi.org/10.18778/0138-0680.47.4.01

Kenneth Kunen. 1987. Negation in Logic Programming. *The Journal of Logic Programming* 4, 4 (Dec. 1987), 289–308. https://doi.org/10.1016/0743-1066(87)90007-0

Olivier Laurent. 2016. Focusing in Orthologic. In *1st International Conference on Formal Structures for Computation and Deduction, FSCD 2016, June 22-26, 2016, Porto, Portugal (LIPIcs, Vol. 52)*, Delia Kesner and Brigitte Pientka (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Porto, Portugal, 25:1–25:17. https://doi.org/10.4230/LIPIcs.FSCD.2016.25

Magnus Madsen, Ming-Ho Yee, and Ondrej Lhoták. 2016. From Datalog to flix: a declarative language for fixed points on lattices. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2016, June 13-17, 2016*, Chandra Krintz and Emery D. Berger (Eds.). ACM, Santa Barbara, CA, USA, 194–208. https://doi.org/10.1145/2908080.2908096

William McCune. 1998. Automatic Proofs and Counterexamples for Some Ortholattice Identities. *Inf. Process. Lett.* 65, 6 (1998), 285–291. https://doi.org/10.1016/S0020-0190(98)00015-5

Andrea Meinander. 2010. A Solution of the Uniform Word Problem for Ortholattices. *Mathematical Structures in Computer Science* 20, 4 (Aug. 2010), 625–638. https://doi.org/10.1017/S0960129510000125

Michel Minoux. 1988. LTUR: A Simplified Linear-Time Unit Resolution Algorithm for Horn Formulae and Computer Implementation. *Inform. Process. Lett.* 29, 1 (Sept. 1988), 1–12. https://doi.org/10.1016/0020-0190(88)90124-X

Oded Padon, Giuliano Losa, Mooly Sagiv, and Sharon Shoham. 2017. Paxos made EPR: decidable reasoning about distributed protocols. *Proc. ACM Program. Lang.* 1, OOPSLA (2017), 108:1–108:31. https://doi.org/10.1145/3140568

Ruzica Piskac, Leonardo de Moura, and Nikolaj Bjørner. 2010. Deciding Effectively Propositional Logic Using DPLL and Substitution Sets. *Journal of Automated Reasoning* 44, 4 (April 2010), 401–424. https://doi.org/10.1007/s10817-009-9161-6

F. P. Ramsey. 1930. On a Problem of Formal Logic. *Proceedings of the London Mathematical Society* s2-30, 1 (1930), 264–286. https://doi.org/10.1112/plms/s2-30.1.264

J. P. Rawling and S. A. Selesnick. 2000. Orthologic and Quantum Logic: Models and Computational Elements. *J. ACM* 47, 4 (July 2000), 721–751. https://doi.org/10.1145/347476.347481

John Alan Robinson. 1965. A Machine-Oriented Logic Based on the Resolution Principle. *J. ACM* 12, 1 (1965), 23–41. https://doi.org/10.1145/321250.321253

John Alan Robinson and Andrei Voronkov (Eds.). 2001. *Handbook of Automated Reasoning (in 2 Volumes)*. Elsevier and MIT Press, MIT.

Jürgen Schulte Mönting. 1981. Cut Elimination and Word Problems for Varieties of Lattices. *Algebra Universalis* 12, 1 (Dec. 1981), 290–321. https://doi.org/10.1007/BF02483891

M.A.E.H. Sherif. 1997. Decision Problem for Orthomodular Lattices. *Algebra Universalis* 37, 1 (Jan. 1997), 70–76. https://doi.org/10.1007/PL00000328

Richard Statman. 1979. Intuitionistic Propositional Logic is Polynomial-Space Complete. *Theor. Comput. Sci.* 9 (1979), 67–72. https://doi.org/10.1016/0304-3975(79)90006-9

G. S. Tseitin. 1983. On the Complexity of Derivation in Propositional Calculus. In *Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970*, Jörg H. Siekmann and Graham Wrightson (Eds.). Springer, Berlin, Heidelberg, 466–483. https://doi.org/10.1007/978-3-642-81955-1_28

Jeffrey D. Ullman. 1988. *Principles of Database and Knowledge-Base Systems, Volume I.* Principles of Computer Science Series, Vol. 14. Computer Science Press, New Yor, United States.

Jeffrey D. Ullman. 1989. *Principles of Database and Knowledge-Base Systems, Volume II.* Computer Science Press, New Yor, United States.

Pawel Urzyczyn. 1997. Inhabitation in Typed Lambda-Calculi (A Syntactic Approach). In *Typed Lambda Calculi and Applications, Third International Conference on Typed Lambda Calculi and Applications, TLCA '97, April 2-4, 1997, Proceedings (Lecture Notes in Computer Science, Vol. 1210)*, Philippe de Groote (Ed.). Springer, Nancy, France, 373–389. https://doi.org/10.1007/3-540-62688-3_47

Niki Vazou, Eric L. Seidel, Ranjit Jhala, Dimitrios Vytiniotis, and Simon Peyton-Jones. 2014. Refinement Types for Haskell. In *Proceedings of the 19th ACM SIGPLAN International Conference on Functional Programming (ICFP '14)*. Association for Computing Machinery, New York, NY, USA, 269–282. https://doi.org/10.1145/2628136.2628161

Philip M. Whitman. 1941. Free Lattices. *Annals of Mathematics* 42, 1 (1941), 325–330. https://doi.org/10.2307/1969001 jstor:1969001