New Algebraic Normative Theories for Ethical and Legal Reasoning in the LogiKEy Framework

Ali Farjami
farjami110@gmail.com, University of Luxembourg

Abstract
In order to design and engineer ethical and legal reasoners and responsible systems, Benzmüller, Parent and van der Torre introduced the LogiKEy methodology, based on the semantical embedding of deontic logics into classic higher-order logic. This article considerably extends the LogiKEy deontic logics and dataset using an algebraic approach, and develops a theory of input/output operations for normative reasoning on top of Boolean algebras.

Keywords: LogiKEy framework; Normative reasoning

1. Introduction
Benzmüller, Parent and van der Torre [1] introduced the LogiKEy framework for the formalization and automation of new ethical reasoners, normative theories and deontic logics. The LogiKEy framework uses higher-order logic (HOL) as a metalogic to embed other logics. A logic embedded in HOL can thus be tracked by automated theorem provers (ATP), interactive automated provers (ITP) and HOL model finders. The LogiKEy methodology allows a user to simultaneously combine and experiment with underlying logics (and their combinations), ethico-legal domain theories, and concrete examples.

Earlier work presented semantical embedding of two traditions in deontic logic in the LogiKEy framework, namely Åqvist’s dyadic deontic logic $E$ [2] and Makinson and van der Torre’s input/output (I/O) logic [3]. Subsequent work provided the Isabelle/HOL dataset for the LogiKEy workbench [4]. This article considerably extends the LogiKEy deontic logics and dataset using an algebraic approach. In particular, it extends the theory of input/output operations [3] and corresponding proof systems on top of Boolean
algebras and, more generally, abstract logics [5].

The article is structured as follows: Section 2 and 3 provide the soundness and completeness results of I/O operations for deriving permissions and obligations on top of Boolean algebras. Section 4 shows how I/O operations can be generalized over any abstract logic. Section 5 integrates a conditional theory into input/output logic. Section 6 introduces semantical embedding of I/O logic into HOL, including soundness and completeness (faithfulness). Section 7 discusses related work, and Section 8 concludes the article. Appendix A shows how the semantical embedding described in Section 6 is implemented in the Isabelle/HOL proof assistant. Some experiments are provided to show that this logic implementation enables interactive and automated reasoning. All the proofs are in the appendices. Appendix B provides proofs relating to Sections 2 and 3, Appendix C proofs relating to Section 5 and Appendix D proofs relating to Section 6.

2. Permissive norms: input/output operations

Deontic logic investigates logical relations among normative concepts [6]. Input/output (I/O) logic was initially introduced by Makinson and van der Torre [7] to study conditional norms viewed as relations between logical formulas [7]. In this setting, the meaning of normative concepts is given in terms of a set of procedures yielding outputs for inputs. Suppose that $N^O$ denotes a set of obligatory norms and $N^P$ a set of permissive norms. The formula $(a, x) \in N^O$ means “given $a$, it is obligatory that $x$”, while the formula $(a, x) \in N^P$ means “given $a$, it is permitted that $x$.” The formula $x \in out(N^P, A)$ means “given normative system $N^P$ and input set $A$ (state of affairs), $x$ (permission) is in the output”. The output operations resemble inferences, where inputs need not be included among outputs, and outputs need not be reusable as inputs [7]. The proof system of an I/O logic is specified via a number of derivation rules acting on pairs $(a, x)$ of formulas. Given a set $N$ of pairs, $(a, x) \in deriv_1(N)$ is written to say that $(a, x)$ can be derived from $N$ using these rules. The term “input/output logic” is used broadly to refer to a family of related systems such as simple-minded, basic, and reusable systems [8,7]. This section uses similar terminology, and introduces some input/output systems for deriving permissions on top of Boolean algebras. Each derivation system is closed under a set of rules, including for instance the weakening of the output (WO) rule or the strengthening of the input (SI) rule. A bottom-up approach is used to characterize different derivation
systems. The AND rule, for the output, is absent in the derivation systems presented in this section.

**Definition 1 (Boolean algebra).** A structure $\mathcal{B} = \langle B, \wedge, \lor, \neg, 0, 1 \rangle$ is a Boolean algebra if and only if it satisfies the following identities:

- $x \lor y \approx y \lor x$, $x \land y \approx y \land x$
- $x \lor (y \lor z) \approx (x \lor y) \lor z$, $x \land (y \land z) \approx (x \land y) \land z$
- $x \lor 0 \approx x$, $x \land 1 \approx x$
- $x \lor \neg x \approx 1$, $x \land \neg x \approx 0$
- $x \lor (y \land z) \approx (x \lor y) \land (x \lor z)$, $x \land (y \lor z) \approx (x \land y) \lor (x \land z)$

**Definition 2 (Syntax).** For a set of variables $X$, the set of Boolean terms is denoted over $X$ by $\text{Ter}(X)$ as follows:

$$\text{Ter}(X) = \bigcup_{n \in \mathbb{N}} \text{Ter}_n(X)$$

where

- $\text{Ter}_0(X) = X \cup \{0, 1\}$
- $\text{Ter}_{n+1}(X) = \text{Ter}_n(X) \cup \{a \land b, a \lor b, \neg a : a, b \in \text{Ter}_n(X)\}$.

Given a Boolean algebra $\mathcal{B}$, the elements of $\text{Ter}(\mathcal{B})$ are ordered as $a \leq b$ iff $a \land b = a$. Since $\leq$ is antisymmetric, $a \leq b$ and $b \leq a$ imply $a = b$.

**Definition 3 (Upward-closed set).** Given a Boolean algebra $\mathcal{B}$, a set $A \subseteq \text{Ter}(\mathcal{B})$ is called upward-closed if it satisfies the following property:

For all $x, y \in \text{Ter}(\mathcal{B})$, if $x \leq y$ and $x \in A$, then $y \in A$.

The least upward-closed set that includes $A$ by $\text{Up}(A)$ is denoted. The $\text{Up}$ operator satisfies the following properties:

- $A \subseteq \text{Up}(A)$ (Inclusion)
- $A \subseteq B \Rightarrow \text{Up}(A) \subseteq \text{Up}(B)$ (Monotony)

---

1. An equation $t \approx t'$ holds in an algebra $\mathcal{A}$ if its universal closure $\forall x_0 \ldots x_n : t \approx t'$ is a sentence that is true in $\mathcal{A}$.
2. The symbol “=” is used to express that both sides name the same object. The elements of the variable set $(B)$ that are represented by different letters are supposed to be independent in the algebra $(B)$ w.r.t $\leq$. 

3
• $Up(A) = Up(Up(A))$  \hspace{1cm} \textit{(Idempotence)}

An operator that satisfies these properties is called a closure operator.

The “$Up$” operator, for a given set $A$, sees all the elements that are in a higher or equal position to the elements of $A$ in terms of their usual ordering in lattices. Unlike the propositional logic consequence relation (“$Cn$”) operator, the “$Up$” operator is not closed under conjunction so that we do not have $a \land \neg a \in Up(a, \neg a)$.

**Definition 4 (Semantics).** In input/output logic, the main semantic construct for normative propositions is the output operation, which represents the set of normative propositions related to a normative system $N$, regarding state of affairs $A$, namely $out(N, A)$. Normative system $N$ denotes a set of norms $(a, x)$ in which the body and head are Boolean terms. Let $N(A) = \{ x \mid (a, x) \in N \text{ for some } a \in A \}$. In a Boolean algebra $B$ for $X \subseteq \text{Ter}(B)$, the equation $Eq(X) = \{ x \in \text{Ter}(B) \mid \exists y \in X, x = y \}$ is defined. Given a Boolean algebra $B$, a normative system $N \subseteq \text{Ter}(B) \times \text{Ter}(B)$ and an input set $A \subseteq \text{Ter}(B)$, I/O Boolean operations are defined as follows:

**Zero Boolean I/O operation:**

$$out^B_0(N, A) = Eq(N(Eq(A)))$$

$$out^B_R(N, A) = Eq(N(A)) \quad out^B_I(N, A) = N(Eq(A))$$

**Simple-I Boolean I/O operation:**

$$out^B_I(N, A) = Eq(N(Up(A)))$$

**Simple-II Boolean I/O operation:**

$$out^B_{II}(N, A) = Up(N(Eq(A)))$$

**Simple-minded Boolean I/O operation:**

$$out^B_1(N, A) = Up(N(Up(A)))$$

\footnote{Sometimes $Up(a, b, \ldots)(Eq(a, b, \ldots))$ is written instead of $Up\{a, b, \ldots\}(Eq\{a, b, \ldots\})$ and $out(N, a)$ (derive($N$, $a$)) is written instead of $out(N, \{a\})$ (derive($N$, $\{a\}$)).}
Basic Boolean I/O operation:

\[ \text{out}_2^B(N, A) = \bigcap \{ \text{Up}(N(V)), A \subseteq V, \text{V is saturated} \} \]

Reusable Boolean I/O operation:

\[ \text{out}_3^B(N, A) = \bigcap \{ \text{Up}(N(V)), A \subseteq V = \text{Up}(V) \supseteq N(V) \} \]

Put \( \text{out}_1^B(N) = \{(A, x) : x \in \text{out}_1^B(N, A) \} \).

We turn to the proof theory. A derivation of a pair \((a, x)\) from \(N\), given a set \(X\) of rules, is understood to be a tree with \((a, x)\) at the root, each non-leaf node related to its immediate parents by the inverse of a rule in \(X\), and each leaf node an element of \(N\).

**Definition 5 (Proof system).** Given a Boolean algebra \(B\) and a normative system \(N \subseteq \text{Ter}(B) \times \text{Ter}(B)\), it is defined that \((a, x) \in \text{derive}_i^B(N)\) if and only if \((a, x)\) is derivable from \(N\) using \(\text{EQI}, \text{EQO}, \text{SI}, \text{WO}, \text{OR}, \text{T}\) as follows:

| Rule Set | Rule       | Derivation Example |
|----------|------------|--------------------|
| \(\text{derive}_i^B\) | \{EQO\} | \(x = y \quad (a, x) \quad (a, y)\) |
| \(\text{derive}_2^B\) | \{EQI\} | \(a = b \quad (a, x) \quad (b, x)\) |
| \(\text{derive}_0^B\) | \{EQI, EQO\} | \(a = b \quad (a, x) \quad (b, x)\) |
| \(\text{derive}_1^B\) | \{SI, EQO\} | \(b \leq a \quad (a, x) \quad (b, x)\) |
| \(\text{derive}_1^B\) | \{WO, EQI\} | \(a \vee b \quad (a, x) \quad (b, x)\) |
| \(\text{derive}_2^B\) | \{SI, WO, OR\} | \(x \leq y \quad (a, x) \quad (a, y)\) |
| \(\text{derive}_3^B\) | \{SI, WO, T\} | \(a \leq x \quad (a, x) \quad (a, y)\) |

Given a set of \(A \subseteq \text{Ter}(B)\), then \((A, x) \in \text{derive}_i^B(N)\) whenever \((a, x) \in \text{derive}_i^B(N)\) for some \(a \in A\). Put \(\text{derive}_i^B(N, A) = \{x : (A, x) \in \text{derive}_i^B(N)\}\).

4 A set \(V\) is saturated in a Boolean algebra \(B\) iff
- If \(a \in V\) and \(b \geq a\), then \(b \in V\);
- If \(a \lor b \in V\), then \(a \in V\) or \(b \in V\).

5 EQI stands for equivalence of the input, EQO for equivalence of the output, and T for transitivity.

6 In the work of Makinson and van der Torre, it is for some conjunction \(a\) of elements in \(A\).
Theorem 1 (Soundness and completeness). $\text{out}^B_i(N) = \text{derive}^B_i(N)$.

Example 1. For the conditionals $N = \{(1, g), (g, t)\}$ and the input set $A = \{\}$ we have $\text{out}^B_i(N, A) = \{\}$, and for the input set $C = \{g\}$ we have $\text{out}^B_i(N, C) = U_p(t)$.

Example 2. For the conditionals $N = \{(1, g), (g, t), (\neg g, \neg t), (a, b)\}$ and the input set $A = \{\neg g\}$ we have $\text{out}^B_3(N, A) = U_p(g, t, \neg t)$.

3. Obligatory norms: input/output operations

This section adds the AND and cumulative transitivity (CT) rules to the derivation systems introduced, with the aim of rebuilding the derivation systems introduced by Makinson and van der Torre [7] for deriving obligations.

Definition 6 (Proof system). Given a Boolean algebra $B$ and a normative system $N \subseteq \text{Ter}(B) \times \text{Ter}(B)$, it is defined that $(a, x) \in \text{derive}^X_i(N)$ if and only if $(a, x)$ is derivable from $N$ using EQO, EQI, SI, WO, OR, AND, CT as follows:

| derive$^X_i$ | Rules |
|-------------|-------|
| derive$^{AND}_{II}$ | $\{\text{WO}, \text{EQI}, \text{AND}\}$ |
| derive$^{AND}_I$ | $\{\text{SI}, \text{WO}, \text{AND}\}$ |
| derive$^{AND}_{II}$ | $\{\text{SI}, \text{WO}, \text{OR}, \text{AND}\}$ |
| derive$^{CT}_I$ | $\{\text{SI}, \text{EQO}, \text{CT}\}$ |
| derive$^{CT}_I$ | $\{\text{WO}, \text{EQI}, \text{CT}\}$ |
| derive$^{CT}_I$ | $\{\text{SI}, \text{WO}, \text{CT}\}$ |
| derive$^{CT,AND}_I$ | $\{\text{SI}, \text{WO}, \text{CT}, \text{AND}\}$ |

Given a set of $A \subseteq \text{Ter}(B)$, $(A, x) \in \text{derive}^X_i(N)$ whenever $(a, x) \in \text{derive}^X_i(N)$ for some $a \in A$. Put $\text{derive}^X_i(N, A) = \{x : (A, x) \in \text{derive}^X_i(N)\}$.

Makinson and van der Torre [7] noticed that in some cases, the order of application of two derivation rules is reversible. For instance, any application of AND followed by WO (SI) may be replaced by one in which WO (SI) is followed by AND. Based on this observation, new output operations are defined by, for example, rearranging the derivation $(a, x)$ in the proof system $\{\text{SI}, \text{WO}, \text{AND}\}$ such that the AND rule applies only at the end. The $\{\text{SI}, \text{WO}\}$ system has been characterized as the simple-minded
I/O operation $\text{out}^B$. Now by using (finite) iterations of AND on top of $\text{out}_1^B$, a new output operation is defined that characterizes the proof system \{SI, WO, AND\}. Three kinds of such output operations are defined—$\text{out}_1^{\text{AND}}$, $\text{out}_1^{\text{CT}}$, and $\text{out}_1^{\text{CT,AND}}$—that can characterize the proof systems introduced in Definition 6. Note that there are some non-reversible orders, such as the WO rule followed by OR rule, for which no transformation appears to be available.

**Definition 7 (Semantics $\text{out}_i^{\text{AND}}$).** Given a Boolean algebra $B$, a normative system $N \subseteq \text{Ter}(B) \times \text{Ter}(B)$ and an input set $A \subseteq \text{Ter}(B)$, the AND operation is defined as follows:

\[
\begin{align*}
\text{out}_i^{\text{AND}}(N, A) &= \text{out}_i^B(N, A) \\
\text{out}_i^{\text{AND}}(N, A) &= \text{out}_i^{\text{AND}}(N, A) \\
\text{out}_i^{\text{AND}}(N, A) &= \bigcup_{n \in N} \text{out}_i^{\text{AND}}(N, A)
\end{align*}
\]

Put $\text{out}_i^{\text{AND}}(N) = \{(A, x) : x \in \text{out}_i^{\text{AND}}(N, A)\}$.

**Definition 8 (Semantics $\text{out}_i^{\text{CT}}$).** Given a Boolean algebra $B$, a normative system $N \subseteq \text{Ter}(B) \times \text{Ter}(B)$ and an input set $A \subseteq \text{Ter}(B)$, the CT operation is defined as follows:

\[
\begin{align*}
\text{out}_i^{\text{CT}}(N, A) &= \text{out}_i^B(N, A) \\
\text{out}_i^{\text{CT}}(N, A) &= \text{out}_i^{\text{CT}}(N, A) \\
\text{out}_i^{\text{CT}}(N, A) &= \bigcup_{n \in N} \text{out}_i^{\text{CT}}(N, A)
\end{align*}
\]

Put $\text{out}_i^{\text{CT}}(N) = \{(A, x) : x \in \text{out}_i^{\text{CT}}(N, A)\}$.

**Definition 9 (Semantics $\text{out}_i^{\text{CT,AND}}$).** Given a Boolean algebra $B$, a normative system $N \subseteq \text{Ter}(B) \times \text{Ter}(B)$ and an input set $A \subseteq \text{Ter}(B)$, the CT,AND operation is defined as follows:

\[
\begin{align*}
\text{out}_i^{\text{CT,AND}}(N, A) &= \text{out}_i^{\text{CT}}(N, A) \\
\text{out}_i^{\text{CT,AND}}(N, A) &= \text{out}_i^{\text{CT,AND}}(N, A) \\
\text{out}_i^{\text{CT,AND}}(N, A) &= \bigcup_{n \in N} \text{out}_i^{\text{CT,AND}}(N, A)
\end{align*}
\]

Put $\text{out}_i^{\text{CT,AND}}(N) = \{(A, x) : x \in \text{out}_i^{\text{CT,AND}}(N, A)\}$.
Theorem 2. Given a Boolean algebra \( B \), for every normative system \( N \subseteq \text{Ter}(B) \times \text{Ter}(B) \) we have \( \text{out}^{\text{AND}}_{i}(N) = \text{derive}^{\text{AND}}_{i}(N), \; i \in \{II, 1, 2\} \);
\( \text{out}^{\text{CT}}_{i}(N) = \text{derive}^{\text{CT}}_{i}(N), \; i \in \{I, II, 1\} \); and \( \text{out}^{\text{CT}, \text{AND}}_{1}(N) = \text{derive}^{\text{CT}, \text{AND}}_{1}(N) \).

Similarly, it is possible to define the \( \text{out}^{\text{OR}}_{i}(N) \) operation and characterize some other proof systems:

| \( \text{derive}^{\text{AND}}_{i} \) | Rules |
|---|---|
| \( \text{derive}^{\text{OR}}_{I} \) | \{SI, EQO, OR\} |
| \( \text{derive}^{\text{CT}, \text{OR}}_{I} \) | \{SI, EQO, CT, OR\} |
| \( \text{derive}^{\text{CT}, \text{OR}}_{1} \) | \{SI, WO, CT, OR\} |
| \( \text{derive}^{\text{CT}, \text{OR}, \text{AND}}_{1} \) | \{SI, WO, CT, OR, AND\} |

Four systems, namely \( \text{derive}^{\text{AND}}_{1}, \text{derive}^{\text{AND}}_{2} \) (or \( \text{derive}^{\text{OR}, \text{AND}}_{1} \)), \( \text{derive}^{\text{CT}, \text{AND}}_{1} \) and \( \text{derive}^{\text{CT}, \text{OR}, \text{AND}}_{1} \), are introduced by Makinson and van der Torre [7] for reasoning about obligatory norms.

4. I/O mechanism over abstract logics

An abstract logic [5] is a pair \( \mathcal{A} = \langle \mathcal{L}, C \rangle \) where \( \mathcal{L} = \langle L, ... \rangle \) is an algebra and \( C \) is a closure operator, defined on the power set of its universe, that means that for all \( A, B \subseteq L \):

- \( A \subseteq C(A) \)
- \( A \subseteq B \rightarrow C(A) \subseteq C(B) \)
- \( C(A) = C(C(A)) \)

The elements of an abstract logic can be ordered as \( a \leq b \) if and only if \( b \in C(\{a\}) \). Without loss of generality, the algebra of formulas (or terms in the algebraic context) is used where \( \text{Fm}(X) = \langle Fm(X), ... \rangle \) for a set of fixed variables \( X \). Similar to Boolean algebras, the \( \text{Eq} \) and \( \text{Up} \) operators can be defined for \( A \subseteq \text{Fm}(X) \).

Definition 10 (Semantics). Given an abstract logic \( \mathcal{A} = \langle \text{Fm}(X), C \rangle \), a normative system \( N \subseteq \text{Fm}(X) \times \text{Fm}(X) \) and an input set \( A \subseteq \text{Fm}(X) \), the I/O operations are defined as follows:

- \( \text{out}^{A}_{0}(N, A) = \text{Eq}(N(\text{Eq}(A))) \)
can be defined.

we have $M \subseteq \text{it can be defined that}$

out

is

derive

Theorem 3 (Soundness and completeness).

Moreover, other rules such as AND and CT can be added to the systems

in the same way as in Section

Example 3. In a modal logic system KT, for the conditionals $N = \{(p, \Box q), (q, r), (s, t)\}$ and input set $A = \{p\}$, we have $\text{out}^\text{KT}_i(N, A) = \text{Up}(\Box q, r)$.

Put $\text{out}^A_i(N) = \{x : (A, x) \in \text{derive}^A_i(N)\}$.

Theorem 3 (Soundness and completeness). $\text{out}^A_i(N) = \text{derive}^A_i(N)$.

A logical system $L = \langle L, \vdash_L \rangle$ straightforwardly provides an equivalent abstract logic $\langle \text{Fm}(L), C_{\vdash_L} \rangle$. Therefore, an I/O framework can be built over different types of logics including first-order logic, simple type theory, description logic, as well as different kinds of modal logics that are expressive for intentional concepts such as belief and time.

Example 3. In a modal logic system KT, for the conditionals $N = \{(p, \Box q), (q, r), (s, t)\}$ and input set $A = \{p\}$, we have $\text{out}^\text{KT}_i(N, A) = \text{Up}(\Box q, r)$.

Moreover, other rules such as AND and CT can be added to the systems in the same way as in Section

Theorem 4. Every $\text{out}^B_i$, and $\text{out}^A_i$ operation is a closure operator.

Nested I/O operations can be defined. Since $\text{out}^B_i$ is a closure operator, it can be defined that $\text{out}^A_j(M, \text{out}^B_i(N))$ where $N \subseteq \text{Ter}(B) \times \text{Ter}(B)$, $M \subseteq (\text{Ter}(B) \times \text{Ter}(B)) \times (\text{Ter}(B) \times \text{Ter}(B))$ and in the abstract logic $A$ we have $L = N \times N$ and $C = \text{out}^B_i$. The corresponding characterization is $\text{derive}^A_j(M, \text{derive}^B_i(N))$. Similarly, nested $\text{out}^A_j(M, \text{out}^A_i(N))$ operations can be defined.

---

8 For this case, the abstract logic $A = \langle \text{Fm}(X), C \rangle$ should include $\lor$, that is a binary operation symbol, either primitive or defined by a term, and we then have $a \lor b, b \lor a \in C\{a\}$ ($\lor$-Introduction) and if $c \in C\{a\} \cap C\{b\}$ then $c \in C(a \lor b), C(b \lor a)$ ($\lor$-Elimination).
5. Synthesizing normative reasoning and preferences

Input/output logic originally developed on top of classical propositional logic [7]. This section shows that the extension of propositional logic with a set of conditional norms is sound and complete with respect to the class of Boolean algebras that the corresponding I/O operation holds. The language of classical propositional logic consists of the connectives $\mathcal{L}_C = \{\land, \lor, \neg, \top, \bot\}$. Let $X$ be a set of variables; as usual the set of formulas is defined over $X$ and referred to as $Fm(X)$. The algebra of formulas over $X$ is a Boolean algebra as follows:

$Fm(X) = \langle Fm(X), \land_{Fm(X)}, \lor_{Fm(X)}, \neg_{Fm(X)}, \top_{Fm(X)}, \bot_{Fm(X)} \rangle$

where $\land_{Fm(X)}(\phi, \psi) = (\phi \land \psi)$, $\lor_{Fm(X)}(\phi, \psi) = (\phi \lor \psi)$, $\neg_{Fm(X)}(\phi) = \neg \phi$, $\top_{Fm(X)} = \top$, and $\bot_{Fm(X)} = \bot$. It is defined that $\phi \vDash_C \psi$ if and only if $\phi \leq \psi$, and $\phi \vDash_C \phi$ if and only if $\varphi \leq \psi$ and $\psi \leq \varphi$.

**Definition 12.** Let $N \subseteq Fm(X) \times Fm(X)$ where $X$ is a set of propositional variables. The formula $(\phi, \psi) \in \text{derive}_i^Fm(X)(N)$ applies if and only if $(\phi, \psi)$ is derivable from $N$ using $EQO, EQI, SI, WO, OR, T$ as follows:

| derive$^i_{Fm(X)}$ | Rules | $(\phi, \psi)$ | $\psi \vDash_C \phi$ | $(\psi, \phi)$ | $\phi \vDash_C \phi$ | $(\varphi, \psi)$ | $\psi \vDash_C \phi$ |
|---------------------|-------|----------------|---------------------|----------------|---------------------|----------------|---------------------|
| derive$^R_{Fm(X)}$  | $\{EQO\}$ | $\text{EQO}$ | $\psi \vDash_C \phi$ | $(\psi, \phi)$ | $\phi \vDash_C \phi$ | $\text{EQO}$ | $\varphi \vDash_C \phi$ | $(\varphi, \psi)$ | $\psi \vDash_C \phi$ |
| derive$^0_{Fm(X)}$  | $\{EQI\}$ | $\text{EQI}$ | $\varphi \vDash_C \phi$ | $(\varphi, \psi)$ | $\psi \vDash_C \phi$ | $\text{EQI}$ | $\phi \vDash_C \phi$ | $(\phi, \psi)$ | $\psi \vDash_C \phi$ |
| derive$^1_{Fm(X)}$  | $\{SI, EQO\}$ | $\text{SI}, EQO$ | $\varphi \vDash_C \phi$ | $(\varphi, \psi)$ | $\psi \vDash_C \phi$ | $\text{SI}, EQO$ | $\phi \vDash_C \phi$ | $(\phi, \psi)$ | $\psi \vDash_C \phi$ |
| derive$^2_{Fm(X)}$  | $\{WO, EQI\}$ | $\text{WO}, EQI$ | $\phi \vDash_C \phi$ | $(\phi, \psi)$ | $\psi \vDash_C \phi$ | $\text{WO}, EQI$ | $\psi \vDash_C \phi$ | $(\varphi, \psi)$ | $\psi \vDash_C \phi$ |
| derive$^3_{Fm(X)}$  | $\{SI, WO, OR\}$ | $\text{SI}, WO, OR$ | $\psi \vDash_C \phi$ | $(\psi, \phi)$ | $\varphi \vDash_C \phi$ | $\text{SI}, WO, OR$ | $\phi \vDash_C \phi$ | $(\phi, \psi)$ | $\psi \vDash_C \phi$ |

It is defined that $(\Gamma, \psi) \in \text{derive}_i^Fm(X)(N)$ if $(\phi, \psi) \in \text{derive}_i^Fm(X)(N)$ for some $\phi \in \Gamma \subseteq Fm(X)$. Put $\text{derive}_i^Fm(X)(N, \Gamma) = \{\psi : (\Gamma, \psi) \in \text{derive}_i^Fm(X)(N)\}$.

---

9For the precise definition, the auxiliary symbols brackets $(, )$ are used. Apart from the use of brackets, the formulas over $X$ are Boolean terms over $X$: $\text{Ter}(X)$.
Example 4. For the conditional norm set $N = \{ (\top, g), (g, t), (\neg g, \neg t) \}$ and input set $A = \{ \neg g \}$, we have $\text{out}_{3}^{\text{Fm}(X)}(N, A) = \text{Up}(t, \neg t, g)$.

Given $(\text{Fm}(X), \vdash_{C})$, let $\mathcal{B}$ be a Boolean algebra and $X$ be a set of propositional variables. A valuation on $\mathcal{B}$ is a function from $X$ into the universe of $\mathcal{B}$. Any valuation on $\mathcal{B}$ can be extended in a unique way to a homomorphism from the algebra $\text{Fm}(X)$ into $\mathcal{B}$. A valuation $V$ on $\mathcal{B}$ satisfies a formula if $V(\varphi) = 1_{B}$, and it satisfies a set of formulas if it satisfies all its elements [9].

Definition 13. For any Boolean algebra $\mathcal{B}$, the consequence relation $\models_{\mathcal{B}}$ can be defined as follows:

$\Gamma \models_{\mathcal{B}} \varphi$ if and only if for any valuation on $\mathcal{B}$ that $V(\Gamma) = 1_{B}$ then $V(\varphi) = 1_{B}$.

Definition 14. Let $\text{BA}$ be the class of all Boolean algebras. The consequence relation $\models_{\text{BA}}$ can be defined as follows:

$\Gamma \models_{\text{BA}} \varphi$ if and only if for any Boolean algebra $\mathcal{B}$, $\Gamma \vdash_{\mathcal{B}} \varphi$.

Theorem 5. For every set of formulas $\Gamma$ and every formula $\varphi$,

$\Gamma \models_{\text{BA}} \varphi$ if and only if $\Gamma \vdash_{C} \varphi$.

Theorem 6. Let $X$ be a set of propositional variables and $N \subseteq \text{Fm}(X) \times \text{Fm}(X)$. For a given Boolean algebra $\mathcal{B}$ and a valuation $V$ on $\mathcal{B}$, it is defined that $N^V = \{(V(\varphi), V(\psi)) | (\varphi, \psi) \in N\}$. We have

$(\varphi, \psi) \in \text{derive}_{i}^{\text{Fm}(X)}(N)$

if and only if

$V(\psi) \in \text{out}_{i}^{\mathcal{B}}(N^V, \{V(\varphi)\})$ for every $\mathcal{B} \in \text{BA}$ and valuation $V$.

The theorem can be extended for arbitrary input set $\Gamma \subseteq \text{Fm}(X)$. Suppose that $(\Gamma, \psi) \in \text{derive}_{i}^{\text{Fm}(X)}(N)$, then $(\varphi, \psi) \in \text{derive}_{i}^{\text{Fm}(X)}(N)$ for $\varphi \in \Gamma$. As above, we have $V(\psi) \in \text{out}_{i}^{\mathcal{B}}(N^V, \{V(\varphi)\})$ for every $\mathcal{B} \in \text{BA}$ and valuation $V$, so that by definition of $\text{out}_{i}^{\mathcal{B}}$, it can be said that $V(\psi) \in \text{out}_{i}^{\mathcal{B}}(N^V, \{V(\varphi) | V(\varphi) \in V(\Gamma)\})$ for every $\mathcal{B} \in \text{BA}$ and valuation $V$. 

11
Theorem 7. Let $X$ be a set of propositional variables and $N \subseteq \text{Fm}(X) \times \text{Fm}(X)$. For a given Boolean algebra $\mathcal{B}$ and a valuation $V$ on $\mathcal{B}$, it is defined that $N^V = \{(V(\varphi), V(\psi)) | (\varphi, \psi) \in N\}$. We have

\[
(\varphi, \psi) \in \text{derive}_{i^{\text{AND}}}(N)
\]

if and only if

\[
V(\psi) \in \text{out}_{i^{\text{AND}}}(N^V, \{V(\varphi)\}) \text{ for every } \mathcal{B} \in \text{BA} \text{ and valuation } V.
\]

5.1. Consistency check

Constraints can be added to the derivation systems such that the output set of formulas is consistent with the proposed constraint.

Definition 15. Let $X$ be a set of propositional variables and $N \subseteq \text{Fm}(X) \times \text{Fm}(X)$. Given the constraint $\text{Con}$ that is a set of formulas $\text{Con} \subseteq \text{Fm}(X)$, it is defined that $(\varphi, \psi) \in \text{derive}_{i^{\text{Con}}}(N)$ if and only if

\[
(\varphi, \psi) \in \text{derive}_{i^{\text{Fm}(X)}}(N) \text{ and } \text{Con}, \psi \nvdash \bot.
\]

Given a set of $\Gamma \subseteq \text{Fm}(X)$, it is defined that $(\Gamma, \psi) \in \text{derive}_{i^{\text{Con}}}(N)$ if $(\varphi, \psi) \in \text{derive}_{i^{\text{Con}}}(N)$ for some $\varphi \in \Gamma$.

Theorem 8. Let $X$ be a set of propositional variables, $N \subseteq \text{Fm}(X) \times \text{Fm}(X)$, and $\text{Con} \subseteq \text{Fm}(X)$. For a given Boolean algebra $\mathcal{B}$ and a valuation $V$ on $\mathcal{B}$, it is defined that $N^V = \{(V(\varphi), V(\psi)) | (\varphi, \psi) \in N\}$. We have

\[
(\varphi, \psi) \in \text{derive}_{i^{\text{Con}}}(N)
\]

if and only if

\[
V(\psi) \in \text{out}_{i^{\text{Con}}}(N^V, \{V(\varphi)\}) \text{ for every } \mathcal{B} \in \text{BA} \text{ and valuation } V
\]

and

for some $\mathcal{B} \in \text{BA}$, there is a valuation $V$ such that

\[
\forall \delta \in \text{Con}, V(\delta \land \psi) = 1_{\mathcal{B}}.
\]
5.2. Integrating preferences

In the normative systems proposed, it is possible to add a preference relation over the set of valuations and define a new conditional theory. Conditional obligation sentences are analyzed which have the form $a > \bigcirc x$, where $>$ is a (preferential) conditional connective \[10\] \[11\]. Given the set of obligatory norms $N^O$, and suppose that $(a,x) \in N^O$, the new conditionals are defined as follows:

\[ a > \bigcirc x \text{ holds iff } (a,x) \in \text{derive}_i(N^O) \text{ and } a > x \text{ holds } \]

where $\text{derive}_i(N^O)$ is an appropriate derivation system for obligation. Intuitively, the modal translation of $a \to \bigcirc x$ for $(a,x) \in \text{derive}_i(N^O)$ \[7\] \[8\] is considered. This makes the definition a compositional definition of monadic obligation operators and conditionals.

For a given set of permissive norms $N^P$, by choosing a plausible derivation system for permission—$\text{derive}_i(N^P)$—that is similar to the definition of a conditional obligation, then conditional permission can be defined as

\[ a > Px \text{ holds iff } (a,x) \in \text{derive}_i(N^P) \text{ and } \neg(a > \neg x) \text{ holds } \]

where $\neg(a > \neg x)$ is the conditional dual of $a > x$. The set of new conditional obligations is denoted by $\text{derive}_i^O$ and the set of new conditional permissions is denoted by $\text{derive}_i^P$. This article does not refer to the subscripts or superscripts of the normative system or derivation systems where they are clear from the context or do not affect our discussion.

**Definition 16.** Let $X$ be a set of propositional variables and $\text{MaxC}$ the set of all the maximal consistent subsets of $\text{Fm}(X)$. Let $f \subseteq \text{MaxC} \times \text{MaxC}$ be a relation over elements of $\text{MaxC}$ and $\text{opt}_f(\varphi) = \{M \in \text{MaxC} \mid \varphi \in M, \forall K(\varphi \in K \to (M,K) \in f)\}$. It is defined that $\varphi > \bigcirc \psi \in \text{derive}_i^O(N)$ if and only if

\[ (\varphi, \psi) \in \text{derive}_i^{\text{Fm}(X)}(N) \text{ and } \forall M \in \text{opt}_f(\varphi) (\psi \in M). \]

Given a set of $\Gamma \subseteq \text{Fm}(X)$, it is defined that $\Gamma > \bigcirc \psi \in \text{derive}_i^O(N)$ if $\varphi > \bigcirc \psi \in \text{derive}_i^O(N)$ for some $\varphi \in \Gamma$.

**Definition 17.** Let $X$ be a set of propositional variables and $f \subseteq \text{MaxC} \times \text{MaxC}$. A preference Boolean algebra for $\text{Fm}(X)$ is a structure $M = \langle \mathcal{B}, \mathcal{V}, \succeq_f \rangle$ where:
B is a Boolean algebra,
• \( V = \{ V_i \}_{i \in I} \) is the set of valuations from \( \text{Fm}(X) \) on \( B \),
• \( \succeq_f \subseteq V \times V: \succeq_f \) is a betterness or comparative goodness relation over valuations from \( \text{Fm}(X) \) to \( B \) such that \( V_i \succeq_f V_j \) iff 
  \( \{ \varphi | V_i(\varphi) = 1_B \}, \{ \psi | V_j(\psi) = 1_B \} \) \( \in f \).

No specific properties (like reflexivity or transitivity) are considered for the betterness relation. For a given preference Boolean algebra \( M = \langle B, V, \succeq_f \rangle \), it is defined that
\[
\text{opt}_{\succeq_f}(\varphi) = \{ V_i \in V | V_i(\varphi) = 1_B, \forall V_j(V_j(\varphi) = 1_B \rightarrow V_i \succeq_f V_j) \}.
\]

**Theorem 9.** Let \( X \) be a set of propositional variables, where \( N \subseteq \text{Fm}(X) \times \text{Fm}(X) \), and \( f \subseteq \text{MaxC} \times \text{MaxC} \). For a given Boolean algebra \( B \) and a valuation \( V \) on \( B \), it is defined that \( N^V = \{(V(\varphi), V(\psi)) | (\varphi, \psi) \in N \} \). We have
\[
\varphi > \bigcirc \psi \in \text{derive}_{OH}(N)
\]
if and only if
\[
V(\psi) \in \text{out}_i^B(N^V, \{ V(\varphi) \}) \text{ for every } B \in \text{BA and valuation } V,
\]
and
for every preference Boolean algebra \( M = \langle B, V, \succeq_f \rangle \),
for every valuation \( V_i \in \text{opt}_{\succeq_f}(\varphi) \),
it is the case that \( V_i(\psi) = 1_B \).

The theorem can also be rewritten as follows:\[10\]
\[
\varphi > \bigcirc \psi \in \text{derive}_{OH}(N)
\]
if and only if
\[
\psi \in \text{out}_i^{\text{Fm}(X)}(N, \{ \varphi \}) \text{ and in } M = \langle 2, V, \succeq_f \rangle,
\]
for every valuation \( V_i \in \text{opt}_{\succeq_f}(\varphi) \), we have \( V_i(\psi) = 1_B \).

---

\[10\] If \( V_i \in \text{opt}_{\succeq_f}(\varphi) \) in \( M = \langle 2, V, \succeq_f \rangle \), then we have \( V_i \in \text{opt}_{\succeq_f}(\varphi) \) in every preference Boolean algebra \( M = \langle B, V, \succeq_f \rangle \).
Example 5. For the conditional norm set \( N = \{ (\top, g), (g, t), (\neg g, \neg t) \} \), the maximal consistent sets can be ordered as follows: the best maximal consistent sets have both \( g \) and \( t \) (type \( s_1 \)); the second best maximal consistency sets are those that have either \( \{ g, \neg t \} \) (type \( s_2 \)) or \( \{ \neg g, \neg t \} \) (type \( s_3 \)); and the worst maximal consistent sets are those that have \( \{ \neg g, t \} \) (type \( s_4 \)).

\[
\begin{array}{c|c}
\text{best} & s_1 \cdot g, t \\
\hline
\text{2nd best} & s_2 \cdot g, s_3 \cdot \\
\hline
\text{worst} & s_4 \cdot t
\end{array}
\]

Since \( \forall M \in \text{opt}_f(\top) (g \in M) \), \( \forall M \in \text{opt}_f(g) (t \in M) \) and \( \forall M \in \text{opt}_f(\neg g) (\neg t \in M) \), we have \( \top > \bigcirc g, g > \bigcirc t, \neg g > \bigcirc \neg t \in \text{derive}^{O_H}_i(N) \).

5.3. Integrating preferences along premise sets

A preference relation can be introduced over a set of valuations by means of a premise set [12, 13]. Valuations play the role of possible worlds here.

Definition 18. Let \( X \) be a set of propositional variables and \( \text{MaxC} \) the set of all maximal consistent subsets of \( Fm(X) \). For \( A \subseteq Fm(X) \), let \( f^A \subseteq \text{MaxC} \times \text{MaxC} \) such that \( f^A = \{ (K, M) | \forall \varphi \in A, (\varphi \in M \rightarrow \varphi \in K) \} \) is a relation over elements of \( \text{MaxC} \). Let \( \text{opt}_{f^A}(\varphi) = \{ M \in \text{MaxC} | \varphi \in M, \forall K (\varphi \in K \rightarrow (M, K) \in f^A) \} \). It is defined that \( \varphi > \bigcirc \psi \in \text{derive}^{O_K}_i(N) \) if and only if

\[
(\varphi, \psi) \in \text{derive}^{Fm(X)}_i(N) \text{ and } \forall M \in \text{opt}_{f^A}(\varphi) (\psi \in M).
\]

Given a set of \( \Gamma \subseteq Fm(X) \), it is defined that \( \Gamma > \bigcirc \psi \in \text{derive}^{O_K}_i(N) \) if \( \varphi > \bigcirc \psi \in \text{derive}^{O_K}_i(N) \) for some \( \varphi \in \Gamma \).

Definition 19. Let \( X \) be a set of propositional variables and \( A \subseteq Fm(X) \). A factual-preference Boolean algebra for \( Fm(X) \) is a structure \( M = \langle B, V, \succeq_A \rangle \), where:

- \( B \) is a Boolean algebra,
- \( V = \{ V_i \}_{i \in I} \) is the set of valuations from \( Fm(X) \) on \( B \),
- \( \succeq_A \subseteq V \times V \) such that \( V_i \succeq_A V_j \iff \forall \varphi \in A \ (V_j(\varphi) = 1_B \rightarrow V_i(\varphi) = 1_B) \).
Here, the betterness relation is reflexive and transitive by definition. For a given preference Boolean algebra $M = \langle B, V, \succeq_A \rangle$, it is defined that $\text{opt}_{\succeq_A}(\varphi) = \{ V_i \in V \mid V_i(\varphi) = 1_B, \forall V_j (V_j(\varphi) = 1_B \rightarrow V_i \succeq_A V_j) \}$.

**Theorem 10.** Let $X$ be a set of propositional variables, where $N \subseteq Fm(X) \times Fm(X)$, and $A \subseteq Fm(X)$. For a given Boolean algebra $B$ and a valuation $V$ on $B$, it is defined that $N^V = \{(V(\varphi), V(\psi)) | (\varphi, \psi) \in N \}$. We have

$$\varphi > \bigcirc \psi \in \text{derive}^{O^K}_i(N)$$

if and only if

$$V(\psi) \in \text{out}^B_i(N^V, \{ V(\varphi) \})$$

for every $B \in \text{BA}$ and valuation $V$

and

for every factual-preference Boolean algebra $M = \langle B, V, \succeq_A \rangle$,

for every valuation $V_i \in \text{opt}_{\succeq_A}(\varphi)$,

it is the case that $V_i(\psi) = 1_B$.

The theorem can be rewritten as follows\(^{11}\)

$$\varphi > \bigcirc \psi \in \text{derive}^{O^K}_i(N)$$

if and only if

$$\psi \in \text{out}^{Fm(X)}_i(N, \{ \varphi \})$$

and for $M = \langle 2, V, \succeq_A \rangle$,

for every valuation $V_i \in \text{opt}_{\succeq_A}(\varphi)$, we have $V_i(\psi) = 1_B$

or

$$\psi \in \text{out}^{Fm(X)}_i(N, \{ \varphi \})$$

and if $\varphi$ is consistent with $A$

then $A, \varphi \vdash \psi$, and if $\varphi$ is inconsistent with $A$, then $\varphi \vdash \psi$.

---

\(^{11}\)By the definition of $\text{opt}_{f_A}(\varphi)$, then $\psi$ is true in all maximal consistent subsets that include both $A$ and $\varphi$, or when $\varphi$ is inconsistent with $A$, in all maximal consistent subsets that include $\varphi$. 

16
The results for the constrained assumptions and preferences can be extended for the other systems introduced, for instance derive\(_{i}^{AND}(N)\).

**Example 6.** For the conditional norm set \(N = \{ (\top, g), (g, t), (\neg g, \neg t) \} \) and premise set \(A = \{ \neg g, \neg g \to \neg t \} \), the best maximal consistent set is type \(s_3\) (see Example 5), while types \(s_1, s_2\) and \(s_4\) are second best.

\[
\begin{array}{cccc}
\text{best} & s_3 \bullet \\
\text{2nd best} & s_2 \bullet g & s_1 \bullet g, t & s_4 \bullet t
\end{array}
\]

Since \(\forall M \in \text{opt}_{fA}(\neg g) (\neg t \in M)\), we have \(\neg g > \bigcirc \neg t \in \text{derive}_{i}^{OK}(N)\).

It is straightforward to rewrite the theorems for conditional permissions.

\[
\begin{align*}
\varphi > P\psi & \in \text{derive}_{i}^{PK}(N) & \varphi > P\psi & \in \text{derive}_{i}^{PH}(N) \\
\text{if and only if} & \quad \text{if and only if} \\
(\varphi, \psi) & \in \text{derive}_{i}^{Fm(X)}(N) & (\varphi, \psi) & \in \text{derive}_{i}^{Fm(X)}(N) \\
\text{for every factual-preference Boolean algebra} & \quad \text{for every preference Boolean algebra} \\
M = \langle B, V, \succeq_A \rangle, & \quad M = \langle B, V, \succeq_f \rangle, \\
\text{there is a valuation} & \quad \text{there is a valuation} \\
V_i & \in \text{opt}_{\succeq_A}(\varphi) & V_i & \in \text{opt}_{\succeq_f}(\varphi) \\
\text{such that} & \quad \text{such that} \\
V_i(\psi) & = 1_B. & V_i(\psi) & = 1_B.
\end{align*}
\]

6. Semantical embedding of input/output logic into HOL

The simple type theory developed by Church [14], a.k.a. classical higher-order logic (HOL), is an expressive language for representing mathematical structures. The syntax and semantics of HOL are well understood [15, 16] (for a brief introduction to HOL see Appendix D). It has roots in Frege’s book “Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens” [17] and Russell’s ramified theory of types [18]. The so-called shallow semantical embedding approach was developed by Benzmüller [19] for translating (the semantics of) classical and non-classical logics into HOL. Examples include propositional and quantified multimodal logics [20, 21] and dyadic deontic logics [22, 2].

Benzmüller et al. [3] devised an indirect approach to embedding two I/O operations in modal logic and consequently into HOL. One advantage of building I/O operations over Boolean algebras is that the I/O logic can be directly embedded in HOL. For normative system \(N\), the structure
$\mathcal{N} = \langle \mathcal{B}, V, N^V \rangle$ is called a Boolean normative model, where $V$ is a valuation from $\text{Fm}(X)$ to $\mathcal{B}$. The semantical embedding of I/O logic is based on Theorem 6, which states that $(\varphi, \psi) \in \text{derive}_{\text{Fm}(X)}^i(N)$ holds if and only if $V(\psi) \in \text{out}_{i}^B(N^V, \{V(\varphi)\})$ holds in all Boolean normative models.

The remainder of this section shows how the embedding works, abbreviating type $i \rightarrow o$ as $\tau$. The HOL signature is assumed to contain the constant symbols $N^i_\tau, \neg^i_\tau, \lor^i_\tau, \land^i_\tau, \top^i_\tau$ and $\bot^i_\tau$. Moreover, for each atomic propositional symbol $p^j \in X$ of $\text{Fm}(X)$, the HOL signature must contain a respective constant symbol $p^j_\tau$. Without loss of generality, it is assumed that besides those symbols and the primitive logical connectives of HOL, no other constant symbols are given in the signature of HOL.

The mapping $[\cdot]$ translates element $\varphi \in \text{Fm}(X)$ into HOL terms $[\varphi]$ of type $i$. The mapping is recursively defined:

- $[p^j] = p^j_\tau$ for $p^j \in X$
- $[\top] = \top_\tau$
- $[\bot] = \bot_\tau$
- $[\neg \varphi] = \neg_{i \tau}( [\varphi] )$
- $[ \varphi \lor \psi ] = \lor_{i \tau}( [\varphi] [\psi] )$
- $[ \varphi \land \psi ] = \land_{i \tau}( [\varphi] [\psi] )$
- $[d_i(N)(\varphi, \psi)]^{12} = ( \circ_i(N)_{\tau \rightarrow \tau} \{ [\varphi] \} [\psi] )$

$\circ_1(N)_{\tau \rightarrow \tau}, \circ_2(N)_{\tau \rightarrow \tau}, \circ_3(N)_{\tau \rightarrow \tau}$ and $\circ_4(N)_{\tau \rightarrow \tau}$ and $\circ_5(N)_{\tau \rightarrow \tau}$ are thereby abbreviated HOL terms:

- $\circ_1(N)_{\tau \rightarrow \tau} = \lambda A_\tau \lambda X_\tau ( \exists U ( \exists Y ( A Z \land Z = Y \land N Y U \land U \leq X )))$
- $\circ_2(N)_{\tau \rightarrow \tau} = \lambda A_\tau \lambda X_\tau ( \exists U ( \exists Y ( A Z \land Z \leq Y \land N Y U \land U = X )))$
- $\circ_3(N)_{\tau \rightarrow \tau} = \lambda A_\tau \lambda X_\tau ( \exists U ( \exists Y ( A Z \land Z \leq Y \land N Y U \land U \leq X )))$
- $\circ_4(N)_{\tau \rightarrow \tau} = \lambda A_\tau \lambda X_\tau ( \forall U ( Saturated V \land \forall U ( A U \rightarrow V U )) \rightarrow \exists Y ( \exists Z ( Z \leq X \land N Y Z \land V Y )))$
- $\circ_5(N)_{\tau \rightarrow \tau} = \lambda A_\tau \lambda X_\tau ( \forall V ( \forall U ( A U \rightarrow V U ) \land V = U p V \land \forall W ( \exists Y ( V Y \land N Y W ) \rightarrow V W )) \rightarrow \exists Y ( \exists Z ( Z \leq X \land N Y Z \land V Y )))$
where

\[
\leq = \lambda X_i \lambda Y_i (X \land_{i} Y = X)
\]

\[
\text{Saturated} = \lambda A_i (\forall X \forall Y ((A (X \lor_{i} Y) \rightarrow A X \lor A Y) \\
\land (A X \land X \leq Y \rightarrow A Y)))
\]

\[
Up = \lambda A_i \lambda X_i (\exists Z (A Z \land Z \leq X)).
\]

No further specification is needed for \(N_{i \rightarrow \tau}, \neg_{i \rightarrow \tau}, \lor_{i \rightarrow \tau}, \land_{i \rightarrow \tau}, \top_i \) and \(\bot_i\).

6.1. Soundness and completeness

To prove the soundness and completeness, that is, faithfulness, of the above embedding, a mapping from Boolean normative models into Henkin models is employed.

**Definition 20 (Henkin model \(H^N\) for Boolean normative model \(N\)).**

For any Boolean normative model \(N = \langle B, V, N V \rangle\), a corresponding Henkin model \(H^N\) is defined. Thus, let a Boolean normative model \(N = \langle B, V, N V \rangle\) be given. Moreover, assume that the finite set \(X = \{p^1, \ldots, p^m\}\), for \(m \geq 1\), are the only atomic symbols in \(\text{Fm}(X)\). The embedding requires the corresponding signature of HOL to provide constant symbols \(p^j_i\) such that \(\lfloor p^j \rfloor = p^j_i\).

A Henkin model \(H^N = \langle \{D_\alpha\}_{\alpha \in T}, I \rangle\) for \(N\) is now defined as follows: \(D_i\) is chosen as the set of \(B\); all other sets \(D_{\alpha \rightarrow \beta}\) are chosen as (not necessarily full) sets of functions from \(D_\alpha\) to \(D_\beta\). For all \(D_{\alpha \rightarrow \beta}\), the rule that every term \(t_{\alpha \rightarrow \beta}\) must be denoted in \(D_{\alpha \rightarrow \beta}\) must be obeyed (Denotatpflicht). In particular, it is required that \(D_i, D_{i \rightarrow i}, D_{i \rightarrow i} \) and \(D_{i \rightarrow \tau}\) should contain the elements \(Ip^j_i\), \(I\top_i\), \(I\bot_i\), \(I\neg_{i \rightarrow i}\), \(I\lor_{i \rightarrow i}, I\land_{i \rightarrow i}\) and \(IN_{i \rightarrow \tau}\). The interpretation function \(I\) of \(H^N\) is defined as follows:

1. For \(j = 1, \ldots, m\): \(Ip^j_i \in D_i\) is chosen such that \(Ip^j_i = V(p^j)\) in \(N\).
2. \(I\top_i \in D_i\) is chosen such that \(I\top_i = V(\top)\) in \(N\).
3. \(I\bot_i \in D_i\) is chosen such that \(I\bot_i = V(\bot)\) in \(N\).
4. \(I\neg_{i \rightarrow i} \in D_{i \rightarrow i}\) is chosen such that \(I(\neg_{i \rightarrow i} \varphi) = \psi\) iff \(\neg V(\varphi) = V(\psi)\) in \(N\).

\[\text{Note: } d_i(N)(\varphi, \psi) \text{ is an abbreviation of } (\varphi, \psi) \in \text{derive}^\text{Fm}(X)(N).\]
5. \( I \lor_{i \in I_1} \phi \psi = \phi \iff V(\phi) \lor V(\psi) = V(\phi) \) in \( N \).

6. \( I \land_{i \in I_1} \phi \psi = \phi \iff V(\phi) \land V(\psi) = V(\phi) \) in \( N \).

7. \( I N_{i \tau} \in D_{i \tau} \) is chosen such that \( I N_{i \tau} \phi \psi = T \iff (V(\phi), V(\psi)) \in N^V \)
in \( N \).

8. For the logical connectives \( \neg, \land, \lor, \Pi \) and \( = \) of HOL, the interpretation

The existence of valuation \( V \), which is a Boolean homomorphism from the
Boolean algebra \( \text{Fn}(X) \) into the Boolean algebra \( \mathcal{B} \), guarantees the existence
of \( I \) and its above-mentioned requirements. Since it is assumed that there are
no other symbols (apart from \( \top, \bot, \neg, \land, \lor, \Pi \) and \( = \) in the signature of HOL, \( I \) is a total function. Moreover, the
above construction guarantees that \( H^N \) is a Henkin model: \( \langle D, I \rangle \) is a frame, and the choice of \( I \) in combination with the Denotatpflicht ensures that for
arbitrary assignments, \( g, \| \cdot \|^{H^M,g} \) is a total evaluation function.

**Lemma 1.** Let \( H^M = \{D_\alpha\}_{\alpha \in \tau}, I \rangle \) be a Henkin model for Boolean normative
model \( N \). We have \( H^N \models^{HOL} \Sigma \) for all \( \Sigma \in \{\text{COM} \lor, \text{COM} \land, \text{ASS} \land, \text{IDE} \lor, \text{IDE} \land, \text{COMP} \lor, \text{COMP} \land, \text{Dis} \lor \land, \text{Dis} \land \lor\} \), where:

- \( \text{COM} \lor \) is \( \forall X_i Y_i \left( X \lor Y = Y \lor X \right) \)
- \( \text{COM} \land \) is \( \forall X_i Y_i \left( X \land Y = Y \land X \right) \)
- \( \text{ASS} \lor \) is \( \forall X_i Y_i Z_i \left( X \lor (Y \lor Z) = (X \lor Y) \lor Z \right) \)
- \( \text{ASS} \land \) is \( \forall X_i Y_i Z_i \left( X \land (Y \land Z) = (X \land Y) \land Z \right) \)
- \( \text{IDE} \lor \) is \( \forall X_i \left( X \lor \top = X \right) \)
- \( \text{IDE} \land \) is \( \forall X_i \left( X \land \bot = X \right) \)
- \( \text{COMP} \lor \) is \( \forall X_i \left( X \lor \neg X = \top \right) \)
- \( \text{COMP} \land \) is \( \forall X_i \left( X \land \neg X = \bot \right) \)
- \( \text{Dis} \lor \land \) is \( \forall X_i Y_i Z_i \left( X \lor (Y \land Z) = (X \lor Y) \land (X \lor Z) \right) \)
- \( \text{Dis} \land \lor \) is \( \forall X_i Y_i Z_i \left( X \land (Y \lor Z) = (X \land Y) \lor (X \land Z) \right) \)

**Lemma 2.** Let \( H^N \) be a Henkin model for Boolean normative model
\( N = \langle \mathcal{B}, V, N^V \rangle \). For all conditional norms \( \langle \varphi, \psi \rangle \) with arbitrary variable assignments \( g \), it holds that \( V(\psi) \in \text{out}^R_i(N, \{V(\varphi)\}) \) if and only if
\( \| [d_i(N)(\varphi, \psi)] \|^{H^N,g} = T \).
Lemma 3. For every Henkin model $H = \langle \{ D_\alpha \}_{\alpha \in T}, I \rangle$ such that $H \models^{\text{HOL}} \Sigma$ for all $\Sigma \in \{ \text{COM} \lor, \text{COM} \land, \text{ASS} \lor, \text{ASS} \land, \text{IDE} \lor, \text{IDE} \land, \text{COMP} \lor, \text{COMP} \land, \text{Dis} \lor \land, \text{Dis} \land \lor \}$, there exists a corresponding Boolean normative model $N$. Corresponding means that for all conditional norms $(\varphi, \psi)$ and for all $g$ assignments, then $\| d_i(N)(\varphi, \psi) \|^{H, g} = T$ if and only if $V(\psi) \in \text{out}_i^B(N^V, \{ V(\varphi) \})$.

Theorem 11 (Soundness and completeness of the embedding).

For every Boolean normative model $N$, $V(\psi) \in \text{out}_i^B(N^V, \{ V(\varphi) \})$ if and only if

$$\{ \text{COM} \lor, ..., \text{Dis} \land \lor \} \models^{\text{HOL}} [d_i(N)(\varphi, \psi)].$$
7. Related work

Gabbay, Parent and van der Torre [23] proposed building an I/O framework on top of lattices. They have results only for the simple-minded output operation. This article has shown that for an input set \( A \), by using the upward-closed set of \( A \) operator instead of the upward-closed set of the infimum of \( A \) [23], many new and old derivation systems can be built over Boolean algebras, Heyting algebras, and generally any abstract logic. The algebrization of the I/O framework shows more similarity with the theory of joining-systems [24], an algebraic approach for the study of normative systems over Boolean algebras. It can be said that norms in the I/O framework play the same role as joining in the theory of Lindahl and Odelstal [24] [25]. There are important similarities between input/output logic and the theory of joining-systems, such as studying normative systems as deductive systems and representing norms as ordered pairs. Moreover, both frameworks can generally be built on top of algebraic structures such as Boolean algebras and lattices. While the focus in input/output logic is deontic and factual detachment, the central themes of the theory of joining-systems are intermediate concepts and representing normative systems as a network of subsystems and their inter-relationships (for more details, see [24]). Sun [25] built Boolean joining systems that characterize I/O logic in a sense that a norm is derivable from a set of norms if and only if it is in the set of norms algebraically generated in the Lindenbaum-Tarski algebra for propositional logic. As in the Bochman approach [26], the work of Sun [25] has no direct connection to input/output operations. In this article, algebraic I/O operations were built directly over Boolean algebras and, more generally, abstract logics. There is a similar result for building the simple-minded I/O operation over Tarskian consequence relations in [27] (see the discussion about abstract input/output logic in [28]).

This article defined two groups of operations similar to the possible world semantics characterization of box and diamond, where box is closed under AND \( (((\Box \varphi \land \Box \psi) \rightarrow \Box(\varphi \land \psi)) \), and not in diamond:

**Derivations systems that do not admit the AND rule:** In the main literature of input/output logic developed by Makinson and van der Torre [7], Parent, Gabbay, and van der Torre [29], Parent and van der Torre [30, 31, 32], and Stolpe [33, 34, 35], at least one form of AND inference rule is present. Sun [28] analyzed norm derivation rules of input/output logic in isolation. Still, it is not clear how to combine them and build new logical systems,
specifically systems that do not admit the AND rule. This article has shown
how to remove the AND rule from the proof system and build new I/O
operations to produce permissible propositions. Unlike minimal deontic log-
ics [36, 37], and similar approaches such as that of Ciabattoni, Gulisano and
Lellmann [38] that do not have deontic aggregation principles, the approach
presented in this article validates deontic and factual detachment.

Derivations systems that admit the AND rule: In accordance with the
reversibility of inference rules in the I/O proof systems, this article has shown
how it is possible to add AND and other rules required for obligation [7] to
the proof systems, and find I/O operations for them.

There are other abstract approaches: I/O operations over semigroups [39],
which does not admit AND, and a detachment mechanism over an arbitrary
set [40] that admits a kind of AND, called cumulative aggregation. However,
it is not clear how these approaches can be used for logical purposes.

Constrained I/O logic [41] was introduced for reasoning about contrary-
to-duty problems. In this article, constraints are preferences. In this sense,
the article has presented a semantic characterization for constrained I/O
logic. There are syntactic [41], (c.f. Section 6) and proof theoretical [42] (c.f.
Section 3) characterizations for constrained I/O logic. The semantic charac-
terization is more flexible than the syntactic characterization mentioned since
the approach presented here does not necessarily depend on the AND, SI, and
(EQI, EQO) rules required for syntactic characterization of I/O operations
in modal logic [42].
8. Conclusion

This article presented new algebraic systems developed in the LogiKEy normative reasoning framework. A dataset of semantical embeddings of deontic logics in HOL is available (see Appendix A). The dataset can be used for ethical and legal reasoning tasks. In summary, this article characterizes a class of proof systems over Boolean algebras for a set of explicitly given conditional norms as follows:

| derive \textsuperscript{B} | Rules | \textsuperscript{EQO} | \frac{(a,x)}{(a,y)} | \frac{x = y}{(a,y)} |
|--------------------------|-------|-----------------|----------------|----------------|
| derive \textsuperscript{B} \textsuperscript{R} | \{EQO\} | \textsuperscript{T} | \frac{(a,x)}{(a,y)} | \frac{(x,y)}{(a,y)} |
| derive \textsuperscript{B} \textsuperscript{L} | \{EQI\} | \textsuperscript{EQI} | \frac{(a,x)}{(b,x)} | \frac{a = b}{(b,x)} |
| derive \textsuperscript{B} \textsuperscript{0} | \{EQI, EQO\} | \textsuperscript{OR} | \frac{(a,x)}{(a \lor b, x)} | \frac{(b, x)}{(a \lor b, x)} |
| derive \textsuperscript{B} \textsuperscript{1} | \{SI, EQQ\} | \textsuperscript{SI} | \frac{(a,x)}{(b,x)} | \frac{b \leq a}{(b,x)} |
| derive \textsuperscript{B} \textsuperscript{2} | \{SI, WO, OR\} | \textsuperscript{WO} | \frac{(a,x)}{(a,y)} | \frac{x \leq y}{(a,y)} |
| derive \textsuperscript{B} \textsuperscript{3} | \{SI, WO, T\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |

Each proof system is sound and complete for an input/output (I/O) operation. The I/O operations resemble inferences, where inputs need not be included among outputs, and outputs need not be reusable as inputs. Moreover, this article has shown how to add the two rules AND and CT to the proof systems and find corresponding I/O operations for them.

| derive \textsuperscript{X} | Rules | \textsuperscript{AND} | \frac{(a,x)}{(a,y)} | \frac{(a,y)}{(a_x \wedge y)} |
|--------------------------|-------|-----------------|----------------|----------------|
| derive \textsuperscript{AND} \textsuperscript{1} | \{WO, EQI, AND\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{AND} \textsuperscript{2} | \{SI, WO, AND\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{AND} \textsuperscript{3} | \{SI, WO, OR, AND\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{1} | \{SI, EQQ, CT\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{2} | \{WO, EQI, CT\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{3} | \{SI, WO, CT\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{4} | \{SI, EQQ, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{5} | \{SI, WO, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{6} | \{SI, EQQ, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{7} | \{SI, WO, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{8} | \{SI, EQQ, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{9} | \{SI, WO, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |
| derive \textsuperscript{CT} \textsuperscript{10} | \{SI, EQQ, CT, OR\} | \textsuperscript{CT} | \frac{(a,x)}{(a,y)} | \frac{(a \wedge x, y)}{(a,y)} |

The input/output logic is inspired by a view of logic as a secretarial...
assistant tasked with preparing inputs before they go into the motor engine and are unpacked as outputs, rather than logic as an inference motor [7]. The only input/output logics investigated so far in the literature are built on top of classical propositional logic and intuitionist propositional logic [29]. The algebraic construction presented has shown how to build input/output operations on top of any abstract logic.

Finally, this article has proved that the extension of propositional logic with a set of conditional norms is sound and complete with respect to the class of Boolean algebras, and that the corresponding I/O operation holds for all of them. Based on this result, a conditional theory has been integrated into input/output logic.

Acknowledgments

I would like to thank Leon van der Torre, Dov Gabbay, Christoph Benzmüller, Xavier Parent, and Majid Alizadeh for comments that greatly improved the manuscript. Thanks also to Llio Humphreys for her proofreading of the English in this article.

References

[1] C. Benzmüller, X. Parent, L. van der Torre, Designing normative theories for ethical and legal reasoning: LogiKEy framework, methodology, and tool support, Artificial Intelligence 237 (2020) 103348.

[2] C. Benzmüller, A. Farjami, X. Parent, Åqvist’s dyadic deontic logic E in HOL, Journal of Applied Logics – IfCoLoG Journal of Logics and their Applications (Special Issue on Reasoning for Legal AI) 6 (5) (2019) 733–755.

[3] C. Benzmüller, A. Farjami, P. Meder, X. Parent, I/O logic in HOL, Journal of Applied Logics – IfCoLoG Journal of Logics and their Applications (Special Issue on Reasoning for Legal AI) 6 (5) (2019) 715–732.

[4] C. Benzmüller, A. Farjami, D. Fuenmayor, P. Meder, X. Parent, A. Steen, L. van der Torre, V. Zahoransky, LogiKEy workbench: Deontic logics, logic combinations and expressive ethical and legal reasoning (Isabelle/HOL dataset), Data in Brief (106409) (2020) 1–15.
[5] J. M. Font, R. Jansana, A general algebraic semantics for sentential logics, Cambridge University Press, 2017.

[6] P. McNamara, Deontic logic, in: E. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, winter 2014 Edition, Metaphysics Research Lab, Stanford University, 2014.

[7] D. Makinson, L. van der Torre, Input/output logics, Journal of Philosophical Logic 29 (4) (2000) 383–408.

[8] X. Parent, L. van der Torre, Introduction to deontic logic and normative systems, College Publications, 2018.

[9] R. Jansana, Algebraic propositional logic, in: E. N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, winter 2016 Edition, Metaphysics Research Lab, Stanford University, 2016.

[10] B. Hansson, An analysis of some deontic logics, Nous (1969) 373–398.

[11] D. Lewis, Counterfactuals, Blackwell, Oxford, 1973.

[12] D. Lewis, Ordering semantics and premise semantics for counterfactuals, Journal of Philosophical Logic 10 (2) (1981) 217–234.

[13] A. Kratzer, Modals and conditionals: New and revised perspectives, Vol. 36, Oxford University Press, 2012.

[14] A. Church, A formulation of the simple theory of types, Journal of Symbolic Logic 5 (2) (1940) 56–68.

[15] C. Benzmüller, C. Brown, M. Kohlhase, Higher-order semantics and extensionality, Journal of Symbolic Logic 69 (4) (2004) 1027–1088. doi: 10.2178/jsl/1102022211. URL http://christoph-benzmueller.de/papers/J6.pdf

[16] C. Benzmüller, P. Andrews, Church’s type theory, in: E. N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, summer 2019 Edition, Metaphysics Research Lab, Stanford University, 2019, pp. 1–62 (in pdf version). URL https://plato.stanford.edu/entries/type-theory-church/
[17] G. Frege, Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens, Halle, 1879.

[18] B. Russell, Mathematical logic as based on the theory of types, American Journal of Mathematics 30 (3) (1908) 222–262.

[19] C. Benzmüller, Universal (meta-) logical reasoning: Recent successes, Science of Computer Programming 172 (2019) 48–62.

[20] C. Benzmüller, L. Paulson, Quantified multimodal logics in simple type theory, Logica Universalis (Special Issue on Multimodal Logics) 7 (1) (2013) 7–20. doi:10.1007/s11787-012-0052-y. URL http://christoph-benzmueller.de/papers/J23.pdf

[21] C. Benzmüller, Automating quantified conditional logics in HOL in: F. Rossi (Ed.), 23rd International Joint Conference on Artificial Intelligence (IJCAI-13), AAAI Press, Beijing, China, 2013, pp. 746–753. URL http://christoph-benzmueller.de/papers/C37.pdf

[22] C. Benzmüller, A. Farjami, X. Parent, A dyadic deontic logic in HOL, in: J. Broersen, C. Condoravdi, S. Nair, G. Pigozzi (Eds.), Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018, Vol. 9706, College Publications, 2018, pp. 33–50.

[23] D. Gabbay, X. Parent, L. van der Torre, A geometrical view of I/O logic, coRR abs/1911.12837 (2019).

[24] L. Lindahl, J. Odelstad, The theory of joining-systems, in: D. Gabbay, J. Horty, X. Parent, R. van der Meyden, L. van der Torre (Eds.), Handbook of deontic logic and normative systems., Vol. 1, College Publications, 2013, pp. 545–634.

[25] X. Sun, Proof theory, semantics and algebra for normative systems, Journal of Logic and Computation 28 (8) (2015) 1757–1779.

[26] A. Bochman, Explanatory nonmonotonic reasoning, World scientific, 2005.

[27] W. Carnielli, M. E. Coniglio, L. van der Torre, Input/output consequence relations: reasoning with intensional contexts, Unpublished report, 2009.
[28] X. Sun, Logic and games of norms: a computational perspective, Ph.D. thesis, University of Luxembourg, Luxembourg (2016).

[29] X. Parent, D. Gabbay, L. van der Torre, Intuitionistic basis for input/output logic, in: David Makinson on Classical Methods for Non-Classical Problems, Springer, 2014, pp. 263–286.

[30] X. Parent, L. van der Torre, Sing and dance!, in: F. Cariani, D. Grossi, J. Meheus, X. Parent (Eds.), Deontic Logic and Normative Systems — 12th International Conference, DEON 2014, Ghent, Belgium, July 12-15, 2014, Springer, 2014, pp. 149–165.

[31] X. Parent, L. van der Torre, The pragmatic oddity in norm-based deontic logics, in: G. Governatori, J. Keppens (Eds.), Proceedings of the 16th edition of the International Conference on Artificial Intelligence and Law, ACM-SIAM, 2017, pp. 169–178.

[32] X. Parent, L. van der Torre, I/O logics with a consistency check., in: J. Broersen, C. Condoravdi, S. Nair, G. Pigozzi (Eds.), Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018, College Publications, 2018, pp. 285–299.

[33] A. Stolpe, Normative consequence: The problem of keeping it whilst giving it up, in: R. van der Meyden, L. van der Torre (Eds.), Deontic Logic and Normative Systems — 9th International Conference, DEON 2008, Luxembourg, Luxembourg, July 15-18, 2008, Springer, 2008, pp. 174–188.

[34] A. Stolpe, Norms and norm-system dynamics, Ph.D. thesis, Department of Philosophy, University of Bergen, Norway (2008).

[35] A. Stolpe, A concept approach to input/output logic, Journal of Applied Logic 13 (3) (2015) 239–258.

[36] B. F. Chellas, Modal logic, Cambridge university press, 1980.

[37] L. Goble, Prima facie norms, normative conflicts, and dilemmas, in: D. Gabbay, J. Horty, X. Parent, R. van der Meyden, L. van der Torre (Eds.), Handbook of Deontic Logic, Vol. 1, College Publications, 2013, pp. 499–544.
[38] A. Ciabattoni, F. Gulisano, B. Lellmann, Resolving conflicting obligations in Mīmāṁsā: a sequent-based approach, in: J. Broersen, C. Condoravdi, S. Nair, G. Pigozzi (Eds.), Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018, 2018, pp. 91–109.

[39] S. C. Tosatto, G. Boella, L. van der Torre, S. Villata, Abstract normative systems: Semantics and proof theory, in: G. Brewka, T. Eiter, S. A. McIlraith (Eds.), Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning, AAAI Press, 2012, pp. 358–368.

[40] D. A. Ambrossio, Non-monotonic logics for access control: Delegation revocation and distributed policies, Ph.D. thesis, University of Luxembourg, Luxembourg (2017).

[41] D. Makinson, L. van der Torre, Constraints for input/output logics, Journal of Philosophical Logic 30 (2) (2001) 155–185.

[42] C. Straßer, M. Beirlaen, F. van de Putte, Adaptive logic characterizations of input/output logic, Studia Logica 104 (5) (2016) 869–916.

[43] T. Nipkow, L. Paulson, M. Wenzel, Isabelle/HOL — A proof assistant for higher-order logic, Vol. 2283 of Lecture Notes in Computer Science, Springer, 2002.

[44] J. Blanchette, T. Nipkow, Nitpick: A counterexample generator for higher-order logic based on a relational model finder, in: M. Kaufmann, L. C. Paulson (Eds.), First International Conference on Interactive Theorem Proving, no. 6172 in Lecture Notes in Computer Science, Springer, 2010, pp. 131–146.

[45] J. C. Blanchette, S. Böhme, L. C. Paulson, Extending sledgehammer with SMT solvers, Journal of Automated Reasoning 51 (1) (2013) 109–128.

[46] W. J. Blok, D. Pigozzi, Algebraizable logics, Vol. 77, American Mathematical Society, 1989.

[47] L. Henkin, Completeness in the theory of types, Journal of Symbolic Logic 15 (2) (1950) 81–91.
Appendix A

The semantical embedding outlined in Section 6 has been implemented in the higher-order proof assistant Isabelle/HOL. Figures 1 and 2 display their respective encoding. Figure 1, after introducing type $i$ for representing the elements of Boolean algebra, introduces the algebraic operators as constants in higher-order logic. The algebraic operators are also characterized in accordance with the definition of Boolean algebra.

**Figure 1: Semantical embedding of Boolean algebra in Isabelle/HOL**

```plaintext
theory IOBoolean
  imports Main

begin

typedec i (* type for boolean elements *)
type_synonym τ = "(i⇒bool)"
type_synonym a = "(i⇒i⇒bool)"
consts N :: "i⇒i⇒bool" ("N") (* Normative system *)
consts dis :: "i⇒i⇒i" (infixr "∨" 50)
consts con :: "i⇒i⇒i" (infixr "∧" 60)
consts neg :: "i⇒i" ("¬"[52][53])
consts top :: i ("T")
consts bot :: i ("⊥")

axiomatization where
  COMdis : "∀X. ∀Y. (X ∨ Y) = (Y ∨ X)" and
  COMcon : "∀X. ∀Y. (X ∧ Y) = (Y ∧ X)" and
  ASSdis : "∀X. ∀Y. ∀Z. (X ∨ (Y ∨ Z)) = (X ∨ (Y ∨ Z))" and
  ASScon : "∀X. ∀Y. ∀Z. (X ∧ (Y ∧ Z)) = (X ∧ (Y ∧ Z))" and
  IDEdis : "∀X. (X ∨ ⊥) = X" and
  IDEcon : "∀X. (X ∧ T) = X" and
  COMPdis : "∀X. (X ∨ ¬X) = T" and
  COMPcon : "∀X. (X ∧ ¬X) = ⊥" and
  Ddiscon : "∀X. ∀Y. ∀Z. (X ∨ (Y ∨ Z)) = ((X ∨ Y) ∨ (X ∨ Z))" and
  Dcondis : "∀X. ∀Y. ∀Z. (X ∧ (Y ∧ Z)) = ((X ∧ Y) ∧ (X ∧ Z))"
```

Figure 2 displays the semantical embedding of I/O operations ($out_i$) in HOL, including the definition of the upward-closed set operator and saturated set.

Figure 3 shows some experiments via the model and countermodel finder Nitpick, and prove some facts about I/O operations using automatic theorem provers (auto and meson) via the Sledgehammer tool.

30
Figure 2: Semantical embedding of \textit{out} in Isabelle/HOL

```
definition orde1OB :: "1 ⇒ τ" (infixr "≤" 50) where "X ≤ Y = ((X ∧ Y) = X)"
definition satu1OB :: "τ ⇒ bool" ("Saturated") where "Saturated V = ∀X. ∀Y. (((V (X ∨ Y)) = ((V X ∨ V Y)) ∧ ((V X ∧ (V ≤ Y)) = V Y))"
definition Upward1OB :: "τ ⇒ τ" ("Up") where "Up V = ∀X. (∃Z. (V Z ∧ Z ≤ X))"
```

```
definition out1 :: "a ⇒ τ ⇒ τ" ("₃<_<>_") where "₃<_<>_(A;x) = ∃X. (∀Y. (∀Z. (Z ∧ (Z Y) ∧ M Y U ∧ (U ≤ X) )))
```

```
definition out11 :: "a ⇒ τ ⇒ τ" ("₃<_<>_") where "₃<_<>_(A;x) = ∃X. (∀Y. (∀Z. (Z ∧ (Z Y) ∧ M Y U ∧ (U ≤ X) )))
```

```
definition out1 :: "a ⇒ τ ⇒ τ" ("₃<_<>_") where "₃<_<>_(A;x) = ∃X. (∀Y. (∀Z. (Z ∧ (Z Y) ∧ M Y U ∧ (U ≤ X) )))
```

```
definition out2 :: "a ⇒ τ ⇒ τ" ("₃<_<>_") where "₃<_<>_(A;x) = ∃X. (∀Y. (∀Z. (Z ∧ (Z Y) ∧ M Y U ∧ (U ≤ X) )))
```

```
definition out3 :: "a ⇒ τ ⇒ τ" ("₃<_<>_") where "₃<_<>_(A;x) = ∃X. (∀Y. (∀Z. (Z ∧ (Z Y) ∧ M Y U ∧ (U ≤ X) )))
```
```

Figure 3: Some experiments on \textit{out} in Isabelle/HOL

```
consts a :: i
c consts b :: i
c consts c :: i
c consts M :: τ

lemma "₃<_<>_(λX. X = a) > a" nitpick [user_axioms,expect=genuine,show_all] oops

lemma "₃<_<>_(λX. X = a) > X ∧ (b ≤ a) → ₃<_<>_(λX. X = b) > X" nitpick [satisfy,user_axioms,show_all,expect=genuine,card=4] oops

lemma "₃<_<>_(λX. X = a) > X ∧ ₃<_<>_(λX. X = b) > X → ₃<_<>_(λX. (X ∨ a)) > X" nitpick [user_axioms,expect=genuine,show_all] oops

lemma "₃<_<>_(λX. X = a) > X ∧ ₃<_<>_(λX. X = a) > y → ₃<_<>_(λX. X = a) > (X ∧ y)" nitpick [user_axioms,expect=genuine,show_all] oops

lemma "₃<_<>_(λX. X = a) > X → ₃<_<>_(λX. X = a) > X" nitpick [user_axioms,expect=genuine,show_all] oops

lemma "₃<_<>_(λX. X = a) > X → ₃<_<>_(λX. X = a) > X" unfoldingDefs by auto

lemma "₃<_<>_(λX. X = a) > X → ₃<_<>_(λX. X = a) > X" unfoldingDefs by meson

lemma "₃<_<>_(λX. X = a) > X → ₃<_<>_(λX. X = a) > X" nitpick [user_axioms,expect=genuine,show_all] oops
```

In Figure 4, the first two lemmas prove the soundness of \textit{out}₁. The next two lemmas show the factual detachment of this output operation. The last
two lemmas illustrate the soundness of out\textsubscript{1} and out\textsubscript{II} where the depth of inference is one.

Figure 4: Soundness of out\textsubscript{1} in Isabelle/HOL

```plaintext
(*Soundness for Out1*)
lemma "(\circ \circ \circ \circ N ; N ; ((AX. X= a))> X \wedge (X \leq Y)) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= a))> Y" unfolding Defs
by (metis COMPcon COMPdis COMcon COMdis Dcondis Ddiscon IDEcon IDEdis ordeIOB_def)

lemma "(\circ \circ \circ \circ N ; N ; ((AX. X= a))> X \wedge (b \leq a)) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= b))> X" unfolding Defs
by (metis COMPcon COMPdis COMcon COMdis Dcondis Ddiscon IDEcon IDEdis)

lemma "((N a b) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= a))> b))" unfolding Defs
by (metis COMPcon COMPdis COMcon COMdis Dcondis Ddiscon IDEdis)

lemma "((N a b \wedge W a) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= a))> b))" unfolding Defs
by (metis COMPcon COMPdis COMcon COMdis Dcondis Ddiscon IDEdis)

lemma "((N a b \wedge (b \leq c)) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= a))> c))" unfolding Defs
by (metis COMPcon COMPdis COMcon COMdis Dcondis Ddiscon IDEcon)

lemma "((N a b \wedge (c \leq a)) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= c))> b))" unfolding Defs
by (metis COMPcon COMPdis COMcon COMdis Dcondis Ddiscon IDEdis)

lemma "((N a b \wedge (b \leq c)) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= a))> c))" unfolding Defs using ordeIOB_def outI_def by auto

lemma "((N a b \wedge (c \leq a)) \rightarrow (\circ \circ \circ \circ N ; N ; ((AX. X= c))> b))" unfolding Defs using ordeIOB_def outII_def by auto
```

Figure 5 shows the soundness of out\textsubscript{2} and out\textsubscript{3} for a depth of one. The input/output operations introduced by Makinson and van der Torre \cite{7} are implemented in Figure 6. The implementations are based on the reversibility of rules in the derivation systems. The four input/output operations introduced in \cite{7} were built over the simple-minded output operation (see Section \[3\]).
Figure 5: Soundness of $out_2$ and $out_3$ in Isabelle/HOL

```isabelle
(* out2 depth1-soundness *)

lemma "((N a b ∧ (b ≤ c)) → (∃2N;((∀X. X = a)) > c))" unfolding Defs by auto

lemma "((N a b ∧ (c ≤ a)) → (∃2N;((∀X. X = c)) > b))" unfolding Defs by (metis COMPcon COMPcon COMDis Dcondis Discondis IDEdis)

lemma "((N a b ∧ N c b) → (∃2N;((∀X. X = (a ∨ c))) > b))" unfolding Defs by (metis COMPDis COMPcon COMDis Dcondis Discondis IDEcon)

(* out3 depth1-soundness *)

lemma "((N a b ∧ (b ≤ c)) → (∃3N;((∀X. X = a)) > c))" unfolding Defs by auto

lemma "((N a b ∧ (c ≤ a)) → (∃3N;((∀X. X = c)) > b))" unfolding Defs by (metis COMPDis COMPcon COMDis Dcondis Discondis IDEcon)

lemma "((N a b ∧ N b c) → (∃3N;((∀X. X = a)) > c))" unfolding Defs by (metis COMPCon COMPDis COMDis Dcondis Discondis IDEdis)
```

Figure 6: Semantical embedding of output operations in Isabelle/HOL

```isabelle
theory output operation imports IO boolean
begin

definition Route :: "a ⇒ i ⇒ i ⇒ bool" ("Route<_,__>")
where "Route<_,__> = λX. λX. λY. (λZ. λZ. (λ≤Y. (Z ≤ Y ∧ X ≤ Y)))"

definition sub_rel :: "rel ⇒ rel ⇒ rel" where "sub_rel R Q = Y u v. (Q u v w. (Q u v w. (Q v w))" ("Route<_,__>")

definition outAND :: "a ⇒ τ ⇒ τ" ("outAND<_,__>") where "outAND<_,__> = λX. λY. TCAND (Route<_,__>) Y X"

definition outOR :: "a ⇒ a ⇒ a" ("outOR<_,__>") where "outOR<_,__> = λX. λY. TCOR (Route<_,__>) Y X"

definition outXOR :: "a ⇒ τ ⇒ τ" ("outXOR<_,__>") where "outXOR<_,__> = λX. λY. TCXOR (Route<_,__>) Y X"

definition outORI :: "a ⇒ τ ⇒ τ" ("outORI<_,__>") where "outORI<_,__> = λX. λY. TCOR (Route<_,__>) Y X"

definition outXORI :: "a ⇒ τ ⇒ τ" ("outXORI<_,__>") where "outXORI<_,__> = λX. λY. TCXOR (Route<_,__>) Y X"
```

33
The following lemmas (see Fig. 7) show the automation capability of implemented output operations for the simple-minded output operation \( \text{out}^{\text{AND}}_1(N) \) as introduced by Makinson and van der Torre [7].

Figure 7: Semantical embedding of output operations in Isabelle/HOL

```
lemma imp : "\(\exists x \neg N; (\forall X. X = a) \) \rightarrow b \rightarrow \bigwedge \neg N; (\forall X. X = a) \) \rightarrow b"
using out1_def Rout_def Sub_re_def close_AND_def TCAND_def unfolding Defst outAND_def
by auto

lemma "N \ a \ b \rightarrow (\bigwedge \neg N; (\forall X. X = a) \) \rightarrow b)"
using imp Rout_def Sub_re_def close_AND_def TCAND_def unfolding Defst outAND_def
by (metis COMPcon COMPdis CONcon COMPdis Dcondis IDEcon IDEdis ordeI0B_def )

lemma "N \ a \ b \ \wedge \ N \ a \ c \rightarrow (\bigwedge \neg N; (\forall X. X = a) \) \rightarrow (b \ \wedge \ c))"
using imp Rout_def Sub_re_def close_AND_def TCAND_def
unfolding Defst outAND_def TCAND_def
by (metis COMPcon COMPdis Dcondis IDEcon IDEdis ordeI0B_def )

lemma imp2 : "\(\exists x \neg N; (\forall X. X = a) \) \rightarrow b \rightarrow \bigwedge \neg N; (\forall X. X = a) \) \rightarrow b"
using out1_def Rout_def Sub_re_def close_OR_def TCOR_def unfolding Defst outOR_def
by auto

lemma "(\exists x \neg N; (\forall X. X = a) \) \rightarrow b \ \wedge \ (\exists x \neg N; (\forall X. X = a) \) \rightarrow c \rightarrow \bigwedge \neg N; (\forall X. X = a) \) \rightarrow (b \ \wedge \ c)"
unfolding Defst outAND_def TCAND_def close_AND_def out1_def
by metis
```

The proof system of input/output logic can be implemented directly in Isabelle/HOL—see Fig. 8 and 9. The idea is based on an (universal) order of rules in a derivation. The ordering of rules and closure operation are the main ways of defining the derivation systems (for more details, see Section 3). For example, in line 27 of Fig.9, \texttt{derSIEQO} introduces the derivation system \texttt{derive}, with the rules in \{\texttt{SI, EIQO}\} and in lines 51–52, \texttt{derSIWOCTORAND} introduce the derivation system \texttt{derive4} with the rules in \{\texttt{SI, WO, CT, OR, AND}\}. 

34
35

Figure 8: Semantical embedding of I/O proof systems in Isabelle/HOL

```isabelle
theory systems imports IOBoolean
begin

definition Close EQO :: "α ⇒ bool" where "Close EQO Q ≡ ∀ u v w. (Q u v ∧ (v = w) −→ (Q u w))"
definition Close EQI :: "α ⇒ bool" where "Close EQI Q ≡ ∀ u v w. (Q u v ∧ (u = w) −→ (Q w v))"
definition Close SI :: "α ⇒ bool" where "Close SI Q ≡ ∀ u v w. (Q u v ∧ (v ≤ w) −→ (Q w v))"
definition Close WQ :: "α ⇒ bool" where "Close WQ Q ≡ ∀ u v w. (Q u v ∧ (v ≤ w) −→ (Q u w))"
definition Close AND :: "α ⇒ bool" where "Close AND Q ≡ ∀ u v w. (Q u v ∧ Q u w −→ (Q u (v ∧ w)))"
definition Close OR :: "α ⇒ bool" where "Close OR Q ≡ ∀ u v w. (Q u v ∧ Q u w −→ (Q (v v w)))"
definition Close CT :: "α ⇒ bool" where "Close CT Q ≡ ∀ u v w. (Q u v ∧ Q (v u w) −→ (Q v w))"

definition Sub rel :: "α⇒α⇒bool" where "Sub rel R Q ≡ ∀ u v. R u v −→ Q u v"
definition TCEQO :: "α ⇒ α" where "TCEQO R ≡ λ x y. λ Q. Close EQO Q −→ (Sub rel R Q −→ Q x y)"
definition TCQI :: "α ⇒ α" where "TCQI R ≡ λ x y. λ Q. Close EQI Q −→ (Sub rel R Q −→ Q x y)"
definition TCWI :: "α ⇒ α" where "TCWI R ≡ λ x y. λ Q. Close SI Q −→ (Sub rel R Q −→ Q y x)"
definition TAND :: "α ⇒ α" where "TAND R ≡ λ x y. λ Q. Close AND Q −→ (Sub rel R Q −→ Q x y)"
definition TCOR :: "α ⇒ α" where "TCOR R ≡ λ x y. λ Q. Close OR Q −→ (Sub rel R Q −→ Q y x)"
definition TCCT :: "α ⇒ α" where "TCCT R ≡ λ x y. λ Q. Close CT Q −→ (Sub rel R Q −→ Q y x)"

definition derSI :: "α⇒α⇒α" where "derSI α β = TCSI (TCQI β)"
definition derWQ :: "α⇒α⇒α" where "derWQ α β = TCWI β"
definition derAND :: "α⇒α⇒α" where "derAND α β = TAND β"
definition derOR :: "α⇒α⇒α" where "derOR α β = TCOR β"
definition derCT :: "α⇒α⇒α" where "derCT α β = TCCT β"
```

Figure 9: Semantical embedding of I/O proof systems in Isabelle/HOL

```isabelle
definition derSEQO :: "α⇒α" where "derSEQO α = TCSI (TCEQO α)"
definition derWOEI :: "α⇒α" where "derWOEI α = TCWI (TCQI α)"
definition derSEQI :: "α⇒α" where "derSEQI α = TCSI (TCEQI α)"
definition derANDO :: "α⇒α" where "derANDO α = TAND α"
definition derORO :: "α⇒α" where "derORO α = TCOR α"
definition derCTO :: "α⇒α" where "derCTO α = TCCT α"
```

```isabelle
definition derDERIV1_Per :: "α⇒α⇒α" where "derDERIV1_Per α β = TCSI (TCQI β)"
definition derDERIV1_Ord :: "α⇒α⇒α" where "derDERIV1_Ord α β = TCWI β"
definition derDERIV2_Per :: "α⇒α⇒α" where "derDERIV2_Per α β = TCSI (TCEQI β)"
definition derDERIV2_Ord :: "α⇒α⇒α" where "derDERIV2_Ord α β = TCWI (TCQI β)"
definition derDERIV3_Per :: "α⇒α⇒α" where "derDERIV3_Per α β = TCSI (TCCT β)"
definition derDERIV3_Ord :: "α⇒α⇒α" where "derDERIV3_Ord α β = TCWI (TCQI β)"
definition derDERIV4_Per :: "α⇒α⇒α" where "derDERIV4_Per α β = TCSI (TCCT β)"
definition derDERIV4_Ord :: "α⇒α⇒α" where "derDERIV4_Ord α β = TCWI (TCQI β)"
```
One advantage of implementing the proof system of I/O logic, besides the output operations, is that completeness theorems can be checked. For example, the completeness of out1, as shown in Fig. 10, is checked in lines 70–73. Lines 61 and 62 show the AND closure. Lines 64–67 demonstrate automation of the implementation for a normative system $M$.

The proof theoretical difference of different I/O systems can be examined (cf. Fig. 11). For example, lines 81–85 show that the implemented derivation system derSIWOOR ($derive_2$) is sound for the OR rule for a depth of one.

---

**Figure 10: Completeness checking of out1 in Isabelle/HOL**

```isabelle
lemma "Close AND (TCAND N)" unfolding Defst TCAND def by metis

lemma "\((M a b \lor \{x y\}. M b \land \{a \leq y\}) \rightarrow \text{derSI-M} a b\)"
using Sub_rel_def Close_SI_def TCSI_def unfolding Defst and Defs derSI_def by metis

\text{(OUT1 completeness)}

lemma "(\forall x. x \neq a) \rightarrow y \rightarrow \text{derSIWOOR-a y}"
using Sub_rel_def Close_SI_def Close_M0_def TCSI_def TCWO_def unfolding Defst and Defs derSI_def Sub_rel_def TCWO_def TCSI_def by metis
```

**Figure 11: Some experiments on I/O proof systems in Isabelle/HOL**

```isabelle
lemma "\((N a b \land N a c) \land (N x y \rightarrow (N a b \lor N a c))\)"
   \rightarrow derSIWAND\rightarrow a (b\land c)\ (* AND Closed *)
using Sub_rel_def Close_SI_def TCSI_def TCWO_def unfolding Defst and Defs derSI_def Sub_rel_def TCWO_def by auto

lemma "\((N a b \land N c b) \land (N x y \rightarrow (N a b \lor N c a))\)"
   \rightarrow derSIWOOR\rightarrow (a \lor c) b\ (*OR Closed *)
using Sub_rel_def Close_SI_def TCSI_def TCWO_def unfolding Defst and Defs derSI_def Sub_rel_def TCWO_def by auto

lemma "\((N a b \land N (a\land b)) \land (N x y \rightarrow (N a b \lor N (a\land b)))\)"
   \rightarrow derSIWOCT\rightarrow a c\ (*CT Closed *)
using Sub_rel_def Close_SI_def TCSI_def TCWO_def unfolding Defst and Defs derSI_def Sub_rel_def TCWO_def by (simp Close_CT_def Sub_rel_def TCT_def TCWO_def derSIWOCT_def)

lemma "\((N a b \land N (a\land b)) \land (N x y \rightarrow (N a b \lor N (a\land b)))\)"
   \rightarrow derSIWCTAND\rightarrow a c\ (*CT Closed *)
using Close_CT_def Sub_rel_def TCT_def TCWO_def derSIWOCT_def unfolding Defst and Defs derSI_def Sub_rel_def TCWO_def TCT_def by (metis (no_types, hide_lams) Sub_rel_def TCSI_def)
```

36
Appendix B

Proof for Theorem 1: Zero Boolean I/O operation

Outline of proof for soundness: for the input set $A \subseteq \text{Ter}(B)$, it is shown that if $(A,x) \in \text{derive}_0^B(N)$, then $x \in \text{out}_1^B(N,A)$. By definition, $(A,x) \in \text{derive}_0^B(N)$ iff $(a,x) \in \text{derive}_0^B(N)$ for some $a \in A$. By induction on the length of derivation and since $\text{out}_1^B(N)$ validates EQI and EQO, if $(a,x) \in \text{derive}_0^B(N)$, then $x \in \text{out}_1^B(N,\{a\})$. Thus, by definition of $\text{out}_1^B(N)$, we have $x \in \text{out}_1^B(N,A)$. If $A = \{\}$, then by definition $(A,x) \notin \text{derive}_0^B(N)$. The outline works for the soundness of other systems presented in this appendix as well.

Soundness: $\text{out}_1^B(N)$ validates EQI and EQO.

EQI: It needs to be shown that

$$\text{EQI} \quad \frac{x \in \text{Eq}(N(\text{Eq}(a))))}{a = b} \quad x \in \text{Eq}(N(\text{Eq}(b)))$$

If $x \in \text{Eq}(N(\text{Eq}(a))))$, then there are $t_1$ and $t_2$ such that $t_1 = a$ and $t_2 = x$ and $(t_1, t_2) \in N$. If $a = b$ then $t_1 = b$. Hence, by definition, $x \in \text{Eq}(N(\text{Eq}(b)))$.

EQO: It needs to be shown that

$$\text{EQO} \quad \frac{x \in \text{Eq}(N(\text{Eq}(a))))}{x = y} \quad y \in \text{Eq}(N(\text{Eq}(a)))$$

If $x \in \text{Eq}(N(\text{Eq}(a))))$, then there are $t_1$ and $t_2$ such that $t_1 = a$ and $t_2 = x$ and $(t_1, t_2) \in N$. If $x = y$ then $t_2 = y$. Hence, by definition, $y \in \text{Eq}(N(\text{Eq}(a)))$.

Completeness: $\text{out}_1^B(N) \subseteq \text{derive}_0^B(N)$

It is shown that if $x \in \text{Eq}(N(\text{Eq}(a)))$, then $(A,x) \in \text{derive}_0^B(N)$. Suppose that $x \in \text{Eq}(N(\text{Eq}(A)))$, then there are $t_1$ and $t_2$ such that $t_1 = a$ and $a \in A$, and $t_2 = x$ such that $(t_1, t_2) \in N$.

\[\text{For the completeness proofs, if } A = \{\}, \text{ then by definition of } \text{Eq}(\{\}) = \{\} \text{ and } \text{Up}(\{\}) = \{\}, \text{ we have } x \notin \text{out}_1^B(N,\{\}) = \{\}.\]
Thus, $x \in \text{derive}_0^B(N, a)$ and then $x \in \text{derive}_0^B(N, A)$.

Proof for Theorem 1: Simple-I Boolean I/O operation

Soundness: $\text{out}_I^B(N)$ validates SI and EQO.

SI: It needs to be shown that

$$x \in \text{Eq}(N(\text{Up}(a))) \quad b \leq a$$

If $x \in \text{Eq}(N(\text{Up}(a)))$, then $\exists t_1$ such that $a \leq t_1$ and $((t_1, x) \in N$ or $(t_1, y) \in N$ and $y = x)$. Hence, if $b \leq a$, we have $b \leq t_1$ and then $x \in \text{Eq}(N(\text{Up}(b)))$.

EQO: It needs to be shown that

$$x \in \text{Eq}(N(\text{Up}(a))) \quad y \in \text{Eq}(N(\text{Up}(a)))$$

If $x \in \text{Eq}(N(\text{Up}(a)))$, then by definition of $\text{Eq}(X)$, if $x = y$, we have $y \in \text{Eq}(N(\text{Up}(a)))$.

Completeness: $\text{out}_I^B(N) \subseteq \text{derive}_I^B(N)$.

It is shown that if $x \in \text{Eq}(N(\text{Up}(A)))$, then $(A, x) \in \text{derive}_I^B(N)$. Suppose that $x \in \text{Eq}(N(\text{Up}(A)))$, then there is $t_1$ such that $a \leq t_1$ and $(t_1, x) \in N$ or $((t_1, y) \in N$ and $y = x$) for $a \in A$. There are two cases:

$$\text{SI} \quad \frac{(t_1, x)}{(a, x)} \quad \frac{a \leq t_1}{(a, x)}$$

$$\text{EQO} \quad \frac{(t_1, y)}{(a, x)} \quad \frac{y = x}{(a, x)}$$

Thus, $x \in \text{derive}_I^B(N, a)$ and then $x \in \text{derive}_I^B(N, A)$. 
Proof for Theorem 1: Simple-II Boolean I/O operation

Soundness: $\text{out}^B_{II}(N)$ validates WO and EQI.

WO: It needs to be shown that

$$\text{WO} \frac{x \in Up(N(Eq(a)))}{y \in Up(N(Eq(a)))} x \leq y$$

If $x \in Up(N(Eq(a)))$, then there is $t_1$ such that $t_1 \leq x$ and $(a,t_1) \in N$ or $((b,t_1) \in N$ and $a = b)$. If $x \leq y$, then $t_1 \leq y$ and we have $y \in Up(N(Eq(a)))$.

EQI: It needs to be shown that

$$\text{EQI} \frac{x \in Up(N(Eq(a)))}{x \in Up(N(Eq(b)))} a = b$$

If $x \in Up(N(Eq(a)))$, then there is $t_1$ such that $t_1 \leq x$ and $(a,t_1) \in N$ or $((c,t_1) \in N$ and $a = c)$. Hence, if $a = b$, then by definition $x \in Up(N(Eq(b)))$.

Completeness: $\text{out}^B_{II}(N) \subseteq \text{derive}^B_{II}(N)$.

It is shown that if $x \in Up(N(Eq(a)))$, then $(A,x) \in \text{derive}^B_{II}(N)$. Suppose that $x \in Up(N(Eq(A)))$, then there is $t_1$ such that $t_1 \leq x$ and $(a,t_1) \in N$ or $((b,t_1) \in N$ and $a = b)$ for $a \in A$. There are two cases:

$$\text{WO} \frac{(a,t_1)}{(a,x)} t_1 \leq x \quad \text{EQI} \frac{(b,t_1)}{(a,x)} a = b \quad \text{WO} \frac{(a,t_1)}{(a,x)} t_1 \leq x$$

Thus, $x \in \text{derive}^B_{II}(N,a)$ and then $x \in \text{derive}^B_{II}(N,A)$.

Proof for Theorem 1: Simple-minded Boolean I/O operation

Soundness: $\text{out}^B_{I}(N)$ validates SI and WO.

SI: It needs to be shown that

$$\text{SI} \frac{x \in Up(N(Up(a)))}{x \in Up(N(Up(b)))} b \leq a$$
Since \( b \leq a \) we have \( \text{Up}(a) \subseteq \text{Up}(b) \). Hence, \( N(\text{Up}(a)) \subseteq N(\text{Up}(b)) \) and therefore \( \text{Up}(N(\text{Up}(a))) \subseteq \text{Up}(N(\text{Up}(b))) \).

WO: It needs to be show shown that

\[
\begin{align*}
\text{WO} & \quad \frac{x \in \text{Up}(N(\text{Up}(a))) \quad x \leq y}{y \in \text{Up}(N(\text{Up}(a)))}
\end{align*}
\]

Since \( \text{Up}(N(\text{Up}(a))) \) is upward-closed and \( x \leq y \), we have \( y \in \text{Up}(N(\text{Up}(a))) \).

Completeness: \( \text{out}^B_1(N) \subseteq \text{derive}^B_1(N) \).

It is shown that if \( x \in \text{Up}(N(\text{Up}(A))) \), then \( (A, x) \in \text{derive}^B_1(N) \). Suppose that \( x \in \text{Up}(N(\text{Up}(A))) \), then there is \( y_1 \) such that \( y_1 \in N(\text{Up}(A)) \), \( y_1 \leq x \), and there is \( t_1 \) such that \( (t_1, y_1) \in N \) and \( a \leq t_1 \) for \( a \in A \).

\[
\begin{align*}
\text{SI} & \quad a \leq t_1 \quad \text{WO} \quad \frac{(t_1, y_1) \quad y_1 \leq x}{(t_1, x)} \\
& \quad (a, x)
\end{align*}
\]

Thus, \( x \in \text{derive}^B_1(N, a) \) and then \( x \in \text{derive}^B_1(N, A) \).

Proof for Theorem 1: Basic Boolean I/O operation

Soundness: \( \text{out}^B_2(N) \) validates SI, WO and OR.

OR: It needs to be shown that

\[
\begin{align*}
\text{OR} & \quad \frac{x \in \text{out}^B_2(N, \{a\}) \quad x \in \text{out}^B_2(N, \{b\})}{x \in \text{out}^B_2(N, \{a \lor b\})}
\end{align*}
\]

Suppose that \( \{a \lor b\} \subseteq V \), since \( V \) is saturated we have \( a \in V \) or \( b \in V \). Suppose that \( a \in V \), in this case since \( \text{out}^B_2(N, \{a\}) \subseteq \text{Up}(N(V)) \), we have \( x \in \text{out}^B_2(N, \{a \lor b\}) \).
Completeness: $\text{out}_2^B(N) \subseteq \text{derive}_2^B(N)$.

Suppose that $x \notin \text{derive}_2^B(N,A)$, then by monotony of the derivability operation, there is a maximal set $V$ such that $A \subseteq V$ and $x \notin \text{derive}_2^B(N,V)$. Then $V$ is saturated because:

(a) Suppose that $a \in V$ and $a \leq b$, by definition of $V$ we have $(a,x) \notin \text{derive}_2^B(N)$. It needs to be shown that $x \notin \text{derive}_2^B(N,b)$ and since $V$ is maximal, we have $b \in V$. Suppose that $(b,x) \in \text{derive}_2^B(N,V)$. We have

$$\text{SI} \quad (b,x) \quad a \leq b \quad (a,x)$$

That is a contradiction of $(a,x) \notin \text{derive}_2^B(N)$.

(b) Suppose that $a \lor b \in V$, by definition of $V$ we have $x \notin \text{derive}_2^B(N,a \lor b)$. It needs to be shown that $x \notin \text{derive}_2^B(N,a)$ or $x \notin \text{derive}_2^B(N,b)$. Suppose that $x \in \text{derive}_2^B(N,a)$ and $x \in \text{derive}_2^B(N,b)$, then we have

$$\text{OR} \quad (a,x) \quad (b,x) \quad (a \lor b, x)$$

That is a contradiction of $x \notin \text{derive}_2^B(N,a \lor b)$.

Therefore, we have $x \notin U^P(N(V))$ (that is equal to $x \notin \text{out}_1^B(N,V)$) and so $x \notin \text{out}_2^B(N,A)$.

Proof for Theorem 1: Reusable Boolean I/O operation

Soundness: $\text{out}_3^B(N)$ validates $\text{SI}$, $\text{WO}$ and $T$.

T: It needs to be shown that

$$\text{T} \quad x \in \text{out}_3^B(N,\{a\}) \quad y \in \text{out}_3^B(N,\{x\}) \quad y \in \text{out}_3^B(N,\{a\})$$

Suppose that $X$ is the smallest set such that $\{a\} \subseteq X = U^P(X) \supseteq N(X)$. Since $x \in \text{out}_3^B(N,\{a\})$ we have $x \in X$, and from $y \in \text{out}_3^B(N,\{x\})$ we have $y \in X$. Thus, $y \in \text{out}_3^B(N,\{a\})$.

---

\[14\] Consider the set $E = \{V : A \subseteq V$ and $x \notin \text{deriv}(G,V)\}$. This set is a partially ordered set which is ordered by the monotony property of derivation. Every chain (any set linearly ordered by set-theoretic inclusion) has an upper bound (the union of the sets) in $E$. So set $E$ has at least a maximal element by Zorn’s lemma.
Completeness: $\text{out}^R_3(N) \subseteq \text{derive}^R_3(N)$.

Suppose that $x \not\in \text{derive}^R_3(N,a)$. It is necessary to find $B$ such that $a \in B = \text{Up}(B) \supseteq N(B)$ and $x \not\in \text{Up}(N(B))$. Put $B = \text{Up}({a} \cup \text{derive}^R_3(N,a))$. It is shown that $N(B) \subseteq B$. Suppose that $y \in N(B)$, then there is $b \in B$ such that $(b,y) \in N$. It is shown that $y \in B$. Since $b \in B$, there are two cases:

- $b \geq a$: in this case we have $(a,y) \in \text{derive}^R_3(N)$ since $(b,y) \in \text{derive}^R_3(N)$ and we have

$$SI \frac{(b,y)}{(a,y)} \frac{a \leq b}{(a,y)}$$

- $\exists z \in \text{derive}^R_3(N,a), b \geq z$: in this case we have

$$T \frac{(a,z)}{(a,y)} SI \frac{(b,y)}{(z,y)} \frac{z \leq b}{(a,y)}$$

It only needs to shown that $x \not\in \text{Up}(N(B)) = \text{out}^R_1(N, \{a\} \cup \text{derive}^R_3(N,a))$. Suppose that $x \in \text{Up}(N(B))$, then there is $y_1$ such that $x \geq y_1$ and $\exists t_1,$ $(t_1,y_1) \in N$ and $t_1 \in \text{Up}({a} \cup \text{derive}^R_3(N,a))$. There are two cases:

- $t_1 \geq a$: in this case we have

$$SI \frac{(t_1,y_1)}{(a,y_1)} \frac{a \leq t_1}{(a,x)} \frac{y_1 \leq x}{(a,x)}$$

- $\exists z_1 \in \text{derive}^R_3(N,a), z_1 \leq t_1$: in this case we have

$$T \frac{(a,z_1)}{(a,y_1)} SI \frac{(t_1,y_1)}{(z_1,y_1)} \frac{z_1 \leq t_1}{(a,y_1)} \frac{y_1 \leq x}{(a,x)}$$

Thus, in both cases, $(a,x) \in \text{derive}^R_3(N)$ and then $x \in \text{derive}^R_3(N,a)$, and that is a contradiction.
Proof for Theorem 2

The proof is based on the reversibility of inference rules, as studied by Makinson and van der Torre [7].

Lemma 4. Let $D$ be any derivation using at most EQI, SI, WO, OR, AND, CT. Then, there is a derivation $D'$ of the same root from a subset of leaves that applies AND only at the end.

Proof 1. See Observation 18 [7].

The main point of the observation is that it is possible to reverse the order of rules AND, WO to WO, AND; AND, SI to SI, AND; AND, OR to OR, AND and finally AND, CT to SI, CT or CT, AND. It is also possible to reverse the order of the AND and EQI rules as follows:

\[
\begin{align*}
\text{AND} & \quad (a, x) \quad (a, y) \\
\text{EQI} & \quad (a, x \land y) \quad a = b \\
\end{align*}
\]

\[
\begin{align*}
\text{EQI} & \quad (a, x \land y) \quad (b, x \land y) \\
\end{align*}
\]

\[
\begin{align*}
\text{AND} & \quad (b, x) \quad (b, y) \\
\end{align*}
\]

Hence, in each system of \{WO, EQI, AND\}, \{SI, WO, AND\} and \{SI, WO, OR, AND\}, the AND rule can be applied only at the end. Thus, it is possible to characterize $\text{deriv}_{\text{AND}}(N)$ using the fact $\text{deriv}_{\text{SI}}(N) = \text{out}_{\text{SI}}(N)$ and the iterations of AND.

It is easy to check that CT can be reversed with SI, EQI, WO, and EQI by the fact that it is similarly possible to characterize $\text{deriv}_{\text{CT}}(N)$.

Finally, since AND can be reversed with ST, WO and CT, it is possible to characterize $\text{deriv}_{\text{CT,AND}}(N)$ by applying (finite) iterations of AND over $\text{out}_{\text{CT}}(N)$ that means $\text{out}_{\text{CT,AND}}(N)$.

Proof for Theorem 3

The proofs are the same as the soundness and completeness proofs in Theorem 1.

Proof for Theorem 4

This only looks at I/O operations over Boolean algebras since the argument for abstract logics is similar. It needs to be shown that

- $N \subseteq \text{out}_{\text{SI}}(N)$
- $N \subseteq M \Rightarrow \text{out}_{\text{SI}}(N) \subseteq \text{out}_{\text{SI}}(M)$
• $\text{out}_i^B(N) = \text{out}_i^B(\text{out}_i^B(N))$

By the soundness and completeness theorems, we have $\text{out}_i^B(N) = \text{derive}_i^B(N)$. So $\text{derive}_i^B(N)$ is studied, which is more simple than $\text{out}_i^B(N)$. The first two properties are clear from the definition of $\text{derive}_i^B$. For the last property, it needs to be shown that $\text{derive}_i^B(N) = \text{derive}_i^B(\text{derive}_i^B(N))$. We have $\text{derive}_i^B(\text{derive}_i^B(N)) = \text{derive}_i^B(\{(A, x)|(a, x) \in \text{derive}_i^B(N) \text{ for some } a \in A\}) = \{(A, x)|(a, x) \in \text{derive}_i^B(N) \text{ for some } a \in A\}$ since $N \subseteq \{(A, x)| (a, x) \in \text{derive}_i^B(N) \text{ for some } a \in A\}$ and the same rules apply over $\text{derive}_i^B(N)$. Actually, it needs to be shown that if $(a, x) \in \text{derive}_i^B(N)$, then $\text{derive}_i^B(N) = \text{derive}_i^B(N \cup \{(a, x)\})$ holds for $\text{derive}_i^B$.

Appendix C

Proof for Theorem 5

See [46, 9].

Proof for Theorem 6

Here is the proof for the case of $i = 1$.

• Suppose that $(\varphi, \psi) \in \text{derive}_1^{\text{Fm}(X)}(N)$. For an arbitrary valuation $V$ and arbitrary Boolean algebra $B \in \text{BA}$, it needs to be shown that $V(\psi) \in \text{out}_1^B(N^V, \{V(\varphi)\})$. The proof is by induction on the length of the proof $(\varphi, \psi) \in \text{derive}_1^{\text{Fm}(X)}(N)$.

Base case: If $(\varphi, \psi) \in N$, then $(V(\varphi), V(\psi)) \in N^V$ by definition, and we have $V(\psi) \in \text{out}_1^B(N^V, \{V(\varphi)\})$.

Inductive step: It is shown that for $n > 0$, if $V(\psi) \in \text{out}_1^B(N^V, \{V(\varphi)\})$ holds for $n$, then $V(\psi) \in \text{out}_1^B(N^V, \{V(\varphi)\})$ also holds for $n + 1$.

Suppose that the length of proof $(\varphi, \psi) \in \text{derive}_1^{\text{Fm}(X)}(N)$ is $n + 1$. There are two possibilities:

– Using SI in the last step: There is $\phi$ such that $(\varphi, \psi) \in \text{derive}_1^{\text{Fm}(X)}(N)$ and $\varphi \vdash_C \phi$. In this case, by the induction step we have $V(\psi) \in \text{out}_1^B(N^V, \{V(\phi)\})$, and by the completeness of the simple-minded operation we have $(V(\phi), V(\psi)) \in \text{derive}_1^B(N)$. Since $\varphi \vdash_C \phi$, then by Theorem 5 we have $\varphi \vdash_{\text{BA}} \phi$. So $V(\varphi) \land V(\phi) = V(\varphi)$.

44
Proof for Theorem 7

\[ \text{derive}^L_{\text{Fm}(X)}(N), \text{and by the soundness of the simple-minded operation we have } V(\psi) \in \text{out}^B_{\text{Fm}(X)}(N, \{V(\varphi)\}). \]

- **Using WO in the last step:** There is \( \phi \) such that \( (\varphi, \phi) \in \text{derive}^\text{Fm}(X)_{\text{Fm}(X)}(N) \) and \( \phi \vdash_C \psi \). In this case, by the induction step we have \( V(\phi) \in \text{out}^B_{\text{Fm}(X)}(N^V, \{V(\varphi)\}) \), and by the completeness of the simple-minded operation we have \( (V(\varphi), V(\phi)) \in \text{derive}^B_{\text{Fm}(X)}(N) \).

\[ \text{Since } \phi \vdash_C \psi, \text{ then by Theorem 5 we have } \phi \models_{\text{BA}} \psi. \]

\[ \text{So } V(\phi) \land V(\psi) = V(\phi). \]

Then from \((V(\varphi), V(\psi)) \in \text{derive}^B_{\text{Fm}(X)}(N)\) and \(V(\phi) \leq V(\psi)\) using the WO rule we have \((V(\varphi), V(\psi)) \in \text{derive}^B_{\text{Fm}(X)}(N)\), and by the soundness of the simple-minded operation we have \( V(\psi) \in \text{out}^B_{\text{Fm}(X)}(N^V, \{V(\varphi)\}) \).

\[ \bullet \text{ The proof in the other direction is by contraposition. Suppose that } (\varphi, \psi) \notin \text{derive}^\text{Fm}(X)_{\text{Fm}(X)}(N), \text{ if } \text{Fm}(X) \text{ is taken as a Boolean algebra, then by the completeness of } \text{derive}^\text{Fm}(X)_{\text{Fm}(X)}(N), \text{ we have } \psi \notin \text{out}^\text{Fm}(X)_{\text{Fm}(X)}(N, \{\varphi\}). \]

Then it is enough that the valuation function is put as the identity function on the Boolean algebra \( \text{Fm}(X) \) which means \( \psi \notin \text{out}^\text{Fm}(X)_{\text{Fm}(X)}(N, \{\varphi\}) \).

The proof is similar for the other derivation systems: \( \text{derive}^{\text{Fm}(X)}_{\text{Fm}(X)}(N) \), \( \text{derive}^L_{\text{Fm}(X)}(N) \), \( \text{derive}^1_{\text{Fm}(X)}(N) \), \( \text{derive}^{11}_{\text{Fm}(X)}(N) \), \( \text{derive}^2_{\text{Fm}(X)}(N) \), and \( \text{derive}^3_{\text{Fm}(X)}(N) \).

**Proof for Theorem 2**

\[ \bullet \text{ The proof from right to left is similar to Theorem 6. It just needs to be checked that for the case when AND is the last step of the derivation, that there are } \delta_1 \text{ and } \delta_2 \text{ such that } (\varphi, \delta_1), (\varphi, \delta_2) \in \text{derive}^{1\text{AND}}_{\text{Fm}(X)}(N) \]

\[ \text{and } \psi = \delta_1 \land \delta_2. \]

In this case, by the induction step we have \( V(\delta_1) \in \text{out}^{\text{AND}}_{\text{Fm}(X)}(N^V, \{V(\varphi)\}) \) and \( V(\delta_2) \in \text{out}^{\text{AND}}_{\text{Fm}(X)}(N^V, \{V(\varphi)\}) \). By Theorem 2, we have \((\varphi, \delta_1) \in \text{derive}^{1\text{AND}}_{\text{Fm}(X)}(N) \) and \((\varphi, \delta_2) \in \text{derive}^{1\text{AND}}_{\text{Fm}(X)}(N) \).

Then by using the AND rule, we have \((\varphi, \delta_1 \land \delta_2) \in \text{derive}^{1\text{AND}}_{\text{Fm}(X)}(N) \), and then by Theorem 2, we have \( V(\psi) \in \text{out}^{1\text{AND}}_{\text{Fm}(X)}(N^V, \{V(\varphi)\}) \).

\[ \bullet \text{ The proof in the other direction is by contraposition. Suppose that } (\varphi, \psi) \notin \text{derive}^{1\text{AND}}_{\text{Fm}(X)}(N), \text{ if } \text{Fm}(X) \text{ is taken as a Boolean algebra, then by Theorem 2 we have } \psi \notin \text{out}^{1\text{AND}}_{\text{Fm}(X)}(N, \{\varphi\}), \text{ then if the valuation function is put as the identity function on the algebra } \text{Fm}(X), \text{ we have } \psi \notin \text{out}^{1\text{AND}}_{\text{Fm}(X)}(N, \{\varphi\}). \]
It is possible to extend the proof for the arbitrary input set \( \Gamma \subseteq Fm(X) \) and to extend this theorem for other addition rule operators.

**Proof for Theorem 8**

- From left to right: Suppose that \((\varphi, \psi) \in derive_{i}^{Con}(N)\), then by definition, \((\varphi, \psi) \in derive_{i}^{Fm(X)}(N)\) and \(Con, \psi \not\subset C\perp\). From Theorem 6 we have "\(V(\psi) \in out_{i}^{B} (N^V, \{V(\varphi)\})\) for every \(B \in BA\) and valuation \(V\)". From Theorem 5 there is a Boolean algebra \(B\) such that \(Con, \psi \not\subset B\perp\). So there is a valuation \(V\) on \(B\) such that \(\forall \delta \in Con, V(\delta \land \psi) = 1_{B}\).

- The proof from right to left is similar.

By the definition of \(derive_{i}^{Con}(N)\), it is possible to extend the theorem for the case of \((\Gamma, \psi) \in derive_{i}^{Con}(N)\) where \(\Gamma \subseteq Fm(X)\).

**Proof for Theorem 9**

- From left to right: Suppose that \(\varphi > \bigcirc \psi \in derive_{i}^{OH}(N)\), by definition, \((\varphi, \psi) \in derive_{i}^{Fm(X)}(N)\) and from Theorem 6 we have "\(V(\psi) \in out_{i}^{B} (N^V, \{V(\varphi)\})\) for every \(B \in BA\) and valuation \(V\)". For the second part, notice that every maximal consistent subset defines a valuation and vice versa. So "\(\forall M \in opt_{f}(\psi) (\psi \in M)\)" is equivalent to that for any valuation \(V_{i} \in opt_{\geq f}(\varphi)\), so that we have \(V_{i}(\psi) = 1_{B}\) and vice versa.

- From right to left, the proof is similar.

By the definition of \(derive_{i}^{OH}(N)\), it is possible to extend the theorem for the case of \(\Gamma > \bigcirc \psi \in derive_{i}^{OH}(N)\) where \(\Gamma \subseteq Fm(X)\).

**Proof for Theorem 10**

- From left to right: Suppose that \((\varphi, \psi) \in derive_{i}^{OK}(N)\), by definition, \((\varphi, \psi) \in derive_{i}^{Fm(X)}(N)\) and from Theorem 6 we have "\(V(\psi) \in out_{i}^{B} (N^V, \{V(\varphi)\})\) for every \(B \in BA\) and valuation \(V\)". For the second part, notice that every maximal consistent subset defines a valuation and vice versa. So "\(\forall M \in opt_{fA}(\varphi) (\psi \in M)\)" is equivalent to that for any valuation \(V_{i} \in opt_{\geq fA}(\varphi)\), so that we have \(V_{i}(\psi) = 1_{B}\) and vice versa.

- From right to left, the proof is similar.

By the definition of \(derive_{i}^{OK}(N)\), it is possible to extend the theorem for the case of \(\Gamma > \bigcirc \psi \in derive_{i}^{OK}(N)\) where \(\Gamma \subseteq Fm(X)\).
Appendix D

Introduction to higher-order logic

This brief introduction to classical higher-order logic (HOL) is adapted from [3].

HOL is based on simple typed \( \lambda \)-calculus. It is assumed that the set \( \mathcal{T} \) of simple types is freely generated from a set of basic types \( \{ o, i \} \) using the function type constructor \( \to \). Type \( o \) denotes the set of Booleans where type \( i \) refers to a non-empty set of individuals.

For \( \alpha, \beta, o \in \mathcal{T} \), the language of HOL is generated as follows:

\[
s, t ::= p_\alpha | X_\alpha | (\lambda X_\alpha s_\beta)_{\alpha \to \beta} | (s_{\alpha \to \beta} t_\alpha)_\beta
\]

where \( p_\alpha \) represents a typed constant symbol (from a possibly infinite set \( \mathcal{P}_\alpha \) of such constant symbols) and \( X_\alpha \) represents a typed variable symbol (from a possibly infinite set \( \mathcal{V}_\alpha \) of such symbols). \( (\lambda X_\alpha s_\beta)_{\alpha \to \beta} \) and \( (s_{\alpha \to \beta} t_\alpha)_\beta \) are called abstraction and application respectively. HOL is a logic of terms in the sense that the formulas of HOL are given as terms of type \( o \). Moreover, a sufficient number of primitive logical connectives are required in the signature of HOL, i.e., these logical connectives must be contained in the sets \( \mathcal{P}_\alpha \) of constant symbols. The primitive logical connectives of choice in this article are \( \neg o \to o \), \( \lor o \to o \to o \), \( \Pi (\alpha \to o) \to o \) and \( = \alpha \to \alpha \to o \). The symbols \( \Pi (\alpha \to o) \to o \) and \( = \alpha \to \alpha \to o \) are generally assumed for each type \( \alpha \in \mathcal{T} \). From the selected set of primitive connectives, other logical connectives can be introduced as abbreviations. Type information as well as brackets may be omitted if obvious from the context, and infix notation may also be used to improve readability. For example, \( (s \lor t) \) may be written instead of \( ((\lor o \to o s_\alpha) t_\alpha)_o \). Often, \( \forall X_\alpha s_\alpha \) is written as syntactic sugar for \( \Pi (\alpha \to o) \to o (\lambda X_\alpha s_\alpha) \).

The notions of free variables, \( \alpha \)-conversion, \( \beta\eta \)-equality and substitution of a term \( s_\alpha \) for a variable \( X_\alpha \) in a term \( t_\beta \), denoted as \( [s/X]t \), are defined as usual.

The semantics of HOL are well understood and thoroughly documented [15]. In this article, the semantics of choice is Henkin’s general models [47].

A frame \( D \) is a collection \( \{ D_\alpha \}_{\alpha \in \mathcal{T}} \) of nonempty sets \( D_\alpha \) such that \( D_o = \{ T, F \} \), denoting truth and falsehood respectively. \( D_{\alpha \to \beta} \) represents a collection of functions mapping \( D_\alpha \) into \( D_\beta \).

A model for HOL is a tuple \( M = \langle D, I \rangle \), where \( D \) is a frame and \( I \) is a family of typed interpretation functions mapping constant symbols \( p_\alpha \) to appropriate elements of \( D_\alpha \) called the denotation of \( p_\alpha \). The logical connectives
\(\neg, \lor, \Pi \text{ and } =\) are always given in their expected standard denotations. A variable assignment \(g\) maps variables \(X_\alpha\) to elements in \(D_\alpha\) while \(g[d/W]\) denotes the assignment that is identical to \(g\), except for the variable \(W\), which is now mapped to \(d\). The denotation \(\|s_\alpha\|^M,g\) of a HOL term \(s_\alpha\) on a model \(M = \langle D, I \rangle\) under assignment \(g\) is an element \(d \in D_\alpha\) and is defined in the following way:

\[
\begin{align*}
\|p_\alpha\|^M,g &= I(p_\alpha) \\
\|X_\alpha\|^M,g &= g(X_\alpha) \\
\|(s_\alpha \to t_\beta)\|^M,g &= \|s_\alpha\|^M,g(\|t_\alpha\|^M,g) \\
\|(\lambda X_\alpha s_\beta\alpha \to \beta\|^M,g &= \text{the function } f \text{ from } D_\alpha \text{ to } D_\beta \text{ such that } \\
f(d) &= \|s_\beta\|^M,g[d/X_\alpha]\text{ for all } d \in D_\alpha
\end{align*}
\]

Since \(I(\neg o \to o), I(\lor o \to o), I(\Pi(o \to o) \to o)\) and \(I(=o \to o)\) always denote the standard truth functions, we have:

1. \(\|(\neg o \to o)\|^M,g = T\) iff \(\|s_o\|^M,g = F\).
2. \(\|(\lor o \to o)\|^M,g = T\) iff \(\|s_o\|^M,g = T\) or \(\|t_o\|^M,g = T\).
3. \(\|(\forall X_\alpha s_\beta\alpha \to \beta\|^M,g = \|(\Pi(o \to o) \to o(\lambda X_\alpha s_o))\|^M,g = T\) iff for all \(d \in D_\alpha\) we have \(\|s_o\|^M,g[d/X_\alpha] = T\).
4. \(\|(=o \to o)\|^M,g = T\) iff \(\|s_o\|^M,g = \|t_o\|^M,g\).

From the selected set of primitive connectives, other logical connectives can be introduced as abbreviations. An HOL formula \(s_o\) is true in a Henkin model \(M\) under the assignment \(g\) if and only if \(\|s_o\|^M,g = T\). This is also expressed with the notation \(M, g \models^{HOL} s_o\). An HOL formula \(s_o\) is called valid in \(M\), denoted as \(M \models^{HOL} s_o\), if and only if \(M, g \models^{HOL} s_o\) for all assignments \(g\). Moreover, a formula \(s_o\) is called valid, denoted as \(\models^{HOL} s_o\), if and only if \(s_o\) is valid in all Henkin models \(M\). Finally, \(\Sigma \models^{HOL} s_o\) is defined for a set of HOL formulas \(\Sigma\) if and only if \(M \models^{HOL} s_o\) for all Henkin models \(M\) with \(M \models^{HOL} t_o\) for all \(t_o \in \Sigma\).

**Proof for Lemma 7**

The proof is straightforward. For example, for COM \(\lor\) we have the following:
COM $\lor$:

For all $a, b \in D_i$: $I \lor_{i \to i} a b = I \lor_{i \to i} b a$
(from the definition of $I \lor_{i \to i}$ and $\lor$)

$\Leftrightarrow$

For all assignments $g$, for all $a, b \in D_i$

$\| X \lor Y = Y \lor X \|^{|H^N,g|a/X|b/Y|} = T$

$\Leftrightarrow$

For all $g$, we have $\| \forall X \forall Y (X \lor Y = Y \lor X) \|^{|H^N,g|} = T$

$\Leftrightarrow$

$H^N \models^{\text{HOL}} \text{COM} \lor$

Proof for Lemma 3

Fact: notice that for all $\varphi \in \text{Fm}(X)$ and for all assignments $g$ by induction on the structure of $\varphi$, we have $\| [\varphi] \|^{|H^N,g|} = V(\varphi)$.

For simplification, the term abbreviations are used for the saturated set, the $\leq$ ordering and upward set. It is easy to see that these terms abbreviations have the same corresponding sets in the corresponding Henkin model as in the Boolean algebra.

Here then is the proof:

$(d_1(N))$

$\| [d_1(N)(\varphi, \psi)] \|^{|H^N,g|} = T$

$\Leftrightarrow$ $\| ([\bigcirc_1(N)_{\tau \to r} \{[\varphi]\}) \{\psi\} \|^{|H^N,g|} = T$

$\Leftrightarrow$ $\| (\lambda A_1 \lambda X_1 (\exists U (\exists Y (\exists Z (A Z \land Z \leq Y \land N Y U \land U \leq X)))) \{[\varphi]\}) \{\psi\} \|^{|H^N,g|} = T$

$\Leftrightarrow$ $\| (\lambda X_1 (\exists U (\exists Y (\exists Z ([\varphi]) Z \land Z \leq Y \land N Y U \land U \leq X))) \{[\varphi]\}) \{\psi\} \|^{|H^N,g|} = T$

$\Leftrightarrow$ $\| \exists U (\exists Y (\exists Z ([\varphi]) Z \land Z \leq Y \land N Y U \land U \leq [\psi]))) \|^{|H^N,g|} = T$

$\Leftrightarrow$ $\| \forall Y (\exists Z ([\varphi]) Z \land Z \leq Y \land N Y U \land U \leq [\psi]))) \|^{|H^N,g|} = T$

$\Leftrightarrow$ There are elements $b$ and $c$ such that $b, c \in D_i$ and $\| [\varphi] \leq Y \land N Y U \land U \leq [\psi]) \|^{|H^N,g|b/U|c/Y|} = T$

$\Leftrightarrow$ There are elements $b, c \in \bar{B}$ such that $V(\varphi) \leq c \land N^V c b \land b \leq V(\psi)$

$\Leftrightarrow$ $V(\psi) \in Up(N^V (Up\{V(\varphi)\}))$

$\Leftrightarrow$ $V(\psi) \in \text{out}^B(N^V, \{V(\varphi)\})$
(d₂(N))

\[
\| d₂(N)(\varphi, \psi) \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\check{O}_2(N)_{\tau \rightarrow \tau} \{[\varphi]\}) \{[\psi]\} \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\lambda A_τ \lambda X_i (\forall V (\text{Saturated} V \land \forall U (A U \rightarrow V U))
\rightarrow \exists Y (\exists Z (Z \leq X \land N Y Z \land V Y))) \{[\varphi]\}) \{[\psi]\} \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\lambda X_i (\forall V (\text{Saturated} V \land \forall U (\{\varphi\}) U \rightarrow V U))
\rightarrow \exists Y (\exists Z (Z \leq X \land N Y Z \land V Y))) \{[\psi]\} \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\forall V (\text{Saturated} V \land \forall U (\{\varphi\}) U \rightarrow V U)
\rightarrow \exists Y (\exists Z (Z \leq \varphi \land N Y Z \land V Y))) \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\text{There are elements } b \text{ and } c \text{ such that } b, c \in D_i \text{ and } c \leq V(\psi) \land N V b \land V b
\]
\[\iff\]
\[
\text{For every saturated set } V \text{ such that } \{V(\varphi)\} \subseteq V, \text{ we have } V(\psi) \in U p(N^V(V))\]
\[\iff\]
\[
V(\psi) \in \text{out}^S_N(N^V, \{V(\varphi)\})
\]

(d₃(N))

\[
\| d₃(N)(\varphi, \psi) \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\check{O}_3(N)_{\tau \rightarrow \tau} \{[\varphi]\}) \{[\psi]\} \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\lambda A_τ \lambda X_i (\forall V (\forall U (A U \rightarrow V U) \land V = U p V)
\land \forall W (\exists Y (V Y \land N Y W) \rightarrow V W))
\rightarrow \exists Y (\exists Z (Z \leq X \land N Y Z \land V Y))) \{[\varphi]\}) \{[\psi]\} \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\| (\lambda X_i (\forall V (\forall U (\{\varphi\}) U \rightarrow V U) \land V = U p V)
\land \forall W (\exists Y (V Y \land N Y W) \rightarrow V W))
\rightarrow \exists Y (\exists Z (Z \leq \varphi \land N Y Z \land V Y))) \|^{H_{N, g}} = T
\]
\[\iff\]
\[
\text{There are elements } b \text{ and } c \text{ such that } b, c \in D_i \text{ and } c \leq V(\psi) \land N V b \land V b
\]
\[\iff\]
\[
\text{For every saturated set } V \text{ such that } \{V(\varphi)\} \subseteq V, \text{ we have } V(\psi) \in U p(N^V(V))\]
\[\iff\]
\[
V(\psi) \in \text{out}^S(N^V, \{V(\varphi)\})
\]
For every set $V$ that $\text{Up}(V) = V$, $\{V(\varphi)\} \subseteq V$ and $N^V(V) \subseteq V$, there are elements $b, c \in B$ such that

$$c \leq V(\psi) \land N^V b \land V b$$

For every set $V$ that $\text{Up}(V) = V$, $\{V(\varphi)\} \subseteq V$ and $N^V(V) \subseteq V$, we have $V(\psi) \in \text{Up}(N^V(V)))$.

$V(\psi) \in \text{out}^3_B(N, \{V(\varphi)\})$

Proof for Lemma 3

Suppose that $H = \langle \{D_\alpha\}_{\alpha \in T}, I \rangle$ is a Henkin model such that $H \models^{\text{HOL}} \Sigma$ for all $\Sigma \in \{\text{COM} \lor, ..., \text{Dis} \land \lor\}$. Without loss of generality, it can be assumed that the domains of $H$ are denumerable. The corresponding Boolean normative model $N$ is constructed as follows:

- $B = D_i$.
- $1 = I \top_i$.
- $0 = I \bot_i$.
- $a \lor b = c$ for $a, b, c \in B$ iff $I \lor_{i \leftarrow i} ab = c$.
- $a \land b = c$ for $a, b, c \in B$ iff $I \land_{i \leftarrow i} ab = c$.
- $a = \neg b$ for $a, b \in B$ iff $I \neg_{i \leftarrow i} a = b$.
- The valuation on $B$ is defined such that for all $p^j \in X$, $V(p^j) = I(p^j)$.
- $(a, b) \in N^V$ for $a, b \in B$ iff $IN_{i \leftarrow i} ab = T$.

Since $H \models^{\text{HOL}} \Sigma$ for all $\Sigma \in \{\text{COM} \lor, ..., \text{Dis} \land \lor\}$, it is straightforward (but tedious) to verify that $\land, \lor, \neg, 0$ and $1$ satisfy the conditions required for a Boolean algebra.

Moreover, the above construction ensures that $H$ is a Henkin model $H^N$ for Boolean normative model $N$. Hence, Lemma 2 applies. This ensures that for all conditional norms $(\varphi, \psi)$, and for all assignment $g$, we have:

$$\|d_i(N)(\varphi, \psi)\|^{H,g} = T$$

if and only if $V(\psi) \in \text{out}^3_B(N^V, \{V(\varphi)\})$.

Proof for Theorem 11

Soundness

The proof is by contraposition. Suppose that for a Boolean normative model $\langle B, V, N^V \rangle$, we have $V(\psi) \notin \text{out}^3_B(N^V, \{V(\varphi)\})$. Now let $H^N$ be a Henkin model for Boolean normative model $N$. Then by Lemma 2 for
an arbitrary assignment $g$, it is held that $\|d_i(N)(\varphi, \psi)\|^H_{N,g} = F$, but $\|\text{COM} \lor^H_{N,g} = T$, $\ldots$, $\|\text{Dis} \land \lor^H_{N,g} = T$, and that is a contradiction.

Completeness

The proof is again by contraposition. If it is assumed that $\{\text{COM} \lor, \ldots, \text{Dis} \land \lor\} \not\models^\text{HOL} [d_i(N)(\varphi, \psi)]$, then there is a Henkin model $H = \langle\{D_\alpha\}_{\alpha \in T}, I\rangle$ such that $H \models^\text{HOL} \Sigma$ for all $\Sigma \in \{\text{COM} \lor, \ldots, \text{Dis} \land \lor\}$, but $\|d_i(N)(\varphi, \psi)\|^H_{g} = F$ for some assignment $g$. By Lemma 3, there is a Boolean normative model $N$ such that $V(\psi) \not\in \text{out}_B^N(N^V, \{V(\varphi)\})$, and that is a contradiction.