I/O Logic in HOL*

Ali Farjami[1], Paul Meder[1], Xavier Parent[1], and Christoph Benzmüller[1,2]

[1] University of Luxembourg, Luxembourg
[2] Freie Universität Berlin, Germany
`farjami110@gmail.com`
`paul.meder.001@student.uni.lu`
`xavier.parent@uni.lu`
`c.benzmueller@gmail.com`

Abstract. A shallow semantical embedding of Input/Output logic in classical higher-order logic is presented, and shown to be faithful (sound an complete). This embedding has been implemented in the higher-order proof assistant Isabelle/HOL. We provide an empirical regulative framework for assessing General Data Protection Regulation.

Keywords: Input/output logic · Isabelle/HOL · General Data Protection Regulation.

1 Introduction

Deontic logic is a reasoning framework about normative concepts such as obligation, permission and prohibition. While in traditional deontic logic the focus is on inference patterns, in the modern approach ("norm-based") the logical result is based on application of a specific normative system (a set of conditional norms) on a given input (e.g. a fact). Input/Output (I/O) logic is a seminal rule based system that is addressed in the Handbook of deontic logic as a chapter. The advantage of I/O logic is that there is no need truth function (in terms of truth-values or possible worlds) for a set of conditional norms. The framework is expressive enough for dealing with legal concepts such as constitutive, prescriptive and defensible rules [12].

We propose a deontic reasoner based on I/O logic. *Basic Output* and *Basic Reusable Output* are two important I/O semantics that can be formulated with possible worlds semantics. We encode these two I/O semantics in classical higher-order logic (HOL). The syntax and semantics of HOL are well understood [6] and there exist automated proof tools for it; examples include Isabelle/HOL [22] and Leo-II [11]. For embedding *Basic Output* and *Basic Reusable Output* in HOL, we use the shallow semantical embedding of Kripke semantics (**K** and **KT**) in classical higher-order logic. The semantical embedding is faithful.

Our embedding has been encoded in Isabelle/HOL to enable experiments for regulative frameworks. We examined General Data Protection Regulation

* This work has been supported by the European Unions Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 690974.

(GDPR) in our implementation. The experiments with this environment provide evidence that this logic implementation fruitfully enables interactive and automated reasoning at the meta-level and the object level.

The article is structured as follows: Sec. 2 is a quick review about modal logic and higher-order logic and Sec. 3 introduces I/O logic. The semantical embedding of *Basic Output* and *Basic Reusable Output* in HOL is then devised and studied in Sec. 4. This section also shows the faithfulness (viz. soundness and completeness) of the embedding. In Sec. 5 we describe GDPR in our framework.

2 Preliminaries

In this section we assume familiarity with modal logic and simple type theory. In the following we briefly mention the most important notions. The syntax of modal logic **K** is based on propositional logic plus a modal operator $\Box$.

$$\varphi, \psi ::= p | \neg\varphi | \varphi \vee \psi | \Box\varphi$$

where p denotes atomic formulas and other logical connectives can be defined in the usual way. For the axiomatization take all the classical tautologies and all the formula of the form $\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$ (called axiom K) and as rules modes pones (from φ, $\varphi \to \psi$ to ψ) and necessitation rule (from φ to $\Box\varphi$ for all the formula).

A Kripke frame for **K** is a pair $\langle S, R\rangle$, where S is a non-empty set of possible worlds and R is a binary relation on S (called accessibility relation). A Kripke model for **K** is a triple $M = \langle S, R, V\rangle$, where $\langle S, R\rangle$ is a Kripke frame, and V is a function assigning a set of worlds to each atomic formula, that is, $V(p) \subseteq S$.

Satisfiability of a formula φ for a model $M = \langle S, R, V\rangle$ and a world $s \in S$ is expressed by writing that $M, s \models \varphi$ and we define $V^M(\varphi) = \{s \in S | M, s \models \varphi\}$.

$$\begin{array}{lll} M, s \models & p & \text{if and only if } s \in V(p) \\ M, s \models & \neg\varphi & \text{if and only if } M, s \not\models \varphi \text{ (that is, not } M, s \models \varphi) \\ M, s \models & \varphi \vee \psi & \text{if and only if } M, s \models \varphi \text{ or } M, s \models \psi \\ M, s \models & \Box\varphi & \text{if and only if for all } t,\ sRt\ M, t \models \varphi \end{array}$$

As usual, a modal formula φ is valid in a Kripke model $M = \langle S, R, V\rangle$, i.e. $M \models^K \varphi$, if and only if for all worlds $s \in S$ we have $M, s \models \varphi$. A formula φ is *valid*, denoted $\models^K \varphi$, if and only if it is valid in every Kripke model.

System **K** is sound and complete respect the class of all Kripke models. If we add other axioms to system **K** such as $T : \Box\varphi \to \varphi$, $4 : \Box\varphi \to \Box\Box\varphi$, $5 : \Diamond\varphi \to \Box\Diamond\varphi$ the corresponding class of Kripke models should be reflexive, transitive, euclidean.

Classical higher-order logic (HOL) is based on simple typed λ-calculus. We assume that the set T of simple types is freely generated from a set of basic types $\{o, i\}$ using function type constructor $\to$. Type o denotes the set of Booleans, and i a non-empty set of individuals.

The *language of HOL* is defined by $(\alpha, \beta, o \in T)$

$$\begin{aligned}s,t ::= p_\alpha | X_\alpha | (\lambda X_\alpha s_\beta)_{\alpha\to\beta} | (s_{\alpha\to\beta}\, t_\alpha)_\beta | (\neg_{o\to o} s_o)_o | \\ (s_o \vee_{o\to o\to o} t_o)_o | (\Pi_{(\alpha\to o)\to o} s_{\alpha\to o})_o\end{aligned}$$

where p_α denotes typed constant symbol and X_α typed variable symbol. HOL is a logic of terms in the sense that the *formulas of HOL* are given as the terms of type o. Our selected primitive logical connectives are $\neg_{o\to o}$, $\vee_{o\to o\to o}$ and $\Pi_{(\alpha\to o)\to o}$.

The notions of *free variables*, *α-conversion*, *$\beta\eta$-equality* (denoted as $=_{\beta\eta}$) and *substitution* of a term s_α for a variable X_α in a term t_β (denoted as $[s/X]t$) are defined as usual.

The semantics of HOL is well understood and thoroughly documented. The semantics of choice for the remainder is Henkin's general models [20]. For more details we refer to literature [6].

A *frame* D is a collection $\{D_\alpha\}_{\alpha\in T}$ of nonempty sets D_α, such that $D_o = \{T, F\}$ (for truth and falsehood). The $D_{\alpha\to\beta}$ are collections of functions mapping D_α into D_β.

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_α to appropriate elements of D_α, called the *denotation of* p_α. The logical connectives $\neg$, $\vee$, Π and $=$ are always given their expected, standard denotations. A *variable assignment* g maps variables X_α to elements in D_α. $g[d/W]$ denotes the assignment that is identical to g, except for variable W, which is now mapped to d. The *denotation* $\|s_\alpha\|^{M,g}$ of an HOL term s_α on a model $M = \langle D, I\rangle$ under assignment g is an element $d \in D_\alpha$ defined in the following way:

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

Since $I\neg_{o\to o}$, $I\vee_{o\to o\to o}$ and $I\Pi_{(\alpha\to o)\to o}$ always denote the standard truth functions we have

1. $\|(\neg_{o\to o}\, s_o)_o\|^{M,g} = T$ iff $\|s_o\|^{M,g} = F$.
2. $\|((\vee_{o\to o\to o}\, s_o)\, t_o)_o\|^{M,g} = T$ iff $\|s_o\|^{M,g} = T$ or $\|t_o\|^{M,g} = T$.
3. $\|(\forall X_\alpha s_o)_o\|^{M,g} = \|(\Pi_{(\alpha\to o)\to o}(\lambda X_\alpha s_o))_o\|^{M,g} = T$ iff for all $d \in D_\alpha$ we have $\|s_o\|^{M,g[d/X_\alpha]} = T$.

An HOL formula s_o is *true* in an Henkin model M under assignment g if and only if $\|s_o\|^{M,g} = T$; this is also expressed by writing that $M, g \models^{\mathrm{HOL}} s_o$. An HOL formula s_o is called *valid* in M, which is expressed by writing that $M \models^{\mathrm{HOL}} s_o$, if and only if $M, g \models^{\mathrm{HOL}} s_o$ for all assignments g. Moreover, a formula s_o is called *valid*, expressed by writing that $\models^{\mathrm{HOL}} s_o$, if and only if s_o is valid in all Henkin models M. Finally, we define $\Sigma \models^{\mathrm{HOL}} s_o$ for a set of HOL formulas Σ if and only if $M \models^{\mathrm{HOL}} s_o$ for all Henkin models M with $M \models^{\mathrm{HOL}} t_o$ for all $t_o \in \Sigma$.

3 Input/Output logic

Input/Output logic was initially developed by Makinson and van der Torre, and introduced in [21]. I/O logic focuses on the reasoning and studying of conditional norms. The literature has presented several different I/O operators. For this work, we will focus on *Basic Output* and *Basic Reusable Output.*

3.1 Semantics

$G \subseteq \mathcal{L} \times \mathcal{L}$ is called a normative system, with $\mathcal{L}$ representing the set of all the formulas of propositional logic. A pair $(a, x) \in G$ is referred to as a conditional norm or obligation, where a and x are formulas of propositional logic. a is called the body and represents some situation or condition, whereas x is called the head and represents what is obligatory or desirable in that situation. Thus the pair (a, x) is read as 'given a, it is obligatory that x'. For a set of formulas A, we have that $G(A) = \{x \mid (a, x) \in G \text{ for some } a \in A\}$ and $Cn(A) = \{x \mid A \vdash x\}$ with $\vdash$ denoting the classical propositional consequence relation. A set of formulas is considered as *complete* if it is either *maximal consistent* or equal to $\mathcal{L}$.

Definition 1 (Basic Output). *Given a set of conditional norms G and an input set A of propositional formulas,*

$$out_2(G, A) = \bigcap\{Cn(G(V)) \mid A \subseteq V, V \ \ complete\}$$

Definition 2 (Basic Reusable Output). *Given a set of conditional norms G and an input set A of propositional formulas,*

$$out_4(G, A) = \bigcap\{Cn(G(V)) \mid A \subseteq V \supseteq G(V), V \ \ complete\}$$

Besides those traditional formulations of the operators, [21] documents modal formulations for out_2 and out_4.

Theorem 1. *$x \in out_2(G, A)$ if and only if $x \in Cn(G(\mathcal{L}))$ and $G^\Box \cup A \vdash_S \Box x$ for any modal logic $\boldsymbol{S}$ with $\boldsymbol{K_0} \subseteq \boldsymbol{S} \subseteq \boldsymbol{K45}$.*

Theorem 2. *$x \in out_4(G, A)$ if and only if $x \in Cn(G(\mathcal{L}))$ and $G^\Box \cup A \vdash_S \Box x$ for any modal logic $\boldsymbol{S}$ with $\boldsymbol{K_0 T} \subseteq \boldsymbol{S} \subseteq \boldsymbol{KT45}$.*

The notation $G^\Box$ denotes the set of all modal formulas of the form $b \to \Box y$, such that $(b, y) \in G$. We have that $G^\Box \cup A \vdash_S \Box x$ if for a finite subset Y of $G^\Box \cup A$, it holds that $(\bigwedge Y \to \Box x) \in \mathbf{S}$. The notation $\bigwedge Y$ stands for the conjunction of all the elements $y_1, y_2, \ldots, y_n$ in Y i.e. $y_1 \wedge y_2 \wedge \cdots \wedge y_n$.

3.2 Proof Theory

In terms of proof theory, I/O logics are characterized by derivation rules about norms. Given a set of norms G, a derivation system is the smallest set of norms which extends G and is closed under certain derivation rules.

- (SI) Strengthening the input: from (a, x) to (b, x) whenever we have $\vdash b \rightarrow a$
- (WO) Weakening the output: from (a, x) to (a, y) whenever we have $\vdash x \rightarrow y$
- (AND) Conjunction of the output: from (a, x) and (a, y) to $(a, x \wedge y)$
- (OR) Disjunction of the input: from (a, x) and (b, x) to $(a \vee b, x)$
- (CT) Cumulative transitivity: from (a, x) and $(a \wedge x, y)$ to (a, y)

$deriv_2$ denotes the derivation system for the I/O logic operator out_2 and is formed by the rules SI, WO, AND and OR. The derivation system for out_4 is called $deriv_4$ and it is closed under all of the five rules.

4 Shallow semantical embedding

The so-called *shallow semantical embedding* is an approach proposed by Christoph Benzmüller [5], that uses classical higher-order logic (HOL) as a meta-logic in order to represent and model the syntactic and semantical elements of a specific target logic. This methodology is documented and studied for Kripke semantics in [9] and for neighborhood semantics in [7]. This section presents shallow semantical embeddings of the I/O operators out_2 and out_4 in HOL and provides proofs for the soundness and completeness of both operators. To realize these embeddings, we use the provided modal formulations of the operators.

4.1 Semantical embedding of K in HOL

By introducing a new type i to denote a possible world, the propositions of **K** are identified with certain HOL terms (predicates) of type $i \rightarrow o$. The type $i \rightarrow o$ is abbreviated as τ in the remainder. This allows us to represent the propositional formulas of **K** as functions from possible worlds to truth values in HOL and therefore the truth of a formula can explicitly be evaluated in a particular world. The HOL signature is assumed to further contain the constant symbol $r_{i \rightarrow \tau}$. Moreover, for each propositional symbol p^i of **K** , the HOL signature must contain the corresponding constant symbol p^i_τ. Without loss of generality, we assume that besides those symbols and the primitive logical connectives of HOL, no other constant symbols are given in the signature of HOL.

The mapping $\lfloor \cdot \rfloor$ translates a formula φ of **K** into a formula $\lfloor \varphi \rfloor$ of HOL of type τ. The mapping is defined recursively:

$$\begin{aligned}
\lfloor p \rfloor &= p_\tau \\
\lfloor \neg s \rfloor &= \neg_\tau \lfloor s \rfloor \\
\lfloor s \vee t \rfloor &= \vee_{\tau \rightarrow \tau \rightarrow \tau} \lfloor s \rfloor \lfloor t \rfloor \\
\lfloor \Box s \rfloor &= \Box_{\tau \rightarrow \tau} \lfloor s \rfloor
\end{aligned}$$

$\neg_{\tau}$, $\vee_{\tau\to\tau\to\tau}$, $\Box_{\tau\to\tau}$, abbreviate the following formulas of HOL:

$$\begin{aligned}
\neg_{\tau\to\tau} &= \lambda A_\tau \lambda X_i \neg(A\,X)\\
\vee_{\tau\to\tau\to\tau} &= \lambda A_\tau \lambda B_\tau \lambda X_i (A\,X \vee B\,X)\\
\Box_{\tau\to\tau} &= \lambda A_\tau \lambda X_i \forall Y_i (\neg(r_{i\to\tau} X\,Y) \vee A\,Y)
\end{aligned}$$

Analyzing the truth of a translated formula $\lfloor s \rfloor$ in a world represented by term w_i corresponds to evaluating the application $(\lfloor s \rfloor\, w_i)$. In line with previous work [10], we define $\text{vld}_{\tau\to o} = \lambda A_\tau \forall S_i (A\,S)$. With this definition, validity of a formula s in $\mathbf{K}$ corresponds to the validity of the formula $(\text{vld}\, \lfloor s \rfloor)$ in HOL, and vice versa.

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

Lemma 1 (Aligning Henkin models H^M with Kripke models M). *For every Kripke model $M = \langle S, R, V\rangle$ there exists a corresponding Henkin model H^M such that for all formula δ of $\mathbf{K}$, all assignment g and world s it holds:*

$$M, s \models \delta \textit{ if and only if } \|\lfloor \delta \rfloor\, S_i\|^{H^M, g[s/S_i]} = T$$

Proof. See [9].

Lemma 2 (Aligning Kripke models M_H with Henkin models H). *For every Henkin model $H = \langle \{D_\alpha\}_{\alpha\in T}, I\rangle$ there exists a corresponding Kripke model M_H such for all formula δ of $\mathbf{K}$ and for all assignment g and world s it holds*

$$\|\lfloor \delta \rfloor S_i\|^{H, g[s/S_i]} = T \textit{ if and only if } M_H, s \vDash \delta$$

Proof. See [9].

The following table summarizes the alignment of Henkin models and Kripke models. For the class of Kripke models $\langle S, R, V\rangle$ that satisfies some conditions (such as reflexivity) the corresponding Henkin models also satisfies the higher-order counter part of this condition. For example for the reflexivity, the higher-order representation is $\forall X_i R X_i X_i$.

Kripke model $\langle S, R, V\rangle$	Henkin model $\langle D, I\rangle$
Possible worlds $s \in S$	Set of individuals $s_i \in D_i$
Acceptability relation R sRu	Binary predicates $r_{i\to i\to o}$ $Ir_{i\to i\to o}(s_i, u_i) = \top$
Propositional letters p^j Valuation function $s \in V(p^j)$	Unary predicates $p^j_{i\to o}$ Interpretation function $Ip^j_{i\to o}(s_i) = \top$

Theorem 3 (Faithfulness of the embedding of System K in HOL).

$$\models^K \varphi \textit{ if and only if } \models^{HOL} \textit{vld}\, \lfloor \varphi \rfloor$$

Proof. See [9].

4.2 Semantical embedding of input/output logic in HOL

Given a finite set of conditional norms G and an input set A of propositional formulas, for the embedding of out_2 and out_4 we use their translation in modal logic and after that we use theorem 3 for the faithfulness.

Theorem 4 (Faithfulness of the embedding of out_2 in HOL).

$$\varphi \in out_2(G, A)$$

if and only if

$$\models^{HOL} vld \lfloor \bigwedge (G^{\Box} \cup A) \to \Box\varphi \rfloor \textit{ and } \models^{HOL} \bigwedge G(\mathcal{L}) \to \varphi$$

Proof. The proof is directly by choosing $\mathbf{S} = \mathbf{K}$ in theorem 1 and then apply theorem 3.

$$\varphi \in out_2(G, A)$$

if and only if

$$G^{\Box} \cup A \vdash_K \Box\varphi \text{ and } \varphi \in Cn(G(\mathcal{L}))$$

if and only if

$$\models^{\mathrm{K}} \bigwedge (G^{\Box} \cup A) \to \Box\varphi \text{ and } \bigwedge G(\mathcal{L}) \vdash \varphi$$

if and only if

$$\models^{\mathrm{HOL}} \mathrm{vld} \lfloor \bigwedge (G^{\Box} \cup A) \to \Box\varphi \rfloor \text{ and } \models^{\mathrm{HOL}} \bigwedge G(\mathcal{L}) \to \varphi$$

Theorem 5 (Faithfulness of the embedding of out_4 in HOL).

$$\varphi \in out_4(G, A)$$

if and only if

$$T \models^{HOL} vld \lfloor \bigwedge (G^{\Box} \cup A) \to \Box\varphi \rfloor \textit{ and } \models^{HOL} \bigwedge G(\mathcal{L}) \to \varphi$$

Proof. The proof is directly by choosing $\mathbf{S} = \mathbf{KT}$ in theorem 2 and then apply theorem 3.

$$\varphi \in out_4(G, A)$$

if and only if

$$G^{\Box} \cup A \vdash_{KT} \Box\varphi \text{ and } \varphi \in Cn(G(\mathcal{L}))$$

if and only if

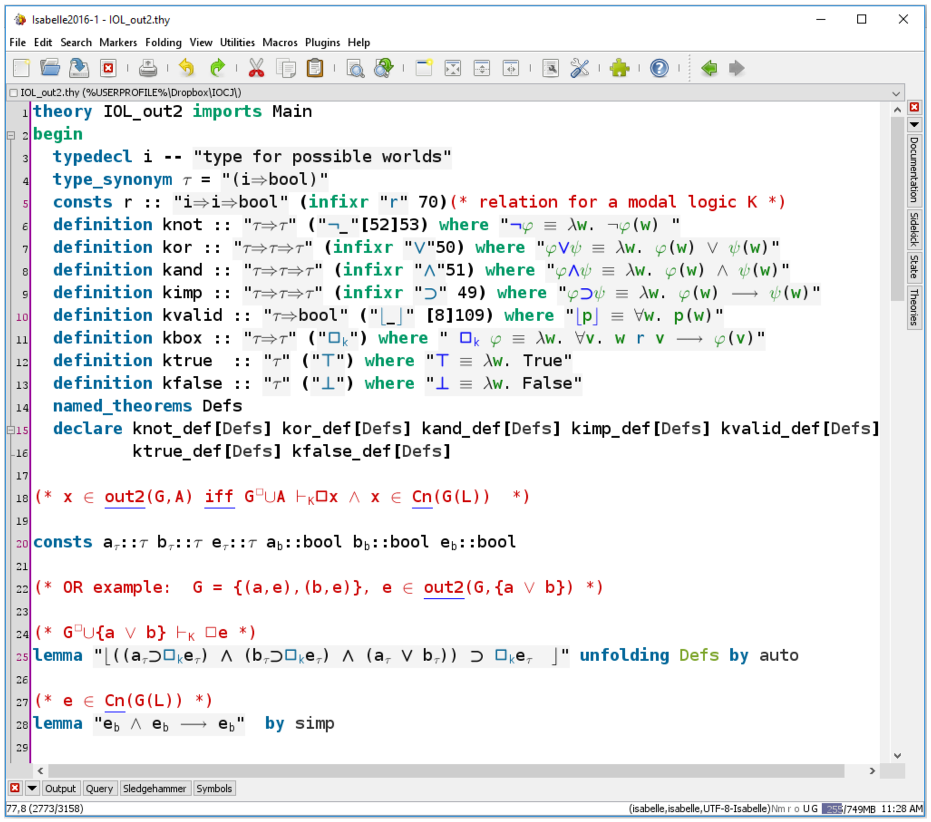
$$\models^{\mathrm{KT}} \bigwedge (G^{\Box} \cup A) \to \Box\varphi \text{ and } \bigwedge G(\mathcal{L}) \vdash \varphi$$

if and only if

$$T \models^{\mathrm{HOL}} \mathrm{vld} \lfloor \bigwedge (G^{\Box} \cup A) \to \Box\varphi \rfloor \text{ and } \models^{\mathrm{HOL}} \bigwedge G(\mathcal{L}) \to \varphi$$

4.3 Implementation of input/output logic in Isabelle/HOL

The semantical embeddings of out_2 and out_4 as devised in the previous section have been implemented into the higher-order proof assistant Isabelle/HOL [22]. The embedding of the operator out_2 is based on a system **K**. We declare the type i to denote possible worlds, introduce an Isabelle/HOL constant r to represent the relation R for the corresponding class of Kripke models, and define the lifted connectives of a modal logic.



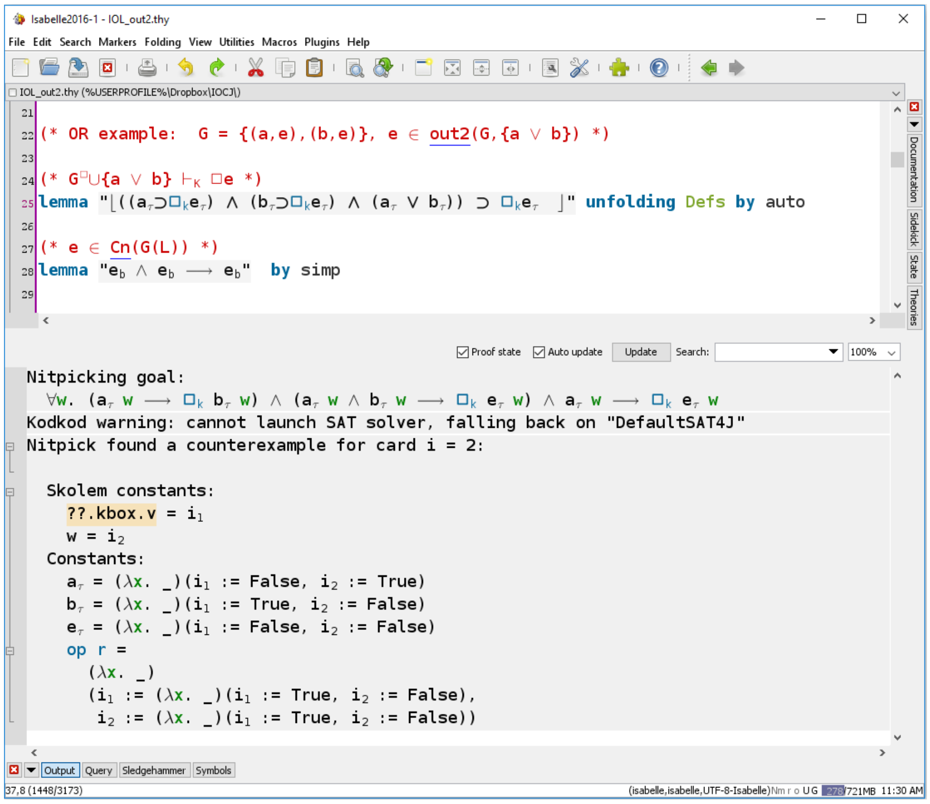
Isabelle2016-1 - IOL_out2.thy

```
theory IOL_out2 imports Main
begin
  typedecl i -- "type for possible worlds"
  type_synonym τ = "(i⇒bool)"
  consts r :: "i⇒i⇒bool" (infixr "r" 70)(* relation for a modal logic K *)
  definition knot :: "τ⇒τ" ("¬_"[52]53) where "¬φ ≡ λw. ¬φ(w) "
  definition kor :: "τ⇒τ⇒τ" (infixr "∨"50) where "φ∨ψ ≡ λw. φ(w) ∨ ψ(w)"
  definition kand :: "τ⇒τ⇒τ" (infixr "∧"51) where "φ∧ψ ≡ λw. φ(w) ∧ ψ(w)"
  definition kimp :: "τ⇒τ⇒τ" (infixr "⊃" 49) where "φ⊃ψ ≡ λw. φ(w) ⟶ ψ(w)"
  definition kvalid :: "τ⇒bool" ("⌊_⌋" [8]109) where "⌊p⌋ ≡ ∀w. p(w)"
  definition kbox :: "τ⇒τ" ("□_k") where " □_k φ ≡ λw. ∀v. w r v ⟶ φ(v)"
  definition ktrue  :: "τ" ("⊤") where "⊤ ≡ λw. True"
  definition kfalse :: "τ" ("⊥") where "⊥ ≡ λw. False"
  named_theorems Defs
  declare knot_def[Defs] kor_def[Defs] kand_def[Defs] kimp_def[Defs] kvalid_def[Defs]
          ktrue_def[Defs] kfalse_def[Defs]

(* x ∈ out2(G,A) iff G^□∪A ⊢_K□x ∧ x ∈ Cn(G(L))  *)

consts a_τ::τ b_τ::τ e_τ::τ a_b::bool b_b::bool e_b::bool

(* OR example:  G = {(a,e),(b,e)}, e ∈ out2(G,{a ∨ b}) *)

(* G^□∪{a ∨ b} ⊢_K □e *)
lemma "⌊((a_τ⊃□_ke_τ) ∧ (b_τ⊃□_ke_τ) ∧ (a_τ ∨ b_τ)) ⊃ □_ke_τ  ⌋" unfolding Defs by auto

(* e ∈ Cn(G(L)) *)
lemma "e_b ∧ e_b ⟶ e_b"  by simp
```

77,8 (2773/3158)   (isabelle,isabelle,UTF-8-Isabelle)Nm r o UG 255/749MB 11:28 AM

Fig. 1: Semantical embedding of out_2 in Isabelle/HOL

Let the set of conditional norms G be composed of the elements (a, e) and (b, e), where a, b and e are propositional symbols, and let the input set A correspond to the singleton set containing $a \vee b$. According to the provided translation, $e \in out_2(G, A)$ if and only if $G^{\Box} \cup A \vdash_K \Box e$ and $e \in Cn(G(\mathcal{L}))$. The first part makes use of a modal logic of type **K** whereas the second part uses classical propositional logic. Regarding the implementation, the propositional symbols a, b and e have to be declared as constants of type τ and of type *bool*. The former constants can then be applied to the lifted connectives to formulate the statement $G^{\Box} \cup A \vdash_K \Box e$, the later are used with the higher-order connectives to express the statement $e \in Cn(G(\mathcal{L}))$. The framework's integrated automatic theorem provers (ATPs), called via the Sledgehammer tool [14], are able to prove both of the statements. This is depicted in figure 1. This small example shows that our encoded out_2 operator satisfies the rule of disjunction (OR).



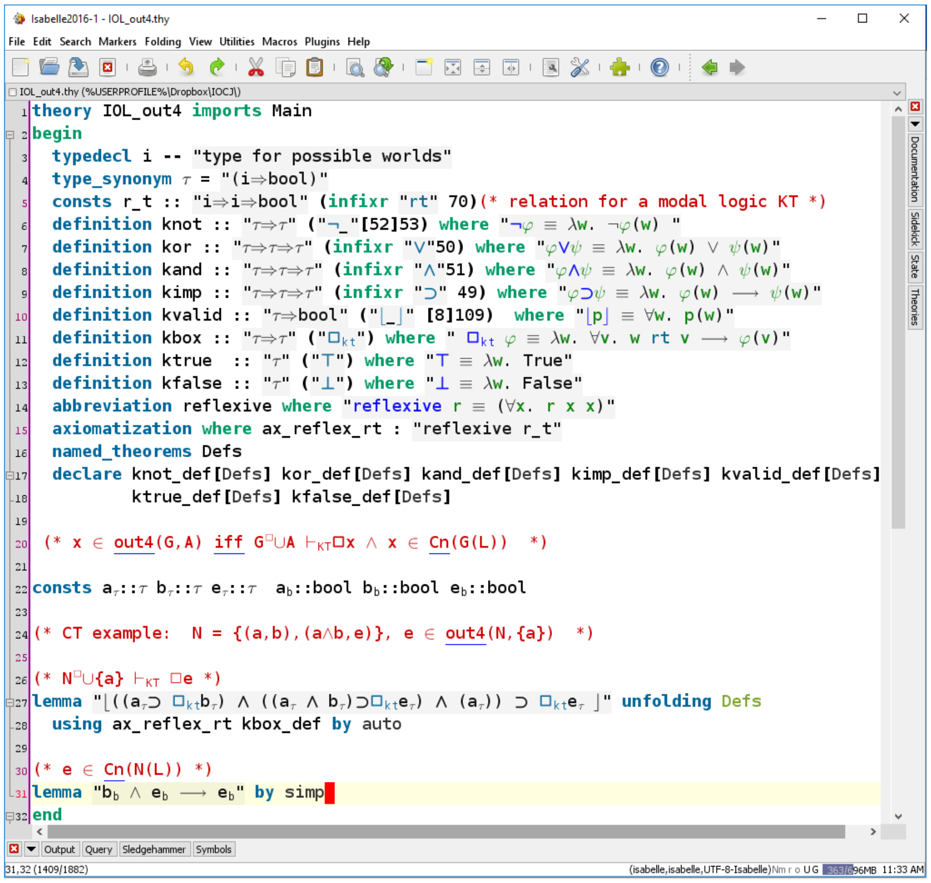
```
(* OR example:  G = {(a,e),(b,e)}, e ∈ out2(G,{a ∨ b}) *)

(* G□∪{a ∨ b} ⊢_K □e *)
lemma "⌊((a_τ⊃□_ke_τ) ∧ (b_τ⊃□_ke_τ) ∧ (a_τ ∨ b_τ)) ⊃ □_ke_τ  ⌋" unfolding Defs by auto

(* e ∈ Cn(G(L)) *)
lemma "e_b ∧ e_b ⟶ e_b"  by simp
```

```
Nitpicking goal:
  ∀w. (a_τ w ⟶ □_k b_τ w) ∧ (a_τ w ∧ b_τ w ⟶ □_k e_τ w) ∧ a_τ w ⟶ □_k e_τ w
Kodkod warning: cannot launch SAT solver, falling back on "DefaultSAT4J"
Nitpick found a counterexample for card i = 2:

  Skolem constants:
    ??.kbox.v = i_1
    w = i_2
  Constants:
    a_τ = (λx. _)(i_1 := False, i_2 := True)
    b_τ = (λx. _)(i_1 := True, i_2 := False)
    e_τ = (λx. _)(i_1 := False, i_2 := False)
    op r =
      (λx. _)
      (i_1 := (λx. _)(i_1 := True, i_2 := False),
       i_2 := (λx. _)(i_1 := True, i_2 := False))
```

Isabelle2016-1 - IOL_out2.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
IOL_out2.thy (%USERPROFILE%\Dropbox\IOCJ\)
Documentation
Sidekick
State
Theories
Proof state
Auto update
Update
Search:
100%
Output
Query
Sledgehammer
Symbols
37,8 (1448/3173)
(isabelle,isabelle,UTF-8-Isabelle)Nm r o UG 278/721MB 11:30 AM

Fig. 2: Further experiments with the embedding of out_2 in Isabelle/HOL

Consider a set of conditional norms $N = \{(a, b), (a \wedge b, e)\}$ and input set $A = \{a\}$. The rule of cumulative transitivity (CT) is not satisfied for the out_2 operator. This can also be verified with our implementation. The model finder Nitpick [13] is able to generate a countermodel for the statement $N^{\Box} \cup A \vdash_K \Box e$ and therefore show that $e \notin out_2(N, A)$. In particular, Nitpick came up with a model consisting of two possible worlds i_1 and i_2. We have that a is evaluated to true at i_2 by the valuation function V, and $V(b) = \{i_1\}$ and $V(e) = \emptyset$. Further, we have for the relation r, the world i_1 is related to itself and i_2 is related to i_1. This is denoted by the set $r = \{(i_1, i_1), (i_2, i_1)\}$. Since the relation r is not reflexive in a system **K**, the formula $((a \to \Box b) \wedge ((a \wedge b) \to \Box e) \wedge a) \to \Box e$ is not valid. The formulation of the example and the generation of the countermodel is illustrated in Figure 2.


Isabelle2016-1 - IOL_out4.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
IOL_out4.thy (%USERPROFILE%\Dropbox\IOCJ\)
Documentation
Sidekick
State
Theories
Output
Query
Sledgehammer
Symbols
31,32 (1409/1882)
(isabelle,isabelle,UTF-8-Isabelle)Nm r o UG 363/696MB 11:33 AM

```
theory IOL_out4 imports Main
begin
  typedecl i -- "type for possible worlds"
  type_synonym τ = "(i⇒bool)"
  consts r_t :: "i⇒i⇒bool" (infixr "rt" 70)(* relation for a modal logic KT *)
  definition knot :: "τ⇒τ" ("¬_"[52]53) where "¬φ ≡ λw. ¬φ(w) "
  definition kor :: "τ⇒τ⇒τ" (infixr "∨"50) where "φ∨ψ ≡ λw. φ(w) ∨ ψ(w)"
  definition kand :: "τ⇒τ⇒τ" (infixr "∧"51) where "φ∧ψ ≡ λw. φ(w) ∧ ψ(w)"
  definition kimp :: "τ⇒τ⇒τ" (infixr "⊃" 49) where "φ⊃ψ ≡ λw. φ(w) ⟶ ψ(w)"
  definition kvalid :: "τ⇒bool" ("⌊_⌋" [8]109)  where "⌊p⌋ ≡ ∀w. p(w)"
  definition kbox :: "τ⇒τ" ("□_kt") where " □_kt φ ≡ λw. ∀v. w rt v ⟶ φ(v)"
  definition ktrue  :: "τ" ("⊤") where "⊤ ≡ λw. True"
  definition kfalse :: "τ" ("⊥") where "⊥ ≡ λw. False"
  abbreviation reflexive where "reflexive r ≡ (∀x. r x x)"
  axiomatization where ax_reflex_rt : "reflexive r_t"
  named_theorems Defs
  declare knot_def[Defs] kor_def[Defs] kand_def[Defs] kimp_def[Defs] kvalid_def[Defs]
          ktrue_def[Defs] kfalse_def[Defs]

 (* x ∈ out4(G,A) iff G^□∪A ⊢_KT□x ∧ x ∈ Cn(G(L))  *)

consts a_τ::τ b_τ::τ e_τ::τ  a_b::bool b_b::bool e_b::bool

(* CT example:  N = {(a,b),(a∧b,e)}, e ∈ out4(N,{a})  *)

(* N^□∪{a} ⊢_KT □e *)
lemma "⌊((a_τ⊃ □_kt b_τ) ∧ ((a_τ ∧ b_τ)⊃□_kt e_τ) ∧ (a_τ)) ⊃ □_kt e_τ ⌋" unfolding Defs
  using ax_reflex_rt kbox_def by auto

(* e ∈ Cn(N(L)) *)
lemma "b_b ∧ e_b ⟶ e_b" by simp
end
```

Fig. 3: Semantical embedding of out_4 in Isabelle/HOL

The embedding of the out_4 operator is based on a system **KT** which means that the corresponding class of Kripke models satisfy the property of reflexivity. In our implementation, the relation for this system is denoted by the constant r_t and we, therefore, declare it as reflexive. Due to this property, the Sledgehammer tool is able to prove the statement $N^{\Box} \cup A \vdash_{KT} \Box e$ and thus we could verify that $e \in out_4(N, A)$. Figure 3 shows the encoding of the out_4 and the verification of the CT example.

5 Application in legal reasoning

The work [8] already documents some practical experiments of automated I/O logic in the domain of legal reasoning. In particular, General Data Protection Regulation (GDPR, Regulation EU 2016/679) is used as an application scenario. By this regulation, the European Parliament, the Council of the European Union and the European Commission aim to strengthen and unify data protection for all the individuals within the European Union. Below two norms of GDPR:

1. Personal data shall be processed lawfully (Art. 5). For example, the data subject must have given consent to the processing of his or her personal data for one or more specific purposes (Art. 6/1.a).
2. If the personal data have been processed unlawfully (none of the requirements for a lawful processing applies), the controller has the obligation to erase the personal data in question without delay (Art. 17.d, right to be forgotten).

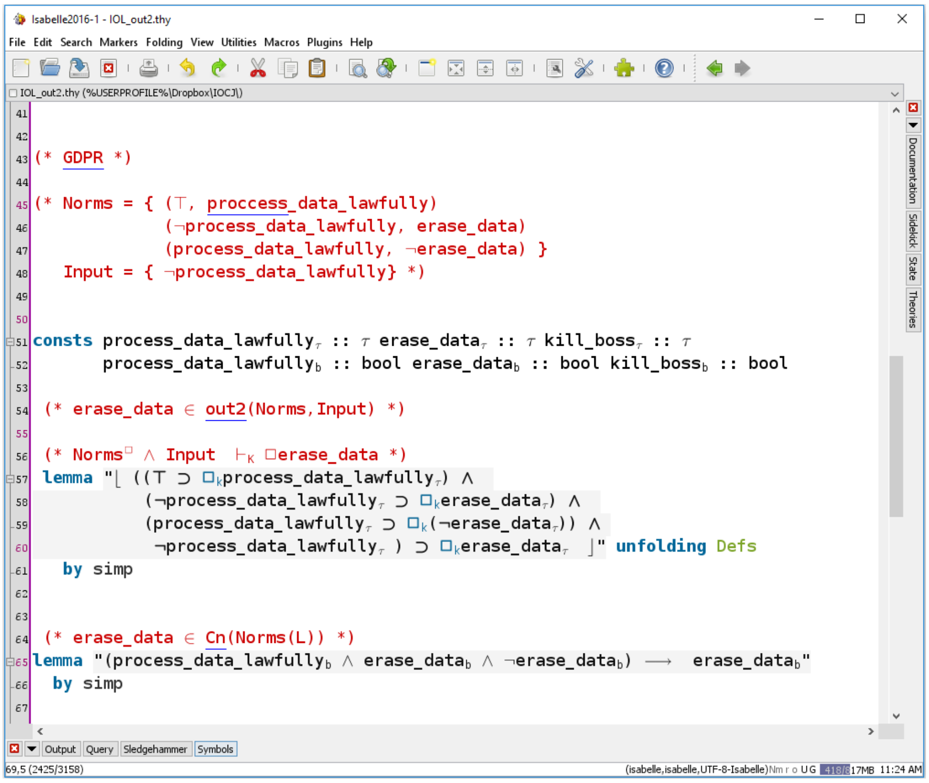
The authors added the following two knowledge units. This establishes a typical Contrary-To-Duty (CTD) scenario which was then analyzed in the context of I/O logic.

3. It is obligatory e.g. as part of a respective agreement between a customer and a company) to keep the personal data (as relevant to the agreement)provided that it is processed lawfully.
4. Some data in the context of such an agreement has been processed unlawfully

We formulated this GDPR scenario in Isabelle/HOL as an application scenario for the embedded out_2 operator. See Figure 4. The lines 45-47 show the set of *Norms* which is composed of

- $(\top, process\_data\_lawfully)$
  This norm states that it is obligatory to process data lawfully.
- $(\neg process\_data\_lawfully, erase\_data)$
  This norms states that if the data was not processed lawfully then it is obligatory to erase the data.
- $(process\_data\_lawfully, \neg erase\_data)$
  This norms states that if the data has been processed lawfully then it is obligatory to keep the data.

Line 48 shows the *Input* set. We assume a situation where the data has not been processed lawfully. Formally $Input = \{\neg process\_data\_lawfully\}$. In lines 51-52, we introduce the constants symbols which are representing the propositions. Just like for the previous examples, each proposition is declared as a constant of type τ and of type *bool*. Next, we want to check that $erase\_data$ is outputted in the context of $\neg process\_data\_lawfully$, meaning that the data should be erased in the situation when the data has not been processed lawfully. So we have to verify that $erase\_data \in out_2(Norms, Input)$. By the modal translation of this operator, we have to check that $Norms^{\Box} \cup Input \vdash_K \Box erase\_data$



Isabelle2016-1 - IOL_out2.thy

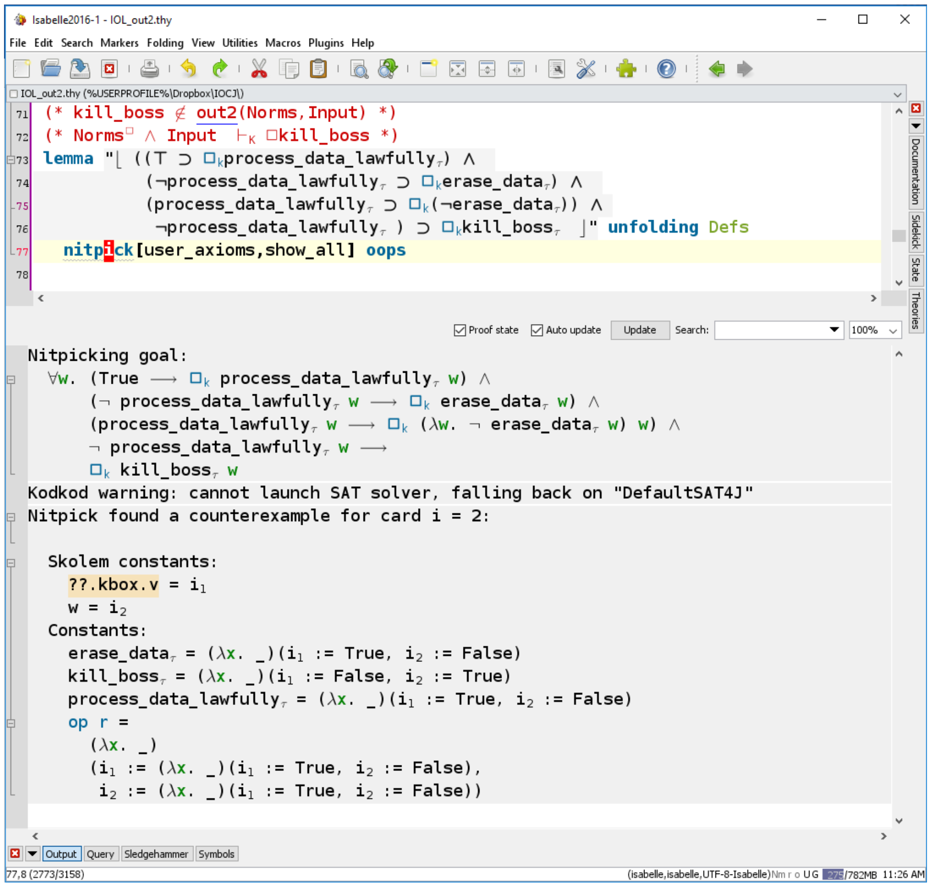
IOL_out2.thy (%USERPROFILE%\Dropbox\IOCJ\)

```
(* GDPR *)

(* Norms = { (⊤, proccess_data_lawfully)
             (¬process_data_lawfully, erase_data)
             (process_data_lawfully, ¬erase_data) }
   Input = { ¬process_data_lawfully} *)

consts process_data_lawfullyτ :: τ erase_dataτ :: τ kill_bossτ :: τ
       process_data_lawfullyb :: bool erase_datab :: bool kill_bossb :: bool

 (* erase_data ∈ out2(Norms,Input) *)

 (* Norms□ ∧ Input  ⊢K □erase_data *)
 lemma "⌊ ((⊤ ⊃ □kprocess_data_lawfullyτ) ∧
           (¬process_data_lawfullyτ ⊃ □kerase_dataτ) ∧
           (process_data_lawfullyτ ⊃ □k(¬erase_dataτ)) ∧
            ¬process_data_lawfullyτ ) ⊃ □kerase_dataτ  ⌋" unfolding Defs
   by simp


 (* erase_data ∈ Cn(Norms(L)) *)
lemma "(process_data_lawfullyb ∧ erase_datab ∧ ¬erase_datab) ⟶  erase_datab"
  by simp
```

69,5 (2425/3158) (isabelle,isabelle,UTF-8-Isabelle)Nm r o UG 418/817MB 11:24 AM

Fig. 4: Further experiments with the embedding of out_2 in Isabelle/HOL

and *earse_data* $\in Cn(Norms(\mathcal{L}))$. The lines 56-66 show the formulation for this statements. Both of them could be by proven by the integrated ATPs of Isabelle/HOL.



```
(* kill_boss ∉ out2(Norms,Input) *)
(* Norms□ ∧ Input  ⊢K □kill_boss *)
lemma "⌊ ((⊤ ⊃ □kprocess_data_lawfullyτ) ∧
           (¬process_data_lawfullyτ ⊃ □kerase_dataτ) ∧
           (process_data_lawfullyτ ⊃ □k(¬erase_dataτ)) ∧
            ¬process_data_lawfullyτ ) ⊃ □kkill_bossτ  ⌋" unfolding Defs
  nitpick[user_axioms,show_all] oops
```

```
Nitpicking goal:
  ∀w. (True ⟶ □k process_data_lawfullyτ w) ∧
      (¬ process_data_lawfullyτ w ⟶ □k erase_dataτ w) ∧
      (process_data_lawfullyτ w ⟶ □k (λw. ¬ erase_dataτ w) w) ∧
      ¬ process_data_lawfullyτ w ⟶
      □k kill_bossτ w
Kodkod warning: cannot launch SAT solver, falling back on "DefaultSAT4J"
Nitpick found a counterexample for card i = 2:

  Skolem constants:
    ??.kbox.v = i1
    w = i2
  Constants:
    erase_dataτ = (λx. _)(i1 := True, i2 := False)
    kill_bossτ = (λx. _)(i1 := False, i2 := True)
    process_data_lawfullyτ = (λx. _)(i1 := True, i2 := False)
    op r =
      (λx. _)
      (i1 := (λx. _)(i1 := True, i2 := False),
       i2 := (λx. _)(i1 := True, i2 := False))
```

Fig. 5: Further experiments with the embedding of out_2 in Isabelle/HOL

Just like in the work of [8], we also showed that we are not able to derive any weird or unethical conclusions such as killing the boss. See Figure 5. The model finder Nitpick is able to generate a countermodel for the the statement $Norms^{\Box} \cup Input \vdash_K \Box kill\_boss$ and therefore it showed that $kill\_boss \notin out_2(Norms, Input)$. Nitpick found a model M consisting of two possible worlds i_1 and i_2. For the valuation function V, we have that $V(erase\_data) = \{i_1\}$, $V(proccess\_data\_lawfully) = \{i_1\}$ and $V(kill\_boss) = \{i_2\}$ and for the relation R, we have that $R = \{(i_1, i_1), (i_2, i_1)\}$. The world i_2 satisfies the formulas $\top \to \Box process\_data\_lawfully$, $\neg process\_data\_lawfully \to \Box erase\_data$,

$process\_data\_lawfully \rightarrow \Box\neg erase\_data$ and $\neg process\_data\_lawfully$. However, the formula $\Box kill\_boss$ is not satisfied in i_2.

6 Conclusion

We have presented a straightforward embedding of I/O logic in HOL and we have shown that this embedding is sound and complete. We provide an empirical reasoner for I/O logic. The works presented here and in [7] provide the theoretical foundation for the implementation and automation of deontic logic within existing theorem provers and proof assistants for HOL. A future direction is whether the provided implementation already supports non-trivial applications in practical normative reasoning such as legal reasoning or multi-agent systems, or whether further improvements are required. There is still a lot of room for future work. We could employ our implementation to systematically study some meta-logical properties of I/O logic within Isabelle/HOL. Moreover, we could similarly implement the intuitionistic input/output logic [24].

References

1. Andrews, P.B.: General models and extensionality. Journal of Symbolic Logic **37**(2), 395–397 (1972)
2. Andrews, P.B.: Church's type theory. In: Zalta, E.N. editor, The Stanford Encyclopedia of Philosophy. Metaphysics Research Lab, Stanford University, spring 2014 edition, (2014)
3. Åqvist, L.: Deontic logic. In: Handbook of philosophical logic, pp. 147–264. Springer, Dordrecht (2002)
4. Benzmüller, C.: Cut-elimination for quantified conditional logic. Journal of Philosophical Logic, **46**(3), 333–353, (2017)
5. Benzmüller, C.: Recent successes with a meta-logical approach to universal logical reasoning (extended abstract). In: da Costa Cavalheiro, S.A., Fiadeiro, J.L. (eds.) Formal Methods: Foundations and Applications, volume 10623 of Lecture Notes in Computer Science, pp. 7–11. Springer, (2017)
6. Benzmüller, C., Brown, C., Kohlhase, M.: Higher-order semantics and extensionality. Journal of Symbolic Logic, **69**(4), 1027–1088, (2004)
7. Benzmüller, C., Farjami, A., Parent, X.: A dyadic deontic logic in HOL. In: Broersen, J., Condoravdi, C., Nair, S., Pigozzi, G. (eds.) Deontic Logic and Normative Systems — 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3-6 July, 2018, pp. 33–50, College Publications, UK, (2018)
8. Benzmüller, C., Parent, X., van der Torre, L.: A Deontic Logic Reasoning Infrastructure. In: Manea, F., Miller, R., Nowotka, D. (eds.) 14th Conference on Computability in Europe, CiE 2018, Kiel, Germany, July 30-August, 2018, Springer, LNCS, volume 10936, (2018)
9. Benzmüller, C., Paulson, L.: Multimodal and intuitionistic logics in simple type theory. The Logic Journal of the IGPL **18**(6), 881–892 (2010). https://doi.org/10.1093/jigpal/jzp080.
10. Benzmüller, C., Paulson, L.C.: Quantified multimodal logics in simple type theory. Logica Universalis (Special Issue on Multimodal Logics), **7**(1), 7–20, (2013)

11. Benzmüller, C.,Sultana, N., Paulson, L. C., Theiß, F.: The higher-order prover LEO-II. Journal of Automated Reasoning, **55**(4), 389–404, (2015)
12. Boella, G., van der Torre, L.: Regulative and Constitutive Norms in Normative Multiagent Systems. In: Dubois, D., and Welty, C., Williams, M. (eds.) Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004), Whistler, Canada, June 2-5, 2004, pp. 255–266, AAAI Press, USA, (2004)
13. Blanchette, J.C., Nipkow, T.: Nitpick: A counterexample generator for higher-order logic based on a relational model finder. In: ITP 2010, number 6172 in Lecture Notes in Computer Science, pp. 131–146. Springer, (2010)
14. Blanchette, J. C., Paulson, L. C.: Hammering Away - A User's Guide to Sledgehammer for Isabelle/HOL, (2017)
15. Chisholm, R.M.: Contrary-to-duty imperatives and deontic logic. Analysis, **24**(2), 33–36 (1963)
16. Church, A.: A formulation of the simple theory of types. Journal of Symbolic Logic, **5**(2), 56–68, (1940)
17. Gabbay, D., Horty, J., Parent, X., van der Meyden, R., van der Torre, L.: Handbook of deontic logic and normative systems. Volume 1. College Publications, UK, (2013)
18. Gabbay, D., Horty, J., Parent, X., van der Meyden, R., van der Torre, L.: Handbook of deontic logic and normative systems. Volume 2. College Publications, UK, (2018)
19. Hansson, B.: An analysis of some deontic logics. Nous, 373–398 (1969)
20. Henkin, L.: Completeness in the theory of types. Journal of Symbolic Logic, **5**(2), 81–91, (1950)
21. Makinson, D., van der Torre, L.: Input/output logics. Journal of Philosophical Logic, **29**(4), 383408, (2000)
22. Nipkow, T., Paulson, L.C., Wenzel., M.: Isabelle/HOL — A Proof Assistant for Higher-Order Logic, volume 2283 of Lecture Notes in Computer Science. Springer, (2002)
23. Parent, X.: Completeness of Åqvists systems E and F. The Review of Symbolic Logic, **8**(1), 164–177 (2015)
24. Parent, X.: A modal translation of an intuitionistic I/O operation. Presented at the 7th Workshop on Intuitionistic Modal Logic and Applications (IMLA 2017), organized by V. de Paiva and S. Artemov at the University of Toulouse (France), 17-28 July, 2017 (satellite workshop of ESSLI'17). Post-conference version of the paper under review.