Three Implementations of the Categorical Imperative

Lavanya Singh

September 13, 2021

Contents

1 Introduction

As artifical reasoners become increasingly powerful, computers become capable of performing complex ethical reasoning. The field of machine ethics [39] is interesting for two reasons. First, the proliferation of artifically autonomous agents is creating and will continue to create a demand for automated ethics. These agents must be able to reason about complex ethical theories that withstand philosophical scrutiny. Second, just as automated mathematical reasoning gives mathematicians new powers, automated ethical reasoning is a tool that philosophers can use when reasoning about ethics. Many contradictions or paradoxes with an ethical system may not be obvious to the human eye, but can be easily tested using an automated theorem prover.

Modelling ethics without sacrificing the intracacies of an ethical theory is a challenging computational and philosophical problem. Simple and intuitive computational approaches, such as encoding ethics as a constraint satisfaction problem, fail to capture the complexity of most philosophically plausible systems. On the other hand, it is not immediately clear how to formalize many complex moral theories, like virtue ethics.

Computational ethics also requires a sophisticated ethical theory to model. Constraint satisfaction systems often default to some version of utiliarianism, the principle of doing the most good for the most people. Alternatively, they model basic moral principles such as “do not kill,” without modelling the theory that these principles originated from. Modelling a more complex ethical theory will not only enable smarter philosophical machines, it will also empower philosophers to study more complex ethical issues with the computer’s help. The entire field of philosophy is devoted to developing and testing robust ethical theories. Plausible machine ethics must draw on plausible moral philosophy.

The ideal candidate ethical theory will be both philosophically interesting and easy to formalize. Kantian ethics, often described as “formal,” has been often floated as such a candidate [33, 6, 27]. The categorical imperative, Kant’s universal law of morality, is a moral rule that can be used to guide action.

This project’s objective is to automate Kantian ethics. I present two different formalizations of Kant’s categorical imperative implemented and tested in the Isabelle/HOL [30] theorem prover. I model each formalization as an extension of Carmo and Jones’ Dyadic Deontic Logic (DDL) [12]. I then embed the corresponding DDL formalization in higher-order logic (HOL) and implement it in Isabelle. Section 3.1 implements and tests the naive formalization, a “control group” that is clearly implausible but demonstrates the methods and tools of my approach. Section 3.2 implements a more sophisticated formalization inspired by Moshe Kroy’s partial formalization of the categorical imperative.

I contribute implementations of two different interpretations of the categorical imperative, examples of how each implementation can be used to model and solve ethical scenarios, and tests that examine ethical and logical properties of the system, including logical consistency, consistency of obligation, and possibility of permissibility. The implementations themselves are usable models of ethical principles and the tests represent the kind of philosophical work that formalized ethics can contribute.

2 System Overview

The goal of this project is to automate sophisticated ethical reasoning. This requires three components. First, the choice of an ethical theory that is both intuitively attractive and lends itself to formalization. Second, the choice of formal logic to model the theory in. Third, the choice of automation engine to implement the formal model in. Section 2.1 introduces Kantian ethics, Section 2.2 explains Carmo and Jones's Dyadic Deontic Logic [12] as a base logic, and Section 2.3 presents the Isabelle/HOL implementation of the logic.

2.1 Kantian Ethics

Kantian ethics is an attractive choice of ethical theory to automates. Kant's writings inspired much of Western analytic philosophy. In addition to the rich neo-Kantian program, almost all major philosophical traditions after Kant have engaged with his work. Much of Western libertarian political thought is inspired by Kant's deontology, and his ethics have bled into household ethical thought. Deontology is seen as one of the three major schools of Western analytic ethics.

Understanding the ethic's potential for formalization requires understanding Kant's system. Kant argues that if morality exists, it must take the form of an inviolable rule, which he calls the "categorical imperative." He presents three formulations of the categorical imperative, as well as a robust defense of them in his seminal work on ethics [19]. He argues that all three formulations are equivalent.

The first formulation of the categorical imperative is the "Formula of Universal Law."

Definition *(Formula of Universal Law)*

I ought never to act except in such a way that I could also will that my maxim should become a universal law [19]

A "maxim" is a moral rule such as, "I can murder someone to take their job". "Willing" a maxim is equivalent to acting on that rule. The FUL creates a thought experiment called the universalizability test: to decide if a maxim is permissible, imagine what would happen if everyone willed that maxim. If your imagined world yields a contradiction, the maxim is prohibited[1]. Intuitively, the FUL asks the

[1] There is another case here. What happens if the imagined world does not yield a logical contradiction, but is still undesirable? Kant distinguishes between these cases as contradictions in conception and contradictions in will. Both yield moral prohibitions, but contradictions in conception generate stronger contradictions and therefore stronger obligations. This paper will focus entirely

question, ”What would happen if everyone did that?” [24].

As an example, let’s apply the universalizability test to the maxim of murdering others to take their job. Universalized, anyone who wants a job can murder the person currently holding that job. It is contradictory to simultaneously will that I acquire a job (through murder) and also that someone can take that job from me whenever they want (also by murder). The maxim thwarts its own end, and is thus internally contradictory. Therefore, this maxim is prohibited.

The universalizability test makes the “formal” nature of Kant’s ethics immediately clear. Formal logic has the tools to universalize a maxim (apply a universal quantifier) and to test for contradictions (test for inconsistency).

Kant also presents two additional formulations of the categorical imperative.

Definition *(Formula of Humanity)*

The Formula of Humanity (FUH) is to act in such a way “that you use humanity, whether in your own person or in the person of any other, always at the same time as an end, never merely as a means.”[19]

Definition *(Kingdom of Ends)*

The third formulation of the categorical imperative states that “we should so act that we may think of ourselves as legislating universal laws through our maxims.”[24]

The last two formulations are not as obviously formal as the FUL, but they can still be modelled in logic. Because Kantian ethics presents a series of rules, a logical system can encode the theory by modelling each rule as an axiom.

The above outline is a brief and incomplete picture of a rich philosophical tradition. Kantian scholars debate the meaning of each formulation of the categorical imperative and develop views far more nuanced than those above. For the purposes of this paper, I will adopt Kant’s three original formulations presented above. Note additionally that Kantian ethics is widely disputed. I do not present a defense of Kant’s ethic in this paper. My approach to formalizing ethics can be applied to other theories as well.

This paper aims to formalize Kant’s ethic as faithfully as possible. This is an important choice. While it is tempting to modify or simplify an ethical theory in seemingly insignificant way, these choices often have ramifications. The entire field of neo-Kantian thought has been exploring versions of Kant’s ethical theory for centuries. I will not attempt to present a radically new conception of Kant’s ethic, but will instead draw on philosophical expertise regarding the content and justification of the categorical imperative.

on contradictions in conception, for this is the only kind of contradiction for which a strict logical interpretation makes sense. For a complete treatment of the difference, see [19, 24]

2.2 Dyadic Deontic Logic

2.2.1 Deontic Logic

Traditional modal logics include the necessitation operator, denoted as $\Box$. In simple modal logic using the Kripke semantics s [14], $\Box p$ is true at a world w if p is true at all of w's neighbors. These logics also usually contain the $\diamond$ operator, representing possibility, where $\diamond p \iff \sim \Box \sim p$. Additionally, modal logics include operators of propositional logic like $\sim, \wedge, \vee, \rightarrow$.

A deontic logic is a special kind of modal logic designed to reason about obligation. Standard deontic logic [14, 28] replaces $\Box$ with the obligation operator O, and $\diamond$ with the permissibility operator P. Using the Kripke semantics for O, Op is true at w if p is true at all ideal deontic alternatives to w. The O operator in SDL takes a single argument (the formula that is obligatory), and is thus called a monadic deontic operator.

While SDL is appreciable for its simplicity, it suffers a variety of well-documented paradoxes, including contrary-to-duty paradoxes [2]. In situations where duty is violated, the logic breaks down and produces paradoxical results. Thus, I use an improved deontic logic instead of SDL for this work.

2.2.2 Dyadic Deontic Logic

I use as my base logic Carmo and Jones's dyadic deontic logic, or DDL, which improves on SDL [12]. It introduces a dyadic obligation operator $O\{A|B\}$ to represent the sentence "A is obligated in the context B". This gracefully handles contrary-to-duty conditionals. The obligation operator uses a neighborhood semantics [35, 29], instead of the Kripke semantics. Carmo and Jones define a function *ob* that maps from worlds to sets of sets of worlds. Intuitively, each world is mapped to the set of propositions obligated at that world, where a proposition p is defined as the worlds at which the p is true.

DDL also includes other modal operators. In addition to $\Box$ and $\diamond$, DDL also has a notion of actual obligation and possible obligation, represented by operators O_a and O_p respectively. These notions are accompanied by the corresponding modal operators $\Box_a, \diamond_a, \Box_p, \diamond_p$. These operators use a Kripke semantics, with the func-

[2] The paradigm case of a contrary-to-duty paradox is the Chisholm paradox. Consider the following statements:

1. It ought to be that Tom helps his neighbors
2. It ought to be that if Tom helps his neighbors, he tells them he is coming
3. If Tom does not help his neighbors, he ought not tell them that he is coming
4. Tom does not help his neighbors

These premises contradict themselves, because items (2)-(4) imply that Tom ought not help his neighbors. The contradiction results because the logic cannot handle violations of duty mixed with conditionals. [13, 34]

tions av and pv mapping a world w to the set of corresponding actual or possible versions of w.

For more of fine-grained properties of DDL see [12] or this project's source code. DDL is a heavy logic and contains modal operators that aren't necessary for my analysis. While this expressivity is powerful, it may also cause performance impacts. DDL has a large set of axioms involving quantification over complex higher-order logical expressions. Proofs involving these axioms will be computationally expensive. Benzmueller and Parent warned me that this may become a problem if Isabelle's automated proof tools begin to time out.

2.3 Isabelle/HOL Implementation

Isabelle/HOL is an interactive proof assistant [30] built on Haskell and Scala. It allows the user to define types, functions, definitions, and axiom systems. It has built-in support for both automatic and interactive/manual theorem proving.

I started my project by reimplementing Benzmueller, Farjami, and Parent's [7, 9] implementation of DDL in Isabelle/HOL. This helped me learn how to use Isabelle/HOL, and the implementation showcased in the next few sections demonstrates the power of Isabelle.

BFP use a shallow semantic embedding. This kind of embedding models the semantics of DDL as constants in HOL and axioms as constraints on DDL models. This document will contain a subset of my implementation that is particularly interesting and relevant to understanding the rest of the project. For the complete implementation, see the source code in `paper22.thy`.

2.3.1 System Definition

The first step in embedding a logic in Isabelle is defining the relevant terms and types.

typedecl *i* — i is the type for a set of worlds.

type-synonym *t* = (*i* ⇒ *bool*) — t represents a set of DDL formulae.
— A set of formulae is defined by its truth value at a set of worlds. For example, the set {True} would be true at any set of worlds.

The main accessibility relation that I will use is the *ob* relation:

consts *ob*::*t* ⇒ (*t* ⇒ *bool*) — set of propositions obligatory in this context
— ob(context)(term) is True if the term is obligatory in this context

2.3.2 Axiomatization

For a semantic embedding, axioms are modelled as restrictions on models of the system. In this case, a model is specificied by the relevant accessibility relations, so it suffices to place conditions on the accessibility relations. These axioms can be quite unweildy, so luckily I was able to lift BFP's [7] implementation of Carmo and Jones's original axioms directly. Here's an example of an axiom:

and $\textit{ax-5d}$: $\forall X\, Y\, Z.\ ((\forall w.\ Y(w) \longrightarrow X(w)) \wedge ob(X)(Y) \wedge (\forall w.\ X(w) \longrightarrow Z(w)))$
$\longrightarrow ob(Z)(\lambda w.(Z(w) \wedge \neg X(w)) \vee Y(w))$
— If some subset Y of X is obligatory in the context X, then in a larger context Z, any obligatory proposition must either be in Y or in Z-X. Intuitively, expanding the context can't cause something unobligatory to become obligatory, so the obligation operator is monotonically increasing with respect to changing contexts.

2.3.3 Syntax

The syntax that I will work with is defined as abbreviations. Each DDL operator is represented as a HOL formula. Isabelle automatically unfolds formulae defined with the `abbreviation` command whenever they are applied. While the shallow embedding is performant (because it uses Isabelle's original syntax tree), abbreviations may hurt performance. In some complicated proofs, we want to control definition unfolding. Benzmueller and Parent told me that the performance cost of abbreviations can be mitigated using a definition instead.

Modal operators will be useful for my purposes, but the implementation is pretty standard.

abbreviation $\textit{ddlbox}::t \Rightarrow t\ (\Box)$
where $\Box\, A \equiv \lambda w. \forall y.\ A(y)$
abbreviation $\textit{ddldiamond}::t \Rightarrow t\ (\Diamond)$
where $\Diamond A \equiv \neg(\Box(\neg A))$

The most important operator for our purposes is the obligation operator.

abbreviation $\textit{ddlob}::t \Rightarrow t \Rightarrow t\ (O\{\text{-}|\text{-}\})$
where $O\{B|A\} \equiv \lambda\ w.\ ob(A)(B)$
— O$\{B|A\}$ can be read as "B is obligatory in the context A"

While DDL is powerful because of its support for a dyadic obligation operator, in many cases we need a monadic obligation operator. Below is some syntactic sugar for a monadic obligation operator.

abbreviation $\textit{ddltrue}::t\ (\top)$
where $\top \equiv \lambda w.\ \textit{True}$
abbreviation $\textit{ddlob-normal}::t \Rightarrow t\ (O\ \{\text{-}\})$
where $(O\ \{A\}) \equiv (O\{A|\top\})$

— Intuitively, the context `True` is the widest context possible because `True` holds at all worlds.

Validity will be useful when discussing metalogical/ethical properties.

abbreviation *ddlvalid*::*t*⇒*bool* (⊨-)
  where ⊨*A* ≡ ∀ *w*. *A w*

2.3.4 Syntactic Properties

One way to show that a semantic embedding is complete is to show that the syntactic specification of the theory (axioms) are valid for this semantics - so to show that every axiom holds at every world. BFP [7] provide a complete treatment of the completeness of their embedding, but I will include selected axioms that are particularly interesting here. This section also demonstrates many of the relevant features of Isabelle/HOL for my project.

Consistency

lemma *True* **nitpick** [*satisfy*,*user-axioms*,*format*=*2*] **by** *simp*
— Isabelle has built-in support for Nitpick, a model checker. Nitpick successfully found a model satisfying these axioms so the system is consistent.
— Nitpick found a model for card i = 1:
Empty assignment

Nitpick [11] can generate models or countermodels, so it's useful to falsify potential theorems, as well as to show consistency. by simp indicates the proof method. In this case, simp indicates the Simplification proof method, which involves unfolding definitions and applying theorems directly. HOL has $True$ as a theorem, which is why this theorem was so easy to prove.

Modus Ponens

lemma *modus-ponens*: **assumes** ⊨ *A* **assumes** ⊨ (*A* ⟶ *B*)
  shows ⊨*B*
  using *assms*(*1*) *assms*(*2*) **by** *blast*
— Because I have not defined a "derivable" operator, inference rules are written using assumptions.
— The rule blast is a classical reasoning method that comes with Isabelle out of the box. [30]
— This is an example of a metalogical proof in this system using the validity operator.

Another relevant operator for our purposes is □, the modal necessity operator. In this system, □ behaves as an S5 [14] modal necessity operator.

lemma *K*:
  shows ⊨ ((□(*A* ⟶ *B*)) ⟶ ((□*A*) ⟶ (□*B*))) **by** *blast*

lemma *T*:
  shows ⊨ ((□*A*) ⟶*A*) **by** *blast*

```
lemma 5:
  shows ⊨ ((◇A) → (□(◇A))) by blast
```

As mentioned earlier, the obligation operator is most interesting for my purposes. Here are some of its properties.

```
lemma O-diamond:
  shows ⊨ (O{A|B} → (◇(B ∧ A)))
  using ax-5b ax-5a
  by metis
```

— A is only obligatory in a context if it can possibly be true in that context. This is meant to prevent impossible obligations.

```
lemma O-nec:
  shows ⊨(O{B|A} → (□O{B|A}))
  by simp
```

— Obligations are necessarily obligated. This axiom is faithful to Kant's interpretation of ethics and is evidence of DDL's power in representing Kant's theory. Kant claimed that the categorical imperative was not contingent on any facts about the world, but instead a property of the concept of morality itself [19]. Under this view, obligation should not be world-specific.

Below is an example of a more involved proof in Isabelle. This proof was almost completely automatically generated. The property itself here is not very interesting for my purposes because I will rarely mix the dyadic and monadic obligation operators.

```
lemma O-to-O:
  shows ⊨(O{B|A}→O{(A→B)|⊤})
proof–
  have ∀X Y Z. (ob X Y ∧ (∀w. X w ⟶ Z w)) ⟶ ob Z (λw.(Z w ∧ ¬X w) ∨ Y w)
```

— I had to manually specify this subgoal, but once I did Isabelle was able to prove it automatically.

```
    by (smt ax-5d ax-5b ax-5b'')
```

— Isabelle's proof-finding tool, Sledgehammer [32], comes with out-of-the-box support for smt solving [10].

```
  thus ?thesis
  proof –
    have f1: ∀p pa pb. ((¬ (ob p pa)) ∨ (∃i. (p∧(¬ pb)) i)) ∨ (ob pb ((pb∧(¬ p))∨ pa))
      using ‹∀X Y Z. ob X Y ∧ (⊨(X→Z)) ⟶ ob Z ( (Z∧(¬ X))∨ Y)› by force
    obtain ii :: (i ⇒ bool) ⇒ (i ⇒ bool) ⇒ i where
      ∀x0 x2. (∃v3. (x2∧(¬ x0)) v3) = (x2∧(¬ x0)) (ii x0 x2)
      by moura
    then have ∀p pa pb. ((¬ ob p pa) ∨ (p∧(¬ pb)) (ii pb p)) ∨ ob pb ( (pb∧(¬ p))∨ pa)
      using f1 by presburger
    then show ?thesis
      by fastforce
  qed
```

— This entire Isar style proof was automatically generated using Sledgehammer.
qed

The implementation of DDL showcases some of the useful features of Isabelle. Abbreviations allow us to embed the syntax of DDL into HOL without defining an entire abstract sytax tree. Automated support for proof-finding using Sledgehammer makes proving lemmas trivial, and proving more complex theorems far easier. Nitpick's model finding ability is useful to check for consistency and create countermodels.

3 The Categorical Imperative

In this section, I will present two formulations of the categorical imperative. In Section 3.1, I will consider a simple, naive formulation of the formula of universal law. This formulation is, as I will show, clearly not a good ethical rule. The purpose of this section is to explore the kinds of ethical tests that Isabelle can carry out. In Section 3.2, I will explore Moshe Kroy's [26] partial formalization of the first two formulations of the categorical imperative.

3.1 Naive Formulation of the Formula of Universal Law

This section presents a simple and intuitive formalization of the formula of universal law, which is to will only those maxims that you would simultaneously will universalized. The universalizability test creates negative obligations: if a maxim passes the universalizability test, it is permissible. Else, it is prohibited. In order to appropriately formalize this, we need some notion of permissibility.

abbreviation *ddlpermissable*::*t*⇒*t* (*P*-)
  where $(P\,A) \equiv (\neg(O\,\{\neg A\}))$
— An act A is permissible if its negation is not obligated. For example, buying a red folder is permissible because I am not required to refrain from buying a red folder.

This naive formalization will require very little additional logical machinery, but more complex formalizations may require additional logic concepts.

Let's now consider a naive reading of the Formula of Universal Law (FUL): 'act only in accordance with that maxim through which you can at the same time will that it become a universal law' [19]. An immediate translation to DDL is that if A is not necessary permissible then it is prohibited. In other words, if we cannot universalize PA (where universalizing is represented by the modal necessity operator), then A is prohibited. Let's add this as an axiom to our logic.

axiomatization where
FUL-1: $\models ((\neg(\Box\,(P\,A))) \rightarrow (O\,\{(\neg A)\}))$

Why add the categorical imperative as an axiom of this logic? The purpose of this logic is to perform ethical reasoning. Kant's ethical theory is rule based, so it involves applying the categorical imperative to solve ethical dilemmas. In logic, this is equivalent to adopting the categorical imperative as an axiom and then reasoning in the newly formed logic to come to ethical conclusions. Adding the categorical imperative as an axiom makes it impossible to violate it.

Note that in this system, reasoning about violations of obligation is difficult. Any violation of the categorical imperative immediately results in a contradiction. Developing a Kantian account of contrary- to-duty obligations is a much larger philosophical project that is still open [25]. This paper will focus on the classical Kantian notion of an ideal moral world [31].

The immediate test for any formalization is consistency, which we can check with Nitpick.

lemma *True* **nitpick** [*satisfy*,*user-axioms*,*format=2*] **oops**
— Nitpick found a model for card i = 1:
Empty assignment
— Nitpick tells us that the FUL is consistent[3]

3.1.1 Specifying the Model

One category of tests involves specified models to encode certain facts into the system and then ask questions about obligations. Without specifying the model, we are limited to showing high-level metaethical facts. Let's start with analyzing an obvious example - that murder is wrong.

consts $M::t$
abbreviation *murder-wrong*::*bool* **where** *murder-wrong* $\equiv \models(O\ \{\neg\, M\})$

abbreviation *possibly-murder-wrong*::*bool* **where** *possibly-murder-wrong* $\equiv (\Diamond\ (O\ \{\neg\, M\}))$ *cw*
— These are very simple properties. *poss-murder-wrong* is an abbreviation for the axiom that there is some world where murder might be prohibited. Even this is quite a strong assumption - ideally we'd want to give the system nonmoral facts about murder (like a definition) and then make moral claims.

lemma *wrong-if-possibly-wrong*:
  shows *possibly-murder-wrong* $\longrightarrow$ *murder-wrong*
  by *simp*
— This lemma gets to the "heart" of this naive interpretation. We really want to say that if something isn't necessarily obligated, it's not obligated anywhere.

The above example does exactly what we expect it to: we show that if something is wrong somewhere it's wrong everywhere. That being said, it seems like quite a weak claim. We assumed a very strong, moral fact about murder (that it is wrong somewhere), so it's not surprise that we were able to reach our desired conclusion.

[3] "oops" at the end of a lemma indicates that the proof is left unfinished. It does not indicate that an error occurred. In this case, we aren't interested in proving True (the proof is trivial and automatic), hence the oops.

Let's try a weaker assumption: Not everyone can lie.

typedecl *person*
consts *lie*::*person*⇒*t*
consts *me*::*person*
— Notice that this machinery is quite empty. We don't give axioms about what a person can or can't do.

abbreviation *lying-not-universal*::*bool* **where** *lying-not-universal* $\equiv \forall w.\ \neg\ ((\forall x.\ \textit{lie}(x)\ w) \wedge (\textit{lie}(\textit{me})\ w))$

This is a rough translation of failure of the universalizability test: we will test the maxim universally, as represented by the universal quantifier in the first conjunct, and simultaneously [22], as represented by the second conjunct. The FUL tells us that if this sentence is true, then lying should be prohibited. Let's test it.

lemma *breaking-promises*:
  assumes *lying-not-universal*
  shows $(O\ \{\neg\ (\textit{lie}(\textit{me}))\})\ \textit{cw}$
  nitpick [*user-axioms*]
  oops
— Nitpick found a counterexample for card person = 2 and card i = 2:
Free variable: lie = $(\lambda x.\_)(p_1$:= $(\lambda x.\_)(i_1$:= True, i_2 := False), p_2 := $(\lambda x.\_)(i_1$:= False, i_2 := False))
— Quick note on how to read Nitpick results. Nitpick will either say that it found a "model" or a "counterexample" in the first line. It will then provide a model by specifying model components. For readability, all except for the free variables are hidden. This model has cardinality 2 for the person and world (i) types. The term `lie` is defined for people p_1 and p_2. p_1 lies at world i_1 and does not lie at world i_2. p_2 does the opposite.
— These details will be elided for most Nitpick examples, but this provides guidance on how to interpret the output.

This formula isn't valid. While the FUL should tell us that lying is prohibited, the fact that it doesn't demonstrates the weakness of this naive formulation of the categorical imperative. Kant's version of the FUL universalizes across people, as we did in the definition of *lying-not-universal* $\equiv \forall w.\ \neg\ ((\forall x.\ \textit{lie}\ x\ w) \wedge \textit{lie}\ \textit{me}\ w)$. The naive formalization universalizes across worlds using the □ operator, so it makes sense that it can't handle this example appropriately.

The above implies that the FUL should prescribe consistent obligations across people. If our formalization doesn't, clearly something has gone wrong somewhere. Let's test that!

lemma *universalizability*:
  assumes $\models O\ \{(\textit{lie}(\textit{me}))\}$
  shows $\forall x.\ \models (O\ \{(\textit{lie}(x))\})$
  nitpick [*user-axioms*] **oops**
— Nitpick found a counterexample for card person = 2 and card i = 2:
Free variable: lie = $(\lambda x.\_)(p_1$:= $(\lambda x.\_)(i_1$:= False, i_2 := True), p_2 := $(\lambda x.\_)(i_1$:= False, i_2 := False)) Skolem constant: x = p_2

This lemma demonstrates the problem with the naive interpretation. The FUL universalizes across people but the naive formalization universalizes across worlds. Because this interpretation is so naive, it is limited in its power. However, this serves as an example of the kind of reasoning that Isabelle empowers us to do. Even this simple argument has philosophical consequences. It tells us that reading the FUL as a claim about consistency across possible worlds, instead of consistency across agents, leads to counterintuitive conclusions.

3.1.2 Metaethical Properties

The above section specified the model to simulate practical ethical reasoning, or the kind of reasoning that is useful when an agent has to decide what to do. Formalizations of the categorical imperative can also be used to do metaethical reasoning, which can evaluate a particular ethical theory as good or bad. In this case, we can analyze properties of the system in the form of theorems. For example, if we can show that, in this system, nothing is ever obligated, that would indicate that we have a bad ethical system. This is not only philosophical work, but is also a useful way to test different ethical reasoning systems.

An initial property that we might be interested in is permissibility itself. More generally, an ethical theory that doesn't allow for permissibility would require that every action is either obligatory or prohibited. In fact, if that is the case, many counterintuitive theorems follow, including that all permissible actions are obligatory.[4]

lemma *permissible*:
shows $\exists A.\ ((\neg\ (O\ \{A\}))\wedge(\neg\ (O\ \{\neg\ A\})))\ w$
nitpick [*user-axioms*, *falsify*=*false*] **oops**

— Nitpick found a model for card i = 1 and card s = 1:
Skolem constant: A =($\lambda x.\_$)(i_1 := False)

— We want to show that there exists a model where there is some formula A that is permissible, or, in English, that permissibility is possible. Nitpick finds a model where the above formula holds, so permissibility is indeed possible.

— Note that it's not clear [21] if Kant actually thought that permissibility was a coherent concept. Either way, in modern ethics, permissibility is a pretty widely accepted phenomenon.

lemma *fixed-formula-is-permissible*:
fixes A
shows $((\neg\ (O\ \{A\}))\wedge(\neg\ (O\ \{\neg\ A\})))\ w$
nitpick [*user-axioms*, *falsify*=*false*] **oops**

— Nitpick found a model for card i = 1:
Free variable: A = ($\lambda x.\_$)(i_1 := False)

— This is a slightly stronger result: for any arbitrary action A, there is a model that makes it permissible. We actually don't want this to hold, because if A is ”murder” then the CI requires that it be prohibited in every world.

[4] Proof is in the appendix.

Another initial test is the possibility of arbitrary obligations. If anything can be shown to be obligatory in this theory, then clearly it doesn't track our intuitions.

```
lemma arbitrary-obligations:
  fixes A::t
  shows O {A} w
  nitpick [user-axioms=true] oops
— Nitpick found a counterexample for card i = 1:
Free variable: A = (λx._)(i_1 := False)
— This is good! Shows us that any arbitrary term isn't obligatory.
```

Conflicting Obligations

A more complex metaethical property is the possibility of conflicting obligations. Many deontological ethics are criticized for prescribing conflicting obligations, but in Kantian ethics, obligations never conflict [38]. In order for morality to be action-guiding, it needs to be free of conflicting obligations. Let's see if we can have contradictary obligations under the naive formalization.

```
lemma conflicting-obligations:
  fixes A
  shows (O {A} ∧ O {¬ A}) w
  nitpick [user-axioms, falsify=false] oops
— Nitpick found a model for card i = 2:
Free variable: A = (λx._)(i_1 := False, i_2 := True)
— Oh no! Nitpick found a model with conflicting obligations - that's bad!
```

This is a property of DDL itself, not necessarily of my formalization specifically. A future, more robust formalization should add an axiom that disallows this. Let's see if that causes any obvious problems.

```
lemma removing-conflicting-obligations:
  assumes ∀A. ⊨ (¬ (O {A} ∧ O {¬ A}))
  shows True
  nitpick [satisfy,user-axioms,format=2] oops
— Nitpick found a model for card i = 1:
Empty assignment
— We can disallow conflicting obligations and the system is still consistent - that's good.
```

The above is a rather weak notion of contradictory obligations. Korsgaard [24] argues that Kantian ethics also has the stronger property that if two maxims imply a contradiction, they must not be willed. Let's see if that fact holds in this formalization.

```
lemma implied-contradiction:
  fixes A::t
  fixes B::t
  assumes ⊨(¬ (A ∧ B))
  shows ⊨(¬ (O {A} ∧ O {B}))
  nitpick [user-axioms]
proof –
```

have $\models(\neg(\Diamond(A \wedge B)))$
by (*simp add*: *assms*)
then have $\models(\neg\ (O\ \{A \wedge B\}))$ **by** (*smt O-diamond*)
— Notice that this is **almost** the property we are interested in. In fact, if $O\{A \wedge B\}$ is equivalent to $O\{A\} \wedge O\{B\}$, then the proof is complete.
thus *?thesis* **oops**
— Nitpick found a counterexample for card i = 2:
Free variables: A = $(\lambda x.\_)(i_1$:= True, i_2 := False) B = $(\lambda x.\_)(i_1$:= False, i_2 := True)
— Sadly the property we're actually interested in doesn't follow.

The above proof yields an interesting observation. $O\{A \wedge B\}$ is not equivalent to $O\{A\} \wedge O\{B\}$. The rough English translation of $O\{A \wedge B\}$ is "you are obligated to do both A and B". The rough English translation of $O\{A\} \wedge O\{B\}$ is "you are obligated to do A and you are obligated to do B." We think those English sentences mean the same thing, so they should mean the same thing in our logic as well. Let's test that.

lemma *distribute-obligations*:
assumes $\models (O\ \{A\} \wedge O\ \{B\})$
shows $\models O\ \{A \wedge B\}$
nitpick [*user-axioms*] **oops**
— Nitpick found a counterexample for card i = 2:
Free variables: A = $(\lambda x.\_)(i_1$:= True, i_2 := False) B = $(\lambda x.\_)(i_1$:= False, i_2 := True)

Note that this is a property of DDL itself, not just my formalization. A future formalization might add this property as an axiom.[5]

Miscellaneous Properties

I named this formalization the naive formulation for a reason. Though it seems to be an immediate translation of the FUL into DDL, it doesn't fully respect the properties of modal logic itself. In particular, the formalization as given is equivalent to the below theorem.

lemma *FUL-alternate*:
shows $\models ((\Diamond\ (O\ \{\neg\ A\})) \longrightarrow (O\ \{\neg\ A\}))$
by *simp*
— This means that if something is possibly prohibited, it is in fact prohibited.
— This is a direct consequence[6] of the naive formalization, but it's not clear to me that this is actually how we think about ethics. For example, we can imagine an alternate universe where smiling at someone is considered an incredibly rude and disrespectful gesture. In this universe, we are probably prohibited from smiling at people, but this doesn't mean that in this current universe, smiling is morally wrong.

The "ought implies can" principle is attributed to Kant[7] and is rather intuitive: you can't be obligated to do the impossible. It is worth noting that deontic log-

[5] For discussion of why this property doesn't hold in DDL, see the Appendix.

[6] For a manual proof, see the Appendix.

[7] The exact philosophical credence of this view is disputed, but the rough idea holds nonetheless. See [23] for more.

ics evolved [14] specifically from this principle, so this should hold in both my modified logic and in DDL.

lemma *ought-implies-can*:
  shows $\forall A. \models (O\ \{A\} \rightarrow (\Diamond A))$
  using *O-diamond* **by** *blast*
— $\models \lambda w.\ ob\ ?B\ ?A \longrightarrow \neg \models \neg\ ?B \wedge ?A$ is an axiom of DDL itself, so this theorem holds in DDL.

3.2 Kroy's Formalization of the Categorical Imperative

This section contains a formalization of the categorical imperative introduced by Moshe Kroy in 1976 [26]. Kroy used Hinktikka's deontic logic to formalize the Formula of Universal Law and the Formula of Humanity. I will first import the additional logical tools that Hintikka's logic contains that Kroy relies on, then examine the differences between his logic and DDL, and finally implement and test both of Kroy's formalizations.

3.2.1 Additional Logical Tools

Kroy's logic relies heavily on some notion of identity or agency. We need some notion of "x does action", which we can write as "x is the subject of the sentence 'does action'". This requires defining a subject.

typedecl *s* — s is the type for a "subject", like the subject of a sentence

Kroy also defines a substitution operator[8]. $P(d/e)$ is read in his logic as "P with e substituted for d." DDL has no such notion of substitution, so I will define a more generalized notion of an "open sentence." An open sentence takes as input a subject and returns a complete or "closed" DDL formula. For example, "does action" is an open sentence that can be instantiated with a subject.

type-synonym $os = (s \Rightarrow t)$
— "P sub (d/e)" can be written as "S(e)", where S(d) = P
— The terms that we substitute into are actually instantiations of an open sentence, and substitution just requires re-instantiating the open sentence with a different subject.

New Operators

Because Isabelle is strongly typed, we need to define new operators to handle open sentences. These operators are similar to DDL's original operators. We could probably do without these abbreviations, but they will simplify the notation and make it look more similar to Kroy's original paper.

abbreviation *os-neg*::$os \Rightarrow os$ (¬-)
  where $(\neg A) \equiv \lambda x.\ \neg(A(x))$
abbreviation *os-and*::$os \Rightarrow os \Rightarrow os$ (-∧-)

[8] See page 196 in Kroy's original paper [26].

```
  where (A ∧ B) ≡ λx. ((A(x)) ∧ (B(x)))
abbreviation os-or::os ⇒ os ⇒ os (-∨-)
  where (A ∨ B) ≡ λx. ((A(x)) ∨ (B(x)))
abbreviation os-ob::os ⇒ os (O{-})
  where O{A} ≡ λx. (O {A(x)})
```

Once again, the notion of permissibility will be useful here.

```
abbreviation ddl-permissible::t⇒t (P {-})
  where P {A} ≡ ¬ (O {¬ A})
abbreviation os-permissible::os⇒os (P {-})
  where P {A} ≡ λx. P {A(x)}
```

3.2.2 Differences Between Kroy's Logic (Kr) and DDL

One complication that arises here is that Kroy's original paper uses a different logic than DDL. His custom logic is a slight modification of Hintikka's deontic logic [18]. In this section, I will determine if some of the semantic properties that Kroy's logic (which I will now call Kr) requires hold in DDL. These differences may become important later and can explain differences in my results and Kroy's.

Deontic alternatives versus the neighborhood semantics

The most faithful interpretation of Kr is that if A is permissible in a context, then it must be true at some world in that context. Kr operates under the "deontic alternatives" or Kripke semantics, summarized by Solt [36] as follows: "A proposition of the sort OA is true at the actual world w if and only if A is true at every deontic alternative world to w." Under this view, permissible propositions are obligated at some deontic alternatives, or other worlds in the system, but not at all of them. Let's see if this holds in DDL.

```
lemma permissible-semantics:
  fixes A w
  shows (P {A}) w ⟶ (∃x. A(x))
  nitpick[user-axioms] oops
— Nitpick found a counterexample for card i = 1:
Free variable: A = (λx._)(i₁ := False)
```

Remember that DDL uses neighborhood semantics, not the deontic alternatives view, which is why this proposition fails in DDL. In DDL, the *ob* function abstracts away the notion of deontic alternatives and completely determines obligations. Even if one belives that permissible statements should be true at some deontic alternative, it's not clear that permissible statements must be realized at some world. In some ways, this also coheres with our understanding of obligation. There are permissible actions like "Lavanya buys a red folder" that might not happen in any universe.

An even stricter version of the semantics that Kr requires is that if something is permissible at a world, then it is obligatory at some world. This is a straightforward application of the Kripke semantics. Let's test this proposition.

lemma *permissible-semantics-2*:
  fixes A w
  shows $P\ \{A\}\ w \longrightarrow (\exists x.\ O\ \{A\}\ x)$
  nitpick[*user-axioms*] **oops**
— Nitpick found a counterexample for card i = 1:
Free variable: A = $(\lambda x.\_)(i_1$:= False)

This also doesn't hold in DDL because DDL uses neighborhood semantics instead of the deontic alternatives or Kripke semantics. This also seems to cohere with our moral intuitions. The statement "Lavanya buys a red folder" is permissible in the current world, but it's hard to see why it would be oblgiatory in any world.

One implication of the Kripke semantics is that Kr disallows "vacuously permissible statements." In other words, if something is permissible it has to be obligated at some deontically perfect alternative. If we translate this to the language of DDL, we expect that if A is permissible, it is obligated in some context.

lemma *permissible-semantic-vacuous*:
  fixes A w
  shows $P\ \{A\}\ w \longrightarrow (\exists x.\ ob(x)(A))$
  nitpick[*user-axioms*] **oops**
— Nitpick found a counterexample for card i = 1:
Free variable: A = $(\lambda x.\_)(i_1$:= False)

In order to make this true, we'd have to require that everything is either obligatory or prohibited somewhere. Sadly, that breaks everything and destroys the notion of permissibility everywhere [9]. If something breaks later in this section, it may be because of vacuous permissibility.

Obligatory statements should be permissible

Kr includes the intuitively appealing theorem that if a statement is obligated at a world, then it is permissible at that world[10]. Let's see if that also holds in DDL.

lemma *permissible-prereq-ob*:
  fixes A w
  shows $O\ \{A\}\ w \longrightarrow P\ \{A\}\ w$
  nitpick [*user-axioms*] **oops**
— Nitpick found a counterexample for card i = 2:
Free variable: A = $(\lambda x.\_)(i_1$:= False, i_2 := True)

This particular difference seems untenable. My custom formalization of the categorical imperative must add this as an axiom.

[9] See Appendix for an examination of a buggy version of DDL that led to this insight.

[10] This follows straightforwardly from the Kripke semantics. If proposition A is obligated at world w, this means that at all of w's neighbors, OA holds. Therefore, $\exists w'$ such that w sees w' and OA holds at w' so A is permissible at w.

3.2.3 The Categorical Imperative

I will now implement Kroy's formalization of the Formula of Universal Law. Recall that the FUL says "act only in accordance with that maxim which you can at the same time will a universal law" [19]. Kroy interprets this to mean that if an action is permissible for a specific agent, then it must be permissible for everyone. This formalizes an important moral intuition: the categorical imperative prohibits free-riding. No one is a moral exception. Formalizing this interpretation requires using open sentences to handle the notion of substitution.

abbreviation *FUL::bool* **where** $FUL \equiv \forall w\, A.\ ((\exists p{::}s.\ ((P\ \{A\ p\})\ w))\ \longrightarrow(\ (\forall p.(\ P\ \{A\ p\})\ w)))$

P A p vs *PA p

— In English, this statement roughly means that, for any person p if action A is permissible for p, then action A must be permissible for all agents x. The notion of "permissible for" is captured by the substitution of x for p.

Let's check if this is already an axiom of DDL. If so, then the formalization is trivial.

lemma *FUL*:
 shows *FUL*
 nitpick[*user-axioms*] **oops**
— Nitpick found a counterexample for card s = 2 and card i = 2:
Skolem constants: A = $(\lambda x.\_)(s_1 := (\lambda x.\_)(i_1 := \text{True}, i_2 := \text{True}), s_2 := (\lambda x.\_)(i_1 := \text{False}, i_2 := \text{False}))$ p = s_1 x = s_2

This formalization doesn't hold in DDL, so adding it as an axiom will change the logic.

axiomatization where *FUL*: *FUL*

Consistency check: is the logic still consistent with the FUL added as an axiom?

lemma *True* **nitpick**[*user-axioms*, *satisfy*, *card=1*] **oops**
— Nitpicking formula... Nitpick found a model for card i = 1:
Empty assignment

This completes my implementation of Kroy's formalization of the first formulation of the categorical imperative. I defined new logical constructs to handle Kroy's logic, studied the differences between DDL and Kr, implemented Kroy's formalization of the Formula of Universal Law, and showed that it is both non-trivial and consistent. Now it's time to start testing!

3.2.4 Application Tests

Recall that in Section 3.1.1, we tested the naive interpretation's ability to show that murder is wrong. We started by showing that if murder is possibly wrong, then it is wrong. Let's test that here.

```
consts M::t
```

— Let the constant M denote murder.

```
lemma wrong-if-possibly-wrong:
  shows ((◇ (O {¬ M})) cw) ⟶ (∀ w. (O {¬ M}) w)
  by simp
```

This is the same result we got in Section 3.1.1—if murder is wrong at some world, it is wrong at every world. Kroy's formulation shouldn't actually mean that this property holds. Kroy interprets the FUL as universalizing across *people*, not worlds. In other words, Kroy's formulation implies that if murder is wrong for someone, then it is wrong for everyone. This strange result is actually a property of DDL itself, not a property of Kroy's formalization. Indeed, repeating this experiment in DDL, with no additional axioms that represent the categorical imperative shows that, in DDL, if something is possible wrong, it is wrong at every world. So this is not a useful example to test any formulation, since it holds in the base logic itself.

Let's try adapting our murder wrong axiom to Kroy's formulation. In particular, let's see if murder being wrong for one person means that it's wrong for everyone.

```
consts M2::os
```

— Let's define murder as an open sentence this time, so that we can substitute in different people.

```
lemma wrong-if-wrong-for-someone:
  shows (∃p. ⊨ O {¬(M2 p)}) ⟶ (∀p. ⊨ O {¬(M2 p)})
  proof
    assume (∃p. ⊨ O {¬(M2 p)})
    show (∀p. ⊨ O {¬(M2 p)})
      using FUL ‹∃p. ⊨O{¬M2} p› by blast
  qed
```

— This lemma gets to the heart of Kroy's formulation of the categorical imperative. If murder is prohobited for a specific person p, then it must be prohibited for all people. This test case also revealed a bug in my original implementation of Kroy's formulation of the FUL, demonstrating the power of such philosophical tests for automated ethics. Just as engineers use TDD to find bugs in their code, philosophers and ethicists can use this approach to find bugs in the precise formulations of their theories.

For the naive implementation, we also tested the slightly stronger proposition that if not everyone can simultaneously lie, then lying is prohibited. We want to show that if lying fails the universalizability test, then the FUL prohibits it.

```
consts lie::os
abbreviation lying-not-universal::bool where lying-not-universal ≡ ∀ w. ¬ (∀ p. lie(p) w)
```

— The formula above reads, "At all worlds, it is not the case that everyone lies."

```
lemma lying-prohibited:
  fixes p
  shows lying-not-universal ⟶ ⊨ (O {¬ (lie(p))})
  nitpick[user-axioms] oops
```

— Nitpick found a counterexample for card i = 1 and card s = 1:
Free variable: p = s_1 Skolem constant: λw. p = (λx. _)(i_1 := s_1)

Kroy's formulation also fails to show that if not everyone can lie, then lying is prohibited. There may be an issue here with our representation of lying not being universal. Specifically, the FUL is violated if it's not $possible$ for everyone to simultaneously lie, but the abbreviation above merely represents that fact that not everyone does, in fact, simultaneously lie. It's entirely possible that everyone can simultaneously lie, but for some reason, maybe out of some moral sense, do not. Let's test a more accurate version of the failure of the universalizability test.

We want to represent the sentence, call it $S \longleftrightarrow$ "At all worlds, it is not possible that everyone lies simultaneously." Consider the two abbreviations below.

abbreviation *everyone-lies*::*t* **where** *everyone-lies* $\equiv \lambda w.\ (\forall p.\ (lie(p)\ w))$
— This represents the term "all people lie". Naively, we might think to represent this as $\forall p.lie(p)$. In HOL, the $\forall$ operator has type $('a \rightarrow bool) \rightarrow bool$, where $'a$ is a polymorphic type representing the type of the argument to $\forall$. This is because the universal quantifier binds the argument of the term given to it, such that it turns an open term into a closed term. In the given example, $\forall$ has the type $(s \rightarrow bool) \rightarrow bool$, so it can only be applied to a formula of type $s \rightarrow bool$. In the abbreviation above, we're applying the quantifier to the sentence $lie(p)w$ for any arbitrary w, so the types cohere.
— The term above is true for a set of worlds i (recall that a term is true at a set of worlds) such that, at all the worlds w in i, all people at w lie.

abbreviation *lying-not-possibly-universal*::*bool* **where** *lying-not-possibly-universal* $\equiv \forall w.\ \neg(\Diamond everyone\text{-}lies\ w)$
— Armed with *everyone-lies* $\equiv \lambda w.\ \forall p.\ lie\ p\ w$, it's easy to represent the sentence S. The abbreviation above reads, "At all worlds, it is not possible that everyone lies."

lemma *lying-prohibited-2*:
shows *lying-not-possible-universal* $\longrightarrow\ \models (O\ \{\neg\ (lie(p))\})$
nitpick[*user-axioms*] **oops**
— Nitpick found a counterexample for card i = 1 and card s = 2:
Free variables: $lying\_not\_possible\_universal$ = True p = s_1

Even with the stronger assumption that it's not possible for everyone to lie simultaneously, Kroy's formulation is still not able to show that lying is prohibited for an arbitrary person. That's a problem! WHY IS THIS HAPPENING

3.2.5 Metaethical Tests

lemma *permissible*:
shows $\exists A.\ ((\neg\ (O\ \{A\}))\wedge(\neg\ (O\ \{\neg A\})))\ w$
nitpick [*user-axioms*, *falsify=false*] **oops**
— Nitpick found a model for card i = 1 and card s = 1:
Skolem constant: A =($\lambda x.\_$)(i_1 := False)
— Just as with the naive interpretation, permissibility is possible.

```
lemma fixed-formula-is-permissible:
  fixes A
  shows ((¬ (O {A})) ∧ (¬ (O {¬ A}))) w
  nitpick [user-axioms, falsify=false] oops
— Nitpick found a model for card i = 1:
Free variable: A = (λx._)(i1 := False)
```

— This bad result also holds under Kroy's formulation. Recall that we want this to fail - if A is "murder" then no model should render it permissible.

```
lemma arbitrary-obligations:
  fixes A::t
  shows O {A} w
  nitpick [user-axioms=true] oops
```

ought implies can

arbitrary obligations

conflicting obligations

obligation and permissible relationship

3.3 Misc

stuff from Kroy's paper

4 Related Work

In 1685, Leibniz dreamed of a universal calculator that could be used to resolve philosophical and theological disputes. At the time, the logical and computational resources necessary to make his dream a reality did not exist. Today, automated ethics is a growing field, spurred in part by the need for ethically intelligent AI agents.

Tolmeijer et al. [39] developed a taxonomy of works in implementing machine ethics. An implementation is characterized by (1) the choice of ethical theory, (2) implementation design decisions (e.g. testing), and (3) implementation details (e.g. choice of logic).

In this paper, I formalize Kantian ethics. There is a long line of work implementing other kinds of ethical theories, like consequentialism [1, 3] or particularism [5, 17]. Kantian ethics is a deontological, or rule based ethic, and there is also prior work implementing other deontological theories [16, 4, 2]. Kantian ethics specifically appears to be an intuitive candidate for formalization and implementation [33, 27]. In 2006, Powers [33] argued that an implementation of of Kantian ethics presented technical challenges, such as automation of a non-monotonic logic, and philosophical challenges, like a definition of the categorical imperative. There has also been prior work in formalizing Kantian metaphysics using I/O logic [37]. Deontic logic itself is inspired by Kant's "ought implies can" principle, but it does not include a robust formalization of the entire categorical imperative [14].

Lindner and Bentzen [6] have presented a formalization and implementation of Kant's second formulation of the categorical imperative using a custom logic. They present their goal as "not to get close to a correct interpretation of Kant, but to show that our interpretation of Kant's ideas can contribute to the development of machine ethics." My work aims to formalize Kant's ethic as faithfully as possible. I draw on the centuries of work in moral philosophy, as opposed to developing my own ethical theory. I also hope to formalize the first and third formulations of the categorical imperative, in addition to the first.

The implementation of this paper builds on Benzmueller, Parent, and Farjami's work with the LogiKey framework for machine ethics [7, 9]. The LogiKey project has been used to implement metaphysics [8, 20]. Fuenmayor and Benzmueller [15] have implemented Gewirth's principle of generic consistency, which is similar to Kant's formula of universal law.

5 Future Work

I intend to continue this research for the next year as part of my senior thesis. To make that process easier, I will sketch some goals for the rest of the project. In Section 3.2, I present a young and unfinished implementation of Kroy's formalization of the categorical imperative. The finished version of my project will ideally include an implementation of Kroy's formalization of the second formulation of the categorical imperative as well. I also hope to write robust tests for both of these implementations to explore their limitations. These tests will help inform my eventual formalization of the categorical imperative.

The ultimate goal of the project is to present my own formalization of the categorical imperative that escapes the limitations of the naive formalization and Kroy's formalization. This formalization will likely require some additional logical machinery to handle the complete notion of a maxim, including an agent, action, and end. My formalization will also patch up some of the holes in DDL itself that have been problematic for my project so far, such as the existence of contradictory obligations. I intend to formalize and implement all three formulations of the categorical imperative.

I will then test my formalization of the categorical imperative. I will create two kinds of tests. First, I will create metaethical tests that show logical properties independent of any model specification, as I did for the first two formalizations. Second, I will create tests that specify models and apply my formalization to real, concrete ethical dilemmas. This part of the project will seek to demonstrate the power and limitations of automated ethical reasoning. Questions to be explored here include: How much model specification is necessary to achieve ethical results? How should models be represented and specified? Does the automation of ethical reasoning provide anything, or is all the ethical work hidden in the model specification itself?

This final question is both technical and philosophical, and will be interesting to explore in the written component of my thesis. This question is related to Kant's distinction between analytic and synthetic reasoning [19]. Analytic statements are true simply by virtue of their meaning, such as "All bachelors are unmarried." Synthetic reasoning involves some contribution by the reasoner, in the form of new insight or facts about the world. Kant presents the statements "All bachelors are alone" and "7+5=12" as examples of synthetic propositions. The analytic/synthetic distinction is hotly debated and has been refined significantly since Kant, and this area will require further research.

Kant believes that ethics is synthetic a priori reasoning, but it is unclear if automated theorem provers like Isabelle are capable of anything more than analytic reasoning. Many of the basic proof solving tools like `simp` or `blast` simply unfold definitions and apply axioms, and they appear to perform analytic reasoning. SMT solvers like `Nitpick` and `z3` (bundled with Isabelle) are candidates for synthetic reasoning. Model finding seems more sophisticated than the simple unfolding of definitions, but this requires further exploration.

Lastly, I hope to explore Kant's argument that the three formulations of the categorical imperative are equivalent. This hypothesis has been the subject of controversy, but many neo-Kantians believe that his claim is plausible, if not true. Armed with formalizations of each formulation, I will have all the tools necessary to test this hypothesis. I would like to either prove or disprove this hypothesis for my formalization, and analyze the philosophical implications of my result.

References

[1] D. Abel, J. MacGlashan, and M. Littman. Reinforcement learning as a framework for ethical decision making. In *AAAI Workshop: AI, Ethics, and Society*, 2016.

[2] M. Anderson and S. Anderson. Geneth: A general ethical dilemma analyzer. volume 1, 07 2014.

[3] M. Anderson, S. Anderson, and C. Armen. Towards machine ethics. 07 2004.

[4] M. Anderson and S. L. Anderson. Ethel: Toward a principled ethical eldercare robot.

[5] K. D. Ashley and B. M. McLaren. A cbr knowledge representation for practical ethics. In *Selected Papers from the Second European Workshop on Advances in Case-Based Reasoning*, EWCBR '94, page 181–197, Berlin, Heidelberg, 1994. Springer-Verlag.

[6] M. M. Bentzen and F. Lindner. A formalization of kant's second formulation of the categorical imperative. *CoRR*, abs/1801.03160, 2018.

[7] C. Benzmüller, A. Farjami, and X. Parent. Dyadic deontic logic in hol: Faithful embedding and meta-theoretical experiments. In M. Armgardt, H. C. Nordtveit Kvernenes, and S. Rahman, editors, *New Developments in Legal Reasoning and Logic: From Ancient Law to Modern Legal Systems*, volume 23 of *Logic, Argumentation & Reasoning*. Springer Nature Switzerland AG, 2021.

[8] C. Benzmüller and B. W. Paleo. Formalization, mechanization and automation of gödel's proof of god's existence. *CoRR*, abs/1308.4526, 2013.

[9] C. Benzmüller, X. Parent, and L. W. N. van der Torre. Designing normative theories of ethical reasoning: Formal framework, methodology, and tool support. *CoRR*, abs/1903.10187, 2019.

[10] J. C. Blanchette, S. Böhme, and L. C. Paulson. Extending sledgehammer with smt solvers. In *Proceedings of the 23rd International Conference on Automated Deduction*, CADE'11, page 116–130, Berlin, Heidelberg, 2011. Springer-Verlag.

[11] J. C. Blanchette and T. Nipkow. *Nitpick: A Counterexample Generator for Higher-Order Logic Based on a Relational Model Finder*, volume 6172, page 131–146. Springer Berlin Heidelberg, 2010.

[12] J. Carmo and A. Jones. Completeness and decidability results for a logic of contrary-to-duty conditionals. *J. Log. Comput.*, 23:585–626, 2013.

[13] R. M. Chisholm. Contrary-to-duty imperatives and deontic logic. *Analysis (Oxford)*, 24(2):33–36, 1963.

[14] M. J. Cresswell and G. E. Hughes. *A New Introduction to Modal Logic*. Routledge, 1996.

[15] D. Fuenmayor and C. Benzmüller. Formalisation and evaluation of alan gewirth's proof for the principle of generic consistency in isabelle/hol. *Archive of Formal Proofs*, Oct. 2018. https://isa-afp.org/entries/GewirthPGCProof.html, Formal proof development.

[16] N. S. Govindarajulu and S. Bringsjord. On automating the doctrine of double effect. *CoRR*, abs/1703.08922, 2017.

[17] M. Guarini. Particularism and the classification and reclassification of moral cases. *IEEE Intelligent Systems*, 21(4):22–28, 2006.

[18] J. Hintikka. *Knowledge and Belief*. Cornell University Press, 1962.

[19] I. Kant. *Groundwork of the Metaphysics of Morals*. Cambridge University Press, Cambridge, 1785.

[20] D. Kirchner, C. Benzmüller, and E. N. Zalta. Computer science and metaphysics: A cross-fertilization. *CoRR*, abs/1905.00787, 2019.

[21] P. Kitcher. Kant's argument for the categorical imperative. *Nous*, 38, 2004.

[22] P. Kleingeld. Contradiction and kant's formula of universal law. *Kant-Studien*, 108(1):89–115, 2017.

[23] M. Kohl. Kant and 'Ought Implies Can'. *The Philosophical Quarterly*, 65(261):690–710, 05 2015.

[24] C. Korsgaard. Kant's Formula of Universal Law. *Pacific Philosophical Quarterly*, 66:24–47, 1985.

[25] C. Korsgaard. The Right to Lie: Kant on Dealing with Evil. *Philosophy and Public Affairs*, 15:325–249, 1986.

[26] M. Kroy. A partial formalization of kant's categorical imperative. an application of deontic logic to classical moral philosophy. *Kant-Studien*, 67(1-4):192–209, 1976.

[27] P. Lin, K. Abney, and G. A. Bekey. *Robotics, Ethical Theory, and Metaethics: A Guide for the Perplexed*, pages 35–52. 2012.

[28] P. McNamara and F. Van De Putte. Deontic Logic. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2021 edition, 2021.

[29] R. MONTAGUE. Universal grammar. *Theoria*, 36(3):373–398, 1970.

[30] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher Order Logic*. Springer-Verlag Berlin Heidelberg, Berlin, 2002.

[31] O. O'Neill. *Bounds of Justice*. Cambridge University Press, December 2009.

[32] L. Paulson and J. Blanchette. Three years of experience with sledgehammer, a practical link between automatic and interactive theorem provers. 02 2015.

[33] T. M. Powers. Prospects for a kantian machine. *IEEE Intelligent Systems*, 21(4):46–51, 2006.

[34] D. Rönnedal. Contrary-to-duty paradoxes and counterfactual deontic logic. *Philosophia*, 47, 09 2019.

[35] D. Scott. Advice on modal logic. In K. Lambert, editor, *Philosophical Problems in Logic: Some Recent Developments*, pages 143–173. D. Reidel, 1970.

[36] K. Solt. Deontic alternative worlds and the truth-value of 'oa'. *Logique et Analyse*, 27(107):349–351, 1984.

[37] A. Stephenson, M. Sergot, and R. Evans. Formalizing kant's rules: a logic of conditional imperatives and permissives. *Journal of Philosophical Logic*, 49, November 2019.

[38] J. Timmermann. Kantian dilemmas? moral conflict in kant's ethical theory. *Archiv für Geschichte der Philosophie*, 95(1):36–64, 2013.

[39] S. Tolmeijer, M. Kneer, C. Sarasua, M. Christen, and A. Bernstein. Implementations in machine ethics. *ACM Computing Surveys*, 53(6):1–38, Feb 2021.