Experimenting with Carmo and Jones' DDL

Lavanya Singh

February 17, 2021

Contents

Referencing Benzmuller and Parent's implementation: https://www.mi.fu-berlin.de/inf/groups/ag-ki/publications/dyadic-deontic-logic/C71.pdf

This theory contains the axiomatization of the system and some useful abbreviations.

theory *carmojones-DDL*
imports
Main

begin

1 System Definition

1.1 Definitions

This section contains definitions and constants necessary to construct a DDL model.

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

type-synonym $t = (i \Rightarrow bool)$
— t represents a set of DDL formulas.
— this set is defined by its truth function, mapping the set of worlds to the formula set's truth value.

— accessibility relations map a set of worlds to:
consts $av::i \Rightarrow t$ — actual versions of that world set
— these worlds represent what is "open to the agent"
— for example, the agent eating pizza or pasta for dinner might constitute two different actual worlds

consts $pv::i \Rightarrow t$ — possible versions of that world set
— these worlds represent was was "potentially open to the agent"
— for example, what someone across the world eats for dinner might constitute a possible world, — since the agent has no control over this

consts $ob::t \Rightarrow (t \Rightarrow bool)$ — set of propositions obligatory in this "context"
— ob(context)(term) is True if t is obligatory in the context

consts $cw::i$ — current world

1.2 Axiomatization

This subsection contains axioms. Because the embedding is semantic, these are just constraints on models.

This axiomatization comes from Carmo and Jones p 6 and the HOL embedding defined in Benzmuller and Parent

axiomatization where
ax-3a: $\exists\, x.\ av(w)(x)$
— every world has some actual version

and *ax-4a*: $\forall\, x.\ av(w)(x) \longrightarrow pv(w)(x)$
— all actual versions of a world are also possible versions of it

and *ax-4b*: $pv(w)(w)$
— every world is a possible version of itself

and *ax-5a*: $\neg ob(X)(\lambda w.\ \mathit{False})$
— in any arbitrary context X, something will be obligatory

and *ax-5b*: $\forall\, w.\ ((X(w)\ \wedge\ Y(w)) \longleftrightarrow (X(w)\ \wedge\ Z(w))) \longrightarrow (ob(X)(Y) \longleftrightarrow ob(X)(Z))$ — note that X(w) denotes w is a member of X
— X, Y, and Z are sets of formulas
— If X ∩ Y = X ∩ Z then the context X obliges Y iff it obliges Z

— ob(X)(λ w. Fw) can be read as F ∈ ob(X)
and *ax-5c*: $(\forall\, Z.\ \beta(Z) \longrightarrow ob(X)(Z) \wedge (\exists\, Z.\ \beta(Z))) \longrightarrow$
$(\exists\, y.(\forall\, Z.\ (\beta(Z) \longrightarrow Z(y))) \wedge X(y)) \longrightarrow ob(X)(\lambda w.\ \forall\, Z.(\beta(Z) \longrightarrow Z(w)))$
— For any nonempty subset β of ob(X), if its members share members with X then it members are all in ob(X)

and *ax-5d*: $(\forall\, w.\ (Y(w){\longrightarrow}X(w)) \wedge (ob(X)(Y)) \wedge (\forall\, w.\ (X(w) \longrightarrow Z(w)))) \longrightarrow (ob(Y)(\lambda w.\ Y(w) \vee (Z(w) \wedge \neg X(w))))$
— If some subset Y of X is in ob(X) then in a larger context Z, any obligatory proposition must either be in Y or in Z-X

and *ax-5e*: $((\forall\, w.\ (Y(w){\longrightarrow}X(w)))\ \wedge\ ob(X)(Z)\ \wedge\ (\exists\, w.(Y(w)\ \wedge\ Z(w)))) \longrightarrow ob(Y)(Z)$
— If Z is obligatory in context X, then Z is obligatory in a subset of X called Y, if Z shares some elements with Y

1.3 Abbreviations

These abbreviations are defined in Benzmuller and Parent, p9

These are all syntactic sugar for HOL expressions, so evaluating these symbols will be light-weight

— propositional logic symbols
abbreviation *ddlneg*::$t{\Rightarrow}t$ (¬)
  where $\neg\ A \equiv \lambda w.\ \neg\ A(w)$
abbreviation *ddlor*::$t{\Rightarrow}t{\Rightarrow}t$ (∨)
  where $\vee\ A\ B \equiv \lambda w.\ (A(w) \vee B(w))$
abbreviation *ddland*::$t{\Rightarrow}t{\Rightarrow}t$ (-∧-)
  where $A\wedge\ B \equiv \lambda w.\ (A(w) \wedge B(w))$
abbreviation *ddlif*::$t{\Rightarrow}t{\Rightarrow}t$ (-→-)
  where $A{\rightarrow}B \equiv \lambda w.\ (\neg\ A(w) \vee B(w))$

abbreviation *ddlequiv*::$t \Rightarrow t \Rightarrow t$ (-≡-)
  where $(A \equiv B) \equiv ((A \rightarrow B) \wedge (B \rightarrow A))$

— modal operators
abbreviation *ddlbox*::$t \Rightarrow t$ ($\Box$)
  where $\Box\ A \equiv \lambda w. \forall y.\ A(y)$
abbreviation *ddldiamond*::$t \Rightarrow t$ ($\Diamond$)
  where $\Diamond A \equiv \neg(\Box(\neg A))$

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

— modal symbols over the actual and possible worlds relations
abbreviation *ddlboxa*::$t \Rightarrow t$ ($\Box_a$)
  where $\Box_a A \equiv \lambda x. \forall y.\ (\neg\ av(x)(y) \vee A(y))$
abbreviation *ddldiamonda*::$t \Rightarrow t$ ($\Diamond_a$)
  where $\Diamond_a A \equiv \neg(\Box_a(\neg A))$
abbreviation *ddlboxp*::$t \Rightarrow t$ ($\Box_p$)
  where $\Box_p A \equiv \lambda x. \forall y.\ (\neg\ pv(x)(y) \vee A(y))$
abbreviation *ddldiamondp*::$t \Rightarrow t$ ($\Diamond_p$)
  where $\Diamond_p A \equiv \neg(\Box_a(\neg A))$

— obligation symbols over the actual and possible worlds
abbreviation *ddloba*::$t \Rightarrow t$ (O_a)
  where $O_a\ A \equiv \lambda x.\ ob(av(x))(A) \wedge (\exists y.(av(x)(y) \wedge \neg A(y)))$
abbreviation *ddlobp*::$t \Rightarrow t$ (O_p)
  where $O_p\ A \equiv \lambda x.\ ob(pv(x))(A) \wedge (\exists y.(pv(x)(y) \wedge \neg A(y)))$

— syntactic sugar for a "monadic" obligation operator
abbreviation *ddltrue*::t ($\top$)
  where $\top \equiv \lambda w.\ True$
abbreviation *ddlob-normal*::$t \Rightarrow t$ (*O*-)
  where $(O\ A) \equiv (O\{A|\top\})$

— validity
abbreviation *ddlvalid*::$t \Rightarrow bool$ ($\models$-)
  where $\models A \equiv \forall w.\ A(w)$
abbreviation *ddlvalidcw*::$t \Rightarrow bool$ ($\models_c$-)
  where $\models_c A \equiv A(cw)$

1.4 Consistency

Consistency is so easy to show in Isabelle!

lemma *True* **nitpick** [*satisfy*,*user-axioms*,*show-all*,*format=2*] **oops**
— Nitpick successfully found a countermodel.
— It's not shown in the document printout, hence the oops.
— If you hover over "nitpick" in JEdit, the model will be printed to output.

end

theory *carmojones-DDL-completeness* **imports** *carmojones-DDL*

begin

This theory shows completeness for this logic with respect to the models presented in carmojonesDDl.thy.

2 Inference Rules

2.1 Basic Inference Rules

These inference rules are common to most modal and propostional logics

lemma *modus-ponens*: **assumes** $\models A$ **assumes** $\models (A \rightarrow B)$
  shows $\models B$
  using *assms*(*1*) *assms*(*2*) **by** *blast*
— Because I have not defined a "derivable" operator, inference rules are written using assumptions.
— For further meta-logical work, defining metalogical operators may be useful

lemma *nec*: **assumes** $\models A$ **shows** $\models(\Box A)$
  by (*simp add*: *assms*)

lemma *nec-a*: **assumes** $\models A$ **shows** $\models(\Box_a A)$
  by (*simp add*: *assms*)
lemma *nec-p*: **assumes** $\models A$ **shows** $\models(\Box_p A)$
  by (*simp add*: *assms*)

2.2 Fancier Inference Rules

These are new rules that Carmo and Jones introduced for this logic.

lemma *Oa-boxaO*:
  assumes $\models(B \rightarrow ((\neg(\Box((O_a\ A) \rightarrow ((\Box_a w) \wedge O\{A|w\}))))))$
  shows $\models(B \rightarrow (\neg(\Diamond(O_a\ A))))$
  by (*metis ax-5a ax-5b*)
lemma *Oa-boxpO*:
  assumes $\models(B \rightarrow ((\neg(\Box((O_p\ A) \rightarrow ((\Box_p w) \wedge O\{A|w\}))))))$
  shows $\models(B \rightarrow (\neg(\Diamond(O_p\ A))))$
  by (*metis ax-5a ax-5b*)
— B and A must not contain w. not sure how to encode that requirement.

3 Axioms

3.1 Box

— is an S5 modal operator, which is where these axioms come from.
lemma *K*:
  shows $\models ((\Box(A \rightarrow B)) \rightarrow ((\Box A) \rightarrow (\Box B)))$

by *blast*

lemma *T*:
shows $\models ((\Box A) \rightarrow A)$
by *blast*

lemma *5*:
shows $\models ((\Diamond A) \rightarrow (\Box(\Diamond A)))$
by *blast*

3.2 O

This characterization of O comes from Carmo and Jones p 593

lemma *O-diamond*:
shows $\models (O\{A|B\} \rightarrow (\Diamond(B \wedge A)))$
using *ax-5b ax-5a*
by *metis*
— A is only obligatory in a context if it can possibly be true in that context.

lemma *O-C*:
shows $\models(((\Diamond(A \wedge (B \wedge C))) \wedge (O\{B|A\} \wedge O\{C|A\})) \rightarrow (O\{B \wedge C|A\}))$
by (*metis* (*no-types*, *lifting*) *ax-5b*)
— The conjunction of obligations in a context is obligatory in that context.
— The restriction $\diamond(ABC)$ is to prevent contradictory obligations and contexts.

lemma *O-SA*:
shows $\models(((\Box(A \rightarrow B)) \wedge ((\Diamond(A \wedge C)) \wedge O\{C|B\})) \rightarrow (O\{C|A\}))$
using *ax-5e* **by** *blast*
— The principle of strengthening the antecedent.

lemma *O-REA*:
shows $\models((\Box(A \equiv B)) \rightarrow (O\{C|A\} \equiv O\{C|B\}))$
using *O-diamond ax-5e* **by** *blast*
— Equivalence for equivalent contexts.

lemma *O-contextual-REA*:
shows $\models ((\Box(C \rightarrow (A \equiv B))) \rightarrow (O\{A|C\} \equiv O\{B|C\}))$
by (*metis ax-5b*)
— The above lemma, but in some context C.

lemma *O-nec*:
shows $\models(O\{B|A\} \rightarrow (\Box O\{B|A\}))$
by *simp*
— Obligations are necessarily obligated.

lemma *O-to-O*:
shows $\models(O\{B|A\} \rightarrow O(A \rightarrow B))$
by (*metis* (*no-types*, *lifting*) *O-REA ax-5a ax-5b*)
— Moving from the dyadic to monadic obligation operators.

3.3 Possible Box

— $_p$ is a KT modal operator.
lemma *K-boxp*:
  shows $\models((\Box_p(A \rightarrow B)) \rightarrow ((\Box_p A) \rightarrow (\Box_p B)))$
  by *blast*
lemma *T-boxp*:
  shows $\models((\Box_p A) \rightarrow A)$
  using *ax-4b* **by** *blast*

3.4 Actual Box

— $_a$ is a KD modal operator.
lemma *K-boxa*:
  shows $\models((\Box_a(A \rightarrow B)) \rightarrow ((\Box_a A) \rightarrow (\Box_a B)))$
  by *blast*
lemma *D-boxa*:
  shows $\models((\Box_a A) \rightarrow (\Diamond_a A))$
  using *ax-3a* **by** *blast*

3.5 Relations Between the Modal Operators

— Relation between , $_a$, and $_p$.
lemma *box-boxp*:
  shows $\models((\Box A) \rightarrow (\Box_p A))$
  by *auto*
lemma *boxp-boxa*:
  shows $\models((\Box_p A) \rightarrow (\Box_a A))$
  using *ax-4a* **by** *blast*

— Relation between actual/possible O and .
lemma *not-Oa*:
  shows $\models((\Box_a A) \rightarrow ((\neg(O_a\ A)) \wedge (\neg(O_a\ (\neg A)))))$
  using *O-diamond* **by** *blast*
lemma *not-Op*:
shows $\models((\Box_p A) \rightarrow ((\neg(O_p\ A)) \wedge (\neg(O_p\ (\neg A)))))$
  using *O-diamond* **by** *blast*
lemma *equiv-Oa*:
  shows $\models((\Box_a(A \equiv B)) \rightarrow ((O_a\ A) \equiv (O_a\ B)\ ))$
  using *O-contextual-REA* **by** *blast*
lemma *equiv-Op*:
  shows $\models((\Box_p(A \equiv B)) \rightarrow ((O_p\ A) \equiv (O_p\ B)\ ))$
  using *O-contextual-REA* **by** *blast*

— relationships between actual/possible O and and O proper.
lemma *factual-detach-a*:
  shows $\models(((O\{B|A\} \wedge (\Box_a A)) \wedge ((\Diamond_a B) \wedge (\Diamond_a(\neg B)))) \rightarrow (O_a\ B))$
  using *O-SA* **by** *auto*
lemma *factual-detach-p*:
  shows $\models(((O\{B|A\} \wedge (\Box_p A)) \wedge ((\Diamond_p B) \wedge (\Diamond_p(\neg B)))) \rightarrow (O_p\ B))$

by (*smt O-SA boxp-boxa*)

end