Proving God's Existence by Automating Leibniz' Algebra of Concepts

M. Bentert, Y. Lewash, D. Streit, S. Stugk

Project of Computational Metaphysics 2016

C. Benzmüller, A. Steen, M. Wisniewski

Freie Universität Berlin, Institute of Computer Science, Germany

Freie Universität Berlin

Berlin Mathematical School

Introduction

From 1679 onwards Leibniz developed a modal Logic of *Concepts*. Concepts play the role of *predicates* in ordinary predicate logic.

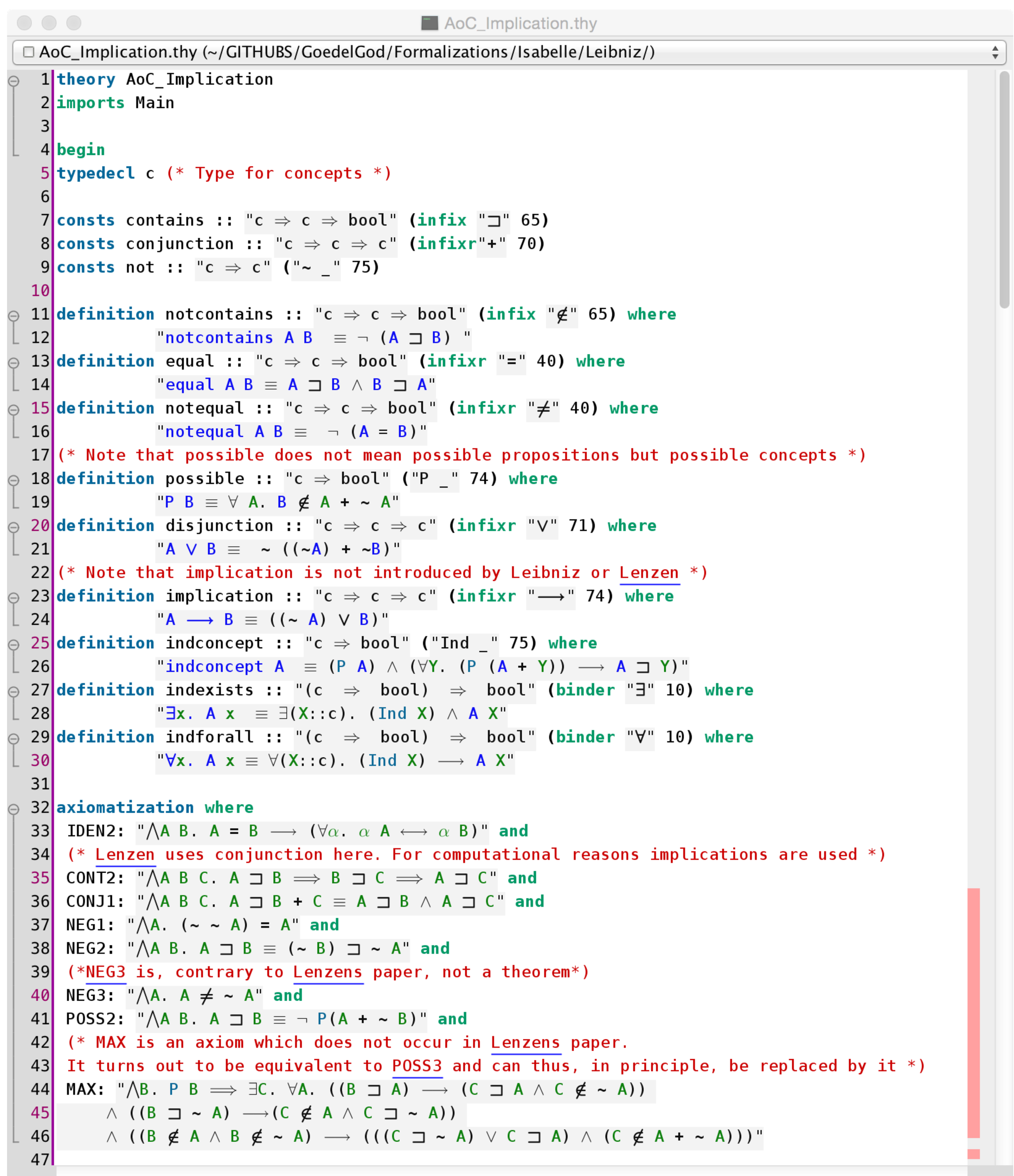
Wolfgang Lenzen showed [2] that one can easily apply Leibniz own logic to formalise a famous ontological argument by Leibniz in his own logic. From this starting point we formulated Lenzen formalisation in the proof assistant Isabelle/HOL and conducted (semi-) *empiric experiments* using Isabelle's automation tools.

Leibniz' Axioms

Leibniz defines three primitive operations on concepts.

- Concept Containment: $\in$
  "Blue" contains "colored"
- Concept Conjunction: $+$
  Combines two concepts into a composite concept
- Concept Negation: $\sim$
  Returns the "opposite concept"

The embedding of concepts, its primitive and definable operations, and Lenzen's axiomatization in Isabelle can be seen in Figure 1.



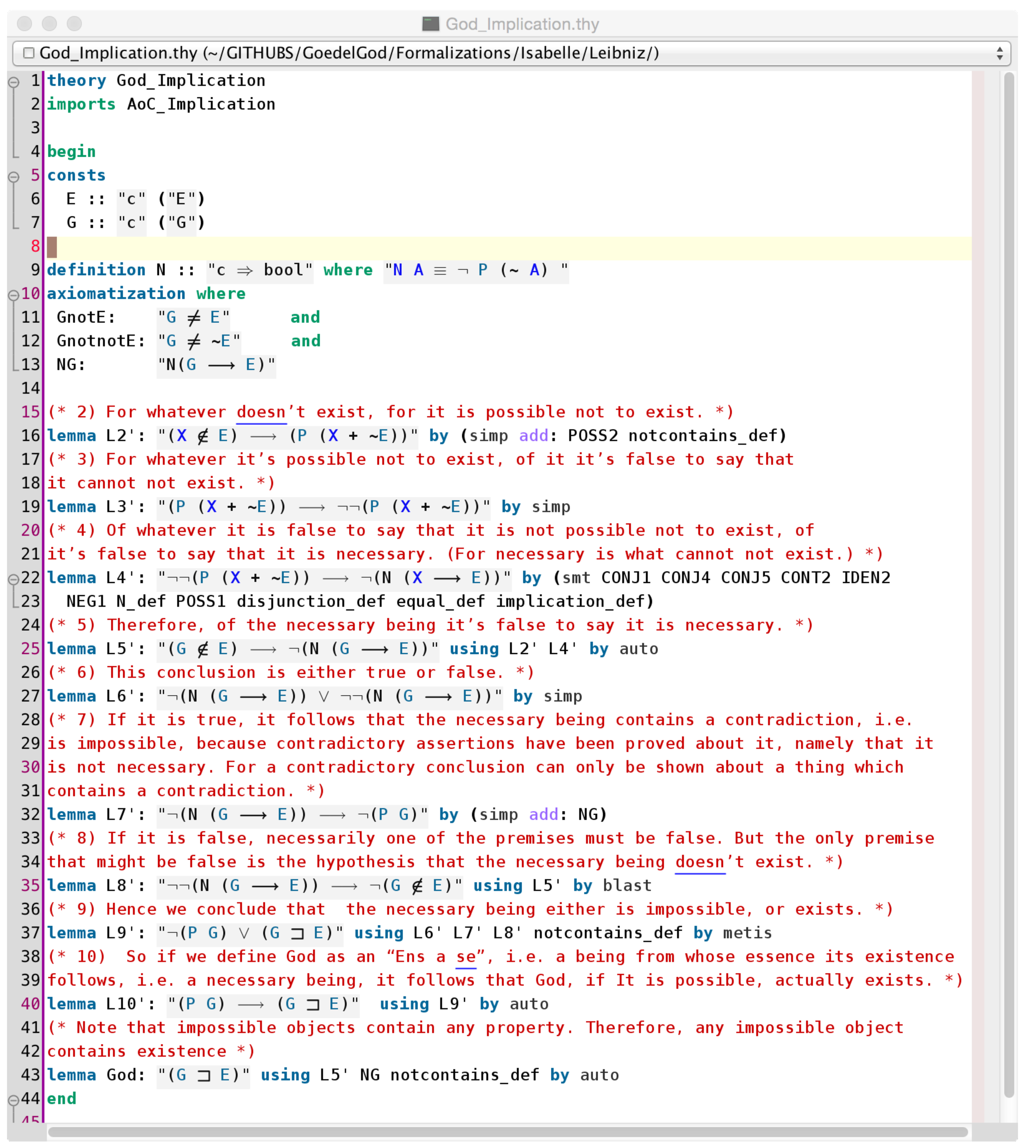
```
AoC_Implication.thy
AoC_Implication.thy (~/GITHUBS/GoedelGod/Formalizations/Isabelle/Leibniz/)

theory AoC_Implication
imports Main

begin
typedecl c (* Type for concepts *)

consts contains :: "c ⇒ c ⇒ bool" (infix "⊐" 65)
consts conjunction :: "c ⇒ c ⇒ c" (infixr"+" 70)
consts not :: "c ⇒ c" ("~ _" 75)

definition notcontains :: "c ⇒ c ⇒ bool" (infix "∉" 65) where
           "notcontains A B  ≡ ¬ (A ⊐ B) "
definition equal :: "c ⇒ c ⇒ bool" (infixr "=" 40) where
           "equal A B ≡ A ⊐ B ∧ B ⊐ A"
definition notequal :: "c ⇒ c ⇒ bool" (infixr "≠" 40) where
           "notequal A B ≡  ¬ (A = B)"
(* Note that possible does not mean possible propositions but possible concepts *)
definition possible :: "c ⇒ bool" ("P _" 74) where
           "P B ≡ ∀ A. B ∉ A + ~ A"
definition disjunction :: "c ⇒ c ⇒ c" (infixr "∨" 71) where
           "A ∨ B ≡  ~ ((~A) + ~B)"
(* Note that implication is not introduced by Leibniz or Lenzen *)
definition implication :: "c ⇒ c ⇒ c" (infixr "⟶" 74) where
           "A ⟶ B ≡ ((~ A) ∨ B)"
definition indconcept :: "c ⇒ bool" ("Ind _" 75) where
           "indconcept A  ≡ (P A) ∧ (∀Y. (P (A + Y)) ⟶ A ⊐ Y)"
definition indexists :: "(c  ⇒  bool)  ⇒  bool" (binder "∃" 10) where
           "∃x. A x  ≡ ∃(X::c). (Ind X) ∧ A X"
definition indforall :: "(c  ⇒  bool)  ⇒  bool" (binder "∀" 10) where
           "∀x. A x ≡ ∀(X::c). (Ind X) ⟶ A X"

axiomatization where
 IDEN2: "⋀A B. A = B ⟶ (∀α. α A ⟷ α B)" and
 (* Lenzen uses conjunction here. For computational reasons implications are used *)
 CONT2: "⋀A B C. A ⊐ B ⟹ B ⊐ C ⟹ A ⊐ C" and
 CONJ1: "⋀A B C. A ⊐ B + C ≡ A ⊐ B ∧ A ⊐ C" and
 NEG1: "⋀A. (~ ~ A) = A" and
 NEG2: "⋀A B. A ⊐ B ≡ (~ B) ⊐ ~ A" and
 (*NEG3 is, contrary to Lenzens paper, not a theorem*)
 NEG3: "⋀A. A ≠ ~ A" and
 POSS2: "⋀A B. A ⊐ B ≡ ¬ P(A + ~ B)" and
 (* MAX is an axiom which does not occur in Lenzens paper.
 It turns out to be equivalent to POSS3 and can thus, in principle, be replaced by it *)
 MAX: "⋀B. P B ⟹ ∃C. ∀A. ((B ⊐ A) ⟶ (C ⊐ A ∧ C ∉ ~ A))
     ∧ ((B ⊐ ~ A) ⟶(C ∉ A ∧ C ⊐ ~ A))
     ∧ ((B ∉ A ∧ B ∉ ~ A) ⟶ (((C ⊐ ~ A) ∨ C ⊐ A) ∧ (C ∉ A + ~ A)))"
```

Figure 1: Axiomatization in *Isabelle/HOL*

Preliminaries to the Divine

Leibniz' ontological argument deals with *necessary*, *possible* and *existence*. These three terms have to be defined before we start the proof itself.

- Existence is just a special concept "E". From today's perspective this seems inappropriate.
- Possibility "P" is a *derived notion*; it attaches to concepts not propositions.
  $P(C) :\Leftrightarrow \forall A : (C \notin (A + \sim A))$
- Leibniz proposed several notions of necessity over the course of his life. We use the straightforward interpretation here.
  $N(C) :\Leftrightarrow \neg P(\sim C)$
- Counterintuitively, Leibniz' modal logic is *extensional*.



```
God_Implication.thy
God_Implication.thy (~/GITHUBS/GoedelGod/Formalizations/Isabelle/Leibniz/)

theory God_Implication
imports AoC_Implication

begin
consts
  E :: "c" ("E")
  G :: "c" ("G")

definition N :: "c ⇒ bool" where "N A ≡ ¬ P (~ A) "
axiomatization where
 GnotE:    "G ≠ E"      and
 GnotnotE: "G ≠ ~E"     and
 NG:       "N(G ⟶ E)"

(* 2) For whatever doesn't exist, for it is possible not to exist. *)
lemma L2': "(X ∉ E) ⟶ (P (X + ~E))" by (simp add: POSS2 notcontains_def)
(* 3) For whatever it's possible not to exist, of it it's false to say that
it cannot not exist. *)
lemma L3': "(P (X + ~E)) ⟶ ¬¬(P (X + ~E))" by simp
(* 4) Of whatever it is false to say that it is not possible not to exist, of
it's false to say that it is necessary. (For necessary is what cannot not exist.) *)
lemma L4': "¬¬(P (X + ~E)) ⟶ ¬(N (X ⟶ E))" by (smt CONJ1 CONJ4 CONJ5 CONT2 IDEN2
  NEG1 N_def POSS1 disjunction_def equal_def implication_def)
(* 5) Therefore, of the necessary being it's false to say it is necessary. *)
lemma L5': "(G ∉ E) ⟶ ¬(N (G ⟶ E))" using L2' L4' by auto
(* 6) This conclusion is either true or false. *)
lemma L6': "¬(N (G ⟶ E)) ∨ ¬¬(N (G ⟶ E))" by simp
(* 7) If it is true, it follows that the necessary being contains a contradiction, i.e.
is impossible, because contradictory assertions have been proved about it, namely that it
is not necessary. For a contradictory conclusion can only be shown about a thing which
contains a contradiction. *)
lemma L7': "¬(N (G ⟶ E)) ⟶ ¬(P G)" by (simp add: NG)
(* 8) If it is false, necessarily one of the premises must be false. But the only premise
that might be false is the hypothesis that the necessary being doesn't exist. *)
lemma L8': "¬¬(N (G ⟶ E)) ⟶ ¬(G ∉ E)" using L5' by blast
(* 9) Hence we conclude that  the necessary being either is impossible, or exists. *)
lemma L9': "¬(P G) ∨ (G ⊐ E)" using L6' L7' L8' notcontains_def by metis
(* 10)  So if we define God as an "Ens a se", i.e. a being from whose essence its existence
follows, i.e. a necessary being, it follows that God, if It is possible, actually exists. *)
lemma L10': "(P G) ⟶ (G ⊐ E)"  using L9' by auto
(* Note that impossible objects contain any property. Therefore, any impossible object
contains existence *)
lemma God: "(G ⊐ E)" using L5' NG notcontains_def by auto
end
```

Figure 2: The entire proof in *Isabelle/HOL*

Proving God

Having defined the logic and basic concepts we can now sate what it means for god to be necessary.

- The term "being god(ly)" is identified with a concept G.
- The desired conclusion "god exists" is identified with $G \in E$ – the concept of god is contained in the concept of existence.
- Interestingly (see the paragraph below) the only working axiomatization for god as the necessary being seems to be $N(\sim G \vee E)$ or with "Concept Implication"
  $N(G \longrightarrow E)$ not $N(G)$

Results and Conclusion

In our work we were able to

- confirm that Leibniz' axiom system is consistent.
- computationally verify and improve upon Lenzen's formalisation.
- easily prove some worrying statements using Leibniz' system (e.g. Whatever possibly exists, exists actually).

We also found *novel* and perhaps *philosophically interesting* facts about Leibniz' ontological argument.

Leibniz uses "ens necessarium" and "ens ex cujus essentia sequitur existentia" interchangeably. In his own system however, there is profound difference between $N(G)$ and $N(G \longrightarrow E)$. If we use the former, the proof fails and Isabelle's nitpick routine quickly finds a countermodel. The latter works as advertised. Our results, especially the countermodels, will be published soon as a book chapter [1].

References

[1] Matthias Bentert, Christoph Benzmüller, David Streit, and Bruno Woltzenlogel-Paleo. Analysis of an ontological proof proposed by Leibniz. In Charles Tandy, editor, *Death and Anti-Death, Volume 14: Four Decades after Michael Polanyi, Three Centuries after G. W. Leibniz*. Ria University Press, 2017. To appear.

[2] Wolfgang Lenzen. Leibnizs ontological proof of the existence of god and the problem of impossible objects. forthcoming.

Leibniz: *Calculemus!*

Computational Metaphysics is a interdisciplinary lecture course designed for advanced students of computer science, mathematics and philosophy. The main objective of the course is to teach the students how modern proof assistants based on expressive higher-order logic support the formal analysis of rational arguments in philosophy (and beyond). In our first course in Summer 2016 the focus has been on ontological arguments for the existence of God. However, some students picked formalisation projects also from other areas (including maths).

Computational Metaphysics was awarded the *Central Teaching Award 2015* of the FU Berlin.