Formally Correct Deduction Methods for Computational Logic

Asta Halkjær From, PhD student, Technical University of Denmark

CICM-13 Doctoral Programme, 30 July 2020

Formally Correct Deduction Methods in Computational Logic

Technical University of Denmark.
February 2020 – January 2023 (a PhD in Denmark is 3 years).

Supervisor is Jørgen Villadsen.
Co-supervisor is Nina Gierasimczuk.

The goal of the project is to strengthen the logical foundations of computer reasoning by ensuring correctness of the deductive methods involved, using state-of-the-art tools like the Isabelle proof assistant.

Two-part talk:

- Overview of the project.
- Recent work on formalizing hybrid logic (IJCAR 2020 talk).

Formalizing Logics

This project takes up the challenge of developing and verifying logics for computer reasoning as well as algorithms based on these logics.

Higher-order logic.
First-order logic.
Natural logic.
Description logic and hybrid logic.
Epistemic logic and doxastic logic.

The Isabelle proof assistant allows exporting verified algorithms into programming languages like Haskell, OCaml and Scala, thus enabling us to produce running systems.

Archive of Formal Proofs

The Archive of Formal Proofs is a collection of proof libraries, examples, and larger scientific developments, mechanically checked in the theorem prover Isabelle. It is organized in the way of a scientific journal, is indexed by dblp and has an ISSN: 2150-914x. Submissions are refereed.

Number of Articles: 545
Number of Authors: 359
Number of lemmas: ~147,600
Lines of Code: ~2,568,200

https://www.isa-afp.org/

See also the Isabelle Formalization of Logic (IsaFoL) project:
https://bitbucket.org/isafol/isafol/

Epistemic Logic, AFP

This work is a formalization of epistemic logic with countably many agents. It includes proofs of soundness and completeness for the axiom system K. The completeness proof is based on the textbook "Reasoning About Knowledge" by Fagin, Halpern, Moses and Vardi (MIT Press 1995).

https://www.isa-afp.org/entries/Epistemic_Logic.html, 2018-10-29, 1672 lines.

A Sequent Calculus for First-Order Logic, AFP

This work formalizes soundness and completeness of a one-sided sequent calculus for first-order logic. The completeness is shown via a translation from a complete semantic tableau calculus, the proof of which is based on the First-Order Logic According to Fitting theory. The calculi and proof techniques are taken from Ben-Ari's Mathematical Logic for Computer Science.

https://www.isa-afp.org/entries/FOL_Seq_Calc1.html, 2019-07-18, 1070 lines.

Formalizing a Seligman-Style Tableau System for Hybrid Logic, AFP

This work is a formalization of soundness and completeness proofs for a Seligman-style tableau system for hybrid logic. The completeness result is obtained via a synthetic approach using maximally consistent sets of tableau blocks.

The formalization differs from previous work in a few ways. First, to avoid the need to backtrack in the construction of a tableau, the formalized system has no unnamed initial segment, and therefore no Name rule. Second, I show that the full Bridge rule is admissible in the system. Third, I start from rules restricted to only extend the branch with new formulas, including only witnessing diamonds that are not already witnessed, and show that the unrestricted rules are admissible. Similarly, I start from simpler versions of the @-rules and show that these are sufficient. The GoTo rule is restricted using a notion of potential such that each application consumes potential and potential is earned through applications of the remaining rules. I show that if a branch can be closed then it can be closed starting from a single unit. Finally, Nom is restricted by a fixed set of allowed nominals. The resulting system should be terminating.

https://devel.isa-afp.org/entries/Hybrid_Logic.html, 2019-12-20, 4909 lines.

Formalizing a Seligman-Style Tableau System for Hybrid Logic

Asta Halkjær From[1], Patrick Blackburn[2] and Jørgen Villadsen[1]

[1] Technical University of Denmark, [2] Roskilde University

IJCAR 2020, July 1.

Introduction

We present a tableau system ST^A for basic hybrid logic with a formalization in Isabelle/HOL (4900+ lines in the Archive of Formal Proofs): https://devel.isa-afp.org/entries/Hybrid_Logic.html

The system is based on ST* by Blackburn, Bolander, Braüner and Jørgensen.

Our short paper described work in progress and now the work has progressed. We show the latest system here.

The focus is on the process and results, not the proofs.

See the paper for details, related work and bibliography.

Hybrid Logic

Modal logic with names for worlds (nominals) and satisfaction operators ($@_i$).

$$\phi, \psi ::= x \mid i \mid \neg\phi \mid \phi \vee \psi \mid \Diamond\phi \mid @_i\phi$$

Semantics based on Kripke models ((*W*, *R*), *V*) and an assignment *g*.

W: underlying set, *R*: binary relation, *V*: unary relation, *g*: maps nominals to worlds.

$$\begin{array}{lll}
\mathfrak{M}, g, w \models x & \text{iff} & w \in V(x) \\
\mathfrak{M}, g, w \models i & \text{iff} & g(i) = w \\
\mathfrak{M}, g, w \models \neg\phi & \text{iff} & \mathfrak{M}, g, w \not\models \phi \\
\mathfrak{M}, g, w \models \phi \vee \psi & \text{iff} & \mathfrak{M}, g, w \models \phi \text{ or } \mathfrak{M}, g, w \models \psi \\
\mathfrak{M}, g, w \models \Diamond\phi & \text{iff} & \text{for some } w', wRw' \text{ and } \mathfrak{M}, g, w' \models \phi \\
\mathfrak{M}, g, w \models @_i\phi & \text{iff} & \mathfrak{M}, g, g(i) \models \phi
\end{array}$$

a nominal only holds at the world it names

satisfactions statements shift perspective

Hybrid Logic in Isabelle – Syntax

Deep embedding of the logic.

The syntax of our object logic becomes a **datatype** in the metalogic.

```
datatype ('a, 'b) fm
  = Pro 'a
  | Nom 'b
  | Neg ‹('a, 'b) fm› (‹¬ _› [40] 40)
  | Dis ‹('a, 'b) fm› ‹('a, 'b) fm› (infixr ‹∨› 30)
  | Dia ‹('a, 'b) fm› (‹◇ _› 10)
  | Sat 'b ‹('a, 'b) fm› (‹@ _ _› 10)
```

We specify the usual infix notation and can give other operators as abbreviations:

```
abbreviation Box (‹□ _› 10) where
  ‹□ p ≡ ¬ (◇ ¬ p)›
```

Hybrid Logic in Isabelle – Semantics

Type variables represent the universes of worlds (‘w), propositional symbols (‘a) and nominals (‘b).

```
datatype ('w, 'a) model =
  Model (R: ‹'w ⇒ 'w set›) (V: ‹'w ⇒ 'a ⇒ bool›)

primrec semantics
  :: ‹('w, 'a) model ⇒ ('b ⇒ 'w) ⇒ 'w ⇒ ('a, 'b) fm ⇒ bool›
  (‹_, _, _ ⊨ _› [50, 50, 50] 50) where
  ‹(M, _, w ⊨ Pro x) = V M w x›
| ‹(_, g, w ⊨ Nom i) = (w = g i)›
| ‹(M, g, w ⊨ ¬ p) = (¬ M, g, w ⊨ p)›
| ‹(M, g, w ⊨ (p ∨ q)) = ((M, g, w ⊨ p) ∨ (M, g, w ⊨ q))›
| ‹(M, g, w ⊨ ◇ p) = (∃v ∈ R M w. M, g, v ⊨ p)›
| ‹(M, g, _ ⊨ @ i p) = (M, g, g i ⊨ p)›
```

We can now talk about the logic in the formal language of higher-order logic.

Seligman-Style Tableau System – Blocks

Formulas are true relative to a given world.

- Internalized calculus: Satisfaction statements only.
- Seligman-style: Group formulas into blocks named by *opening nominals*.

Blocks and branches are modelled naturally:

```
type_synonym ('a, 'b) block = ‹('a, 'b) fm list × 'b›
type_synonym ('a, 'b) branch = ‹('a, 'b) block list›
```

We assume that the initial block is always named to obtain this simple modelling.

$$\begin{array}{c} \vdots \\ @_i\phi_1 \\ @_i\phi_2 \\ \vdots \\ \hline @_j\psi_1 \\ \vdots \end{array} \qquad \begin{array}{c} \vdots \\ \hline i \\ \phi_1 \\ \phi_2 \\ \vdots \\ \hline j \\ \psi_1 \\ \vdots \end{array}$$

(a) Internalized. (b) Seligman-style.

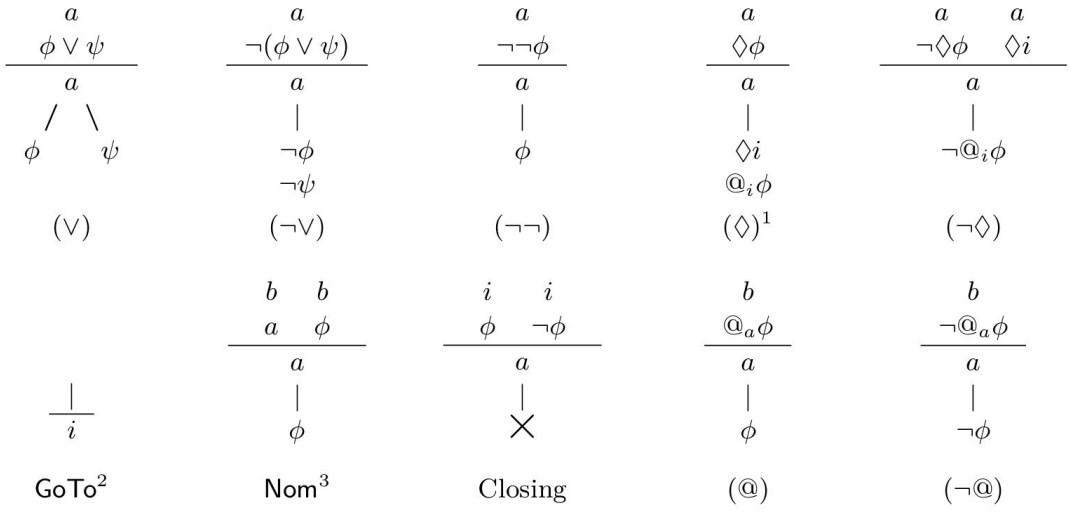
Seligman-Style Tableau System – Rules

Beware: The horizontal line is the block separator.

The vertical lines give extensions / justifications.

Each input formula must appear on a block named by the nominal written above it.

Multiple inputs are written next to each other.



[1] i is fresh, $i \notin A$ and ϕ is not a nominal.
[2] i is not fresh.
[3] If $\phi = i$ or $\phi = \Diamond i$ for some nominal i then $i \in A$.

Seligman-Style Tableau System – Restrictions

We want to rule out the construction of infinite branches.

R1 The output of a (non-GoTo) rule must include a formula new to the current block type.

R2 The ($\Diamond$) rule can only be applied to input $\Diamond\phi$ on an a-block if $\Diamond\phi$ is not already witnessed at a by formulas $\Diamond i$ and $@_i\phi$ for some witnessing nominal i.

R4 The GoTo rule consumes one *potential*. The remaining rules add one unit of potential and we are allowed to start from any amount.

Our @-rules adhere to the **R5** given by Blackburn et al.

R1 and **R2** can be *lifted*.

Potential constrains GoTo without hindering induction.

We restrict the Nom rule by a set of allowed *nominals.*

0.	a		
1.	$\neg(\neg@_i\phi \vee @_i\phi)$		[0]
2.	$\neg\neg@_i\phi$	$(\neg\vee)$ 1	[1]
3.	$\neg@_i\phi$	$(\neg\vee)$ 1	[2]
4.	$@_i\phi$	$(\neg\neg)$ 2	[3]
5.	i	GoTo	[2]
6.	$\neg\phi$	$(\neg@)$ 3	[3]
7.	ϕ	$(@)$ 4	[4]
	$\times$		

Seligman-Style Tableau System – Formalization

The turnstile predicate holds if the branch can be closed (wrt. *A* and *n*). E.g.

```
Close:
‹p at i in branch ⟹ (¬ p) at i in branch ⟹
 A, n ⊢ branch›
```

We can then machine verify results like soundness:

```
theorem soundness_fresh:
  assumes ‹A, n ⊢ [([¬ p], i)]› ‹i ∉ nominals p›
  shows ‹M, g, w ⊨ p›
```

And completeness:

```
theorem completeness:
  fixes p :: ‹('a :: countable, 'b :: countable) fm›
  assumes
    inf: ‹infinite (UNIV :: 'b set)› and
    valid: ‹∀(M :: ('b set, 'a) model) g w. M, g, w ⊨ p›
  shows ‹nominals p, 1 ⊢ [([¬ p], i)]›
```

Concluding Remarks

The nominals of hybrid logic allow us to witness the diamond modality.

Seligman-style rules work on arbitrary formulas due to explicit perspective shifts.

We can model the logic and proof system in Isabelle/HOL in a natural way.

In doing so, we gain absolute trust in our results: no omissions, no ambiguity.

Next steps:

- Show that the system is terminating.
- Verify a decision procedure based on the calculus.

I have an upcoming short presentation at Advances in Modal Logic 2020.