Hybrid Logic in the Isabelle Proof Assistant: Benefits, Challenges and the Road Ahead

From, Asta Halkjær

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
From, A. H. (2020). *Hybrid Logic in the Isabelle Proof Assistant: Benefits, Challenges and the Road Ahead*. 23-27. Paper presented at Advances in Modal Logic 2020.

Hybrid Logic in the Isabelle Proof Assistant: Benefits, Challenges and the Road Ahead

Asta Halkjær From

DTU Compute — Technical University of Denmark

Abstract

We outline benefits of formalizing a proof system for hybrid logic in the proof assistant Isabelle/HOL, showcase how the process of formalization can shape our proofs, and describe our current work on formalizing completeness of a more restrictive system. Formalization: `https://devel.isa-afp.org/entries/Hybrid_Logic.html`

Keywords: Hybrid logic, Seligman-style tableau, Isabelle/HOL

1 Introduction

Basic hybrid logic extends ordinary modal logic with nominals, a special sort of propositional symbol true at exactly one world, and satisfaction statements, $@_i\phi$, which are true if and only if the formula ϕ is true in the world named by nominal i. The well-formed formulas of the basic hybrid logic are defined as follows, where x is a propositional symbol and we use i, j, k, a, b for nominals:

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

The language is interpreted on Kripke models $\mathfrak{M}$, consisting of a frame (W, R) and a valuation of propositional symbols V. Here W is a non-empty set of worlds and R is a binary accessibility relation between them. To interpret nominals we use an assignment g mapping them to elements of W; if $g(i) = w$ we say that nominal i denotes w. Formula satisfiability is defined as follows:

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

We have just presented basic hybrid logic using (semi-formal) natural language, but we could have presented it using a proof assistant like Isabelle/HOL [7] instead. This forces us to be more precise: we would have to define hybrid logic in the proof assistant's logic (here, higher-order logic). But we can then do our metatheory in higher-order logic and machine check its correctness. This leaves no room for ambiguity or mistakes since every statement compiles to the primitives of the proof assistant (that we trust to be correct). Of course, we will have to supply more proof detail which can result in more verbose proofs; nonetheless, used skillfully, formalization can help guide our exploration of metatheory, and suggest new ideas, as we hope to show.

Hybrid logic has received little such treatment. Doczkal and Smolka formalize hybrid logic with nominals but no satisfaction operators in constructive type theory using the proof assistant Coq. They give algorithmic proofs of small model theorems and computational decidability of satisfiability, validity, and equivalence of formulas [3]. In Isabelle/HOL, Linker formalizes the semantic embedding of a spatio-temporal multi-modal logic that includes a hybrid logic-inspired at-operator but has no proof system [6]. The present work is the first sound and complete formalized proof system for hybrid logic that we know of. We have briefly described an earlier version of the formalization in a short paper for an automated reasoning audience [4], but that paper did not cover the notion of "potential" for restricting the GoTo rule.

2 Seligman-Style Tableau System

The proof system must handle the fact that a hybrid logic formula is true relative to a given world. Figures 1a and 1b depict two strategies for this.

$$\begin{array}{c} \vdots \\ \\ @_i\phi_1 \\ @_i\phi_2 \\ \vdots \\ \\ @_j\psi_1 \\ \vdots \end{array}$$

(a) Internalized.

$$\begin{array}{c} \vdots \\ \hline i \\ \phi_1 \\ \phi_2 \\ \vdots \\ \hline j \\ \psi_1 \\ \vdots \end{array}$$

(b) Seligman-style.

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]
	$\times$		

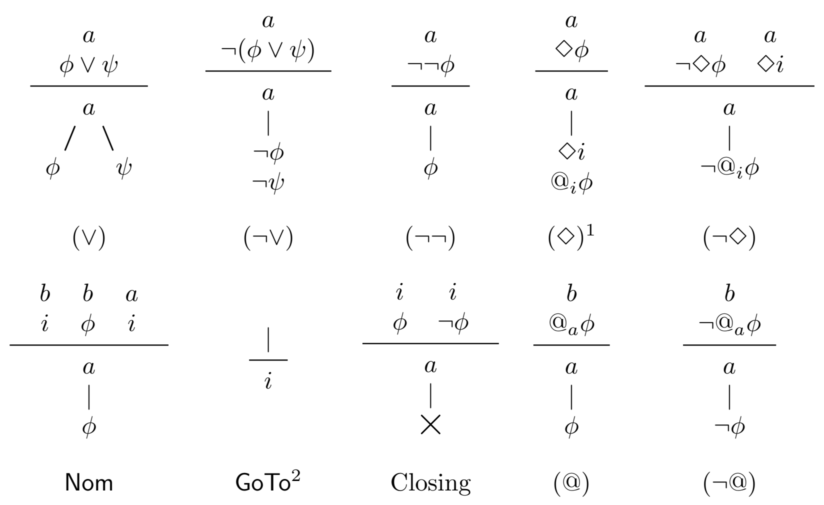
(c) Seligman-style tableau example.

Fig. 1. Tableau styles. (c) displays potential in the fourth column.

Internalized tableau systems work exclusively with satisfaction statements while the Seligman-style tableau system handles arbitrary formulas, giving a more local proof style, by dividing branches into blocks of formulas that are all true at the same world. Each pair of blocks is separated by a horizontal line and every block starts with a nominal dubbed the opening nominal, denoting that world. We call a block with opening nominal i an "i-block."

Figure 2 gives the tableau rules. Every rule has input formulas above the vertical line(s) and output below. The output of GoTo is a new block with corresponding opening nominal, while the other rules extend the last, so-called "current" block. When a rule has multiple input formulas we write them next to each other. Above each input formula, we write the opening nominal of the block it occurs on. Similarly, the opening nominal of the current block is the first thing below the horizontal line. Any formula on the current block may be used as input under the same restrictions on opening nominals. The system resembles (and simplifies) the one developed by Blackburn et al. [1], notably by having single-input (@) and (¬@) rules and assuming that all blocks have an opening nominal causing us to omit a rule.

Figure 1c gives an example tableau for the formula $\neg @_i \phi \vee @_i \phi$ which is negated and placed on a block with an arbitrary opening nominal. Note how the GoTo rule switches perspective to the world denoted by i while consuming a unit of potential in the fourth column.



[1] i is fresh, ϕ is not a nominal.
[2] i is not fresh.

Fig. 2. Tableau rules

We formalize this proof system as an inductive predicate, $\vdash$, in Isabelle by specifying for which branches $\vdash$ holds. For example, the closing condition becomes the following code that allows you to close any branch where, for some p and i, both p and $\neg p$ occur on i-blocks ("at i") in the branch:

Close: ‹*p at i in branch* ⟹ (¬ *p*) *at i in branch* ⟹ *n* ⊢ *branch*›

Here, n is the "potential" from Figure 1c. After defining all cases we can type in a closing tableau and have the computer check that every rule is applied according to our definition: we get a proof checker for free. Moreover, we can machine verify proofs of soundness and completeness.

3 Rule Induction

When we define the proof system, Isabelle provides a principle for proving statements by induction on the construction of a closing tableau. We consider a special case of the principle here, which is used to show lemmas of the form "if the branch Θ closes then so does $f(\Theta)$" where f is some transformation of the branch. Examples of transformations could be to rename nominals or to omit redundant occurrences of formulas.

The induction principle then instructs us, for each rule, to assume that the branch extended by that rule's output has a closing tableau when transformed and show that a closing tableau exists without the extension, typically by applying the rule in question. For instance, in the $(\neg\neg)$ case we assume, first, the premise of the rule, that $\neg\neg\phi$ occurs on an a-block in Θ where a is the opening nominal of the current block. Second: we assume as induction hypothesis that the transformation of Θ extended by ϕ has a closing tableau. To prove the case we need to show that the transformation of just Θ has a closing tableau.

This induction principle is our motivation for rephrasing the following restriction on the proof system by Blackburn et al. [1]:

Original R4 The GoTo rule cannot be applied twice in a row.

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

1.	a	
2.	$\neg\neg\phi$	
	$\vdots$	
3.	a	GoTo
4.	ϕ	$(\neg\neg)$ 2
5.	i	GoTo

(a) Starting point.

1.	a	
2.	$\neg\neg\phi$	
3.	ϕ	
	$\vdots$	
4.	a	GoTo
5.	ϕ	~~$(\neg\neg)$ 2~~
6.	i	~~GoTo~~

(b) Transformed. 5 and 6 are now illegal.

Fig. 3. Unjustified GoTo after weakening on line 3. We assume restriction **R1** [1], that extensions must be new.

The original restriction rules out infinite branches that consist of repeated applications of GoTo. Potential does the same because it decreases with each application. This new formulation, however, works better with the induction principle outlined above, since that principle may force us to apply GoTo twice in a row. Consider Figure 3b where the transformation of the branch means we should not apply GoTo on line 4 as in the tableau we are mimicking but go directly to line 6. With the original **R4** we would need a more intricate transformation of the branch (or a weaker lemma), but with the current restriction we can simply assume that we start with more potential, making the detour benign. The restriction preserves completeness as any closed tableau is finite.

Also, we can always start from a single unit:

Theorem 3.1 (Potential) *If a branch can be closed then it can be closed starting from a single unit of potential. (cf. "No detours" in the formalization.)*

4 Current Work

We have lifted equivalents of the four relevant restrictions by Blackburn et al. [1] (**R1**, **R2** and **R5**) in previous work [4]. Unfortunately, the Nom rule as given can still be used to construct infinite branches [1]. Blackburn et al. replace it with a three-part Nom* rule without this problem and show that it is sufficient for their translation-based completeness proof [1]. Instead of splitting it, we may impose the following, equivalent restriction on the general Nom rule:

Nom* $i = a$ and ϕ is not k or $\Diamond k$ for any k introduced by the $(\Diamond)$ rule.

This restriction means that "$(\Diamond)$-produced" nominals can only appear on their own as opening nominals. This breaks a symmetry otherwise present in exhausted branches: if nominal i appears on a k-block then k also appears on an i-block. The synthetic completeness proof by Jørgensen et al. [5] that we have previously formalized [4] makes use of this symmetry in their modeling of open exhausted branches and their model existence result. We have overcome this by (a) updating the definition of Hintikka sets to model our non-symmetric branches and (b) applying the model existence result by Bolander and Blackburn for a terminating internalized calculus [2] to our synthetic setting.

5 Conclusion

Modern proof assistants are more than capable of handling non-trivial proof systems and their metatheory. It can still be beneficial to shape our proofs such that they work well with the tools provided by the assistant, but in return we gain precision and absolute trust in the correctness of our results.

References

[1] Blackburn, P., T. Bolander, T. Braüner and K. F. Jørgensen, Completeness and Termination for a Seligman-style Tableau System, Journal of Logic and Computation **27** (2017), pp. 81–107.

[2] Bolander, T. and P. Blackburn, Termination for Hybrid Tableaus, Journal of Logic and Computation **17** (2007), pp. 517–554.

[3] Doczkal, C. and G. Smolka, Constructive Formalization of Hybrid Logic with Eventualities, in: Certified Programs and Proofs (CPP). Proceedings, 2011, pp. 5–20.

[4] From, A. H., P. Blackburn and J. Villadsen, Formalizing a Seligman-style Tableau System for Hybrid Logic, in: N. Peltier and V. Sofronie-Stokkermans, editors, Automated Reasoning (2020), pp. 474–481.

[5] Jørgensen, K. F., P. Blackburn, T. Bolander and T. Braüner, Synthetic Completeness Proofs for Seligman-style Tableau Systems, in: Advances in Modal Logic, Volume 11, 2016, pp. 302–321.

[6] Linker, S., Hybrid Multi-Lane Spatial Logic, Archive of Formal Proofs (2017), `http://isa-afp.org/entries/Hybrid_Multi_Lane_Spatial_Logic.html`, Formal proof.

[7] Nipkow, T., L. C. Paulson and M. Wenzel, "Isabelle/HOL - A Proof Assistant for Higher-Order Logic," Lecture Notes in Computer Science **2283**, Springer, 2002.