A Seligman-style Tableau System

Patrick Blackburn[1], Thomas Bolander[2], Torben Braüner[1], and Klaus Frovin Jørgensen[1]

[1]Roskilde University and [2]The Technical University of Denmark

Abstract. Proof systems for hybrid logic typically use @-operators to access information hidden behind modalities; this labeling approach lies at the heart of most resolution, natural deduction, and tableau systems for hybrid logic. But there is another, less well-known approach, which we have come to believe is conceptually clearer. We call this Seligman-style inference, as it was first introduced and explored by Jerry Seligman in the setting of natural deduction and sequent calculus in the late 1990s. The purpose of this paper is to introduce a Seligman-style tableau system.

The most obvious feature of Seligman-style systems is that they work with arbitrary formulas, not just formulas prefixed by @-operators. To achieve this in a tableau system, we introduce a rule called GoTo which allows us to "jump to a named world" on a tableau branch, thereby creating a local proof context (which we call a *block*) on that branch. To the surprise of some of the authors (who have worked extensively on developing the labeling approach) Seligman-style inference is often clearer: not only is the approach more modular, individual proofs can be more direct. We briefly discuss termination and extensions to richer logics, and relate our system to Seligman's original sequent calculus.

1 Introduction

Hybrid logic is a simple extension of ordinary modal logic in which it is possible to name possible worlds (or computational states, or epistemic states, or locations, or times, or situations, or whatever entities are required for the application at hand). Special propositional symbols called *nominals* are added to the underlying modal language. These symbols are true at exactly one world, thus a nominal i 'names' the unique world it is true at. In addition, a collection of modal operators of the form $@_i$ is added. Such a modality wears its intended interpretation on its sleeve: $@_i\varphi$ is true at any world w iff it is true at the (unique) world named by i. Such expressions are called *satisfaction statements*.

It is relatively straightforward to define proof systems for hybrid logic in a range of reasoning styles (including tableau [26, 5, 7, 11, 10], natural deduction [13], and resolution [1, 2]). A resolution theorem prover exists (the HyLoRes system [3]) as well as at least two high-performance tableau provers (namely HTab [20, 21] and Spartacus [19]). Indeed, even the least practical of all proof styles (the humble Hilbert system) turns out to be well-behaved [9, 8]. Moreover, proof systems in different styles can also be given for intuitionistic hybrid

logic, which is obtained by replacing the classical base logic by an intuitionistic one; see [13, 16, 18] for a variety of approaches. In [22], an intuitionistic version of the modal logic S5 extended with @-operators has been proposed as a foundation for distributed functional programming languages; the @-operators are used to reason about the distribution of resources at different locations.

But behind this apparent diversity lies a common strategy, namely *labeling*. Details vary, but in one form or another the basic idea is to use nominals and satisfaction statements to reach behind the modalities and access the information hidden there. Of course, labeled deduction methods are used for many non-classical logics, but the link between labeling and hybrid logic is particularly intimate—*nominals and satisfaction operators provide labeling apparatus built into the object language itself*. So there is a tendency to think that inference in hybrid logic has to be (some form) of labeled deduction. But this is wrong. There is another approach, which we call *Seligman-style* inference, which offers an interesting alternative. The main purpose of this paper is to explore this proof-style in the setting of tableau-based reasoning for hybrid logic.

The difference between label-driven and Seligman-style inference is best introduced by example. Let's consider two ways of formulating the $\Diamond$-elimination rule in a natural deduction framework for hybrid logic. Here's the rule that the label-driven approach naturally leads to:

$$\dfrac{\begin{array}{cc} & [@_i\Diamond j]\ [@_j\varphi] \\ & \vdots \\ @_i\Diamond\varphi & @_k\psi \end{array}}{@_k\psi}\ (\Diamond\mathrm{E})$$

This is easy to explain. We make two assumptions: first that at the world named i we can see a world named j (which is what the satisfaction statement $@_i\Diamond j$ says), and second that φ holds at j (which is what the satisfaction statement $@_j\varphi$ says). We assume nothing else about j beyond these two facts: in effect we have said "let j be an arbitrary world accessible from i at which φ is true". Now, if from these two assumptions we can prove some formula $@_k\psi$ (which says that ψ holds at the world named k) then from a proof of an existential statement $@_i\Diamond\varphi$ (which says that at i it is the case that φ holds at some accessible world) then we get a proof of $@_k\psi$.[1]

A little thought will show that this is a sound rule, but note its *form*. In particular, note that *all the formulas used in this rule are satisfaction statements*. Now, satisfaction statements are global. This is easy to see. If φ is indeed true at the world named i, then $@_i\varphi$ is true at *all* worlds. On the other hand, if φ is false at the world named i, then $@_i\varphi$ is false at *all* worlds. Thus satisfaction statements

[1] For this rule to be correctly applied, j has to be a fresh nominal different from both i and k, and j must not occur in either φ or ψ or in any non-discharged assumptions of the proof other than those specified. The assumptions $@_i\Diamond j$ and $@_j\varphi$ occurring as assumptions in the sub-proof on the right are discharged in the application of the rule. For more on natural deduction in hybrid logic, see Braüner [13].

embody global information. And this means that the labeled natural-deduction rule just formulated controls the reasoning by adopting a *global* perspective.

Contrast this with Seligman systems. In a Seligman-style natural deduction system the $\Diamond$-elimination rule would look like this:

$$\dfrac{\Diamond\varphi \qquad \begin{array}{c}[\Diamond j]\ [@_j\varphi] \\ \vdots \\ \psi\end{array}}{\psi}\ (\Diamond\mathrm{E})$$

Notice the *local* perspective illustrated by the rule.[2] The premises are not packed inside satisfaction statements. We assume that j is a possible world. We may not know the name of the world where we are currently evaluating formulas; we only know that there is a possible world accessible from it (named j) at which φ holds. Now, if it is possible for us, given this information, to prove some formula ψ (in which j doesn't occur), then we actually have a proof of ψ given a proof of $\Diamond\varphi$. The core of the argument is similar to that used in the labeled rule, but (so to speak) we use naked $\Diamond$ information: we don't wrap it up in the protection of satisfaction statements. In particular, we don't bother to specify a global name for the world in which we are working (which is what the $@_i$ operator does in the labeled version of the rule) and, as it turns out, we don't need to. Moreover, the subtree on the right is a free-floating proof context. It is linked to the world in which we are working only by a simple local claim, namely $\Diamond j$ (that is: there is an accessible world called j).

This is interesting for at least two reasons. The first is that it holds out the promise of more modular proof systems: if we don't have to wrap all our rules in a protective cocoon of satisfaction statements, perhaps we can work directly with the original rules for each connective. This is a possibility worth exploring. The second reason is conceptual. Modal logic is sometimes said to be interesting (see, for example, [8]) because of the *local* perspective it takes on possible worlds. But if hybrid logic relies on label-driven deduction, then it seems that its successes are due to the *global* encodings that satisfaction statements make possible. So it is worth investigating whether the more local approach to inference underlying Seligman-style reasoning adapts naturally to tableau systems.

Little has been written on Seligman-style systems. They were introduced in two papers, both by Jerry Seligman, written in the 1990s, namely the natural deduction based [24] and the Gentzen sequent calculus based [25].[3] The first of these papers gave a natural deduction system for a logic of situations, similar to hybrid logic. A characteristic feature of this system is that it has a proof rule enabling travel to another situation, the performance of some hypothetical

[2] The restriction for this rule is that j must not occur in φ, or ψ or in any non-discharged assumptions other than those specified.

[3] Another Gentzen system for hybrid logic that allows arbitrary formulas to occur in derivations can be found in [23]. The latter system makes use of standard Gentzen machinery for the ordinary (non-hybrid) modal logic K, which makes it quite different from the Seligman-style system of [25].

reasoning there, followed by a journey back again (as the reader will see, a similar idea underlies the GoTo rule in our tableau-based approach). This natural deduction system was later modified in Braüner [12] with the aim of obtaining a proof-theoretic property called closure under substitution, which requires keeping more detailed track of hypothetical reasoning. The modified system kept track of hypothetical reasoning, using what are known as *explicit substitutions*, in a modal-logical context. Such explicit substitutions were also used in a natural deduction system for S4 given in [4], and are similar to the "proof boxes" used in linear logic.

The authors of this paper became interested in Seligman-style reasoning because of reasoning problems involving perspective shifts and contextual information. First, Braüner has used his Seligman-style natural deduction system to formalize a well-known false-belief task in cognitive psychology, the Smarties task.[4] To solve such tasks, the subject under investigation has to perform a shift of perspective, either to another person's view of the world, or to the subject's own view at an earlier time. Such shifts lie at the heart of Seligman-style natural deduction, which makes it a natural tool for modeling such problems; see Braüner [14] for further discussion.

More recently, Blackburn and Jørgensen [6] investigated *temporal indexicals*, that is, context-sensitive temporal terms such as *now*, *yesterday*, *today*, and *tomorrow*. They showed they could be modeled in hybrid logic, but did so using a labeled tableau calculus. In the course of adapting their work to other reasoning styles, it became clear that a Seligman-style tableau approach might allow a simpler presentation of the reasoning involved, but no such calculus existed. The present paper arose as an attempt to fill this gap.

2 The Seligman-style Tableau Calculus ST

Let's get down to details. We work with a basic hybrid language which includes a countable set of propositional symbols, a countable set of nominals, the propositional connectives $\neg$ and $\wedge$, the modal operator $\Diamond$, and for each nominal i an $@_i$-operator. This operator takes any formula φ as argument and (as we have already discussed) $@_i\varphi$ says that at the world named i, φ is true. Formulas are built as follows:

$$\varphi ::= i \mid p \mid \bot \mid \neg\varphi \mid \varphi \wedge \psi \mid \Diamond\varphi \mid @_i\varphi.$$

The other propositional connectives are defined in the usual way, and $\Box\varphi$ is defined to be $\neg\Diamond\neg\varphi$. Note that nominals play a special role in hybrid logic as

[4] In one form of the Smarties task, a child is presented with a Smarties tube, the well-known tubes that usually contain the candy-coated chocolate goodies. However this particular tube contains pencils. After the tube is opened, and the context is updated with the knowledge that there are pencils within, the child is asked: *What would your mother think was in the tube if she came in?* This requires a perspectival shift to a context in which the Smarties tube is unopened and its real contents undisclosed. Young and autistic children have difficulty with this—they tend to think that mother will say that there are pencils within, something she cannot possibly know.

they can occur either as subscripts to @ ("in operator position") or as formulas in their own right ("in formula position"). We generally use $i, j, k, \ldots$ to denote nominals and $p, q, r, \ldots$ to denote ordinary propositional symbols.

The semantics for the language of basic hybrid logic is given by interpreting formulas in models based on a frame (W, R) together with a valuation function V. Here W is a non-empty set (we call its elements worlds) and R is a binary relation on this set (the accessibility relation). The valuation V distributes information over the frame; that is, V takes atomic formulas to subsets of W and it satisfies the following two conditions:

1. $V(p)$ is a subset of W, when p is an ordinary propositional symbol.
2. $V(i)$ is a *singleton* subset of W, when i is a nominal.

Satisfiability in a model is defined in the usual way as a relation which obtains between a model $\mathfrak{M} = (W, R, V)$, a point W in the model, and a formula φ:

$$
\begin{array}{ll}
\mathfrak{M}, w \models \bot & \text{never} \\
\mathfrak{M}, w \models a & \text{iff } a \text{ is atomic and } w \in V(a) \\
\mathfrak{M}, w \models \neg\varphi & \text{iff } \mathfrak{M}, w \not\models \varphi \\
\mathfrak{M}, w \models \varphi \wedge \psi & \text{iff } \mathfrak{M}, w \models \varphi \text{ and } \mathfrak{M}, w \models \psi \\
\mathfrak{M}, w \models \Diamond\varphi & \text{iff for some } w',\ wRw' \text{ and } \mathfrak{M}, w' \models \varphi \\
\mathfrak{M}, w \models @_i\varphi & \text{iff } \mathfrak{M}, w' \models \varphi \text{ and } w' \in V(i).
\end{array}
$$

A formula φ is *true in* $\mathfrak{M} = (W, R, V)$ when for all worlds $w \in W$ we have that $\mathfrak{M}, w \models \varphi$. A formula is *valid* if it is true in all models.

Now for our tableau system. As we have already mentioned, one of the pleasant properties of Seligman systems is their modularity. So we simply use standard tableau rules for propositional logic as part of our system. The propositional rules we have chosen are shown in Figure 1.

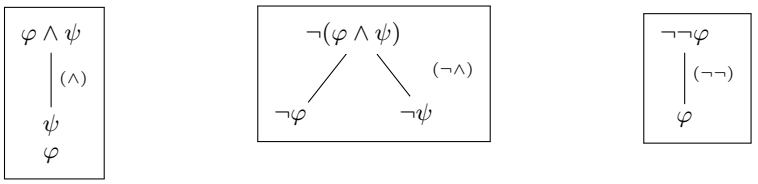


Fig. 1. Module 1: Tableau rules for propositional logic.

But what are the hybrid logical rules? How are we to get away from the global form of the labeled rules that are standardly used in hybrid tableau systems?

(The reader unfamiliar with labeled tableau rules for hybrid logic should consult the Appendix at the end of the paper.) To help us find an answer, let us consider again the Seligman-style rules for natural deduction discussed earlier. How do the rules of natural deduction deal with the local perspective? Well, in natural deduction, the key idea is to allow for *hypothetical* reasoning. This feature of natural deduction has been utilized in the Seligman-style natural deduction as follows: if from the assumption of being (locally) at some world (about which one has assumed nothing) one can conclude something that holds, then one can delete the assumption of being at that particular world and proceed with the reasoning. Putting it in a nutshell: the key deductive concept in natural deduction is hypothetical reasoning, and the Seligman approach finds a way to embody this concept locally.

So what is the key concept in tableau reasoning? The answer is: *branch expansion*. And how can branch expansion be localized? By means of a process which allows the branch to record *shifting perspectives on worlds*. We do this in our tableau calculus by dividing branches into *blocks*. The general idea here is that a block on a branch is a partial description of the information present at a specific world. The rule that allows us to work with multiple blocks is GoTo. The central idea behind our tableau calculus is that, within a given block, one can work freely with the formulas belonging to the block. But in the course of inference we will often need to make use of information about other worlds. The GoTo rule allows us to temporarily shut down our work in one block and shift to another. The rule opens this new block simply by stating a name for it. On the branch of a tableau, application of the GoTo rule is shown by a horizontal line with a nominal, say j, below the line. This notation means: we have just closed whatever block we were working with before and have shifted our attention to a new world, named j, and are now going to start creating a new block (partial description, proof context) involving this world. In short, just as in any tableau system, the fundamental mechanism is branch expansion. But the division of branches into blocks, and the ability to shift our attention between them that the GoTo rule provides, gives us what we need for Seligman-style reasoning.

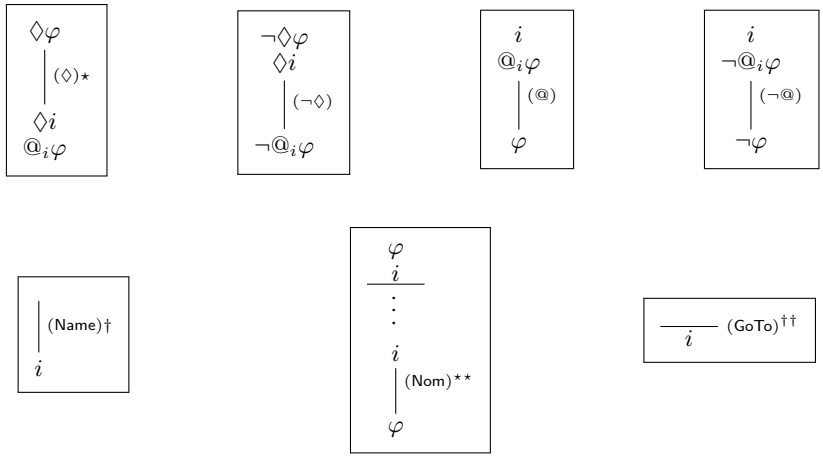
So much for intuition. Let's be more precise. Given a branch Θ in a tableau we define a *block* to be one of the following:

1. The *initial block*, consisting of all the formulas on Θ until the first horizontal line (or all formulas if there is no such line on Θ).
2. The *current block*, consisting of all formulas below the last horizontal line (or all formulas if there is no such line).
3. All formulas that occur between a pair of two consecutive horizontal lines.

The crucial rules of the Seligman-style tableau calculus are given in Figure 2 below. The conditions on rule applications are as follows:

– The propositional rules ($\wedge$), ($\neg\wedge$), ($\neg\neg$) as well as ($\Diamond$) and ($\neg\Diamond$) can only be applied to premises that belong to the current block (that is, these connectives, being local, have local rules).

– In the rules (@) and (¬@), the first premise i has to belong to the current block, whereas the second premise $@_i\varphi$ ($\neg@_i\varphi$) can appear anywhere on the branch (that is, these connectives, being global, transfer information across blocks).
– GoTo and Name can always be applied as they have no premises
– Nom can be applied as described in the rule itself: if φ and i belong to some block distinct from the current block, and i belongs to the current block, then φ can be added to the current block.



⋆ The nominal i is fresh and φ is not a nominal.
† The nominal i is fresh.
⋆⋆ The horizontal line below the two uppermost premises signifies that these premises belong to a block distinct from the current one, whereas the third premise (the lowermost occurrence of i) belongs to the current block.
†† The nominal i must be on the branch.

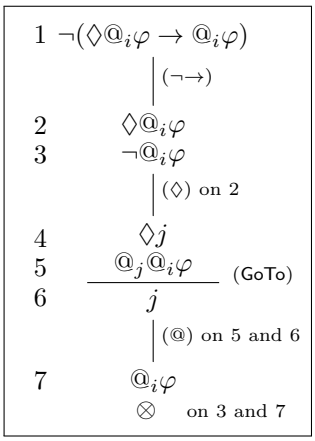
Fig. 2. Module 2: Tableau rules for basic hybrid logic.

It should be clear that the first four rules are simply the obvious (positive and negative) rules for ◊ and @. It is the last three rules that really drive the system. The first of these, Name, simply allows us to give a brand new name to a block. This is reminiscent of what GoTo does, and indeed, with a suitable side condition we could have collapsed Name and GoTo into a single rule. But the two rules play rather different roles in our system. Moreover (as we shall see) the role played by Name, though important, is a relatively restricted: as our completeness proof shows, it is never *necessary* to apply Name except possibly to name the initial block. So we prefer to keep the two rules distinct.

What does the Nom rule do? Recall that the GoTo rule enables us to (temporarily) close down a block and create a new one. But in the course of inference we may create multiple blocks, each of which embodies partial information about some world i. We will often need to integrate this information, and Nom lets us do this. Basically, it says that if i and φ occur together in some block, then, if you later find yourself at some bock that also contains i, you are free to recall that φ is true. The point is simply that both i-containing blocks are partial descriptions of the same world, namely the one named by i.

Summing up, our Seligman-style tableau calculus consists of Modules 1 and 2 given in Figure 1 and 2. We call this system ST. Tableaus are built in the expected way, but we should be explicit about our closure condition: a branch *closes* either by having a *local contradiction* φ and $\neg\varphi$ inside a block, or a *global contradiction* between formulas $@_i\varphi$ and $\neg@_i\varphi$ occurring anywhere on the branch.

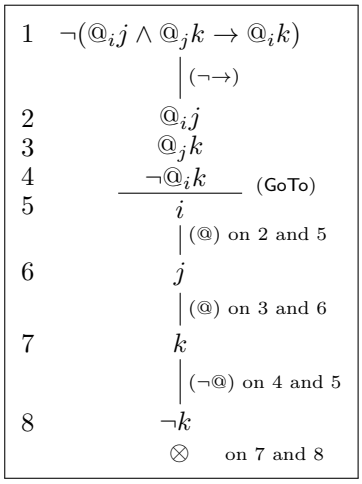
Let's look at an example, a proof in ST of $\Diamond@_i\varphi \to @_i\varphi$, a formula known as the *Back axiom*. Note that this example closes with a global contradiction.

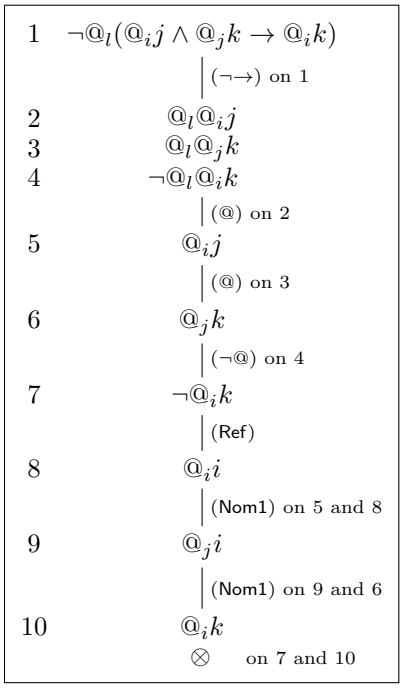


The argument should be clear: we apply an admissible propositional rule ($\neg\to$), and then eliminate the $\Diamond$. For the crucial step at line 5 we apply (GoTo) and jump to j; then apply (@) and the branch closes on $@_i\varphi$ and $\neg@_i\varphi$. The reasoning involved is clear and straightforward.

Indeed, it is instructive to compare proofs in our Seligman-style calculus ST with the proofs yielded by the standard labeled calculus LC (see the Appendix). Consider the following proof of $@_ij \wedge @_jk \to @_ik$, a hybrid validity which says that the world-naming relation is transitive. This is a telling example, as it requires equational reasoning about the identity of worlds. The tableau on the left is in the calculus ST, the one on the right in the calculus LC.[5] The Seligman-style approach makes the form of the argument clearer, and hides tedious bookkeeping details.

[5] Note that in both tableaus we have skipped the obvious application of the conjunctive rule right after ($\neg\to$).





3 Soundness and Completeness

Theorem 1 (Soundness). *If there exists a closed tableau in* ST *having* $\neg\varphi$ *as the root formula, then the formula* φ *is valid.*

Proof. Let Θ be a branch of a tableau of the calculus ST. Let B be a block on Θ and let $\mathfrak{M} = (W, R, V)$ be a model. We say that B is satisfiable by $\mathfrak{M}$ if and only if there exists a world $w \in W$ such that for any formula ψ in B, it is the case that $\mathfrak{M}, w \models \psi$. Moreover, we say that Θ is block-wise satisfiable by $\mathfrak{M}$ if and only if any block on Θ is satisfiable by $\mathfrak{M}$. We say that Θ is block-wise satisfiable if and only if Θ is block-wise satisfiable by some model $\mathfrak{M}$.

Now, the contrapositive of soundness follows from the observation that if a tableau T of the calculus ST has a branch which is block-wise satisfiable, then the tableau obtained by applying a rule to T also has a branch which is block-wise satisfiable. This can be seen simply by inspecting each rule in ST. □

We will now prove completeness of ST. We do so by providing a translation from tableaus in the labeled calculus LC into tableaus in ST.[6] The translation allows us to reduce completeness of ST to the completeness result for LC (see [5]). The approach also clarifies the relationship between the Seligman-style and labeling approaches, and yields some extra insights: for example, that the Name rule only needs to be used once in any ST tableau construction.

[6] Again, see the Appendix for the definition of LC.

Definition 1. *Let Θ be a tableau branch of the calculus* ST*. A formula $@_i\varphi$ ($\neg@_i\varphi$) is said to occur as an* ***induced formula*** *on Θ if there is a block B on Θ such that $i, \varphi \in B$ ($i, \neg\varphi \in B$).*

Lemma 1. *Let φ be any formula and i any nominal not in φ. Assume $T_{\mathtt{LC}}$ is a tableau with root $\neg@_i\varphi$ in the calculus* LC*. Then there exists a tableau $T_{\mathtt{ST}}$ with root $\neg\varphi$ in the calculus* ST*, and a bijection $b : \{\Theta \mid \Theta \text{ is a branch of } T_{\mathtt{LC}}\} \to \{\Theta' \mid \Theta' \text{ is a branch of } T_{\mathtt{ST}}\}$ such that:*

1. *Given any branch Θ of $T_{\mathtt{LC}}$, all formulas $@_j\psi \in \Theta$ and $\neg@_j\psi \in \Theta$ occur as induced formulas on $b(\Theta)$.*
2. *All nominals that occur on $b(\Theta)$ also occur on Θ.*

Proof. By induction of the number of rule applications made on $T_{\mathtt{LC}}$.

Base case. No rules have been applied and $T_{\mathtt{LC}}$ is $\neg@_i\varphi$, where i does not occur in φ. Then let $T_{\mathtt{ST}}$ be the following tableau in ST:

$$\begin{array}{l} \neg\varphi \\ \quad\big|\ (\mathsf{Name}) \\ i \end{array}$$

T and T' both have a single branch, call them Θ and Θ' respectively. Define b by $b(\Theta) = \Theta'$. The branch Θ only contains the formula $\neg@_i\varphi$ and this occurs as an induced formula on Θ', since the current block of Θ' contains both i and $\neg\varphi$. Hence Condition 1 above holds. Condition 2 holds trivially. This concludes the base case. This is the only place in the translation where we use the Name rule.

Induction step. Assume $T_{\mathtt{LC}}$, $T_{\mathtt{ST}}$ and b are given that satisfy the conditions of the lemma, including Conditions 1 and 2. We need to prove that if $T_{\mathtt{LC}}$ is extended into $T'_{\mathtt{LC}}$ by a single rule application, we can construct a similar extension $T'_{\mathtt{ST}}$ of $T_{\mathtt{ST}}$ and a new bijection b' so that Conditions 1 and 2 again hold. We prove this by examining each possible case of a rule application building $T'_{\mathtt{LC}}$ from $T_{\mathtt{LC}}$.

Case ($\wedge$). Suppose $T'_{\mathtt{LC}}$ is obtained from $T_{\mathtt{LC}}$ by an application of ($\wedge$) to a premise $@_j(\psi_1 \wedge \psi_2)$ on a branch $\Theta_{\mathtt{LC}}$ of $T_{\mathtt{LC}}$. In $T'_{\mathtt{LC}}$ the branch $\Theta_{\mathtt{LC}}$ has become extended by formulas $@_j\psi_1$ and $@_j\psi_2$. Call the extended branch $\Theta'_{\mathtt{LC}}$. By the induction hypothesis, $b(\Theta_{\mathtt{LC}})$ contains a block B with $j, \psi_1 \wedge \psi_2 \in B$ (since $@_j(\psi_1 \wedge \psi_2)$ occurs as an induced formula on $b(\Theta_{\mathtt{LC}})$ by Condition 1). Note that this block need not be the current one. We can now extend $b(\Theta_{\mathtt{LC}})$ as shown in Figure 3, using GoTo, then Nom, then ($\wedge$). Call this extended branch $\Theta'_{\mathtt{ST}}$, and let $T'_{\mathtt{ST}}$ denote the tableau in which $b(\Theta_{\mathtt{LC}})$ has been extended into $\Theta'_{\mathtt{ST}}$. Now define $b' = (b - \{(\Theta_{\mathtt{LC}}, b(\Theta_{\mathtt{LC}}))\}) \cup \{(\Theta'_{\mathtt{LC}}, \Theta'_{\mathtt{ST}})\}$, and note that b' is a bijection from the branches of $T'_{\mathtt{LC}}$ onto the branches of $T'_{\mathtt{ST}}$. It now follows immediately from the induction hypothesis and the construction of the extended tableaus, that Conditions 1 and 2 still hold when $T_{\mathtt{LC}}$ is replaced by $T'_{\mathtt{LC}}$ and b by b'.

The rest of the cases are similar and are left to the reader. Condition 2 is only used in the case for the rule ($\Diamond$), where we need to show that the same nominal is fresh on $T_{\mathtt{ST}}$ as the one chosen when applying ($\Diamond$) on $T_{\mathtt{LC}}$. □

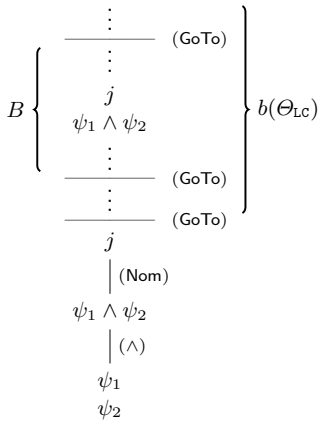


Fig. 3. Case (∧): The extended branch $\Theta'_{\mathtt{ST}}$ of $T'_{\mathtt{ST}}$.

Theorem 2 (Completeness). *If the formula φ is valid, then there exists a closed tableau in* ST *having $\neg\varphi$ as the root formula.*

Proof. Assume φ is valid. As LC is complete [5], there exists a closed LC-tableau $T_{\mathtt{LC}}$ with root $\neg@_i\varphi$, where i is a nominal not occurring in φ. By Lemma 1 there is an ST-tableau $T_{\mathtt{ST}}$ with root $\neg\varphi$ and a bijection $b : \{\Theta \mid \Theta \text{ is a branch of } T_{\mathtt{LC}}\} \to \{\Theta' \mid \Theta' \text{ is a branch of } T_{\mathtt{ST}}\}$ such that:

1. Given any branch Θ of $T_{\mathtt{LC}}$, all formulas $@_j\psi \in \Theta$ and $\neg@_j\psi \in \Theta$ occur as induced formulas on $b(\Theta)$.
2. All nominals that occur on $b(\Theta)$ also occur on Θ.

We now prove that $T_{\mathtt{ST}}$ can be extended into a closed tableau. To this end, let $\Theta_{\mathtt{ST}}$ denote an arbitrary branch on $T_{\mathtt{ST}}$. By definition, $b^{-1}(\Theta_{\mathtt{ST}})$ is a closed branch, meaning that it contains a pair of formulas $@_j\psi$ and $\neg@_j\psi$. Condition 1 implies that these formulas occur induced on $\Theta_{\mathtt{ST}}$. Thus $\Theta_{\mathtt{ST}}$ contains a pair of blocks B_1, B_2 with $j, \psi \in B_1$ and $j, \neg\psi \in B_2$. We can now extend $\Theta_{\mathtt{ST}}$ by applying GoTo once to open a new block containing j, and afterwards applying (Nom) twice to get ψ and $\neg\psi$ in the current block. The extended branch is obviously closed, as the current block will then contain a contradiction. Since $\Theta_{\mathtt{ST}}$ was an arbitrary branch of $T_{\mathtt{ST}}$, this means that every branch of $T_{\mathtt{ST}}$ can be extended to a closing branch, so we can close the entire tableau. □

4 Ongoing work

In this section we briefly discuss ongoing work on termination and extensions to stronger logics. First, we ask whether the system just defined provides a

decision procedure for basic hybrid logic. Second, we note that our system can be extended to a complete system for full first-order hybrid logic.

First, can the tableau system just introduced be used as a decision procedure? It is not difficult to see that unrestricted use of the calculus as presented here can lead to non-terminating computations; indeed, repeated applications of GoTo are a trivial way of doing this. Still, our initial investigations suggest that by imposing natural restrictions on the application of rules, we can get a terminating calculus *without* resorting to loop checks (the first loop check free tableau calculus for hybrid logic was provided in [10]). The first step towards a terminating calculus is to adopt the standard rule application restrictions for tableau calculi to our block-based setting. Usually, the following restrictions are imposed (see e.g. [10]):

($R1$) A rule is never applied twice to the same set of premises on the same branch.
($R2$) A formula is never added to a branch where it already occurs.

The adaptation of these to our block-based setting becomes:

($R1'$) A rule is never applied to a pair of premises φ, ψ at the current block B if, for some nominal $i \in B$, there is a block B' with $i, \varphi, \psi \in B'$ at which the rule instance has already been applied.[7]
($R2'$) A formula is never added to a block where it already occurs.

The only remaining way a branch can be infinite is if it contains infinitely many blocks initialised with the same nominal (the initialising nominal is the one just below the horisontal line). To avoid this kind of non-termination we need a third restriction ($R3'$), explicitly limiting the applicability of the GoTo rule. We are currently working on a termination proof based on these ideas.

A second line of work is extending the system to richer logics. One benefit of the Seligman-style approach is its modularity. So, in principle, it should be relatively easy to obtain complete proof system for richer hybrid logics by adding standard rules for the additional connectives involved. This turns out to be the case for hybrid logics equipped with the tense operators F and P, with the universal modality A, and with the $\downarrow$-binder (see [8] for background information). Indeed, we have also obtained a complete system for full first-order hybrid logic; we shall briefly sketch the main ideas involved.

First-order hybrid logic is what you obtain when when you build hybrid logic over first-order logic instead of over propositional logic. There are a number of syntactic and semantic choices to be made about how to do this; for discussions of the various possibilities see [17, 15]. We adopted the choices made in [7]. The syntax of our language is:

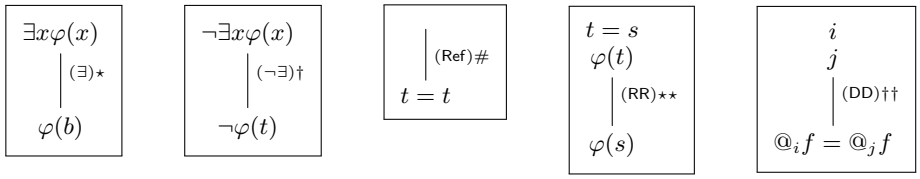
$$\varphi ::= i \mid t = s \mid P(t_1, \ldots, t_n) \mid \neg\varphi \mid \varphi \wedge \psi \mid \Diamond\varphi \mid @_i\varphi \mid \exists x\varphi.$$

Here s, t and $t_1, \ldots, t_n$ are first-order terms. These symbols range over ordinary first-order constants and variables, and also over composite first-order terms of

[7] For rules taking a single premise, we let $\psi = \varphi$.

the form $@_i f$. For example, if f is a symbol standing for the "The President of the United States" (such a symbol is called a *non-rigid designator*) then $@_{2013} f$ is a (composite) first-order term denoting President Obama, and $@_{1962} f$ a term denoting President Kennedy. But apart from this, all is standard. Following [7], we assume a *constant domain* semantics.

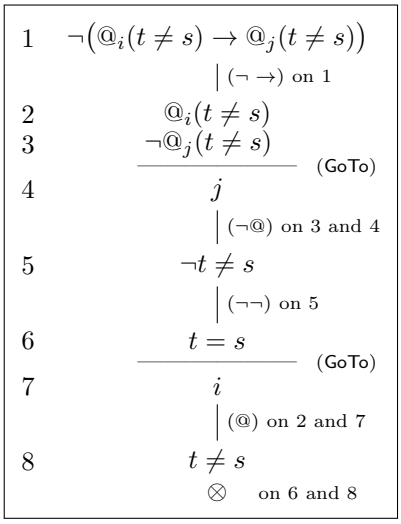
Now, given that the underlying first-order language (modulo the use of composite terms $@_i f$) is standard, we should expect (if the claims about modularity are correct) to obtain a complete proof system by bolting on a collection of standard first-order logic with equality rules, together with a rule for handling composite terms. And this is exactly what happens. All we need to do is add the following third module:

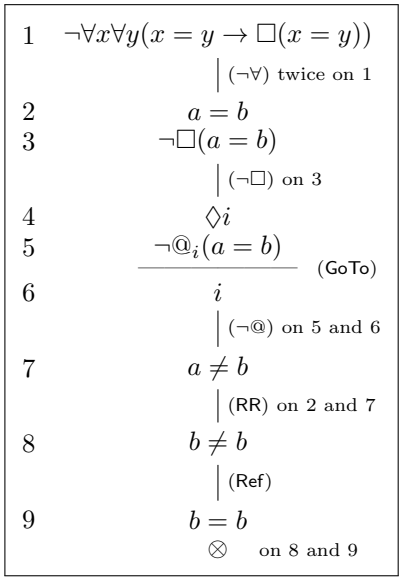


$\star$ $\exists x\varphi(x)$ occurs in the current block and the parameter b is fresh on the branch.
† $\neg\exists x\varphi(x)$ occurs in the current block and the term t occurs on the branch.
\# The term t occurs on the branch.
$\star\star$ $\varphi(t)$ occurs in the current block and $t = s$ is on the branch.
†† The nominals i and j occur together in some block (not necessarily the current one) on the branch and f is a non-rigid symbol occurring on the branch.

Fig. 4. Module 3: First-order tableau rules.

All the rules, except the one at the far right should be familiar. And the rule on the far right clearly captures the way our composite terms work: if i and j name the same world (note that the premises are in the same block) then we can safely conclude that $@_i f$ and $@_j f$ both denote the same first-order entity. Completeness can be proved by translation from the labeled system of [7], much as we did in the propositional case, though the proof is longer and more subtle. So instead of giving a proof sketch, we will simply give two example of the system at work. The first shows that if an inequality is true in some world then it is true in any other world; that is, that $@_i(t \neq s) \rightarrow @_j(t \neq s)$ is derivable. The second shows that "equalities are forever".





5 Concluding Remarks

In his most detailed exposition of his approach, Jerry Seligman [25] states his aim clearly: to obtain "a more egalitarian logic in which there are Rules for All" ([25], page 684). The sequent calculus presented there was the starting point for our work, so to conclude this paper we would like to indicate the main similarities and differences.

As should be clear by now, the crucial rules are those that handle the nominals and the @-operator. Seligman uses the following six rules for this purpose; he calls them Nominal Rules (see [25], page 685):

$$
\begin{array}{lllll}
{}^{\vee}@L & i, \varphi, \Gamma \longrightarrow \Delta & \Rightarrow & i, @_i\varphi, \Gamma \longrightarrow \Delta \\
{}^{\vee}@R & i, \Gamma \longrightarrow \Delta, \varphi & \Rightarrow & i, \Gamma \longrightarrow \Delta, @_i\varphi \\
{}^{\wedge}@L & i, @_i\varphi, \Gamma \longrightarrow \Delta & \Rightarrow & i, \varphi, \Gamma \longrightarrow \Delta \\
{}^{\wedge}@R & i, \Gamma \longrightarrow \Delta, @_i\varphi & \Rightarrow & i, \Gamma \longrightarrow \Delta, \varphi \\
\mathsf{name} & i, \Gamma \longrightarrow \Delta & \Rightarrow & \Gamma \longrightarrow \Delta, \text{ if } i \text{ does not occur in } \Gamma, \Delta \\
\mathsf{term} & i, \Gamma \longrightarrow \Delta & \Rightarrow & \Gamma \longrightarrow \Delta, \text{ if all formulas in } \Gamma, \Delta \text{ are @-prefixed.}
\end{array}
$$

First the easy part. Tableau rules can often be seen as reversed sequent rules, where the formulas on right of the sequent arrow $\longrightarrow$ are negated. If we read the listed rules this way, our (@) and (¬@) rules are simply his ${}^{\vee}@L$ and ${}^{\vee}@R$ rules, and our Name rule is just Seligman's name.

The divergences stem from the remaining three rules. Our first attempt at a tableau system contained the obvious tableau correlates of Seligman's ${}^{\wedge}@L$ and ${}^{\wedge}@R$ rules. The rules introduced @-prefixes and with these rules we were able

to @-prefix whole branches, thereby globalizing the information they contained. Such rules are destructive: they don't simply expand branches, they change them more drastically.

Why did we do this? To try and *directly* capture Seligman's term rule. His term rule is essentially our GoTo rule, but note the side condition: it only lets us jump to a world u if all information is @-prefixed, that is, global. Our first version of GoTo had the same side condition, so proofs in our early systems would typically contain multiple applications of the $^\wedge@L$ and $^\wedge@R$ rules followed by an application of GoTo. But we were dissatisfied notationally; destructive rules are annoying when using a tableau system by hand. Then we noticed a more serious problem: the @-prefixing permitted by the $^\wedge@L$ and $^\wedge@R$ rules interacted badly with (our tableau versions of) the rules $^\vee@L$ and $^\vee@R$. Often we would prefix an @, only to immediately strip it off, a clear proof redundancy.

These interrelated issues led us to introduce blocks. In essence, by making use of blocks, we avoid having to give explicit tableau rules corresponding to rules $^\wedge@L$ and $^\wedge@R$; these rules are absorbed into the concept of a block. This simultaneously eliminates the destructive tableau-rules, and bypasses the proof redundancy just noticed. Moreover, by having GoTo create a local proof context (rather than only be applicable when all the information on the branch has been @-prefixed) we avoid having to impose the side condition.

The drawbacks we discovered in our early tableau systems are *not* present in Seligman's sequent calculus; Seligman's rules and side-conditions elegantly exploit the resources of sequent calculus. But (despite its use of blocks) we believe our system comes close to being a "natural tableau reversal" of Seligman's system. Compare, for example, the sequent derivation of $@_ij \wedge @_jk \rightarrow @_ik$ with the block derivation given earlier:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{i, j, k \longrightarrow k}{i, j, @_jk \longrightarrow k}\,{}^\vee@\mathrm{L}}{i, @_ij, @_jk \longrightarrow k}\,{}^\vee@\mathrm{L}}{i, @_ij, @_jk \longrightarrow @_ik}\,{}^\vee@\mathrm{R}}{@_ij, @_jk \longrightarrow @_ik}\,\mathsf{term}}{@_ij \wedge @_jk \longrightarrow @_ik}\,(\wedge\mathrm{R})$$

This example also illustrates that the term rule in the sequent system is really more of a GoFrom rule than a GoTo rule. Of course, this reflects the fact that tableau rules are, in a sense, reversed sequent rules.

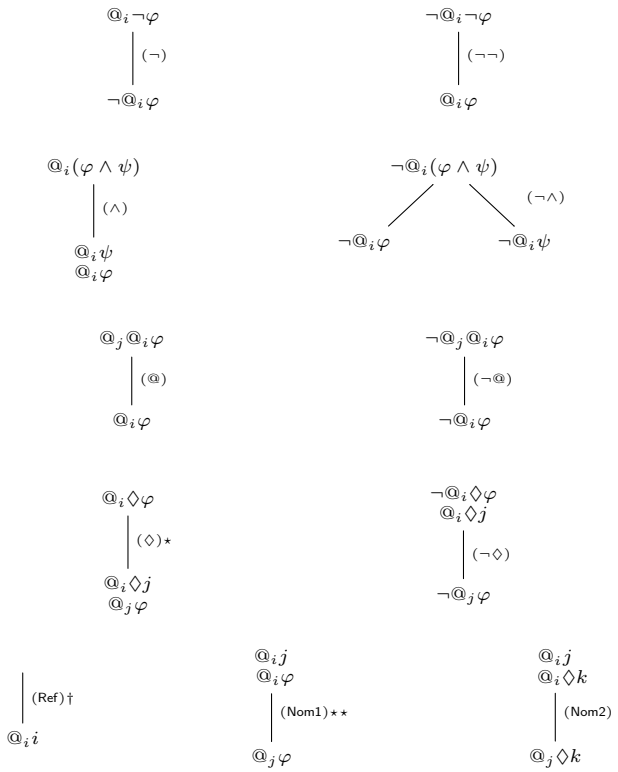
Finally, we remark that the use of blocks reverses a longstanding trend in hybrid logic (reliance on the labeling apparatus in the object language) in favor of imposing more structure at the metalevel. Dividing branches into blocks *externalizes* (passes up to the metalanguage) some of the work done by the @-operator. The use of induced satisfaction statements in our completeness proofs (which reflects the way that Seligman's $^\wedge@L$ and $^\wedge@R$ are absorbed into the concept of a block) is a clear reflection of this externalization.

References

1. Areces, C., Heguiabehere, J.: Direct Resolution for Modal-like Logics. In: Proceedings of the 3rd International Workshop on the Implementation of Logics. pp. 3–16. Tbilisi, Georgia (2002)
2. Areces, C., Gorín, D.: Ordered Resolution with Selection for H(@). In: LPAR. pp. 125–141 (2004)
3. Areces, C., Gorín, D.: Resolution with Order and Selection for Hybrid Logics. J. Autom. Reasoning 46(1), 1–42 (2011)
4. Bierman, G., de Paiva, V.: On an Intuitionistic Modal Logic. Studia Logica 65, 383–416 (2000)
5. Blackburn, P.: Internalizing labelled deduction. Journal of Logic and Computation 10(1), 137–168 (2000)
6. Blackburn, P., Jørgensen, K.F.: Indexical Hybrid Tense Logic. In: Bolander, T., Braüner, T., Ghilardi, S., Moss, L. (eds.) Advances in Modal Logic. vol. 9, pp. 144–60 (2012)
7. Blackburn, P., Marx, M.: Tableaux for quantified hybrid logic. In: Egly, U., Fernmüller, C. (eds.) Automated Reasoning with Analytic Tableaux and Related Methods, LNAI, vol. 2381, pp. 38–52. Springer (2002)
8. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Cambridge University Press, Cambridge (2001)
9. Blackburn, P., Tzakova, M.: Hybrid Languages and Temporal Logic. Logic Journal of the IGPL 7(1), 27–54 (1999)
10. Bolander, T., Blackburn, P.: Termination for Hybrid Tableaus. Journal of Logic and Computation 17(3), 517–554 (2007)
11. Bolander, T., Braüner, T.: Tableau-Based Decision Procedures for Hybrid Logic. Journal of Logic and Computation 16, 737–63 (2006)
12. Braüner, T.: Two natural deduction systems for hybrid logic: A comparison. Journal of Logic, Language and Information 13, 1–23 (2004)
13. Braüner, T.: Hybrid Logic and its Proof-Theory, Applied Logic Series, vol. 37. Springer (2011)
14. Braüner, T.: Hybrid-logical Reasoning in False-Belief Tasks. In: Schipper, B. (ed.) Proceedings of Fourteenth Conference on Theoretical Aspects of Rationality and Knowledge (TARK). pp. 186–195 (2013), available at http://tark.org
15. Braüner, T., Ghilardi, S.: First-order modal logic. In: Handbook of Modal Logic, pp. 549–620. Elsevier (2007)
16. Chadha, R., Macedonio, D., Sassone, V.: A hybrid intuitionistic logic: Semantics and decidability. Journal of Logic and Computation 16, 27–59 (2006)
17. Fitting, M., Mendelsohn, R.: First-Order Modal Logic. Springer (1998)
18. Galmiche, D., Salhi, Y.: Sequent calculi and decidability for intuitionistic hybrid logic. Information and Computation 209, 1447–1463 (2011)
19. Götzmann, D., Kaminski, M., Smolka, G.: Spartacus: A Tableau Prover for Hybrid Logic. Electr. Notes Theor. Comput. Sci. 262, 127–139 (2010)
20. Hoffmann, G., Areces, C.: HTab: A Terminating Tableaux System for Hybrid Logic. In: Proceedings of Methods for Modalities 5 (November 2007)
21. Hoffmann, G.: Tâches de raisonnement en logiques hybrides. Ph.D. thesis, Université Henri Poincaré - Nancy I (Dec 2010), http://tel.archives-ouvertes.fr/tel-00541664
22. Jia, L., Walker, D.: Modal proofs as distributed programs (extended abstract). In: Schmidt, D. (ed.) European Symposium on Programming. Lecture Notes in Computer Science, vol. 2986, pp. 219–233. Springer-Verlag (2004)

23. Kushida, H., Okada, M.: A Proof-Theoretic Study of the Correspondence of Hybrid Logic and Classical Logic. Journal of Logic, Language and Information 16, 35–61 (2007)
24. Seligman, J.: The Logic of Correct Description. In: de Rijke, M. (ed.) Advances in Intensional Logic, Applied Logic Series, vol. 7, pp. 107 – 135. Kluwer (1997)
25. Seligman, J.: Internalisation: The Case of Hybrid Logics. Journal of Logic and Computation 11, 671–689 (2001), special Issue on Hybrid Logics. C. Areces and P. Blackburn (eds.)
26. Tzakova, M.: Tableau Calculi for Hybrid Logics. In: Murray, N. (ed.) Conference on Tableaux Calculi and Related Methods, LNAI, vol. 1617, pp. 278–92. Springer (1999)

Appendix: Labeled Tableau Rules for LC



$\star$ The nominal j is new and φ is not a nominal.
$\dagger$ The nominal i is on the branch.
$\star\star$ The formula φ is an atomic formula, i.e., ordinary propositional symbol or nominal.