

# Completeness and Termination for a Seligman-style Tableau System

**Blackburn, Patrick Rowan; Bolander, Thomas; Braüner, Torben; Jørgensen, Klaus Frovin**

*Published in:*
Journal of Logic and Computation

*Link to article, DOI:*
10.1093/logcom/exv052

*Publication date:*
2015

*Document Version*
Peer reviewed version

# Completeness and Termination for a Seligman-style Tableau System

Patrick Blackburn* Thomas Bolander† Torben Braüner‡
Klaus Frovin Jørgensen§

**Abstract**

Proof systems for hybrid logic typically use @-operators to access information hidden behind modalities; this labelling approach lies at the heart of the best known hybrid resolution, natural deduction, and tableau systems. But there is another 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 natural deduction [22] and sequent calculus [23] in the 1990s. The purpose of this paper is to introduce a Seligman-style tableau system, to prove its completeness, and to show how it can be made to terminate.

The most obvious feature of Seligman-style systems is that they work with arbitrary formulas, not just statements prefixed by @-operators. They do so by introducing machinery for switching to other proof contexts. We capture this idea in the setting of tableaus by introducing a rule called GoTo which allows us to "jump to a named world" on a tableau branch. We first develop a Seligman-style tableau system for basic hybrid logic and prove its completeness. We then prove termination of a restricted version of the system without resorting to loop checking, and show that the restrictions do not effect completeness. Both completeness and termination results are proved constructively: we give translations which transform tableaus in a standard labelled system into tableaus in our Seligman-system and vice-versa.

**Keywords:** Tableau systems, hybrid logic, completeness, termination, Jerry Seligman.

## 1 Introduction

Hybrid logic is an extension of ordinary modal logic in which worlds (or epistemic states, or times, or whatever is required for the application at hand) can be named. Special propositional symbols called *nominals* are added to the underlying modal language. These symbols are true at exactly one world, and a nominal $i$ 'names' the unique world it is true at. In addition, a collection of modal operators of the form $@_i$ is added. These have the obvious interpretation: $@_i\varphi$ is true at any world iff $\varphi$ is true

*Department of Philosophy and Science Studies, Roskilde University. E-mail: patrickb@ruc.dk.
†Department of Applied Mathematics and Computer Science, Technical University of Denmark. E-mail: tobo@dtu.dk.
‡Department of Communication, Business and Information Technologies, Roskilde University. E-mail: torben@ruc.dk.
§Department of Philosophy and Science Studies, Roskilde University. E-mail: frovin@ruc.dk.

at the (unique) world named by the nominal $i$. Expressions of this form are called *satisfaction statements*.

It is relatively straightforward to define proof systems for hybrid logic in a wide range of reasoning styles, including tableau [4, 7, 10, 11, 25], natural deduction [13], and resolution [1, 2]. There is a resolution theorem prover (the HyLoRes system [3]) and at least two high-performance tableau provers (namely HTab [19, 20] and Spartacus [18]). Indeed, even hybrid Hilbert system are well-behaved [8, 9]. Moreover, proof systems in different styles exist for intuitionistic hybrid logic too; see [13, 16, 17] for options.

But behind this apparent diversity lies a common strategy, *labelling*. Of course, labelled deduction methods are used for a wide range of non-classical logics, but the link between labelling and hybrid logic is particularly intimate—nominals and satisfaction operators are essentially labelling 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) labelled deduction. But this is not so. Another approach, which we call *Seligman-style inference*, offers an interesting alternative. The purpose of this paper is to explore Seligman-style inference in the setting of tableau-based reasoning, and to investigate its relationship with the labelling approach.

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:[1]

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

Here 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 $\varphi$ holds at $j$ (which is what the satisfaction statement $@_j\varphi$ says). We assume nothing else about $j$ beyond this: in effect we have said "let $j$ be an arbitrary world accessible from $i$ at which $\varphi$ is true". Now, if from these two assumptions we can prove some formula $@_k\psi$ (which says that $\psi$ 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 $\varphi$ holds at some accessible world) then we get a proof of $@_k\psi$.

This is a sound rule of proof, but note its *form*. In particular, note that *all the formulas used in this rule are satisfaction statements*. Now, satisfaction statements (and their negations) are *global*. This is easy to see. If $\varphi$ is indeed true at the world named $i$, then $@_i\varphi$ is true at *all* worlds. On the other hand, if $\varphi$ is false at the world named $i$, then $@_i\varphi$ is false at *all* worlds. Thus satisfaction statements (and their negations) embody global information. And this means that the labelled 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

[1] For this rule to be correctly applied, $j$ has to be a fresh nominal, that is, $j$ has to be different from both $i$ and $k$, and $j$ must not occur in either $\varphi$ or $\psi$ or in any undischarged assumptions of the proof other than those specified. The formulas $@_i\Diamond j$ and $@_j\varphi$ occurring in the sub-proof on the right are discharged in the application of the rule, which is indicated by putting brackets $[\ldots]$ around the formulas. For more on natural deduction in hybrid logic, see Braüner [13].

the $\Diamond$-elimination rule would look like this:[2]

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

Notice the *local* perspective illustrated by the rule. 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 $\varphi$ holds. Now, if it is possible for us, given this information, to prove some formula $\psi$ (in which $j$ doesn't occur), then we actually have a proof of $\psi$ given a proof of $\Diamond\varphi$. The core of the argument is similar to that used in the labelled 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 labelled 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. 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 may be relying too much on 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, as this is probably the most widely used proof style in hybrid logic.

Little has been written on Seligman-style systems. They were introduced in two papers, both by Jerry Seligman, namely the natural deduction based [22] and the Gentzen sequent calculus based [23].[3] The first of these gives 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 reasoning there, followed by a journey back again (we'll look at this rule shortly; 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, by using what are known as explicit substitutions.

The authors of this paper became interested in Seligman-style reasoning because of reasoning problems involving perspective shifts and contextual information. First, Braüner [14] (later journal version [15]) has used his Seligman-style natural deduction system to formalize two well-known false-belief task in cognitive psychology, the

[2]The side-condition for this rule is that $j$ must not occur in $\varphi$, nor $\psi$ or in any undischarged assumptions other than those specified.

[3]Another sequent system for hybrid logic that allows arbitrary formulas to occur in derivations can be found in the paper [21]. This system makes use of standard sequent machinery for the ordinary (non-hybrid) modal logic K, which makes it quite different from the Seligman-style system of [23].

Smarties task and the Sally-Anne task (see [24]). To solve such tasks, the subject 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. Also recently, Blackburn and Jørgensen [6] used hybrid logic to investigate *temporal indexicals*, context-sensitive terms such as *now*, *yesterday*, *today*, and *tomorrow*. In the course of this work it became clear that a Seligman-style tableau approach might allow a simpler presentation of the reasoning involved, but no such calculus existed. The system described here arose as an attempt to fill this gap.

We took Seligman's sequent system [23] as our starting point. We experimented with a variety of options, some very different from the system ST presented below, before we settled on the key idea: dividing tableau branches into *blocks* which record partial information about worlds. In the course of these experiments, three things became clear. First, the new systems were flexible and natural to work with. Second, precisely because of their flexibility, obtaining a terminating system would not be a trivial task. Third, understanding in detail how the system ST related to standard labelled tableau systems was likely to be both important and enlightening. Hence the technical core of this paper is a detailed specification of how to transform tableau proofs in our Seligman system ST into tableau proofs in a standard labelling system (called LC) and vice-versa.

We proceed as follows. In Section 2 we motivate and define the Seligman-style tableau system ST. In Section 3 we prove its completeness by showing how to transform LC proofs into ST proofs. In Section 4 we briefly compare ST with the sequent calculus that inspired it, before turning, in Section 5, to the issue of termination. For a number of more-or-less trivial reasons it is immediate that ST does not terminate, but there are two deeper reasons as well. We analyze these in detail, and define a restricted system which we call ST$^\star$. We then prove (again using a proof transformation argument) that ST$^\star$ terminates, and that the restrictions we have imposed do not effect completeness. Section 6 concludes with a discussion of ongoing and future work.[4]

## 2 The Seligman-style Tableau Calculus ST

We work with a basic hybrid language built over a countable set of propositional symbols, a countable set of nominals, the propositional connectives $\neg$ and $\wedge$, the modal diamond $\Diamond$, and for each nominal $i$ an $@_i$-operator. Formulas are built as follows (with $i$ ranging over nominals and $p$ over propositional symbols):

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

Other boolean connectives are defined as usual, and the dual box-form $\Box\varphi$ is defined to be $\neg\Diamond\neg\varphi$. Note that nominals can occur either as subscripts to @ ("in operator position") or as formulas in their own right ("in formula position"). We typically use $i, j$ and $k$ for nominals and $p, q$ and $r$ for ordinary propositional symbols.

We interpret the language of basic hybrid logic in models based on frames $(W, R)$. Here $W$ is a non-empty set (we call its elements worlds) and $R$ is a binary relation on this set (the accessibility relation). To fully specify a model, we also need an information distribution, together with a specification of names for worlds of interest.

[4] This paper is a continuation of work initiated in [5]. In the present paper we have refined the ST system and prove termination of it.

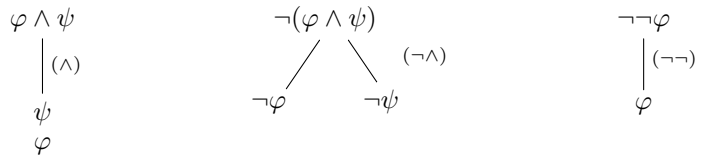



Figure 1: Tableau rules for propositional logic.

Both tasks are performed by a valuation function $V$, which takes ordinary propositional symbols and nominals to subsets of $W$ satisfying the following two conditions:

1. $V(p)$ is an arbitrary 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 between a model $\mathfrak{M} = (W, R, V)$, a world $w \in W$, and a formula $\varphi$:

$$\begin{array}{lll}
\mathfrak{M}, w \models \bot & \text{iff} & \text{falsum} \\
\mathfrak{M}, w \models a & \text{iff} & a \text{ is atomic and } w \in V(a) \\
\mathfrak{M}, w \models \neg\varphi & \text{iff} & \mathfrak{M}, w \not\models \varphi \\
\mathfrak{M}, w \models \varphi \wedge \psi & \text{iff} & \mathfrak{M}, w \models \varphi \text{ and } \mathfrak{M}, w \models \psi \\
\mathfrak{M}, w \models \Diamond\varphi & \text{iff} & \text{for some } w',\ wRw' \text{ and } \mathfrak{M}, w' \models \varphi \\
\mathfrak{M}, w \models @_i\varphi & \text{iff} & \mathfrak{M}, w' \models \varphi \text{ and } w' \in V(i).
\end{array}$$

A formula $\varphi$ 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. An appealing aspect of Seligman systems is their modularity, so we simply use standard tableau rules for propositional logic in our system. The rules we have chosen are shown in Figure 1.

Now for the hybrid logical rules. We want to escape the global form of the labelled rules that are typically used in hybrid tableau systems,[5] but how can we do so? To find an answer, let us consider another Seligman-style natural deduction rule, the Term rule. A central idea of natural deduction is to allow *hypothetical* reasoning. The Term rule adds a hybrid-logical dimension to this: it allows us to perform hypothetical reasoning at a world named by some nominal $i$. Or to put it another way: it allows us switch our perspective to another world, and carry out hypothetical reasoning there:

$$\dfrac{\varphi_1 \quad \cdots \quad \varphi_n \qquad \begin{array}{c}[\varphi_1] \ldots [\varphi_n][i] \\ \vdots \\ \psi\end{array}}{\psi}\,(Term)$$

The rule has a side condition: the formulas $\varphi_1, \ldots, \varphi_n$, and $\psi$ are all @-prefixed, and there must be no undischarged assumptions in the derivation of $\psi$ besides the

[5]The rules are given in Figure 4.

| | | |
|---|---|---|
| 1 | $\Diamond @_i\varphi \wedge \neg @_i\varphi$ | root formula |
| 2 | $\Diamond @_i\varphi$ | ($\wedge$) on 1 |
| 3 | $\neg @_i\varphi$ | ($\wedge$) on 1 |
| 4 | $\Diamond j$ | ($\Diamond$) on 2 |
| 5 | $@_j @_i\varphi$ | ($\Diamond$) on 2 |
| 6 | $j$ | GoTo |
| 7 | $@_i\varphi$ | (@) on 5, 6 |
| 8 | $i$ | GoTo |
| 9 | $\neg\varphi$ | ($\neg$@) on 3, 8 |
| 10 | $\varphi$ | (@) on 7, 8 |
| | $\times$ | |

Figure 2: A tableau in the ST calculus.

specified occurrences of $\varphi_1, \ldots, \varphi_n$, and $i$. This side-condition ensures that the truth-values of the assumptions are unaffected when the evaluation world is shifted (see [13] for more information).

A similar perspective-shifting intuition underlies our Seligman-style tableau system. A central concept of tableau reasoning is branch expansion. We are going to add a hybrid-logical dimension to this and allow ourselves to perform branch expansion at a world named by some nominal $i$. We do so by dividing branches into *blocks*, with blocks separated by horizontal lines. A block on a branch is a partial description of the information in a specific world, and our tableau system is driven by a rule called GoTo which lets us switch between blocks.

Let's see an example. Figure 2 is a tableau showing that $\Diamond @_i\varphi \wedge \neg @_i\varphi$ is unsatisfiable. The tableau has three blocks: lines 1-5 are a block, as are lines 6-7 and 8-10. In the first block, we eliminate the $\Diamond$ in line 2 which gives us $\Diamond j$ and $@_j @_i\varphi$. Now our perspective-shifting intuition comes into play: to use this derived information, we should shift to world $j$. We do so at line 5 by applying GoTo, the core rule of our system. At line 7 we use our new $j$-perspective to retrieve $@_i\varphi$ from line 5. We then realize that if we shift perspective to $i$ we can extract a contradiction, so we apply GoTo again, and the tableau closes because of line 9 and 10. The division of branches into blocks, and the ability GoTo gives us to shift our attention between them, are at the very centre of our approach to Seligman-style tableau reasoning.

Before leaving this example, another remark. Both blocks opened by GoTo start with a nominal: the second block begins with $j$, and the third with $i$. Moreover, note that the initial block is not named in this way. Both points are general: in our tableau system, blocks introduced by GoTo will always open with a nominal. However, although we can always name the initial block with some nominal, we don't always have to, and sometimes we don't want to. For example, if our input formula contains no modalities or @-operators, it would be pointless to start naming worlds: the propositional rules given in Figure 1 are capable of handling such formulas in the familiar fashion.

Bearing this motivating example in mind, let us be more precise about blocks, and how they are created. We work with the usual notion of branch in a (tableau) tree. Given a branch $\Theta$ in a tableau, we define a *block* to be one of the following:

- The *initial block*, consisting of all the formulas on $\Theta$ until the first horizontal line (or all formulas if there is no such line).
- The *current block*, consisting of all formulas below the last horizontal line (or all

formulas if there is no such line).

- All formulas that occur between a pair of two consecutive horizontal lines.

The rule allowing us to close down one block and start up a new one is GoTo. Its precise formulation is given in Figure 3. All blocks except the initial one are opened by an application of GoTo, and hence they all contain a nominal as their first formula. This nominal is called the *opening nominal* of the block. In the tableau of Figure 2, the opening nominal of the block in lines 6–7 is $j$, and the opening nominal of the block in lines 8–10 is $i$. If the initial block contains an application of the Name rule (see Figure 3), then the nominal introduced by this rule will be called the *opening nominal* of the initial block. Otherwise, the initial block will have no opening nominal. The initial block of the tableau in Figure 2 has no opening nominal. A block with opening nominal $i$ is called an *$i$-block*.

The crucial rules of the Seligman-style tableau calculus are given in Figure 3. The general 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 or a previous block with the same opening nominal.

- In the rules (@) and ($\neg$@), the first premise $i$ must either belong to the current block or a previous block with the same opening nominal. The second premise $@_i\varphi$ ($\neg@_i\varphi$) can appear anywhere on the branch.

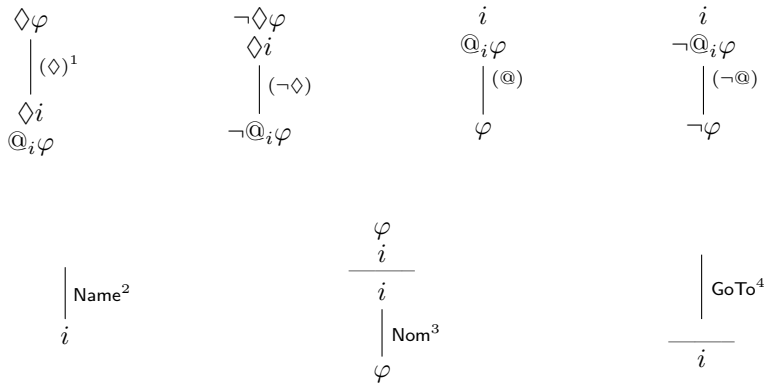
- GoTo and Name can always be applied as they have no premises.

- Nom can be applied as described in the rule itself: if $\varphi$ and $i$ belong to some block distinct from the current block, and $i$ belongs to the current block, then $\varphi$ can be added to the current block.

It should be clear that the first four rules are simply the obvious (positive and negative) rules for $\Diamond$ 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 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 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 particular world named by (say) the nominal $i$. We sometimes need to combine this information, and Nom lets us do this. Basically, it says that if $i$ and $\varphi$ occur together in some block, then, if you later find yourself at some block that also contains $i$, you are free to recall that $\varphi$ 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 3. We call this system ST (short for Seligman-style Tableau). Tableaus are built in the expected way, but we should be explicit about our closure condition:




[1] The nominal $i$ is fresh and $\varphi$ is not a nominal.
[2] The nominal $i$ is fresh.
[3] 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.
[4] The nominal $i$ must already occur on the branch.

Figure 3: ST tableau rules for basic hybrid logic.

> *A branch closes either by having $\varphi$ and $\neg\varphi$ inside a block, or inside two distinct blocks with the same opening nominal.*

It is instructive to compare proofs in our Seligman-style calculus ST with the proofs in the labelled calculus. The rules of the labelled calculus LC are found in Figure 4. These come from Bolander and Blackburn [10] and are going to be used in both the completeness proof and the termination proof of ST. The side condition on the Id rule makes use of the following concept. A formula $@_i\Diamond j$ in an LC tableau is an *accessibility formula* if it is the first conclusion of an application of the $(\Diamond)$ rule. All LC rule applications are subject to the following two constraints:

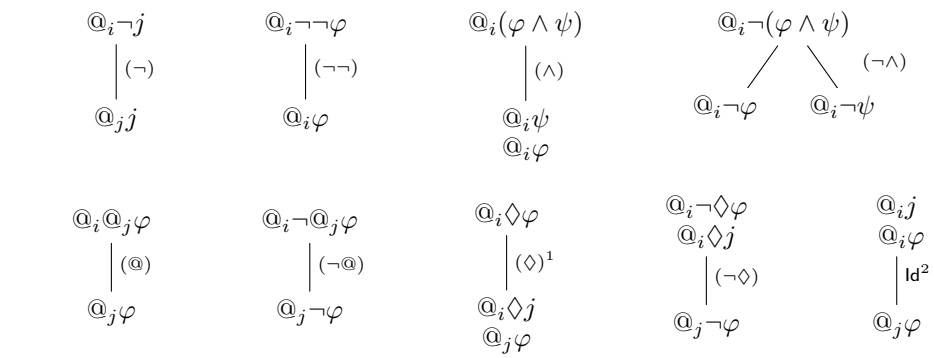
$(R1')$ *A formula is never added to a branch where it already occurs.*

$(R2')$ *The $(\Diamond)$ rule can not be applied twice to the same formula occurrence.*

With these constraints in place, the rules of Figure 4 provide a sound, complete and terminating calculus for propositional hybrid logic [10]. The rules of Figure 4 only differ from the rules of [10] in the side condition on the $(\Diamond)$ rule: here we require that $\varphi$ is not a nominal, whereas the rule of [10] requires that the premise $@_i\Diamond\varphi$ is not an accessibility formula. The rule used here is slightly more restrictive, but it is simple to prove that this doesn't affect the completeness of LC.

Consider the two different proofs given in Figure 5 of $@_ij \wedge @_jk \rightarrow @_ik$, a standard hybrid validity which says that naming with nominals is transitive.[6] This is a nice example, as it requires reasoning about the identity of worlds. The tableau on the left

[6] When giving examples, we make free use of a selection of derivable rules for the defined connectives. Here we give three rules which are particularly useful. In the $(\neg\Box)$ rule the $i$ must be a fresh




[1] The nominal $j$ is fresh and $\varphi$ is not a nominal.
[2] The formula $@_i\varphi$ is not an accessibility formula.

Figure 4: Propositional LC tableau rules.

is in the calculus ST, the one on the right in LC. The Seligman-style proof makes the form of the argument clearer, and hides the book-keeping details involving prefixing @-operators.

| | | |
|---|---|---|
| 1 | $\neg(@_i j \wedge @_j k \to @_i k)$ | |
| 2 | $@_i j$ | $(\neg\to)$, $(\wedge)$ on 1 |
| 3 | $@_j k$ | $(\neg\to)$, $(\wedge)$ on 1 |
| 4 | $\neg@_i k$ | $(\neg\to)$ on 1 |
| 5 | $i$ | GoTo |
| 6 | $j$ | $(@)$ on 2, 5 |
| 7 | $k$ | $(@)$ on 3, 6 |
| 8 | $\neg k$ | $(\neg@)$ on 4, 5 |
| | $\times$ | |

| | | |
|---|---|---|
| 1 | $@_l\neg(@_i j \wedge @_j k \to @_i k)$ | |
| 2 | $@_l@_i j$ | $(\neg\to)$, $(\wedge)$ on 1 |
| 3 | $@_l@_j k$ | $(\neg\to)$, $(\wedge)$ on 1 |
| 4 | $@_l\neg@_i k$ | $(\neg\to)$ on 1 |
| 5 | $@_i j$ | $(@)$ on 2 |
| 6 | $@_j k$ | $(@)$ on 3 |
| 7 | $@_i\neg k$ | $(\neg@)$ on 4 |
| 8 | $@_j\neg k$ | Id on 5, 7 |
| | $\times$ | |

Figure 5: Two tableaus that compare ST and LC.

# 3 Soundness and Completeness

**Theorem 3.1 (Soundness).** *If there exists a closed tableau in* ST *having* $\neg\varphi$ *as the root formula, then the formula* $\varphi$ *is valid.*

*Proof.* Let $\Theta$ be a branch of a tableau of the calculus ST. Let $B$ be a block on $\Theta$ 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 $\psi$ in $B$, it is the case that

nominal. $\neg(\varphi\to\psi)$ $(\neg\to)$ $\varphi$, $\neg\psi$ — $\neg\Box\varphi$ $(\neg\Box)$ $\Diamond i$, $\neg@_i\varphi$ — $\Box\varphi$, $\Diamond i$ $(\Box)$ $@_i\varphi$

$\mathfrak{M}, w \models \psi$. Moreover, we say that $\Theta$ is block-wise satisfiable by $\mathfrak{M}$ if and only if every block on $\Theta$ is satisfiable by $\mathfrak{M}$. We say that $\Theta$ is block-wise satisfiable if and only if $\Theta$ 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 $\mathtt{ST}$ 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 $\mathtt{ST}$. □

We now proceed to prove completeness of $\mathtt{ST}$. We do so by providing a translation from tableaus in the labelled calculus $\mathtt{LC}$ into tableaus in $\mathtt{ST}$. The translation allows us to reduce completeness of $\mathtt{ST}$ to completeness of $\mathtt{LC}$ (see [4]), and thus is a first step towards clarifying the relationship between the Seligman-style and labelling approaches. It also yields some extra information, for example that the Name rule only ever needs to be used once in any $\mathtt{ST}$ tableau construction.

Let $\Theta$ be a tableau branch of the calculus $\mathtt{ST}$. A formula $@_i\varphi$ is said to occur as an *induced formula* on $\Theta$ if there is an $i$-block $B$ on $\Theta$ such that $\varphi \in B$.

**Lemma 3.2.** *Let $\varphi$ be any formula and $i$ any nominal not in $\varphi$. Assume $T_{\mathtt{LC}}$ is a tableau with root $@_i\varphi$ in the calculus $\mathtt{LC}$. Then there exists a tableau $T_{\mathtt{ST}}$ with root $\varphi$ in the calculus $\mathtt{ST}$, and a bijection*

$$\pi : \{\Theta \mid \Theta \text{ is a branch of } T_{\mathtt{LC}}\} \to \{\Theta' \mid \Theta' \text{ is a branch of } T_{\mathtt{ST}}\}$$

*such that:*

1. *Given any branch $\Theta$ of $T_{\mathtt{LC}}$, any formula $@_j\psi \in \Theta$ with $j \neq \psi$ occurs as induced formula on $\pi(\Theta)$.*
2. *All nominals that occur on $\pi(\Theta)$ also occur on $\Theta$.*

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

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

| | | |
|---|---|---|
| 1 | $\varphi$ | |
| 2 | $i$ | Name |

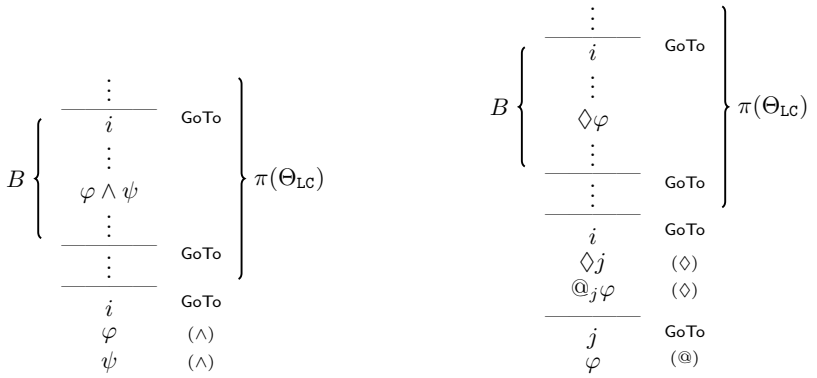
Both $T_{\mathtt{LC}}$ and $T_{\mathtt{ST}}$ have a single branch; call them $\Theta_{\mathtt{LC}}$ and $\Theta_{\mathtt{ST}}$ respectively. Define $\pi$ by $\pi(\Theta_{\mathtt{LC}}) = \Theta_{\mathtt{ST}}$. The branch $\Theta_{\mathtt{LC}}$ only contains the formula $@_i\varphi$ and this occurs as an induced formula on $\Theta_{\mathtt{ST}}$, since the current block of $\Theta_{\mathtt{ST}}$ is an $i$-block containing $\varphi$. 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 $\pi$ are given that satisfy the conditions of the lemma. This means that $\pi$ is a bijection between the branches of $T_{\mathtt{LC}}$ and $T_{\mathtt{ST}}$ satisfying the following conditions:

*ih*1. Given any branch $\Theta$ of $T_{\mathtt{LC}}$, any formula $@_j\psi \in \Theta$ with $j \neq \psi$ occurs as induced formula on $\pi(\Theta)$.

*ih*2. All nominals that occur on $\pi(\Theta)$ also occur on $\Theta$.

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 bijection $\pi'$ such that conditions 1 and 2 still hold. We do so by examining each possible case of a rule application building




(a) Case (∧). (b) Case (◇).

Figure 6: The extended branches $\Theta'_{\mathtt{ST}}$ of $T'_{\mathtt{ST}}$ referred to in the proof of Lemma 3.2.

$T'_{\mathtt{LC}}$ from $T_{\mathtt{LC}}$. For each such rule application, we can assume that at least one of its conclusions can be added without violating $(R1')$ and $(R2')$, since otherwise $T'_{\mathtt{LC}} = T_{\mathtt{LC}}$ and there is nothing to prove.

*Case* $(\wedge)$. Suppose $T_{\mathtt{LC}}$ is extended into $T'_{\mathtt{LC}}$ by an application of $(\wedge)$ to a formula $@_i(\varphi \wedge \psi)$ on a branch $\Theta_{\mathtt{LC}}$ of $T_{\mathtt{LC}}$. In $T'_{\mathtt{LC}}$ the branch $\Theta_{\mathtt{LC}}$ has become extended to include $@_i\varphi$ or $@_i\psi$ or both (depending on whether they already occur on the branch, cf. $(R2')$). We can assume that both $@_i\varphi$ and $@_i\psi$ are being added, the other cases being treated similarly. Call the extended branch $\Theta'_{\mathtt{LC}}$. By $ih1$, $\pi(\Theta_{\mathtt{LC}})$ contains an $i$-block $B$ with $\varphi \wedge \psi$ (since $@_i(\varphi \wedge \psi)$ occurs induced on $\pi(\Theta_{\mathtt{LC}})$). Note that this block need not be the current one. We can now extend $\pi(\Theta_{\mathtt{LC}})$ as shown in Figure 6(a), first using GoTo to start a new block with opening nominal $i$ and then applying $(\wedge)$ to the occurrence of $\varphi \wedge \psi$ in $B$. Call this extended branch $\Theta'_{\mathtt{ST}}$, and let $T'_{\mathtt{ST}}$ denote the tableau in which $\pi(\Theta_{\mathtt{LC}})$ has been extended to $\Theta'_{\mathtt{ST}}$. Now define $\pi' = (\pi - \{(\Theta_{\mathtt{LC}}, \pi(\Theta_{\mathtt{LC}}))\}) \cup \{(\Theta'_{\mathtt{LC}}, \Theta'_{\mathtt{ST}})\}$, and note that $\pi'$ is a bijection from the set of branches of $T'_{\mathtt{LC}}$ onto the set of branches of $T'_{\mathtt{ST}}$.

We now need to show that any formula $@_j\gamma$ with $j \neq \gamma$ occurring on any branch $\Theta$ of $T'_{\mathtt{LC}}$ also occurs as induced formula on $\pi'(\Theta)$. There are two subcases, either *i)* $@_j\gamma$ is $@_i\varphi$ or $@_i\psi$ on $\Theta'_{\mathtt{LC}}$ or *ii)* $@_j\gamma$ is some other formula. In the first case it follows from the construction of $T'_{\mathtt{ST}}$ that $@_j\gamma$ occurs induced; in the latter it follows from $ih1$ together with the fact that $T'_{\mathtt{ST}}$ is constructed as an extension of $T_{\mathtt{ST}}$ and that $\pi'$ is based on $\pi$. Condition 2 of the lemma holds by $ih2$, as no new nominals have been introduced in the construction of $T'_{\mathtt{ST}}$ from $T_{\mathtt{ST}}$.

We omit the proof of the $(\neg\neg)$ case, as the $(\neg\neg)$ rule is structurally a special case of the $(\wedge)$ rule. For the $(\neg)$ rule, there is nothing to prove, as any application of $(\neg)$ on $T_{\mathtt{LC}}$ will only result in the addition of a formula on the form $@_j j$, and these formulas are not part of condition 1.

*Case* $(\neg\wedge)$. Suppose $T_{\mathtt{LC}}$ is extended into $T'_{\mathtt{LC}}$ by an application of $(\neg\wedge)$ to a formula $@_i\neg(\varphi \wedge \psi)$ on a branch $\Theta_{\mathtt{LC}}$. We can assume that neither $@_i\neg\varphi$ nor $@_i\neg\psi$ belongs to $\Theta_{\mathtt{LC}}$, because otherwise the rule application will not lead to branching, and the entire case becomes a simple variant of the $(\wedge)$ case. Under this assumption, the branch $\Theta_{\mathtt{LC}}$ is split into $\Theta^1_{\mathtt{LC}}$ and $\Theta^2_{\mathtt{LC}}$ by adding $@_i\neg\varphi$ and $@_i\neg\psi$, respectively. The resulting tableau is $T'_{\mathtt{LC}}$. By $ih1$, we have an $i$-block $B$ on $\pi(\Theta_{\mathtt{LC}})$ containing $\neg(\varphi \wedge \psi)$. Extend

this branch by applying GoTo with conclusion $i$ and apply $(\neg\wedge)$ to $\neg(\varphi \wedge \psi)$ occurring on $B$. $\pi(\Theta_{\mathtt{LC}})$ thus extends into two different branches. Let $\Theta^1_{\mathtt{ST}}$ be the left branch containing $\neg\varphi$, and let the right branch containing $\neg\psi$ be $\Theta^2_{\mathtt{ST}}$.

Now we define $\pi'$ as the simple modification of $\pi$ that removes $(\Theta_{\mathtt{LC}}, \pi(\Theta_{\mathtt{LC}}))$ and replaces it by $(\Theta^1_{\mathtt{LC}}, \Theta^1_{\mathtt{ST}})$ and $(\Theta^2_{\mathtt{LC}}, \Theta^2_{\mathtt{ST}})$. The extended tableaus are called $T'_{\mathtt{LC}}$ and $T'_{\mathtt{ST}}$, respectively. By the induction hypothesis and the construction of $\pi'$ from $\pi$ we have that $\pi'$ is a bijection between the set of branches of $T'_{\mathtt{LC}}$ and the set of branches of $T'_{\mathtt{ST}}$. Clearly condition 2 is satisfied. For condition 1, we need to prove that any formula $@_j\gamma$ with $j \neq \gamma$ occurring on any branch $\Theta$ of $T'_{\mathtt{LC}}$ occurs induced on $\pi'(\Theta)$. Let $@_j\gamma$ on some $\Theta$ be given with $j \neq \gamma$. If $\Theta$ is different from $\Theta^n_{\mathtt{ST}}$, for $n = 1, 2$ or $@_j\gamma$ is different from $@_i\neg\varphi$ or $@_i\neg\psi$ the claim follows from the induction hypothesis. Say therefore, that $\Theta$ is $\Theta^1_{\mathtt{LC}}$ and $@_j\gamma$ is $@_i\neg\varphi$. By construction, $\pi'(\Theta^1_{\mathtt{LC}}) = \Theta^1_{\mathtt{ST}}$ and on $\Theta^1_{\mathtt{ST}}$ there is an $i$-block (namely the current one) with element $\neg\varphi$. Therefore $@_i\neg\varphi$ occurs on $\pi'(\Theta^1_{\mathtt{LC}})$ in induced form. It's similar when $\Theta$ is $\Theta^2_{\mathtt{LC}}$ and $@_j\gamma$ is $@_i\neg\psi$.

*Case* $(\neg@)$. Suppose $T_{\mathtt{LC}}$ is extended into $T'_{\mathtt{LC}}$ by an application of $(\neg@)$ to a formula $@_i\neg@_j\varphi$ on a branch $\Theta_{\mathtt{LC}}$. In $T'_{\mathtt{LC}}$ the branch $\Theta_{\mathtt{LC}}$ has become extended by the addition of $@_j\neg\varphi$. Call the extended branch $\Theta'_{\mathtt{LC}}$. By $ih1$, the branch $\pi(\Theta_{\mathtt{LC}})$ must contain an $i$-block with $\neg@_j\varphi$. Extend this branch by first applying GoTo with conclusion $j$ and then applying $(\neg@)$ to the premise $\neg@_j\varphi$ to get $\neg\varphi$. Call the extended branch $\Theta'_{\mathtt{ST}}$. As before, let $\pi'$ be constructed from $\pi$ by replacing the pair $(\Theta_{\mathtt{LC}}, \pi(\Theta_{\mathtt{LC}}))$ by $(\Theta'_{\mathtt{LC}}, \Theta'_{\mathtt{ST}})$. Since $\neg\varphi$ occurs in a $j$-block of $\Theta'_{\mathtt{ST}}$, the formula $@_j\neg\varphi$ occurs induced on $\Theta'_{\mathtt{ST}}$. This gives us that condition 1 still holds for the extended tableaus. Condition 2 holds trivially as in the previous cases. Case $(@)$ is completely symmetric and is hence omitted.

*Case* $(\Diamond)$. Suppose $T_{\mathtt{LC}}$ is extended into $T'_{\mathtt{LC}}$ by an application of $(\Diamond)$ to a formula $@_i\Diamond\varphi$ on a branch $\Theta_{\mathtt{LC}}$. In $T'_{\mathtt{LC}}$ the branch $\Theta_{\mathtt{LC}}$ has become extended into a branch $\Theta'_{\mathtt{LC}}$ by the addition of $@_i\Diamond j$ and $@_j\varphi$ for some fresh nominal $j$. By the condition on the $(\Diamond)$ rule, $\varphi$ is not a nominal. By $ih1$, the branch $\pi(\Theta_{\mathtt{LC}})$ contains an $i$-block $B$ with $\Diamond\varphi$. As $j$ is fresh to $\Theta_{\mathtt{LC}}$, it is also, by $ih2$, fresh to $\pi(\Theta_{\mathtt{LC}})$. Extend $\pi(\Theta_{\mathtt{LC}})$ by first applying GoTo with conclusion $i$ to open a new $i$-block. Since $\varphi$ is not a nominal, we can afterwards apply $(\Diamond)$ to the premise $\Diamond\varphi$ occurring at the earlier $i$-block $B$, which will result in the addition of $\Diamond k$ and $@_k\varphi$ for some fresh nominal $k$. Since $j$ is fresh to $\Theta_{\mathtt{LC}}$, we can choose $k = j$, see Figure 6(b). As shown in Figure 6(b), we can now apply GoTo to open a $j$-block, and finally we can apply $(@)$ to the premise $@_j\varphi$ to add $\varphi$ to this $j$-block. Call the resulting branch $\Theta'_{\mathtt{ST}}$. In this branch, both $@_i\Diamond j$ and $@_j\varphi$ occur induced. Hence by letting $\pi'(\Theta'_{\mathtt{LC}}) = \Theta'_{\mathtt{ST}}$, we get that condition 1 is satisfied in the extended tableaus. Condition 2 also holds, as the only new nominal introduced in the construction of the Seligman tableau $T'_{\mathtt{ST}}$ is $j$ which was also introduced in the construction of the labelled tableau $T'_{\mathtt{LC}}$ from $T_{\mathtt{LC}}$.

*Case* $(\neg\Diamond)$. In this case $T'_{\mathtt{LC}}$ is obtained from $T_{\mathtt{LC}}$ by an application of $(\neg\Diamond)$ to formulas $@_i\neg\Diamond\varphi$ and $@_i\Diamond j$ on the branch $\Theta_{\mathtt{LC}}$. In $T'_{\mathtt{LC}}$ the branch $\Theta_{\mathtt{LC}}$ has been extended by $@_j\neg\varphi$. Call the extended branch $\Theta'_{\mathtt{LC}}$. By $ih1$, $\pi(\Theta_{\mathtt{LC}})$ contains $i$-blocks $B_1$ and $B_2$ (not necessarily distinct) such that $\neg\Diamond\varphi \in B_1$ and $\Diamond j \in B_2$. Following the same overall strategy as in the case of $(\Diamond)$, we can extend $\pi(\Theta_{\mathtt{LC}})$ as follows: *1)* First apply GoTo with conclusion $i$; *2)* Then apply $(\neg\Diamond)$ to $\neg\Diamond\varphi \in B_1$ and $\Diamond j \in B_2$ in order to get $\neg@_j\varphi$; *3)* Now apply GoTo with conclusion $j$; *4)* Finally apply $(@)$ to $\neg@_j\varphi$ in order to get $\neg\varphi$. The extended branch contains $@_j\neg\varphi$ in induced form, which leads to satisfaction of condition 1. Condition 2 holds trivially.

*Case* Id. In this case $T'_{\mathtt{LC}}$ is obtained from $T_{\mathtt{LC}}$ by an application of Id to premises $@_ij$ and $@_i\varphi$ occurring on a branch $\Theta_{\mathtt{LC}}$. In $T'_{\mathtt{LC}}$ the branch $\Theta_{\mathtt{LC}}$ has become extended

into a branch $\Theta'_{\mathtt{LC}}$ by the addition of $@_j\varphi$. By the condition on the $\mathsf{Id}$ rule, $@_i\varphi$ is not an accessibility formula. Furthermore, we must have $i \neq j$, since otherwise $@_i\varphi$ and $@_j\varphi$ are the same formula, and then the addition of $@_j\varphi$ to $\Theta_{\mathtt{LC}}$ would be in conflict with $(R2')$. Finally, we can assume $j \neq \varphi$, since otherwise the added formula is $@_jj$, and conditions 1 and 2 then hold by simply letting $T'_{\mathtt{ST}} = T_{\mathtt{ST}}$. By $ih1$, $\pi(\Theta_{\mathtt{LC}})$ contains $i$-blocks $B_1$ and $B_2$ with $j \in B_1$ and $\varphi \in B_2$. We need to show that $\pi(\Theta_{\mathtt{LC}})$ can be extended into a branch containing $\varphi$ in a $j$-block. First step is to extend $\pi(\Theta_{\mathtt{LC}})$ by applying $\mathsf{GoTo}$ with conclusion $j$ to open a new $j$-block. Afterwards we can apply $\mathsf{Nom}$ to the premises $i$ and $j$ of block $B_1$ to add $i$ to the current block. The current block is now a $j$-block containing $i$. If $i = \varphi$, we do not apply additional rules, since then the current block already contains $\varphi$. Otherwise, we apply $\mathsf{Nom}$ again, this time to the premises $i$ and $\varphi$ of block $B_2$ to add $\varphi$ to the current block (this is possible since the current block contains $i$). In both case we now have that $\varphi$ is contained in a $j$-block, as required. From this conditions 1 and 2 of the extended tableaus immediately follow as in the previous cases. □

**Theorem 3.3 (Completeness).** *If the formula $\varphi$ is valid, then there exists a closed tableau in* ST *having $\neg\varphi$ as the root formula.*

*Proof.* Assume $\varphi$ is valid. As LC is complete, there exists a closed LC-tableau $T_{\mathtt{LC}}$ with root $@_i\neg\varphi$, where $i$ is a nominal not occurring in $\varphi$. By Lemma 3.2 there is an ST-tableau $T_{\mathtt{ST}}$ with root $\neg\varphi$ and a bijection $\pi$ from the branches of $T_{\mathtt{LC}}$ into the branches of $T_{\mathtt{ST}}$ such that conditions 1 and 2 of the lemma holds. We now show that $T_{\mathtt{ST}}$ is closed. Let $\Theta_{\mathtt{ST}}$ denote an arbitrary branch of $T_{\mathtt{ST}}$. We need to show that $\Theta_{\mathtt{ST}}$ is closed. As $T_{\mathtt{LC}}$ is a closed LC-tableau, $\pi^{-1}(\Theta_{\mathtt{ST}})$ is a closed branch, meaning that it contains a pair of formulas $@_j\psi$ and $@_j\neg\psi$. If $j \neq \psi$ then condition 1 of the lemma implies that $@_j\psi$ and $@_j\neg\psi$ occur induced on $\Theta_{\mathtt{ST}}$. Thus in this case $\Theta_{\mathtt{ST}}$ contains a pair of $j$-blocks $B_1$ and $B_2$ with $\psi \in B_1$ and $\neg\psi \in B_2$. This means that $\Theta_{\mathtt{ST}}$ is closed, by definition. Assume instead $j = \psi$. Then condition 1 of the lemma only gives us that $@_j\neg\psi$ occurs induced on $\Theta_{\mathtt{ST}}$. But since $j = \psi$, this means that $\Theta_{\mathtt{ST}}$ contains a $j$-block $B$ with $\neg j \in B$. Since $B$ hence contains both $j$ and $\neg j$, we again have that the branch is closed. □

# 4 Remarks on Seligman-style Sequent Calculus

We now know that the tableau system ST is complete, but before we turn to the more demanding task of ensuring termination, we will note some links with Jerry Seligman's sequent calculus. In [23], Seligman develops a first sequent calculus for hybrid logic from a sequent system for classical first-order predicate logic by using a sequence of transformations which step-by-step internalize the semantics of hybrid logic. In the resulting system, the only symbols that occur in formulas are symbols of the hybrid object language. However this first sequent system only takes us halfway, for all the formulas in its rules are prefixed by @-operators (so this first sequent calculus is somewhat reminiscent of the labelled tableau system LC). But in the second stage, Seligman transforms the prefixed system into a system in which formulas are *not* required to be @-prefixed: the rules deal directly with unprefixed formulas. According to [23], page 684, the resulting calculus is "a more egalitarian logic in which there are Rules for All". This calculus inspired our search for a tableau system with similar properties, so let's briefly compare it with ST.

As should by now be clear, the crucial rules in ST are those that handle the nominals and the @-operator in a way that does not require formulas to be @-prefixed. In his system, Seligman uses the following six sequent rules for this purpose; he calls them Nominal Rules (see [23], page 685):

| | | | |
|---|---|---|---|
| $^{\vee}@L$ | $i, \varphi, \Gamma \longrightarrow \Delta$ | $\Rightarrow$ | $i, @_i\varphi, \Gamma \longrightarrow \Delta$ |
| $^{\vee}@R$ | $i, \Gamma \longrightarrow \Delta, \varphi$ | $\Rightarrow$ | $i, \Gamma \longrightarrow \Delta, @_i\varphi$ |
| $^{\wedge}@L$ | $i, @_i\varphi, \Gamma \longrightarrow \Delta$ | $\Rightarrow$ | $i, \varphi, \Gamma \longrightarrow \Delta$ |
| $^{\wedge}@R$ | $i, \Gamma \longrightarrow \Delta, @_i\varphi$ | $\Rightarrow$ | $i, \Gamma \longrightarrow \Delta, \varphi$ |
| name | $i, \Gamma \longrightarrow \Delta$ | $\Rightarrow$ | $\Gamma \longrightarrow \Delta$, if $i$ does not occur in $\Gamma, \Delta$ |
| term | $i, \Gamma \longrightarrow \Delta$ | $\Rightarrow$ | $\Gamma \longrightarrow \Delta$, if all formulas in $\Gamma, \Delta$ are @-prefixed. |

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

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. These rules introduced @-prefixes, and with these rules we were able to @-prefix entire tableau branches, thereby globalizing the information they contain. Our GoTo rule is the tableau corrolate of Seligman's term-rule, but note his side condition: it only lets us jump to a world if all information is @-prefixed, that is, if all information is global (recall that the Seligman-style natural deduction rule Term, which was discussed on page 5, works this way too). 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 (to globalize the information on a tableau branch) followed by an application of GoTo. But we were dissatisfied notationally, and due to the excessive applicability of rules, there was no promising path to a terminating system. Moreover, the @-prefixing permitted by the $^{\wedge}@L$ and $^{\wedge}@R$ rules interacted badly with (our tableau correlates of) the rules $^{\vee}@L$ and $^{\vee}@R$. Sometimes we would be forced to prefix an @, only to immediately strip it off, an obvious proof redundancy which we couldn't get rid of in a principled way.

These inter-related issues led us to introduce blocks. In essence, by making use of blocks, we avoid having to give explicit tableau rules corresponding to sequent rules $^{\wedge}@L$ and $^{\wedge}@R$; these rules are now absorbed into the concept of a block. This simultaneously eliminates these tableau rules, and bypasses the proof redundancy just mentioned. 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.

Despite its use of blocks, we believe our system is a fairly natural tableau analog of Seligman's system. Compare, for example, the following sequent derivation with the tableaux derivation given in Figure 5 on page 9:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{i, j, k \longrightarrow k}{i, j, @_j k \longrightarrow k}\,{}^{\vee}@\mathrm{L}}{i, @_i j, @_j k \longrightarrow k}\,{}^{\vee}@\mathrm{L}}{i, @_i j, @_j k \longrightarrow @_i k}\,{}^{\vee}@\mathrm{R}}{@_i j, @_j k \longrightarrow @_i k}\,\mathsf{term}}{@_i j \wedge @_j k \longrightarrow @_i k}\,\wedge\mathrm{L}$$

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

The use of blocks reverses a longstanding trend in hybrid logic (reliance on the labelling apparatus built into the object language) in favor of imposing more structure at the meta-level. Dividing branches into blocks *externalizes* (passes up to the meta-language) some of the work done by rules dealing with the @-operator. The use of the notion of induced satisfaction statement in our completeness proofs (which reflects the way that Seligman's $^@L$ and $^@R$ are absorbed into the concept of a block) is the clearest expression of this externalization.

# 5 Termination

Can the tableau system ST be used as a decision procedure? It is not difficult to see that unrestricted use of the calculus can lead to non-terminating computations; indeed, repeatedly applying Name is a trivial way of doing this. Still, by imposing natural restrictions on the application of rules, we can get a terminating calculus *without* resorting to loop checks (the first tableau-based decision procedure for hybrid logic, presented in [11], made use of loop checks, but shortly thereafter a procedure not requiring loop checks was devised [10]).

The first step towards a terminating calculus is to adopt standard rule application restrictions for tableau calculi to our block-based setting. On page 8 we presented the restrictions $(R1')$ and $(R2')$ imposed on LC. These restrictions adapt straightforwardly to the block-based setting in ST:

($R1$) *A formula is never added to an i-block if it already occurs in an i-block on the same branch.*

($R2$) *The* ($\Diamond$) *rule can not be applied twice to the same formula occurrence.*

Note that $(R2)$ and $(R2')$ use exactly the same formulation. However, since they appear in the context of two different proof systems, they still express slightly different conditions. In LC, no formula can occur twice on the same branch due to $(R1')$. So in LC, condition $(R2')$ is equivalent to the following: Any two applications of ($\Diamond$) to a formula $@_i\Diamond\varphi$ must occur on distinct branches. This is different in ST. In ST, the ($\Diamond$) rule applies to formulas of the form $\Diamond\varphi$, and $(R1)$ doesn't prevent a branch from containing several occurrences of the same such formula—as long as these occurrences appear in blocks with distinct opening nominals. Hence in ST, the ($\Diamond$) rule might be applied several times to the same formula $\Diamond\varphi$ on the same branch without violating $(R2)$. This can happen if $\Diamond\varphi$ occurs multiple times on the branch in blocks with distinct opening nominals.

As mentioned earlier, the Name rule is only necessary to provide an opening nominal for the initial block. To avoid the trivial form of non-termination that repeated applications of Name can lead to (indefinitely coming up with new names for the same block), we impose the following restriction:

($R3$) *The* Name *rule is only ever applied as the very first rule in a tableau.*

Another rule that can produce non-termination by repeated applications is GoTo; indeed, this can be applied an arbitrary number of times consecutively with the same

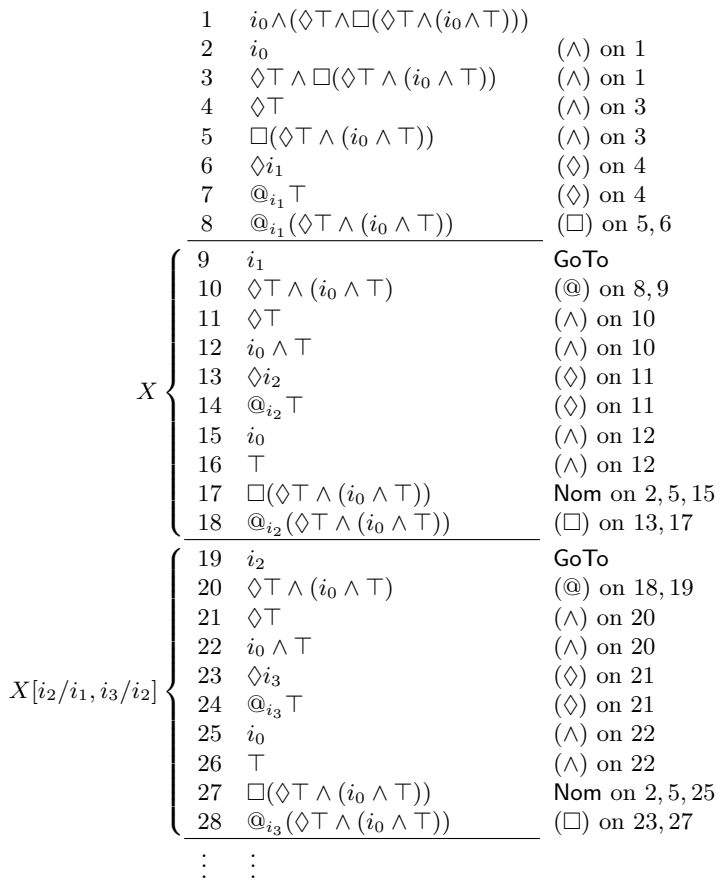



Figure 7: A non-terminating computation under restrictions $(R1)$–$(R4)$.

nominal in the conclusion every time. To avoid this trivial form of non-termination, we add the following restriction:

$(R4)$ *The* GoTo *rule can not be applied twice in a row.*

Restriction $(R4)$ means that GoTo can not be applied on a branch $\Theta$ if the previous rule applied on $\Theta$ was also GoTo. In other words, if we at some point use GoTo to open a new block, we are also required to "do something" with the block before we close it again and open another block.

However the restrictions $(R1)$–$(R4)$ are *not* sufficient to ensure termination, as the example in Figure 7 shows (note that the example makes use of the derived $(\Box)$ rule of Footnote 6). In this tableau, the $(\Diamond)$ rule is applied in lines $13 + 10n$, $n \geq 0$, to produce new successor worlds $\Diamond i_2, \Diamond i_3, \ldots$ Each of these applications of $(\Diamond)$ is to a formula of the form $\Diamond\top$. One might argue that we should see all these applications of $(\Diamond)$ as the *same rule instance*, since they all occur in blocks ending up containing $i_0$, that is, they are all applied in blocks that ultimately end up referring to the same world. But the "trick" used in each block to produce a successor $\Diamond i_{n+2}$ occurs at line $13+10n$. Crucially, this is *before* we realise (at line $15+10n$) that the block refers

to the same world as all the previous ones (because of the presence of $i_0$). Since each of the successors $\Diamond i_{n+2}$ that we produce is fresh, each application of $(\Box)$ in lines $18+10n$ is to a new pair of premises $\Diamond i_{n+2}, \Box(\Diamond\top \wedge (i_0 \wedge \top))$. Such applications are legitimate; there is no principled way to rule out them out.

To understand the problem we are facing, let us look at how termination is usually ensured in *labelled* tableau calculi for hybrid logic. The standard way to prove termination in such calculi without resorting to loop-checks is by a *decreasing length argument*. It goes as follows. We say that a formula $\varphi$ is *true at* a nominal $i$ on a tableau branch $\Theta$ in a labelled calculus if $@_i\varphi$ occurs on $\Theta$.[7] For a pair of nominals $i, j$ on $\Theta$, $j$ is said to be *generated by* $i$ on $\Theta$ if $j$ was introduced to the branch by an application of $(\Diamond)$ to a formula true at $i$. Now the core of the decreasing length argument is to establish that if $j$ is generated by $i$ on $\Theta$, then the syntactic complexity of any formula true at $j$ on $\Theta$ is strictly less than the maximal syntactic complexity of the formulas true at $i$. This immediately implies the non-existence of an infinite sequence of nominals $i_0, i_1, i_2, \ldots$ on $\Theta$, where each nominal is generated by its predecessor. And this can then, ultimately, be shown to ensure termination.

Now consider the case of the Seligman calculus. There are two possible ways to define *true at* in this calculus:

1. $\varphi$ is *true at* $i$ on $\Theta$ if there is an $i$-block on $\Theta$ containing $\varphi$.
2. $\varphi$ is *true at* $i$ on $\Theta$ if there is a block on $\Theta$ containing both $i$ and $\varphi$.

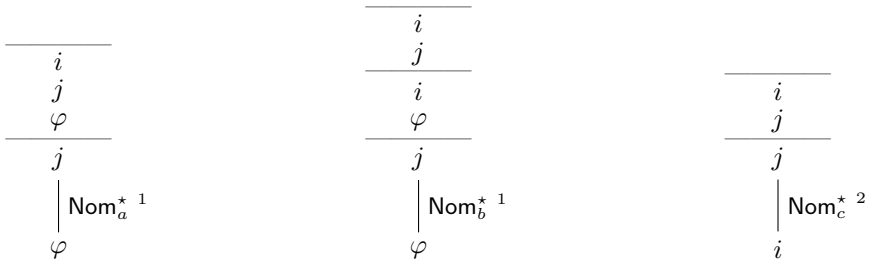
What we are about to show will apply equally to both possible definitions. The definition of generating nominals is as for the labelled calculus: $j$ is *generated by* $i$ on $\Theta$ if $j$ was introduced to $\Theta$ by an application of $(\Diamond)$ to a formula true at $i$.[8] Now consider again the tableau branch of Figure 7. Clearly, $i_0, i_1, i_2, \ldots$ is a sequence of nominals where each has been generated by its predecessor (independent of whether *true at* is defined by 1 or 2 above). Equally clearly, the maximal syntactic complexity of all formulas true at $i_n$ is the same for all $n > 0$. So the ordinary decreasing length argument breaks down in this case.

It is obviously the Nom rule which is responsible for the break-down of the decreasing length argument: this rule allows us to copy the formula $\Box(\Diamond\top \wedge (i_0 \wedge \top))$ to arbitrary blocks containing the nominal $i_0$, independently of any ordering on the nominals. The lesson to be learned is that we, as in labelled calculi, have to enforce some control on the "direction" we allow the copying of formulas, so that we can establish a decreasing length argument. It is OK to copy a formula true at a nominal $i$ to a nominal $j$ if $j$ generated $i$, but not if $i$ generated $j$.

A version of the Nom rule that only allows the copying of formulas in the "right direction", the direction of the generating nominals, is given below. First we need a couple of new definitions. Similar to LC, a formula of the form $\Diamond i$ occurring in an ST tableau is called an *accessibility formula* if it is the first conclusion of an application of the $(\Diamond)$ rule. In the tableau of Figure 7, the formulas in lines 6 and $13+10n$, $n \geq 0$, are accessibility formulas. A *quasi-root subformula* in a tableau is a formula of the form $\varphi$, $\neg\varphi$, $@_i\varphi$ or $\neg@_i\varphi$, where $\varphi$ is a subformula of the root of the tableau. By inspecting the rules of ST, it is clear that the only formulas in a tableau that are not

[7] This is the definition for *internalised* labelled calculi, like the one in [4] and the last calculus of [10]. For prefixed calculi, we would say that $\varphi$ is *true at* the *prefix* $\sigma$ if the prefixed formula $\sigma\varphi$ occurs on $\Theta$ [10].

[8] Note that if we define *true at* by condition 2, then a nominal can be generated by more than one other nominal.




[1] $i$, $j$ and $\varphi$ are all distinct, and $\varphi$ is a quasi-root subformula.
[2] $i$ is a root nominal and $i \neq j$.

Figure 8: The three special cases of $\mathsf{Nom}^\star$.

quasi-root subformulas are accessibility formulas and non-root nominals. We are now ready for the new $\mathsf{Nom}$ rule, called $\mathsf{Nom}^\star$.

$\mathsf{Nom}^\star$ Suppose $i$ and $j$ are nominals, $\varphi$ is a quasi-root subformula and $j \neq i, \varphi$. If $j$ and $\varphi$ both occur in $i$-blocks on a branch $\Theta$, then $\varphi$ can be added to any $j$-block on $\Theta$.

The $\mathsf{Nom}^\star$ rule splits up into three simple cases that we will now investigate. Assume first that $i$, $j$ and $\varphi$ are premises in an application of $\mathsf{Nom}^\star$ with $i \neq \varphi$. Then all of $i$, $j$ and $\varphi$ are distinct, since the rule already requires $j \neq i, \varphi$. The rule requires that both $j$ and $\varphi$ occur in $i$-blocks, which means that they either occur in the same $i$-block or in two distinct ones. This gives us the first two special cases of $\mathsf{Nom}^\star$ shown in Figure 8: $\mathsf{Nom}^\star_a$ and $\mathsf{Nom}^\star_b$. Now assume that we instead have $i = \varphi$. Then $i$ will be a root nominal, since $\varphi$ is required to be a quasi-root subformula. In this case $\mathsf{Nom}^\star$ becomes the last special case, $\mathsf{Nom}^\star_c$, shown in Figure 8. This last special case is a kind of restricted symmetry principle for root nominals.

Note that each of the special cases of $\mathsf{Nom}^\star$ is immediately derivable from the original $\mathsf{Nom}$ rule. The special cases $\mathsf{Nom}^\star_a$ and $\mathsf{Nom}^\star_c$ are just two special cases of $\mathsf{Nom}$, and $\mathsf{Nom}^\star_b$ can be derived by two consecutive applications of $\mathsf{Nom}$: one to add $i$ to the $j$-block, and then another to add $\varphi$. Note also that the non-termination of Figure 7 disappears when $\mathsf{Nom}$ is replaced by $\mathsf{Nom}^\star$: we can not add $\Box(\Diamond\top \wedge (i_0 \wedge \top))$ by $\mathsf{Nom}^\star$ in line 17, because this would require us to have an $i_0$-block containing $i_1$. Even though we have an $i_1$-block containing $i_0$, we can not produce the opposite, since the symmetry principle of $\mathsf{Nom}^\star_c$ only applies to an $i_1$-block containing $i_0$ when $i_1$ is a root nominal.

The requirement of $\varphi$ being a quasi-root subformula in $\mathsf{Nom}^\star$ is essential. If we allowed $\varphi$ to be an accessibility formula, we could produce a non-terminating tableau with root $i_0 \wedge \Diamond\top \wedge \Box(\Diamond\top \wedge i_0)$ (we leave out the derivation to save space). A similar example can be used to show non-termination if $\varphi$ is allowed to be a non-root nominal.

Still, we are not quite finished. Consider the tableau in Figure 9. This tableau is non-terminating, even though no applications of $\mathsf{Nom}$ or $\mathsf{Nom}^\star$ are involved. The problem here is that the (@) rule is used repeatedly to "copy" the formula $\Diamond(\top \wedge i_0)$

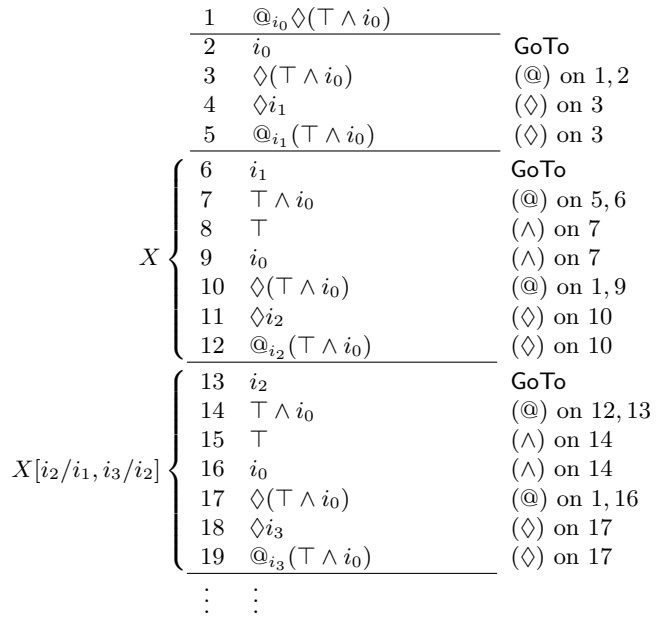



Figure 9: Another non-terminating computation under restrictions $(R1)$–$(R4)$.

to all later nominals, hence destroying the possibility of a decreasing length argument. But can deal with this last problem straightforwardly: when we want to copy a formula $\varphi$ to a block containing a nominal $i$, then $i$ must be the opening nominal of that block; as we saw with the restriction on the Nom rule, opening nominals are special. The corresponding restriction for the (@) rule is the following:

$(R5)$ *(@) and $(\neg@)$ can only be applied to premises $i$ and $@_i\varphi$ $(\neg@_i\varphi)$ when the current block is an $i$-block.*

With $(R5)$ in place, the application of (@) in line 7 of Figure 9 is still OK, but the application in line 10 is not.

We now have the full set of rule restrictions required to show termination. For the sake of the reader, we have collected them all in Figure 10. We are now ready to prove termination and completeness of the restricted calculus. We call this calculus $\mathtt{ST}^\star$. It consists of the same rules as ST, except that Nom is replaced by $\mathsf{Nom}^\star$, and restrictions $(R1)$–$(R5)$ are imposed. To prove termination of $\mathtt{ST}^\star$, we first show that any $\mathtt{ST}^\star$-tableau can be translated into an LC-tableau. This can then be used to show termination of $\mathtt{ST}^\star$, as LC is already known to be terminating [10].

**Lemma 5.1.** *Assume that $\Theta_{\mathtt{ST}}$ is a branch of an $\mathtt{ST}^\star$-tableau in which* Name *is applied as the first rule. Then there exists an* LC*-tableau branch $\Theta_{\mathtt{LC}}$ such that:*

1. *If $@_i\varphi \in \Theta_{\mathtt{ST}}$ then $@_i\varphi \in \Theta_{\mathtt{LC}}$.*
2. *If $\neg@_i\varphi \in \Theta_{\mathtt{ST}}$ then $@_i\neg\varphi \in \Theta_{\mathtt{LC}}$.*
3. *If an occurrence of a formula $\psi$ in a $j$-block of $\Theta_{\mathtt{ST}}$ is not of the form $@_i\varphi$ or $\neg@_i\varphi$ or is $j$ itself, then $@_j\psi \in \Theta_{\mathtt{LC}}$.*

($R1$) *A formula is never added to an $i$-block if it already occurs in an $i$-block on the same branch.*

($R2$) *The ($\Diamond$) rule can not be applied twice to the same formula occurrence.*

($R3$) *The* Name *rule is only ever applied as the very first rule in a tableau.*

($R4$) *The* GoTo *rule can not be applied twice in a row.*

($R5$) *(@) and ($\neg$@) can only be applied to premises $i$ and $@_i\varphi$ ($\neg@_i\varphi$) when the current block is an $i$-block.*

Figure 10: The full set of restrictions, ($R1$)–($R5$), imposed on the terminating calculus $\mathtt{ST}^\star$.

*4. The same nominals occur in $\Theta_{\mathtt{ST}}$ and $\Theta_{\mathtt{LC}}$.*

*Proof.* The proof is by induction on the number of rule applications made on $\Theta_{\mathtt{ST}}$.

*Base case.* We choose our base case to be after the first rule application on $\Theta_{\mathtt{ST}}$, which by assumption is an application of Name. So in the base case, $\Theta_{\mathtt{ST}}$ has the following form:

| | | |
|---|---|---|
| 1 | $\varphi_0$ | |
| 2 | $i_0$ | Name |

where $i_0$ is a nominal not occurring in $\varphi_0$. Let $i_1, \ldots, i_n$ denote the nominals in $\varphi_0$. We let the root of $\Theta_{\mathtt{LC}}$ be the formula $@_{i_0}(\varphi_0 \wedge \bigwedge_{k=0,\ldots,n} @_{i_k} i_k)$. This is the same as the root formula of $\Theta_{\mathtt{ST}}$, except we have added a conjunct of the form $@_{i_k} i_k$ for each root nominal $i_k$. This is for a technical reason that will become clear when we cover the case of $\mathsf{Nom}^\star$ below. We now apply ($\wedge$) to the root of $\Theta_{\mathtt{LC}}$, so that $\Theta_{\mathtt{LC}}$ becomes:

| | | |
|---|---|---|
| 1 | $@_{i_0}(\varphi_0 \wedge \bigwedge_{k=0,\ldots,n} @_{i_k} i_k)$ | |
| 2 | $@_{i_0}\varphi_0$ | ($\wedge$) on 1 |
| 3 | $@_{i_0} \bigwedge_{k=0,\ldots,n} @_{i_k} i_k$ | ($\wedge$) on 1 |

Note that $\Theta_{\mathtt{ST}}$ is an $i_0$-block containing only a single formula $\varphi_0$ distinct from $i_0$ itself. Assume first that $\varphi_0$ is not on the form $@_i\varphi$ or $\neg@_i\varphi$, that is, it is not a satisfaction statement or negated satisfaction statement. In this case, conditions 1 and 2 of the lemma hold trivially. Condition 3 holds by the occurrence of $@_{i_0}\varphi_0$ in $\Theta_{\mathtt{LC}}$. Condition 4 holds trivially by construction of $\Theta_{\mathtt{LC}}$ from $\Theta_{\mathtt{ST}}$. If $\varphi_0$ is a satisfaction statement, we extend $\Theta_{\mathtt{LC}}$ by applying (@) to line 2. This results in adding $\varphi_0$ to $\Theta_{\mathtt{LC}}$, hence satisfying condition 1. Conditions 2–4 are trivially satisfied in this case. The case where $\varphi_0$ is a negated satisfaction statement is completely analogous.

*Induction step.* Assume $\Theta_{\mathtt{ST}}$ and $\Theta_{\mathtt{LC}}$ satisfy conditions 1–4. We need to prove that if $\Theta_{\mathtt{ST}}$ is extended into $\Theta'_{\mathtt{ST}}$ by a single rule application, then we can construct a similar extension $\Theta'_{\mathtt{LC}}$ of $\Theta_{\mathtt{LC}}$ so that 1–4 still hold. We prove this by examining each possible case of a rule application building $\Theta'_{\mathtt{ST}}$ from $\Theta_{\mathtt{ST}}$. For each such rule application, we can assume that at least one of its conclusions can be added without violating ($R1$)–($R5$), since otherwise $\Theta'_{\mathtt{ST}} = \Theta_{\mathtt{ST}}$ and there is nothing to prove.

*Case* ($\neg\neg$). Suppose $\Theta'_{\mathtt{ST}}$ is obtained from $\Theta_{\mathtt{ST}}$ by an application of ($\neg\neg$) to a premise $\neg\neg\varphi$. Then $\Theta'_{\mathtt{ST}}$ is $\Theta_{\mathtt{ST}}$ extended by $\varphi$. The premise $\neg\neg\varphi$ either belongs to the current block of $\Theta_{\mathtt{ST}}$ or a previous block with the same opening nominal. Let $i$ be this nominal. By the induction hypothesis, $@_i\neg\neg\varphi \in \Theta_{\mathtt{LC}}$ (by condition 3). We split into subcases depending on the form of $\varphi$. If $\varphi$ is not a satisfaction statement or negated

satisfaction statement, then conditions 1–2 hold trivially, and we only need to ensure conditions 3–4. Conversely, if $\varphi$ *is* a satisfaction statement or negated satisfaction statement, then condition 3 holds trivially, and we only need to ensure condition 4 and either 1 or 2. We first consider the case where $\varphi$ is not a satisfaction statement or negated satisfaction statement. In this case we satisfy condition 3 by constructing $\Theta'_{\mathtt{LC}}$ such that it contains $@_i\varphi$. If $\Theta_{\mathtt{LC}}$ already contains $@_i\varphi$, we simply let $\Theta'_{\mathtt{LC}} = \Theta_{\mathtt{LC}}$. Otherwise, we can apply $(\neg\neg)$ to $@_i\neg\neg\varphi$ on $\Theta_{\mathtt{LC}}$, which will add $@_i\varphi$ to the branch. We then let $\Theta'_{\mathtt{LC}}$ be the resulting branch. Alternatively, if $\varphi$ is a satisfaction statement, we satisfy condition 1 by constructing $\Theta'_{\mathtt{LC}}$ such that it contains $\varphi$. Let $\Theta'_{\mathtt{LC}}$ be the extension of $\Theta_{\mathtt{LC}}$ obtained by applying first $(\neg\neg)$ to $@_i\neg\neg\varphi$ (if $@_i\varphi$ is not already on $\Theta_{\mathtt{LC}}$), and then applying $(@)$ to $@_i\varphi$ (if $\varphi$ is not already on $\Theta_{\mathtt{LC}}$). If $\varphi$ is a negated satisfaction statement, we apply $(\neg@)$ instead of $(@)$, ensuring that condition 2 is satisfied. Since condition 4 is trivially satisfied, this concludes the case of $(\neg\neg)$. The other propositional cases, as well as the case of $(\neg\Diamond)$, are similar.

*Case* $(@)$. $\Theta'_{\mathtt{ST}}$ is obtained from $\Theta_{\mathtt{ST}}$ by applying $(@)$ to formulas $i$ and $@_i\varphi$. Then $\Theta'_{\mathtt{ST}}$ is $\Theta_{\mathtt{ST}}$ extended by $\varphi$. By $(R5)$, the current block must be an $i$-block. We split into subcases depending on the form of $\varphi$, as for the $(\neg\neg)$ case above. If $\varphi$ is not a satisfaction statement or a negated satisfaction statement, we satisfy condition 3 by constructing $\Theta'_{\mathtt{LC}}$ such that it contains $@_i\varphi$. But by the induction hypothesis we already have $@_i\varphi \in \Theta_{\mathtt{LC}}$, since $@_i\varphi \in \Theta_{\mathtt{ST}}$ (using condition 1). Hence we can let $\Theta'_{\mathtt{LC}} = \Theta_{\mathtt{LC}}$ and we're done. If $\varphi$ is a satisfaction statement, we satisfy condition 1 by applying $(@)$ on $\Theta_{\mathtt{LC}}$ as appropriate. If $\varphi$ is a negated satisfaction statement, we satisfy condition 2 by applying $(\neg@)$ as appropriate. The case of the $(\neg@)$ rule is similar.

*Case* $\mathsf{Nom}^\star$. $\Theta'_{\mathtt{ST}}$ is obtained from $\Theta_{\mathtt{ST}}$ by an application of $\mathsf{Nom}^\star$ of one of the three types shown in Figure 8. Let us consider the cases of $\mathsf{Nom}^\star_a$ and $\mathsf{Nom}^\star_b$ first. In these cases, $\Theta'_{\mathtt{ST}}$ is obtained from $\Theta_{\mathtt{ST}}$ by the addition of a formula $\varphi$ to a $j$-block, where: i) $\Theta_{\mathtt{ST}}$ contains an $i$-block with $j$; ii) $\Theta_{\mathtt{ST}}$ contains an $i$-block with $\varphi$; iii) all of $i$, $j$ and $\varphi$ are distinct; iv) $\varphi$ is a quasi-root subformula. If $\varphi$ is not a satisfaction statement or a negated satisfaction statement, we show that we can extend $\Theta_{\mathtt{LC}}$ into a branch $\Theta'_{\mathtt{LC}}$ containing $@_j\varphi$. By the induction hypothesis, $\Theta_{\mathtt{LC}}$ must contain $@_ij$ and $@_i\varphi$ (from i and ii). Since $\varphi$ is not an accessibility formula, we can apply $\mathsf{Id}$ to the pair of premises $@_ij$, $@_i\varphi$ on $\Theta_{\mathtt{LC}}$ and get $@_j\varphi$. Let $\Theta'_{\mathtt{LC}}$ be $\Theta_{\mathtt{LC}}$ extended by this rule application (again, the addition of $@_i\varphi$ to $\Theta_{\mathtt{LC}}$ will only be blocked if $@_i\varphi$ already occurs there). We now have that conditions 1–4 are satisfied for $\Theta'_{\mathtt{ST}}$ and $\Theta'_{\mathtt{LC}}$. The subcases where $\varphi$ is a satisfaction statement or a negated satisfaction statement are trivial applications of the induction hypothesis. This concludes cases $\mathsf{Nom}^\star_a$ and $\mathsf{Nom}^\star_b$. Now consider $\mathsf{Nom}^\star_c$. In this case, $\Theta_{\mathtt{ST}}$ contains $j$ in an $i$-block, and $\Theta'_{\mathtt{ST}}$ extends $\Theta_{\mathtt{ST}}$ by the addition of $i$ to a $j$-block. Furthermore, $i$ is a root nominal and $i \neq j$. We need to show that we can extend $\Theta_{\mathtt{LC}}$ into a branch $\Theta'_{\mathtt{LC}}$ containing $@_ji$. By the induction hypothesis we have $@_ij \in \Theta_{\mathtt{LC}}$. Now recall from the base case of this proof that line 3 of $\Theta_{\mathtt{LC}}$ is the formula $@_{i_0} \bigwedge_{k=0,\ldots,n} @_{i_k} i_k$. Since $i$ is a root nominal, it must be one of the $i_k$. By repeated applications of $(\wedge)$ to this formula, we can thus extend $\Theta_{\mathtt{LC}}$ into a branch containing $@_{i_0}@_ii$; and by an additional application of $(@)$ into a branch containing $@_ii$. Now we can apply $\mathsf{Id}$ to the pair of premises $@_ij$, $@_ii$ and get $@_ji$. Letting $\Theta'_{\mathtt{ST}}$ be the branch thus extended, we have shown what was required.

*Case* $(\mathsf{GoTo})$. $\Theta'_{\mathtt{ST}}$ is obtained from $\Theta_{\mathtt{ST}}$ by an application of $(\mathsf{GoTo})$. In this case we simply let $\Theta'_{\mathtt{LC}} = \Theta_{\mathtt{LC}}$, and 1–4 will hold trivially by the induction hypothesis.

*Case* $(\Diamond)$. $\Theta'_{\mathtt{ST}}$ is obtained from $\Theta_{\mathtt{ST}}$ by applying $(\Diamond)$ to a formula $\Diamond\varphi$ that either belongs to the current block or a previous block with the same opening nominal $i$. By the side condition on the $(\Diamond)$ rule, $\varphi$ can not be a nominal. By the induction

hypothesis, $@_i\Diamond\varphi \in \Theta_{\mathtt{LC}}$. $\Theta'_{\mathtt{ST}}$ is $\Theta_{\mathtt{ST}}$ extended by $\Diamond j$ and $@_j\varphi$ for some fresh nominal $j$. By condition 4 of the induction hypothesis, $j$ is fresh to $\Theta_{\mathtt{LC}}$ as well. We need to construct $\Theta'_{\mathtt{LC}}$ from $\Theta_{\mathtt{LC}}$ such that it contains $@_i\Diamond j$ and $@_j\varphi$ (to satisfy conditions 1, 3 and 4). Since $j$ is fresh to $\Theta_{\mathtt{LC}}$, it suffices to show that $(\Diamond)$ is applicable to $@_i\Diamond\varphi$ on $\Theta_{\mathtt{LC}}$. Assume for the sake of contradiction that this rule application is blocked. Since $\varphi$ is not a nominal, $\Diamond\varphi$ can not be an accessibility formula, and hence the only condition that can be blocking the rule application is $(R2')$. And this can only happen if $\Theta_{\mathtt{LC}}$ already contains an application of $(\Diamond)$ to this formula occurrence. Since all applications of $(\Diamond)$ on $\Theta_{\mathtt{LC}}$ are induced by applications of $(\Diamond)$ on $\Theta_{\mathtt{ST}}$, this implies that $\Theta_{\mathtt{ST}}$ contains an application of $(\Diamond)$ to an occurrence of $\Diamond\varphi$ in an $i$-block. By $(R1)$, there can only be one occurrence of $\Diamond\varphi$ in an $i$-block on $\Theta_{\mathtt{ST}}$. Hence, we get that $\Theta_{\mathtt{ST}}$ contains an application of $(\Diamond)$ to the $i$-block occurrence of $\Diamond\varphi$. Because of $(R2)$, we can not reapply $(\Diamond)$ to this formula occurrence, contradicting our assumption on how $\Theta'_{\mathtt{ST}}$ is built from $\Theta_{\mathtt{ST}}$. This concludes the case of $(\Diamond)$ and the entire proof. □

We can now prove termination.

**Theorem 5.2.** *Any* $\mathtt{ST}^\star$*-tableau branch is finite.*

*Proof.* Let $\Theta_{\mathtt{ST}}$ be the branch of an $\mathtt{ST}^\star$-tableau. We need to show that $\Theta_{\mathtt{ST}}$ is finite. We can without loss of generality assume that the first rule applied on $\Theta_{\mathtt{ST}}$ is $\mathsf{Name}$. To show this, assume $\Theta'_{\mathtt{ST}}$ is an infinite branch of $\mathtt{ST}^\star$ not including any applications of $\mathsf{Name}$ (recall that by $(R3)$, either $\mathsf{Name}$ is applied as the first rule or never applied). Let $\Theta''_{\mathtt{ST}}$ be obtained from $\Theta'_{\mathtt{ST}}$ by inserting $\mathsf{Name}$ as the first rule application. Then $\Theta''_{\mathtt{ST}}$ is still a valid, infinite tableau branch of $\mathtt{ST}^\star$. This shows that if it is possible to construct an infinite $\mathtt{ST}^\star$ branch, then it is possible to construct one in which $\mathsf{Name}$ is the first rule applied. So in the following we can without loss of generality assume $\mathsf{Name}$ to be the first rule applied on $\Theta_{\mathtt{ST}}$.

By Lemma 5.1, there exists an $\mathtt{LC}$-tableau branch $\Theta_{\mathtt{LC}}$ satisfying conditions 1–4. Since $\mathtt{LC}$ is a terminating calculus, $\Theta_{\mathtt{LC}}$ must be finite and hence only contains finitely many nominals. By condition 4 of Lemma 5.1, $\Theta_{\mathtt{ST}}$ then contains only finitely many nominals as well. Hence the number of distinct formulas on $\Theta_{\mathtt{ST}}$ must also be finite, since any formula on $\Theta_{\mathtt{ST}}$ is either a quasi-root subformula or $\Diamond i$ or $i$ for some nominal. This implies that any block on $\Theta_{\mathtt{ST}}$ must be finite, using $(R2)$. Now assume for the sake of contradiction that $\Theta_{\mathtt{ST}}$ is infinite. Then for some $i$, $\Theta_{\mathtt{ST}}$ must contain infinitely many $i$-blocks (since each block is finite and there are only finitely many nominals). By $(R2)$, none of these $i$-blocks contain a formula already contained in a previous $i$-block—except if it is the opening nominal itself. Hence, each $i$-block either contains a new formula not occurring in any previous $i$-block, or it only contains the opening nominal. Since there are only finitely many distinct formulas but infinitely many $i$-blocks, infinitely many of the $i$-blocks can then only contain the opening nominal. But this contradicts $(R4)$: $\mathsf{GoTo}$ can not be applied twice in a row. □

In Section 3 we proved that the calculus $\mathtt{ST}$ is complete. We now wish to prove that $\mathtt{ST}^\star$ is complete as well, that is, the replacement of $\mathsf{Nom}$ by $\mathsf{Nom}^\star$ and the introduction of restrictions $(R1)$–$(R5)$ doesn't destroy completeness. It suffices to prove an analog of Lemma 3.2 for $\mathtt{ST}^\star$, which we shall now do.

**Lemma 5.3.** *Let* $\varphi$ *be any formula and* $i$ *any nominal not in* $\varphi$*. Assume* $T_{\mathtt{LC}}$ *is a tableau with root* $@_i\varphi$ *in the calculus* $\mathtt{LC}$*. Then there exists a tableau* $T_{\mathtt{ST}}$ *with root* $\varphi$ *in the calculus* $\mathtt{ST}^\star$*, and a bijection*

$$\pi : \{\Theta \mid \Theta \text{ is a branch of } T_{\mathtt{LC}}\} \to \{\Theta' \mid \Theta' \text{ is a branch of } T_{\mathtt{ST}}\}$$

*such that:*

1. *Given any branch $\Theta$ of $T_{\mathtt{LC}}$, any formula $@_j\psi \in \Theta$ with $j \neq \psi$ occurs as induced formula on $\pi(\Theta)$.*

2. *All nominals that occur on $\pi(\Theta)$ also occur on $\Theta$.*

*Proof.* We need to prove that the induction proof of Lemma 3.2 still goes through when $\mathtt{ST}$ is replaced by $\mathtt{ST}^\star$. First we show that any application of $\mathsf{Nom}$ in the induction proof of Lemma 3.2 can be replaced by an application of $\mathsf{Nom}^\star$. The rule $\mathsf{Nom}$ is only applied in the $\mathsf{Id}$ case of the induction step, so let us consider that case.

*Case* $\mathsf{Id}$. As in the $\mathsf{Id}$ case of the proof of Lemma 3.2, we assume the following (see the proof of Lemma 3.2 for details): i) $@_i j, @_i\varphi \in \Theta_{\mathtt{LC}}$; ii) $@_i\varphi$ is not an accessibility formula on $\Theta_{\mathtt{LC}}$; iii) $\Theta'_{\mathtt{LC}}$ is $\Theta_{\mathtt{LC}}$ extended by $@_j\varphi$; iv) $j \neq i, \varphi$; v) $\pi(\Theta_{\mathtt{LC}})$ contains $i$-blocks $B_1$ and $B_2$ with $j \in B_1$ and $\varphi \in B_2$. We need to show that $\pi(\Theta_{\mathtt{LC}})$ can be extended into a branch containing $\varphi$ in a $j$-block. We can assume that $\varphi$ does not already occur in a $j$-block on $\pi(\Theta_{\mathtt{LC}})$, since otherwise there is nothing to prove. First apply $\mathsf{GoTo}$ on $\pi(\Theta_{\mathtt{LC}})$ with conclusion $j$ to open a new $j$-block. Let $@_{i_0}\varphi_0$ denote the root of $\Theta_{\mathtt{LC}}$. The root of $\pi(\Theta_{\mathtt{LC}})$ is then $\varphi_0$, by construction. Since $@_i\varphi$ is not an accessibility formula on $\Theta_{\mathtt{LC}}$, Lemma 6.1 of [10] (Quasi-subformula Property) gives us that $\varphi$ is of the form $\psi$ or $\neg\psi$ where $\psi$ is a subformula of $\varphi_0$. Since $\varphi_0$ is the root of $\pi(\Theta_{\mathtt{LC}})$ this immediately implies that $\varphi$ is a quasi-root subformula on $\pi(\Theta_{\mathtt{LC}})$. Hence on $\pi(\Theta_{\mathtt{LC}})$ we have that $\varphi$ is a quasi-root subformula, $j \neq i, \varphi$ and $j$ and $\varphi$ both occur in $i$-blocks. This is exactly the set of conditions mentioned in the $\mathsf{Nom}^\star$ rule, so we can now apply $\mathsf{Nom}^\star$ at the current $j$-block to add $\varphi$. This shows that $\pi(\Theta_{\mathtt{LC}})$ can be extended into a branch containing $\varphi$ in a $j$-block, as required.

In the following we will use the term "the modified proof of Lemma 3.2" to refer to the proof of Lemma 3.2 where the $\mathsf{Id}$ case has been modified as above to use $\mathsf{Nom}^\star$ instead of $\mathsf{Nom}$. What is now left is to prove that all of the restrictions $(R1)$–$(R5)$ are satisfied in the inductive construction of $T_{\mathtt{ST}}$ in the modified proof of Lemma 3.2. We consider the 5 restrictions in turn below.

$(R1)$. This restriction is not necessarily satisfied in the current construction of $T_{\mathtt{ST}}$, but the construction can easily be modified to satisfy it, as we will now show. Consider the application of a rule distinct from $\mathsf{GoTo}$ on a branch $\Theta_{\mathtt{ST}}$ in the inductive construction of $T_{\mathtt{ST}}$. Let $i$ denote the opening nominal of the current block. If the considered rule application has a conclusion $\varphi$ that already occurs in an $i$-block on $\Theta_{\mathtt{ST}}$, we simply choose not to add $\varphi$ again. This omission doesn't prevent any other rules from being applied later that would otherwise have been applicable, and it obviously doesn't affect conditions 1 and 2 either (the set of formulas occurring induced on $\Theta_{\mathtt{ST}}$ remains the same).

$(R2)$. Assume for the sake of contradiction that $T_{\mathtt{ST}}$ contains a branch $\Theta_{\mathtt{ST}}$ on which $(\Diamond)$ has been applied twice to the same formula occurrence $\Diamond\varphi$ in some $i$-block. Consulting the (modified) proof of Lemma 3.2, it is seen that both of these applications must be induced by applications of $(\Diamond)$ to $@_i\Diamond\varphi$ on $\pi^{-1}(\Theta_{\mathtt{ST}})$ (the only applications of $(\Diamond)$ in the inductive construction of $T_{\mathtt{ST}}$ is in the $(\Diamond)$ case of the proof). However, this directly contradicts that $\pi^{-1}(\Theta_{\mathtt{ST}})$ is built under restriction $(R2')$.

$(R3)$. In the (modified) proof of Lemma 3.2, $\mathsf{Name}$ is only applied as the very first rule, so $(R3)$ is trivially satisfied.

$(R4)$. Consulting the (modified) proof of Lemma 3.2, it is possible to show that $\mathsf{GoTo}$ is never applied twice in a row in the inductive construction of $T_{\mathtt{ST}}$.[9]

[9]But even if $\mathsf{GoTo}$ was applied twice in a row, we would always be able to get rid of it: simply

($R5$). In the modified proof of Lemma 3.2, the (@) and ($\neg$@) rules of ST are being applied only in the ($\neg$@), (@), ($\Diamond$) and ($\neg\Diamond$) cases of the induction step. Inspecting these cases it is seen that the (@) and ($\neg$@) rules are only applied to premises of the form $@_i\varphi$ or $\neg@_i\varphi$ when the current block is an $i$-block. Hence ($R5$) is satisfied. □

**Theorem 5.4** (**Completeness of ST$^\star$**). *If the formula $\varphi$ is valid, then there exists a closed tableau in* ST$^\star$ *having $\neg\varphi$ as the root formula.*

*Proof.* The proof is a copy of the proof of Theorem 3.3, except that ST is replaced by ST$^\star$ and references to Lemma 3.2 are replaced by references to Lemma 5.3. □

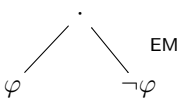
# 6 Concluding Remarks

In this paper we introduced a Seligman-style tableau system for basic hybrid logic. By imposing a block structure on tableaus, and introducing rules (notably the GoTo rule) to exploit this structure we were able to externalize (push up to the meta-level) the basic proof mechanisms. We thus arrived (to use Seligman's words) at "a more egalitarian logic in which there are Rules for All". Indeed, we ended up with two such logics, namely ST and ST$^\star$.

Interesting work remains to be done. For a start, there are tricky questions about rule derivability. For example, some important labelled hybrid tableau systems (as found for instance in [7]) make use of rules not found in LC, such as the Bridge rule:

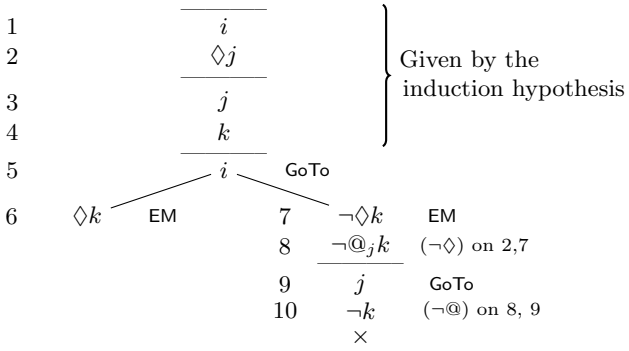
$$\begin{array}{c} @_i\Diamond j \\ @_j k \\ \hline @_i\Diamond k \end{array}\ \text{Bridge}$$

We are interested in finding interpretations of such rules in ST and ST$^\star$ in the way we in these systems have interpreted the rules of LC. A way of approaching such issues might be to show the derivability of an appropriately restricted version of the EM rule (the excluded middle rule, also called the cut rule), which says that for any formula $\varphi$ of some specified type, any leaf of the tableau tree can be split by extending to the left with $\varphi$ and to the right with $\neg\varphi$:




For example, if we could show that EM was derivable in, say, ST for $\varphi$ having the form $@_i\Diamond k$, we could show interpretability of the Bridge rule in the style of Lemma 3.2 as indicated by the following diagram:

omit all applications of GoTo except the last in any sequence of consecutive applications of GoTo. As in the case of ($R2$), this would neither affect the applicability of later rules nor the fulfilment of conditions 1 and 2.




Soundness of the unrestricted EM rule, to be precise, soundness of the system ST + EM, together with our first completeness result, shows that the unrestricted EM rule is admissible in ST (that is, if ST + EM $\vdash \varphi$ then ST $\vdash \varphi$), but the EM rule is not derivable without restrictions on the formula $\varphi$. However, it is an open question whether it is derivable when $\varphi$ has the form $\Diamond k$.

The issue of rule derivability also bears our current work. Our next goal is to provide completeness proofs covering the standard extensions of basic hybrid logic, such as tense logical extensions, the difference operator and universal modalities, the downarrow binder and the strong Priorean binders, and first-order hybrid logic. All these extensions have natural Seligman-style treatments, and in some cases completeness can be established by proving suitable rule derivability results and drawing on the corresponding results for labelled calculi (indeed, we suspect that if we could prove the derivability of the above mentioned restriction version of EM, we could prove completeness of most, perhaps all, of these extensions in this way). But we are currently working on model-theoretic completeness proofs, as we suspect that this may provide more straightforward and general answers. Moreover, although we introduced ST$^\star$ simply as a terminating version of ST, we find the three cases of the Nom$^\star$ rule interesting in their own right. We hope to show model-theoretically that ST$^\star$ can replace ST as the core inference engine in all extended systems.

# 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] Blackburn, P.: Internalizing labelled deduction. Journal of Logic and Computation 10(1), 137–168 (2000)

[5] Blackburn, P., Bolander, T., Braüner, T., Jørgensen, K.F.: A Seligman Tableau System for Hybrid Logic. Lecture Notes in Computer Science 8312, 147–163 (2013)

[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, International Conference, TABLEAUX 2002, pp. 38–52. Copenhagen, Denmark (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.: Hybrid-logical reasoning in the Smarties and Sally-Anne tasks. Journal of Logic, Language and Information 23, 415–439 (2014), revised and extended version of [14]

[16] Chadha, R., Macedonio, D., Sassone, V.: A hybrid intuitionistic logic: Semantics and decidability. Journal of Logic and Computation 16, 27–59 (2006)

[17] Galmiche, D., Salhi, Y.: Sequent calculi and decidability for intuitionistic hybrid logic. Information and Computation 209, 1447–1463 (2011)

[18] Götzmann, D., Kaminski, M., Smolka, G.: Spartacus: A Tableau Prover for Hybrid Logic. Electr. Notes Theor. Comput. Sci. 262, 127–139 (2010)

[19] Hoffmann, G., Areces, C.: HTab: A Terminating Tableaux System for Hybrid Logic. In: Proceedings of Methods for Modalities 5 (November 2007)

[20] 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`

[21] 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)

[22] 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)

[23] 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.)

[24] Stenning, K., van Lambalgen, M.: Human Reasoning and Cognitive Science. MIT Press (2008)

[25] Tzakova, M.: Tableau Calculi for Hybrid Logics. Lecture Notes in Computer Science 1617, 278–92 (1999)