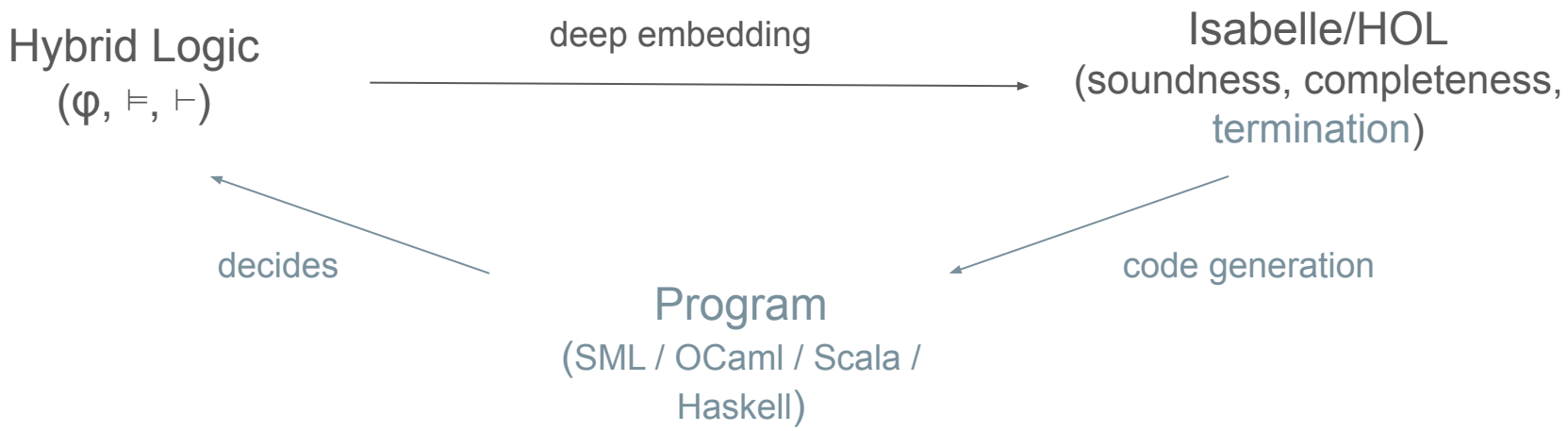
On Termination for Hybrid Tableaux

Asta Halkjær From, PhD student
Technical University of Denmark (DTU Compute)

Introduction

Tableau system ST^A for basic hybrid logic (4900+ lines in AFP):
https://isa-afp.org/entries/Hybrid_Logic.html

Based on ST* by Blackburn, Bolander, Braüner and Jørgensen.

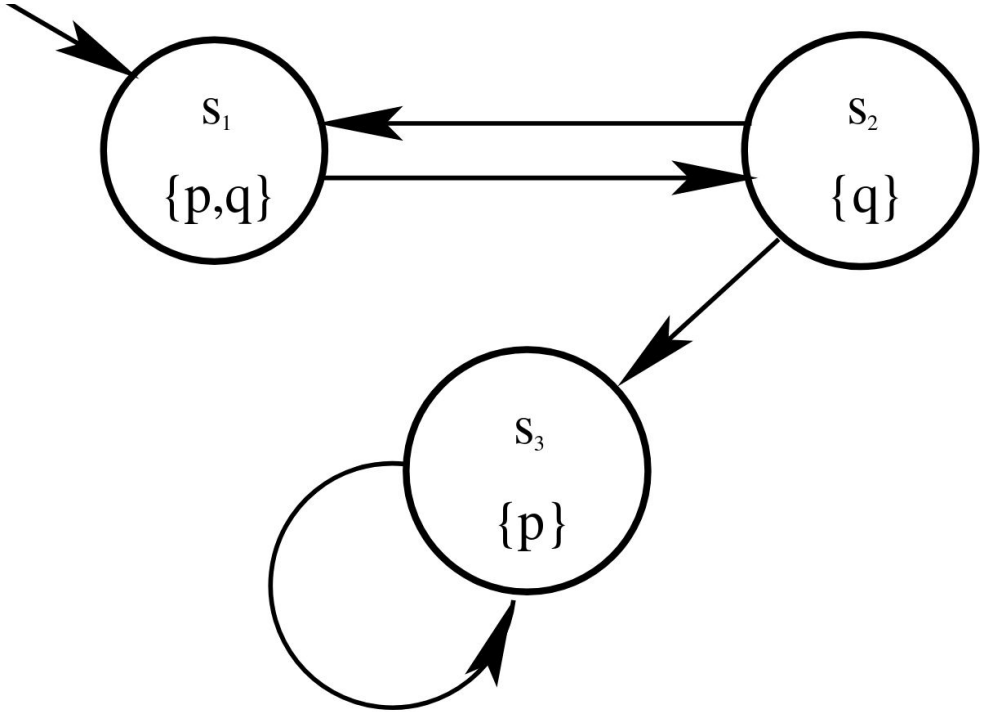


Modal Logic

Talks about *relational structures*
- People
- Program states
- Possible worlds
- …

◊, □ modalities see relations

How do we talk about points?



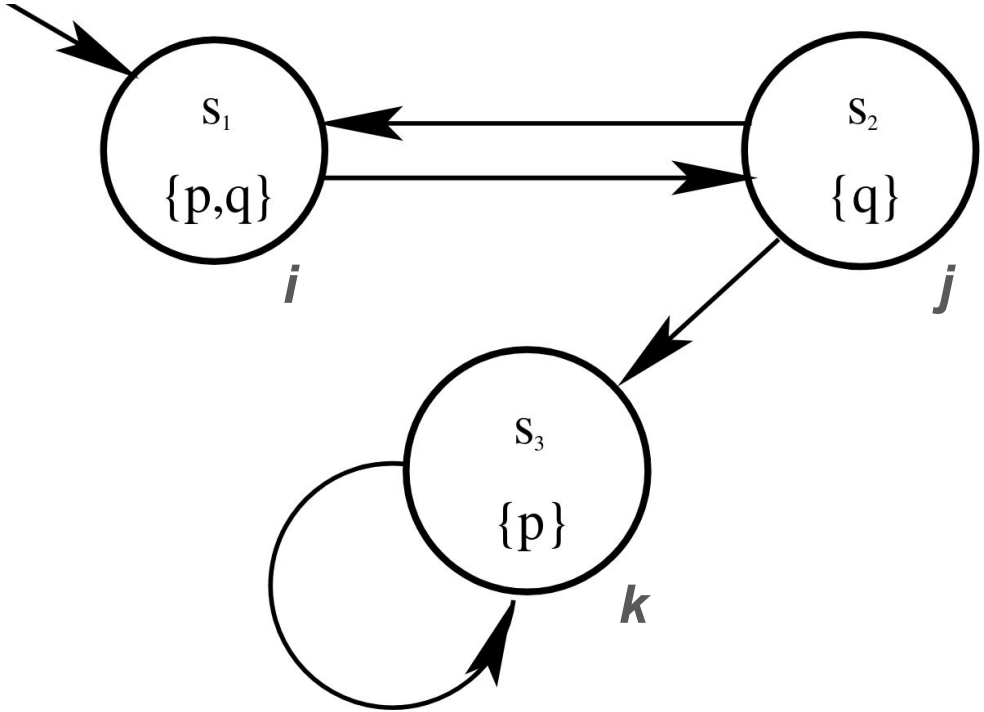
Hybrid Logic

Introduce second sort of propositional symbols

Nominals *i, j, k* name worlds

Satisfaction operator $@_k$ “jumps” to *k*

Now we can prove ◊φ by proving ◊i and $@_i$ φ for some nominal i (cf. ∃



Syntax and Semantics

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

Kripke model ((*W*, *R*), *V*) and assignment *g*.

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

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

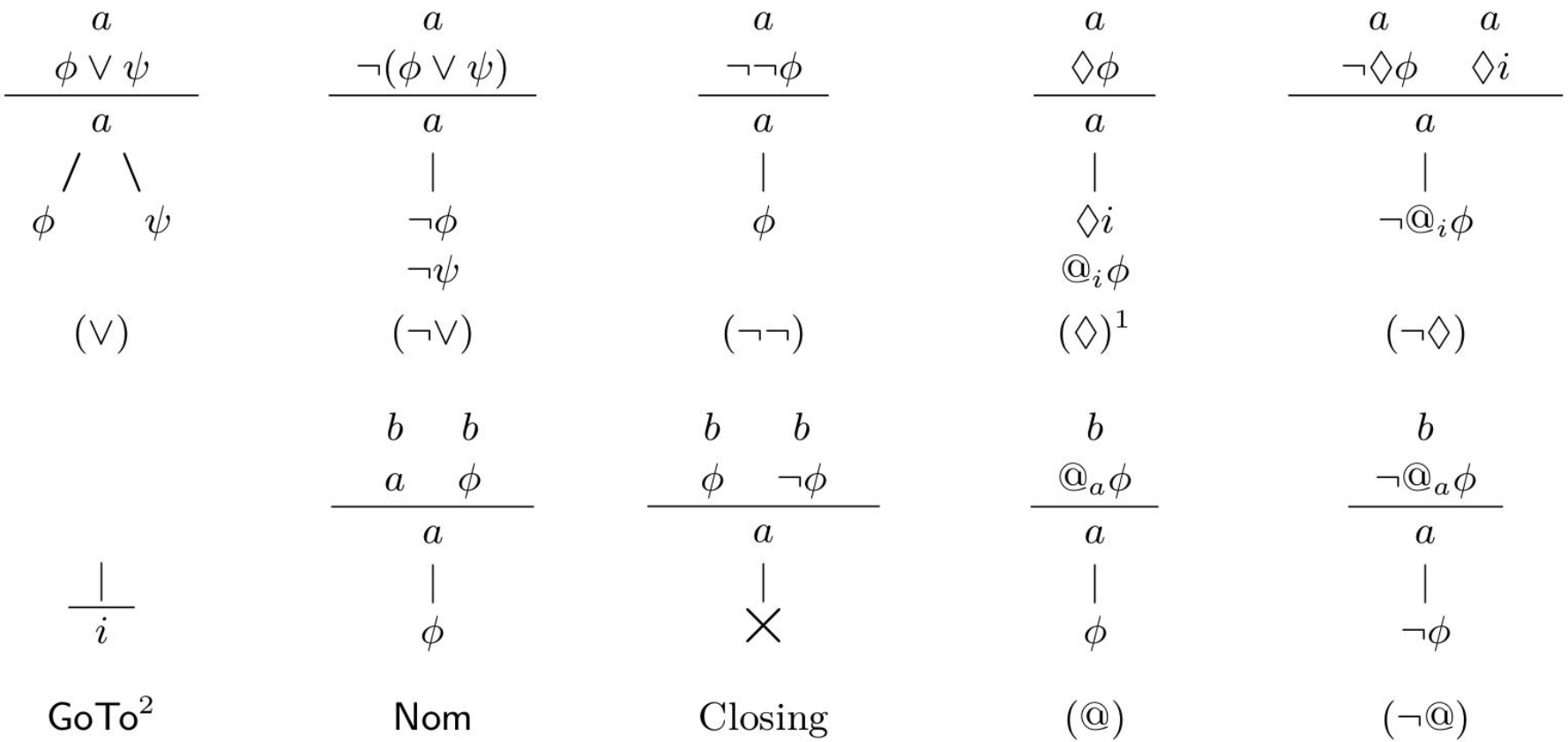
Seligman-Style Tableau

Divide branches into blocks

Each block starts with a nominal
The remaining formulas are true at (@) that nominal

Explicitly open a new block to switch world

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



In (◊) nominal a *generates* i

Restrictions

Don’t apply rules ad infinitum (hopefully)

Requirement: simple to formalize

S1 The output of a non-GoTo rule must include a formula new to the current block type.

S2 The ($\Diamond$) rule can only be applied to input $\Diamond\phi$ on an a-block if $\Diamond\phi$ is not already witnessed at a by formulas $\Diamond i$ and $@_i\phi$ for some witnessing nominal i.

S3 We associate *potential*, a natural number n, with each line in the tableau. GoTo must decrement the number, the other rules increment it and we may start from any amount.

S4 We parameterize the proof system by a fixed set of nominals A and impose the following:

- **a.** The nominal introduced by the ($\Diamond$) rule is not in A.
- **b.** For any nominal i, Nom only applies to a formula $\phi = i$ or $\phi = \Diamond i$ when $i \in A$.

Example Tableau

Start from refuted formula (1)
(at arbitrary nominal (0))

Derive conclusions (2-7)

Search for contradictions (6+7)

We see [potential] restrict **GoTo**

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

In Isabelle/HOL

Blocks are pointed lists, branches are lists of blocks:

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

Inductive predicate `A, n ⊢ b`: branch `b` closes under nominal set `A`, potential `n`

```
| Neg:
    ‹(¬ ¬ p) at a in (ps, a) # branch ⟹
    new p a ((ps, a) # branch) ⟹
    A, Suc n ⊢ (p # ps, a) # branch ⟹
    A, n ⊢ (ps, a) # branch›
```

Tableau Trees I

Need to model the *process* of building a tableau

```
codatatype (labels: 'a) tree = Node (getLabel: 'a) (getSubs: ‹'a tree list›)
```

These can have infinite paths:

```
coinductive ipath :: ‹'a tree ⇒ 'a stream ⇒ bool› where
  IPath: ‹s ∈ set subs ⟹ ipath s tail ⟹ ipath (Node l subs) (l ## tail)›
```

I use a small language of relational properties

```
datatype 'a rel
  = Here ‹'a ⇒ bool›
  | Next ‹'a rel›
  | Impl ‹'a ⇒ bool› ‹'a rel›
```

Tableau Trees II

Interpret the language on both trees and paths (streams):

```
relt :: ‹'a rel ⇒ 'a tree ⇒ bool›
rels :: ‹'a rel ⇒ 'a stream ⇒ bool›
```

For instance:

```
‹relt (Next r) t = (∀s ∈ set (getSubs t). relt r s)›
‹rels (Next r) = nxt (rels r)›
```

Relational properties are inherited by infinite paths:

```
lemma relt_ipath: assumes ‹relt r t› ‹ipath t s› shows ‹rels r s›
```

Tableau Trees III

Tag each kind of rule application:

```
datatype ('a, 'b) tag
  = Close 'b ‹('a, 'b) fm›
  | Open
  | Block 'b
  | Rule 'b ‹('a, 'b) fm list list›
  | Fresh 'b ‹('a, 'b) fm› 'b
```

A tableau is a tree of steps (tag, potential, branch prefix):

```
datatype ('a, 'b) step = Step
 (getTag: ‹('a, 'b) tag›) (getPotential: nat) (getBranch: ‹('a, 'b) branch›)

type_synonym ('a, 'b) tableau = ‹('a, 'b) step tree›
```

Tableau Trees IV

Only some of these trees are well formed:

```
coinductive wft :: ‹'b set ⇒ ('a, 'b) tableau ⇒ bool› for A :: ‹'b set›
where
  | WfNeg:
    ‹(¬ ¬ p) at a in (ps, a) # branch ⟹
    new p a ((ps, a) # branch) ⟹
    getBranch (getLabel s) = (p # ps, a) # branch ⟹
    getPotential (getLabel s) = Suc n ⟹
    wft A s ⟹
    wft A (Node (Step (Rule a [[p]]) n ((ps, a) # branch)) [s])›
 …
```

Essentially the Neg rule from the inductive predicate `A, n ⊢ b`.

Preliminary Results

An infinite path in a well formed tree must generate infinitely many nominals:

```
theorem infinite_generated:
  assumes ‹wft A tab› ‹ipath tab steps›
  shows ‹infinite (generated_ipath steps)›
```

Any fixed nominal (here `a`) only generates finitely many nominals:

```
theorem generated_bound:
  assumes ‹wft A tab› ‹ipath tab steps›
  shows ‹finite {(a, i) |p i. Fresh a p i ∈ getTag ` sset steps}›
```

Lots of work using the newness, (◊) restrictions etc.

Missing Results

Application of König's lemma:

existing results imply an infinite chain of nominals generating each other

Crucially, the measure technique of Bolander and Blackburn:

- **when *i* generates *j*, the formulas on *j*-blocks are smaller than those on *i*-blocks (due to restrictions on the copying Nom rule)**
- formula sizes cannot decrease infinitely
- so such a chain actually cannot exist
- and there can be no infinite paths in a well formed tableau

Trouble

Formalizing the key lemma of the measure technique in Isabelle/HOL

- **when *i* generates *j*, the formulas on *j*-blocks are smaller than those on *i*-blocks (due to restrictions on the copying Nom rule)**

Difficult to reconcile this *global* view of a tableau with the coinduction principle

If anyone has any ideas, pull requests are *very* welcome ;-)

https://github.com/astahfrom/hybrid_termination

References

Blackburn, P. (2000). Representation, reasoning, and relational structures: a hybrid logic manifesto. Logic Journal of the IGPL, 8(3), 339-365.

Bolander, T., & Blackburn, P. (2007). Termination for hybrid tableaus. Journal of Logic and Computation, 17(3), 517-554.

Jørgensen, K. F., Blackburn, P. R., Bolander, T., & Braüner, T. (2016). Synthetic Completeness Proofs for Seligman-style Tableau Systems. In L. Beklemishev, S. Demri, & A. Máté (Eds.), Proceedings of Advances in Modal Logic 2016 (Vol. 11, pp. 302-321). College Publications.

From, A. H. (2021). Synthetic Completeness for a Terminating Seligman-Style Tableau System. In 26th International Conference on Types for Proofs and Programs (TYPES 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

https://github.com/astahfrom/hybrid_termination