Three Lectures on Hybrid Logic

Patrick Blackburn

Section of Philosophy and Science Studies, Roskilde University, Denmark

Days in Logic 2018

25-27 January, Universidade de Aveiro, Aveiro, Portugal

Goals of the course

This course introduces hybrid logic, a form of modal logic in which it is possible to name worlds (or times, or computational states, or situations, or nodes in parse trees, or people - indeed, whatever it is that the elements of Kripke Models are taken to represent. The course has three major goals:

- The first is to convey the ideas and intuitions that have guided the development of hybrid logic.
- The second is to teach something about hybrid deduction and its completeness theory, and to make clear the crucial role played by the basic hybrid language and the Henkin construction.
- The third is to give you a glimpse of more powerful hybrid systems beyond the basic language, such as languages using the downarrow binder and explicit quantification over nominals.

Course Outline

- Lecture 1: From Modal to Hybrid Logic
- Lecture 2: Hybrid deduction
- Lecture 3: Stronger systems

Today: From Modal to Hybrid Logic

In today's lecture we discuss:

- Orthodox modal logic — from an Amsterdam perspective.
- A problem with orthodox modal logic.
- Fixing this problem with basic hybrid logic.
- Technical theme: bisimulations.
- Conceptual theme: What hybrid logic is, and why it is genuinely modal.

What is modal logic?

Slogan 1: Modal languages are simple yet expressive languages for talking about relational structures.

Slogan 2: Modal languages provide an internal, local perspective on relational structures.

Slogan 3: Modal languages are not isolated formal systems.

These slogans pretty much sum up the Amsterdam perspective on modal logic.

Propositional Modal Logic

Given propositional symbols PROP $= \{\mathrm{p}, \mathrm{q}, \mathrm{r}, \ldots\}$, and modality symbols MOD $= \{m, m', m'', \ldots\}$ the basic modal language (over PROP and MOD) is defined as follows:

$$\begin{aligned} \text{WFF} \quad := \quad & \top \mid \bot \mid \mathrm{p} \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \rightarrow \psi \\ & \mid \langle m \rangle \varphi \mid [m]\varphi \end{aligned}$$

If there's just one modality symbol in the language, we usually write $\Diamond$ and $\Box$ for its diamond and box forms.

$\Box\varphi$ can be regarded as shorthand for $\neg\Diamond\neg\varphi$. And $\Diamond$ can be regarded as shorthand for $\neg\Box\neg\varphi$.

Kripke Models

- A Kripke model $\mathcal{M}$ is a triple $(W, \mathcal{R}, V)$, where:
  - W is a non-empty set, whose elements can be thought of possible worlds, or epistemic states, or times, or states in a transition system, or geometrical points, or people standing in various relationships, or nodes in a parse tree — indeed, pretty much anything you like.
  - $\mathcal{R}$ is a collection of binary relation on W (one for each modality)
  - V is a valuation assigning subsets of W to propositional symbols.
- The component $(W, \mathcal{R})$ traditionally call a frame.

Satisfaction Definition

$$
\begin{array}{lll}
\mathcal{M}, w \Vdash \mathrm{p} & \text{iff} & w \in V(\mathrm{p}), \text{ where } \mathrm{p} \in \mathrm{PROP} \\
\mathcal{M}, w \Vdash \neg\varphi & \text{iff} & \mathcal{M}, w \nVdash \varphi \\
\mathcal{M}, w \Vdash \varphi \wedge \psi & \text{iff} & \mathcal{M}, w \Vdash \varphi \text{ and } \mathcal{M}, w \Vdash \psi \\
\mathcal{M}, w \Vdash \varphi \vee \psi & \text{iff} & \mathcal{M}, w \Vdash \varphi \text{ or } \mathcal{M}, w \Vdash \psi \\
\mathcal{M}, w \Vdash \varphi \rightarrow \psi & \text{iff} & \mathcal{M}, w \nVdash \varphi \text{ or } \mathcal{M}, w \Vdash \psi \\
\mathcal{M}, w \Vdash \langle m \rangle \varphi & \text{iff} & \exists w' (w R^m w' \ \& \ \mathcal{M}, w' \Vdash \varphi) \\
\mathcal{M}, w \Vdash [m] \varphi & \text{iff} & \forall w' (w R^m w' \ \Rightarrow \ \mathcal{M}, w' \Vdash \varphi).
\end{array}
$$

Note the internal perspective: we evaluate formulas inside models, at particular states. Modal formulas are like little creatures that explore models by moving between related points. This is a key modal intuition, gives rise to the notion of bisimulation, and is the driving force for at least one traditional application.

Tense logic

- $\langle \mathrm{F} \rangle$ means "at some Future state"
- $\langle \mathrm{P} \rangle$ means "at some Past state".
- $\langle \mathrm{P} \rangle \, \mathrm{mia} - \mathrm{unconscious}$ is true iff we can look back in time from the current state and see a state where Mia is unconscious. Works a bit like the sentence *Mia has been unconscious*.
- $\langle \mathrm{F} \rangle \, \mathrm{mia} - \mathrm{unconscious}$ requires us to scan the states that lie in the future looking for one where Mia is unconscious. Works a bit like the sentence *Mia will be unconscious*.

Feature logic

Consider the following Attribute Value Matrix (AVM):

$$\begin{bmatrix} \textsc{agreement} & \begin{bmatrix} \textsc{person} & \textit{1st} \\ \textsc{number} & \textit{singular} \end{bmatrix} \\ \textsc{case} & -\textit{dative} \end{bmatrix}$$

Feature logic

Consider the following Attribute Value Matrix (AVM):

$$\begin{bmatrix} \textsc{agreement} & \begin{bmatrix} \textsc{person} & \textit{1st} \\ \textsc{number} & \textit{singular} \end{bmatrix} \\ \textsc{case} & -\textit{dative} \end{bmatrix}$$

This is a two-dimensional notational variant of the following modal formula:

$$\begin{aligned} & \langle \textsc{agreement} \rangle \left(\langle \textsc{person} \rangle \, \mathrm{1st} \wedge \langle \textsc{number} \rangle \, \mathrm{singular} \right) \\ \wedge \quad & \langle \textsc{case} \rangle \, \neg \mathrm{dative} \end{aligned}$$

Description logic

And, moving into the heart of ordinary *extensional* logic, consider the following $\mathcal{ALC}$ term:

$$\text{killer} \sqcap \exists \text{EMPLOYER}.\text{gangster}$$

Description logic

And, moving into the heart of ordinary *extensional* logic, consider the following $\mathcal{ALC}$ term:

$$\text{killer} \sqcap \exists \text{EMPLOYER}.\text{gangster}$$

This means exactly the same thing as the modal formula:

$$\text{killer} \wedge \langle \text{EMPLOYER} \rangle \text{gangster}$$

But there's lots of other ways of talking about graphs

- There's nothing magic about frames or Kripke models.
- Frames $(W, \mathcal{R})$, are just a directed multigraphs (or labelled transition systems).
- Valuations simply decorate states with properties.
- So a Kripke model for the basic modal language are just (very simple) relational structures in the usual sense of first-order model theory.
- So we don't have to talk about Kripke models using modal logic — we could use first-order logic, or second-order logic, or infinitary logic, or fix-point logic, or indeed any logic interpreted over relational structures.
- Let's see how...

First-order logic for Kripke models

Suppose we have a Kripke model $(W, \mathcal{R}, V)$, for the modal language over MOD and PROP. We talk about this model in first-order logic by making use of the first-order language built from the following symbols:

- For each propositional symbol p it has a unary predicate symbol P. We'll use V to interpret these predicate symbols.
- For each modality $\langle \mathrm{R} \rangle$, it has a binary relation symbol R. We'll use the binary relations in $\mathcal{R}$ to interpret these symbols.

The first-order language built over these symbols is called the first-order correspondence language (for the modal language over MOD and PROP).

Doing it first-order style (I)

Consider the modal representation

$$\langle \mathrm{F} \rangle \, \mathrm{mia} - \mathrm{unconscious}$$

Doing it first-order style (I)

Consider the modal representation

$$\langle \mathrm{F} \rangle \, \mathrm{mia} - \mathrm{unconscious}$$

we could use instead the first-order representation

$$\exists t(t_o < t \wedge \mathrm{MIA} - \mathrm{UNCONSCIOUS}(t)).$$

Doing it first-order style (II)

And consider the modal representation

$$\text{killer} \wedge \langle \text{EMPLOYER} \rangle \text{gangster}$$

Doing it first-order style (II)

And consider the modal representation

$$\text{killer} \wedge \langle \text{EMPLOYER} \rangle \text{gangster}$$

We could use instead the first-order representation

$$\text{KILLER}(x) \wedge \exists y(\text{EMPLOYER}(x, y) \wedge \text{GANGSTER}(y))$$

Standard Translation

And in fact, any modal representation can by converted into an equi-satisfiable first-order representation:

$$\begin{array}{lcl} \mathrm{ST}_x(\mathrm{p}) & = & \mathrm{P}x \\ \mathrm{ST}_x(\neg\varphi) & = & \neg\,\mathrm{ST}_x(\varphi) \\ \mathrm{ST}_x(\varphi \wedge \psi) & = & \mathrm{ST}_x(\varphi) \wedge \mathrm{ST}_x(\psi) \\ \mathrm{ST}_x(\langle R\rangle\varphi) & = & \exists y(Rxy \wedge \mathrm{ST}_y(\varphi)) \end{array}$$

Note that $\mathrm{ST}_x(\varphi)$ always contains exactly one free variable (namely x).

Proposition: For any modal formula φ, any Kripke model $\mathcal{M}$, and any state w in $\mathcal{M}$ we have that: $\mathcal{M}, w \Vdash \varphi$ iff $\mathcal{M} \models \mathrm{ST}_x(\varphi)[x \leftarrow w]$.

So aren't we better off with first-order logic . . . ?

- We've just seen that any modal formula can be systematically converted into an equi-satisfiable first-order formula.
- And as we'll later see, the reverse is not possible: first-order logic can describe models in far more detail that modal logic can. Some first-order formulas have no modal equivalent. That is, modal languages are weaker than their corresponding first-order languages.
- So why bother with modal logic?

Reasons for going modal

Reasons for going modal

- **Simplicity**. The standard translation shows us that modalities are essentially 'macros' encoding a quantification over related states. Modal notation hides the bound variables, resulting in a compact, easy to read, representations.

Reasons for going modal

- **Simplicity**. The standard translation shows us that modalities are essentially 'macros' encoding a quantification over related states. Modal notation hides the bound variables, resulting in a compact, easy to read, representations.
- **Computability**. First-order logic is undecidable over arbitrary models. Modal logic is decidable over arbitrary models (indeed, decidable in PSPACE). Modal logic trades expressivity for computability.

Reasons for going modal

- **Simplicity**. The standard translation shows us that modalities are essentially 'macros' encoding a quantification over related states. Modal notation hides the bound variables, resulting in a compact, easy to read, representations.
- **Computability**. First-order logic is undecidable over arbitrary models. Modal logic is decidable over arbitrary models (indeed, decidable in PSPACE). Modal logic trades expressivity for computability.
- **Internal perspective**. A natural way of thinking about models. And taken seriously, leads to an elegant characterization of what modal logic can say about models. Let's take a closer look. . .

Bisimulation (I)

Bisimulation (I)

The fundamental notion of equivalence between states for modal logic.

Bisimulations are used in other disciplines besides modal logic. Its role in all of them is to provide an appropriate notion of equivalence.

Social Network Theory Here they capture the notion of two social networks being functionally identical, even though they are not isomorphic. It's the configurations in which the agents stand in various roles that render two social networks "the same".

Theoretical Computer Science Here they embody the notion of behavioural equivalence for processes.

Non-well-founded Set Theory Here they replace the extensionality as the criterion of equality: two non-well-founded sets (graphs) are equal iff they are bisimilar.

Bisimulation (II)

Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic modal language. A relation $Z \subseteq W \times W'$ is a bisimulation between $\mathcal{M}$ and $\mathcal{M}'$ if the following conditions are met:

Bisimulation (II)

Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic modal language. A relation $Z \subseteq W \times W'$ is a bisimulation between $\mathcal{M}$ and $\mathcal{M}'$ if the following conditions are met:

1. Atomic harmony: if wZw' then $w \in V(p)$ iff $w' \in V'(p)$, for all propositional symbols p.

Bisimulation (II)

Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic modal language. A relation $Z \subseteq W \times W'$ is a bisimulation between $\mathcal{M}$ and $\mathcal{M}'$ if the following conditions are met:

1. Atomic harmony: if wZw' then $w \in V(p)$ iff $w' \in V'(p)$, for all propositional symbols p.
2. Forth: if wZw' and wRv then there is a v' such that $w'R'v'$ and vZv'.

Bisimulation (II)

Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic modal language. A relation $Z \subseteq W \times W'$ is a bisimulation between $\mathcal{M}$ and $\mathcal{M}'$ if the following conditions are met:

1. Atomic harmony: if wZw' then $w \in V(p)$ iff $w' \in V'(p)$, for all propositional symbols p.
2. Forth: if wZw' and wRv then there is a v' such that $w'R'v'$ and vZv'.
3. Back: if wZw' and $w'R'v'$ then there is a v such that wRv and vZv'.

Modal formulas are invariant under bisimulation

Proposition: Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic modal language, and let Z be a bisimulation between $\mathcal{M}$ and $\mathcal{M}'$. Then for all modal formulas φ, and all points w in $\mathcal{M}$ and w' in $\mathcal{M}$ such that w is bisimilar to w':

$$\mathcal{M}, w \Vdash \varphi \text{ iff } \mathcal{M}', w' \Vdash \varphi.$$

In words: bisimilar points are modally equivalent, or to put it another way: modal formulas are invariant under bisimulations.

Proof: Induction on the structure of φ.

Not all first-order formulas are bisimulation invariant

- A first-order formula in one free variable $\varphi(x)$ is bisimulation-invariant if for all bisimulations Z between models $\mathcal{M}$ and $\mathcal{M}'$, if wZw' then $\mathcal{M} \models \varphi[w]$ iff $\mathcal{M}' \models \varphi[w']$.
- Not all first-order formulas are bisimulation invariant (which shows that not all first-order formulas can be translated into modal formulas).
- But bisimulation invariance seems to be a natural property in various domains, so it is natural to ask: precisely which first-order formulas are bisimulation invariant? The answer is elegant ...

The van Benthem Characterization Theorem

For all first-order formulas φ (in the correspondence language) containing exactly one free variable, φ is bisimulation-invariant iff φ is equivalent to the standard translation of a modal formula.
In short, modal logic is a simple notation for capturing exactly the bisimulation-invariant fragment of first-order logic.

Proof:
($\Rightarrow$) Immediate from the invariance of modal formula under bisimulation.
($\Leftarrow$) Non-trivial (usually proved using elementary chains or by appealing to the existence of saturated models).

Back to slogan 3

Slogan 3: Modal languages are not isolated formal systems.

Modal languages over models are essentially simple fragments of first-order logic. These fragments have a number of attractive properties such as robust decidability and bisimulation invariance. Traditional modal notation is essentially a nice (quantifier free) 'macro' notation for working with this fragment.

Back to slogan 2

Slogan 2: Modal languages provide an internal, local perspective on relational structures.

This is not just an intuition: the notion of bisimulation, and the results associated with it, shows that this is the key model theoretic fact at work in modal logic.

Back to slogan 1

Slogan 1: Modal languages are simple yet expressive languages for talking about relational structures.

You can use modal logic for just about anything. Anywhere you see a graph, you can use a modal language to talk about it.

That was the good news — now comes the bad

Orthodox modal languages have an obvious drawback for many applications: they don't let us refer to individual states (worlds, times, situations, nodes, ...). That is, they don't allow us to say things like

- this happened *there*; or
- this happened *then*; or
- *this* state has property φ; or
- node *peter's-printer* is marked *out of paper*.

and so on.

Temporal logic

- Temporal representations in Artificial Intelligence (such as Allen's system, and the situation calculus) based around temporal reference, and use some of **Holds**(e, t) operator.
- Worse, standard modal logics of time are completely inadequate for the temporal semantics of natural language. *Vincent accidentally squeezed the trigger* doesn't mean that at some completely unspecified past time Vincent did in fact accidentally squeeze the trigger, it means that at some *particular*, contextually determined, past time he did so. The representation, $\langle \mathrm{P} \rangle$ *vincent* $-$ *accidentally* $-$ *squeeze* $-$ *trigger* fails to capture this.

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

The states described by the first two sentences hold at the same time. The event described by the second takes place a little later. In orthodox modal logics there is no way assert the identity of the times needed for the first two sentences, nor to capture the move forward in time needed by the third.

In fact, for this reason modal languages for temporal representation have not been the tool of choice in natural language semantics for over 25 years.

Feature logic

As we've mentioned, the following Attribute Value Matrix (AVM):

$$\begin{bmatrix} \textsc{agreement} & \begin{bmatrix} \textsc{person} & \textit{1st} \\ \textsc{number} & \textit{plural} \end{bmatrix} \\ \textsc{case} & -\textit{dative} \end{bmatrix}$$

is a notational variant of the following modal formula:

$$\begin{aligned} & \langle \textsc{agreement} \rangle \, (\langle \textsc{person} \rangle \, \text{1st} \wedge \langle \textsc{number} \rangle \, \text{plural}) \\ \wedge \quad & \langle \textsc{case} \rangle \, \neg \text{dative} \end{aligned}$$

Feature logic

But full AVM notation is richer. It can assert re-entrancies:

$$\begin{bmatrix} \textsc{subj} & \boxed{1} \begin{bmatrix} \textsc{agr} & \textit{foo} \\ \textsc{pred} & \textit{bar} \end{bmatrix} \\ \textsc{comp} & [\textsc{subj} \ \boxed{1}\ ] \end{bmatrix}$$

This cannot be captured in orthodox modal logic.

Description logic I

As we have already said, there is a transparent correspondence between simple DL terms and modal formulas:

$$\text{killer} \sqcap \exists \text{EMPLOYER}.\text{gangster}$$

$$\text{killer} \wedge \langle \text{EMPLOYER} \rangle \text{gangster}$$

Nonetheless, this correspondence only involves what description logicians call the TBox (Terminological Box).

Description logic II

Orthodox modal logic does not have anything to say about the ABox (Assertional Box):

$$\text{mia} : \text{Beautiful}$$

$$(\text{jules}, \text{vincent}) : \text{Friends}$$

That is, it can't make assertions about individuals, for it has no tools for naming individuals.

Ambition

- We want to be able to refer to states, but want to do so without destroying the simplicity of propositional modal logic.
- But how can we do this — propositional modal logic has very few moving parts?
- Answer: sort the atomic symbols. Use formulas as terms.
- This will fix the obvious shortcoming — and as we shall learn, it will fix a lot more besides.

Extension #1

- Take a language of basic modal logic (with propositional variables p, q, r, and so on) and add a second sort of atomic formula.
- The new atoms are called nominals, and are typically written i, j, k, and l.
- Both types of atom can be freely combined to form more complex formulas in the usual way; for example,

$$\Diamond(i \wedge \mathrm{p}) \wedge \Diamond(i \wedge \mathrm{q}) \rightarrow \Diamond(\mathrm{p} \wedge \mathrm{q})$$

  is a well formed formula.
- Insist that each nominal be true at exactly one world in any model. A nominal names a state by being true there and nowhere else.

We already have a richer logic

Consider the orthodox formula

$$\Diamond(\mathrm{r} \wedge \mathrm{p}) \wedge \Diamond(\mathrm{r} \wedge \mathrm{q}) \rightarrow \Diamond(\mathrm{p} \wedge \mathrm{q})$$

This is easy to falsify.

We already have a richer logic

Consider the orthodox formula

$$\Diamond(\mathrm{r} \wedge \mathrm{p}) \wedge \Diamond(\mathrm{r} \wedge \mathrm{q}) \rightarrow \Diamond(\mathrm{p} \wedge \mathrm{q})$$

This is easy to falsify.

On the other hand, the hybrid formula

$$\Diamond(i \wedge \mathrm{p}) \wedge \Diamond(i \wedge \mathrm{q}) \rightarrow \Diamond(\mathrm{p} \wedge \mathrm{q})$$

is valid (unfalsifiable). Nominals name, and this adds to the expressive power at our disposal.

Extension #2

- Add formulas of form $@_i\varphi$.
- Such formulas assert that φ is satisfied at the point named by the nominal i.
- Expressions of the form $@_i$ are called satisfaction operators.

Let's make these ideas precise ...

Syntax

- Given ordinary propositional symbols PROP $= \{\mathrm{p}, \mathrm{q}, \mathrm{r}, \ldots\}$, and modalities MOD, let NOM $= \{i, j, k, l, \ldots\}$ be a nonempty set disjoint from PROP.
- The elements of NOM are called nominals; they are second sort of atomic symbol which will be used to name states. g The basic hybrid language (over PROP, MOD and NOM) is defined as follows:

$$\begin{aligned} \mathrm{WFF} \quad := \quad & i \mid p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \\ & \mid \varphi \rightarrow \psi \mid \langle \mathrm{M} \rangle \varphi \mid [\mathrm{M}] \varphi \mid @_i \varphi \end{aligned}$$

Semantics

- As before, a model $\mathcal{M}$ is a triple $(W, \mathcal{R}, V)$.
- As before, $(W, \mathcal{R})$ is just a frame (a labelled transition system).
- The difference lies in V. Now a valuation V is a function with domain PROP$\cup$NOM and range $Pow(W)$ such that for all $i \in$ NOM, $V(i)$ is a singleton subset of W.
- That is, a valuation makes each nominal true at a unique state; the nominal labels this state by being true there and nowhere else.
- We call the unique state w that belongs to $V(i)$ the denotation of i under V.

Satisfaction Definition

$\mathcal{M}, w \Vdash a$	iff	$w \in V(a)$, where $a \in \mathrm{PROP} \cup \mathrm{NOM}$
$\mathcal{M}, w \Vdash \neg\varphi$	iff	$\mathcal{M}, w \nVdash \varphi$
$\mathcal{M}, w \Vdash \varphi \wedge \psi$	iff	$\mathcal{M}, w \Vdash \varphi$ and $\mathcal{M}, w \Vdash \psi$
$\mathcal{M}, w \Vdash \varphi \vee \psi$	iff	$\mathcal{M}, w \Vdash \varphi$ or $\mathcal{M}, w \Vdash \psi$
$\mathcal{M}, w \Vdash \varphi \rightarrow \psi$	iff	$\mathcal{M}, w \nVdash \varphi$ or $\mathcal{M}, w \Vdash \psi$
$\mathcal{M}, w \Vdash \langle \mathrm{M} \rangle \varphi$	iff	$\exists w'(wR^m w' \ \& \ \mathcal{M}, w' \Vdash \varphi)$
$\mathcal{M}, w \Vdash [\mathrm{M}] \varphi$	iff	$\forall w'(wR^m w' \ \Rightarrow \ \mathcal{M}, w' \Vdash \varphi)$.
$\mathcal{M}, w \Vdash @_i\varphi$	iff	$\mathcal{M}, \mathrm{i} \Vdash \varphi$, where i is the denotation of i under V.

Tense logic

- On the road to capturing AI temporal representation formalisms such as Allen's logic of temporal reference; @ can play the role of **Holds**.
- And we can now handle natural language examples more convincingly:
  $\langle \mathrm{P} \rangle (i \wedge \mathit{Vincent} - \mathit{accidentally} - \mathit{squeeze} - \mathit{the} - \mathit{trigger})$
  locates the trigger-squeezing not merely in the past, but at a specific temporal state there, namely the one named by i — better modelling the meaning of *Vincent accidentally squeezed the trigger*. Let's take this a little further...

Reichenbach in hybrid logic

Structure	Name	English example	Representation
E–R–S	Pluperfect	I had seen	$\langle \mathrm{P} \rangle (i \wedge \langle \mathrm{P} \rangle \phi)$
E,R–S	Past	I saw	$\langle \mathrm{P} \rangle (i \wedge \phi)$
R–E–S	Future-in-the-past	I would see	$\langle \mathrm{P} \rangle (i \wedge \langle \mathrm{F} \rangle \phi)$
R–S,E	Future-in-the-past	I would see	$\langle \mathrm{P} \rangle (i \wedge \langle \mathrm{F} \rangle \phi)$
R–S–E	Future-in-the-past	I would see	$\langle \mathrm{P} \rangle (i \wedge \langle \mathrm{F} \rangle \phi)$
E–S,R	Perfect	I have seen	$\langle \mathrm{P} \rangle \phi$
S,R,E	Present	I see	ϕ
S,R–E	Prospective	I am going to see	$\langle \mathrm{F} \rangle \phi$
S–E–R	Future perfect	I will have seen	$\langle \mathrm{F} \rangle (i \wedge \langle \mathrm{P} \rangle \phi)$
S,E–R	Future perfect	I will have seen	$\langle \mathrm{F} \rangle (i \wedge \langle \mathrm{P} \rangle \phi)$
E–S–R	Future perfect	I will have seen	$\langle \mathrm{F} \rangle (i \wedge \langle \mathrm{P} \rangle \phi)$
S–R,E	Future	I will see	$\langle \mathrm{F} \rangle (i \wedge \phi)$
S–R–E	Future-in-the-future	(Latin: abiturus ero)	$\langle \mathrm{F} \rangle (i \wedge \langle \mathrm{F} \rangle \phi)$

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

$$P(i \wedge \text{vincent-wake-up})$$

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

$$P(i \wedge \text{vincent-wake-up})$$

$$\wedge \quad P(j \wedge \text{something-feel-very-wrong})$$

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

$$P(i \wedge \text{vincent-wake-up})$$

$$\wedge \quad P(j \wedge \text{something-feel-very-wrong}) \qquad \wedge \quad @_j i$$

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

$$P(i \wedge \text{vincent-wake-up})$$

$$\wedge \quad P(j \wedge \text{something-feel-very-wrong}) \quad \wedge \quad @_j i$$

$$\wedge \quad P(k \wedge \text{vincent-reach-under-pillow-for-uzi})$$

Tense in text

Vincent woke up. Something felt very wrong. Vincent reached under his pillow for his Uzi.

$$
\begin{array}{lll}
 & P(i \wedge \text{vincent-wake-up}) & \\
\wedge & P(j \wedge \text{something-feel-very-wrong}) & \wedge \ @_j i \\
\wedge & P(k \wedge \text{vincent-reach-under-pillow-for-uzi}) & \wedge \ @_k Pi
\end{array}
$$

Feature logic

$$\begin{bmatrix} \textsc{subj} & \boxed{1} \begin{bmatrix} \textsc{agr} & \mathit{foo} \\ \textsc{pred} & \mathit{bar} \end{bmatrix} \\ \textsc{comp} & [\textsc{subj} \ \boxed{1}\ ] \end{bmatrix}$$

This corresponds to the following hybrid wff:

$$\begin{aligned} & \langle \textsc{subj} \rangle \, (i \wedge \langle \textsc{agr} \rangle \, \mathrm{foo} \wedge \langle \textsc{pred} \rangle \, \rangle \mathrm{bar}) \\ \wedge \quad & \langle \textsc{comp} \rangle \, \langle \textsc{subj} \rangle \, i \end{aligned}$$

Description logic (I)

We can now make ABox statements. For example, to capture the effect of the (conceptual) ABox assertion

$$\mathsf{mia} : \mathsf{Beautiful}$$

we can write

$$@_{\mathsf{mia}}\mathsf{Beautiful}$$

Description logic (II)

Similarly, to capture the effect of the (relational) ABox assertion

$$(\text{jules}, \text{vincent}) : \text{Friends}$$

we can write

$$@_{\text{jules}} \langle \text{Friends} \rangle \text{vincent}$$

Basic hybrid language clearly modal

Neither syntactical nor computational simplicity, nor general 'style' of modal logic, has been compromised.

- Nominals just atomic formulas.
- Satisfaction operators are normal modal operators. That is, for any nominal i we have that:
  - $@_i(\varphi \to \psi) \to (@_i\varphi \to @_i\psi)$ is valid.
  - If φ is valid, then so is $@_i\varphi$.
- Indeed, satisfaction operators are even self-dual modal operators: $@_i\phi$ and $\neg@_i\neg\phi$ say exactly the same thing.

Basic hybrid logic is computable

Enriching ordinary propositional modal logic with both nominals and satisfaction operators does not effect computability. The basic hybrid logic is decidable. Indeed we even have:

Theorem: The satisfiability problem for basic hybrid languages over arbitrary models is PSPACE-complete. (Areces, Blackburn, and Marx).

That is (up to a polynomial) the hybridized language has the same complexity as the orthodox modal language we started with.

Standard Translation

Any basic hybrid formula can by converted into an equi-satisfiable first-order formula. All we have to do is add a first-order constant (or variable) i for each nominal i and translate as follows (note the use of equality):

$$\begin{array}{lcl}
\mathrm{ST}_x(p) & = & Px \\
\mathrm{ST}_x(i) & = & (i = x) \\
\mathrm{ST}_x(\neg\varphi) & = & \neg\,\mathrm{ST}_x(\varphi) \\
\mathrm{ST}_x(\varphi \wedge \psi) & = & \mathrm{ST}_x(\varphi) \wedge \mathrm{ST}_x(\psi) \\
\mathrm{ST}_x(\langle R\rangle\varphi) & = & \exists y(Rxy \wedge \mathrm{ST}_y(\varphi)) \\
\mathrm{ST}_x(@_i\varphi) & = & \mathrm{ST}_i(\varphi)
\end{array}$$

Note that $\mathrm{ST}_x(\varphi)$ always contains at most free variable (namely x).

Proposition: For any basic hybrid formula φ, any Kripke model $\mathcal{M}$, and any state w in $\mathcal{M}$ we have that: $\mathcal{M}, w \Vdash \varphi$ iff $\mathcal{M} \models \mathrm{ST}_x(\varphi)[x \leftarrow w]$.

Basic hybrid logic can specify Robinson Diagrams

Basic hybrid logic can specify Robinson Diagrams

- $@_i p$ says that the states labelled i bears the information p, while $\neg @_i p$ denies this. That is, we can specify how atomic properties are distributed modally.

Basic hybrid logic can specify Robinson Diagrams

- $@_i p$ says that the states labelled i bears the information p, while $\neg @_i p$ denies this. That is, we can specify how atomic properties are distributed modally.
- $@_i j$ says that the states labelled i and j are identical, while $\neg @_i j$ says they are distinct. That is, we can specify theories of state equality modally.

Basic hybrid logic can specify Robinson Diagrams

- $@_i p$ says that the states labelled i bears the information p, while $\neg @_i p$ denies this. That is, we can specify how atomic properties are distributed modally.
- $@_i j$ says that the states labelled i and j are identical, while $\neg @_i j$ says they are distinct. That is, we can specify theories of state equality modally.
- $@_i \Diamond j$ says that the state labelled j is a successor of the state labelled i, and $\neg @_i \Diamond j$ denies this. That is, we can specify theories of state succession modally.

Basic hybrid logic can specify Robinson Diagrams

- $@_i p$ says that the states labelled i bears the information p, while $\neg @_i p$ denies this. That is, we can specify how atomic properties are distributed modally.
- $@_i j$ says that the states labelled i and j are identical, while $\neg @_i j$ says they are distinct. That is, we can specify theories of state equality modally.
- $@_i \Diamond j$ says that the state labelled j is a successor of the state labelled i, and $\neg @_i \Diamond j$ denies this. That is, we can specify theories of state succession modally.

That is, we have all the tools needed to completely describe models (that is, what model theorists call Robinson diagrams). This makes life very straightforward when it comes to proving completeness and interpolation results.

But what *is* basic hybrid logic?

We have seen many examples of what basic hybrid logic can do in various applications.

We've also seen that a number of the properties we liked about modal logic are inherited by the basic hybrid language.

This is all very nice — but none of it gives us a clear mathematical characterization of what basic hybrid logic actually is.

And it is possible to give such a characterization, and a genuinely modal one at that. Let's take a look ...

Bisimulation-with-constants

Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic hybrid language. A relation $Z \subseteq W \times W'$ is a bisimulation-with-constants between $\mathcal{M}$ and $\mathcal{M}'$ if the following conditions are met:

Bisimulation-with-constants

Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic hybrid language. A relation $Z \subseteq W \times W'$ is a bisimulation-with-constants between $\mathcal{M}$ and $\mathcal{M}'$ if the following conditions are met:

1. Atomic harmony: if wZw' then $w \in V(p)$ iff $w' \in V'(p)$, for all propositional symbols p, and all nominals i.
2. Forth: if wZw' and wRv then there is a v' such that $w'R'v'$ and vZv'.
3. Back: if wZw' and $w'R'v'$ then there is a v such that wRv and vZv'.
4. All points named by nominals are related by Z.

Basic hybrid formulas are invariant under bisimulations-with-constants

Proposition: Let $\mathcal{M} = (W, \mathcal{R}, V)$ and $\mathcal{M}' = (W', \mathcal{R}', V')$ be models for the same basic hybrid language, and let Z be a bisimulation-with-constants between $\mathcal{M}$ and $\mathcal{M}'$. Then for all basic hybrid formulas φ, and all points w in $\mathcal{M}$ and w' in $\mathcal{M}$ such that w is bisimilar to w':

$$\mathcal{M}, w \Vdash \varphi \text{ iff } \mathcal{M}', w' \Vdash \varphi.$$

Proof: Induction on the structure of φ.

Lifting the van Benthem Characterization theorem

For all first-order formulas φ (in the correspondence language with constants and equality) containing at most one free variable, φ is bisimulation-with-constants invariant iff φ is equivalent to the standard translation of a basic hybrid formula iff (Areces, Blackburn, ten Cate, and Marx)

In short, basic hybrid logic is a simple notation for capturing exactly the bisimulation-invariant fragment of first-order logic when we make use of constants and equality.

Proof:
($\Rightarrow$) Immediate from the invariance of hybrid formulas under bisimulation.
($\Leftarrow$) Can be proved using elementary chains or by appealing to the existence of saturated models.

Summing up . . .

- We learned about some of the good points of orthodox modal logic, but also saw that it's inability to refer to states is a weakness for various applications.
- We saw that adding nominals and satisfaction operators fixes these weaknesses without sacrificing what we liked about modal logic in the first place. Basic hybrid logic is a natural generalization of orthodox modal logic.

Summing up . . .

- We learned about some of the good points of orthodox modal logic, but also saw that it's inability to refer to states is a weakness for various applications.
- We saw that adding nominals and satisfaction operators fixes these weaknesses without sacrificing what we liked about modal logic in the first place. Basic hybrid logic is a natural generalization of orthodox modal logic.
- But as we shall soon learn, hybridization has fixed some less obvious shortcomings of orthodox modal logic too. In particular, it has given us a logical formalism that is is easy to use deductively — as we shall see tomorrow