Hybrid Logics

Carlos Areces
carlos.areces@gmail.com

Spring Term 2018
Stanford

What we want to cover

- Review the definition of the basic hybrid logics $\mathcal{H}(@)$.
- Talk a bit about decidability / complexity.
- We will define a tableax algorithm for satisfiability of formulas in the basic modal logic.
  - How do we check whether a formula has a model?
  - What can we learn from tableaux?
- Transform the tableau for the basic modal logic into one for $\mathcal{H}(@)$.

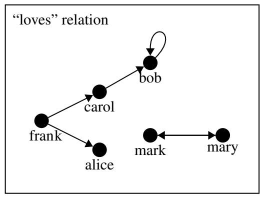
Relevant Bibliography

- Blackburn, P., de Rijke, M. and Venema, Y. Chapter 7, Section 3 of “Modal Logic”, Cambridge Tracts in Theoretical Computer Science, 53, Cambridge University Press, 2001.
- Blackburn, P. Representation, Reasoning, and Relational Structures: a Hybrid Logic Manifesto. Logic Journal of the IGPL, 8(3), 339–625, 2000.
- Areces, C. and ten Cate, B.. Hybrid Logics. In Blackburn, P., Wolter, F., and van Benthem, J., editors, Handbook of Modal Logics, 821–868, Elsevier, 2006.
- Article on “Hybrid Logics” at the Stanford Encycopedia of Philosophy, `http://plato.stanford.edu/entries/logic-hybrid`
- Hybrid Logics Web Page, `http://hylo.loria.fr`

Modal Logics

Syntax: Propositional Logic + modalities

Semantics: Interpreted in terms of relational structures (Graphs)



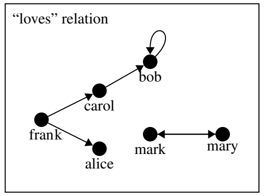
Query: “Does somebody loves a loner?”
(is it true somewhere in the graph?)

The Modal Way: $\varphi := \langle loves \rangle \neg \langle loves \rangle \top$

- MLs can be (usually) thought of as fragments of FO
- Modal logics are (usually) decidable
  - SAT for the basic modal logic is PSpace-complete

The Limits of Modal Expressivity

Some properties can't be expressed in the basic modal language...



Query: "Does Frank love Alice?"

Query: "Is there somebody who loves himself/herself?"

Query: "Are there two people who loves each other?"

- What do we need?
  - constants
  - identity

The Limits of Modal Expressivity

- There is an asymmetry at the heart of modal logic: although states are crucial to modal semantics, nothing in modal syntax can talk about them. Modal logics has no mechanism for referring to or reasoning about the individual states in the structure.
- This leads to two kinds of problem:
  - A representation problem: for some applications modal logic is not adequate as a representation formalism, and
  - A reasoning problem: modal reasoning systems are difficult to devise.
- These limitations motivated the work on Hybrid Logics.

The Basic Recipe: $\mathcal{H}(@)$

basic modal logic			
+	nominals	$\rightarrow$	a new atomic sort
+	@	$\rightarrow$	the 'at' operator
	$\mathcal{H}(@)$	$\rightarrow$	the basic hybrid logic

- Nominals denote elements (nodes) in the model
- $@_i\varphi$ is true iff φ is true in the element denoted by i.
  In particular $@_i j$ says that i and j denote the same point in the model (i.e., $i = j$).

The Hybrid Logic $\mathcal{H}(@)$

Syntax:

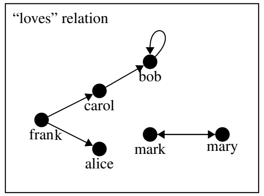
$$\text{FORM} := p \mid i \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle R\rangle\varphi \mid [R]\varphi \mid @_i\varphi, i \in \text{NOM}$$

Semantics: Restrict valuation V so that $V(i)$ is a singleton for $i \in \text{NOM}$.

We define

$$\begin{array}{rll} \mathcal{M}, w \models i & \text{iff} & w \in V(i) \text{ (iff } V(i) = \{w\}) \\ \mathcal{M}, w \models @_i\varphi & \text{iff} & \mathcal{M}, w' \models \varphi \text{ for } w' \in V(i) \end{array}$$

The Expressive Power of $\mathcal{H}(@)$



Query: "Are there two people who loves each other?"

In $\mathcal{H}(@)$: $\varphi := @_{mark}\langle loves\rangle mary \wedge @_{mary}\langle loves\rangle mark \wedge @_{mary}\neg mark$

For $i, j, k \in$ NOM: $\left| \begin{array}{l} @_i i \\ @_i j \rightarrow @_j i \\ @_i j \wedge @_j k \rightarrow @_i k \\ @_i j \rightarrow (@_i \varphi \leftrightarrow @_j \varphi) \end{array} \right.$

Expressive Power (an aside on bisimulation)

The standard tool to measure expressive power of modal logics is bisimulation.

Definition. A bisimulation for BML is a non empty binary relation Z between the domains of two models $\mathcal{M}_1 = \langle W_1, R_1, V_1\rangle$ and $\mathcal{M}_2 = \langle W_2, R_2, V_2\rangle$ s.t.

- Harmony: If $w_1 Z w_2$ then $w_1 \in V(p)$ iff $w_2 \in V(p)$.
- Zig: If $w_1 Z w_2$ and $w_1 R_1 v_1$ then there is v_2 s.t. $w_2 R_2 v_2$ and $v_1 Z v_2$.
- Zag: If $w_1 Z w_2$ and $w_2 R_2 v_2$ then there is v_1 s.t. $w_1 R_1 v_1$ and $v_1 Z v_2$.

Theorem: If Z is a bisimulation and $w_1 Z w_2$ then $\mathcal{M}_1, w_1 \models \varphi$ iff $\mathcal{M}_2, w_2 \models \varphi$.

This is not the correct notion of bisimulation for, e.g., $BML + \Diamond^-$. Why? What is the correct notion of bisimulation for $\mathcal{H}(@)$?

Bisimulations for $\mathcal{H}@$

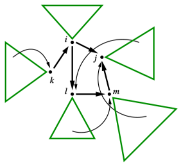
- Consider BML extended with nominals. Do we need any change to the notion of bisimulation? No, the Harmony condition takes care of nominals.
- Consider BML extended with @. Do we need any change? Yes, $@_i$ moves evaluation to the point named by i, that might not be reachable by the Zig/Zag conditions.
- Bisimulation for $\mathcal{H}(@)$: To the conditions for BML-bisimulation add the followng conditions:
  - For all $i \in$ NOM, $i^{\mathcal{M}_1} Z i^{\mathcal{M}_2}$.

Something to ponder

- We were able to say that *mark* and *mary* loves each other (how nice!). But can we say (in $\mathcal{H}(@)$) that there are two people who loves each other (i.e., without naming them)?
- Can we say that somebody loves himself? (again, if we don't know his/her name).
- Think about it for next class.

Some properties of $\mathcal{H}(@)$

- Complexity: Still PSpace-complete.
  - But BML + $\Diamond^-$ + 1 nominal is ExpTime-complete!
- We lost the "Tree Model Property."
  - But we still have a forest model property:



  - (The hybrid μ-calculus with past and the universal modality is ExpTime-complete, and the proof uses tree-automata).

Decidability and Complexity

- Let us consider a problem P having just YES/NO answers (our main concern will be logical questions like "Is a formula φ of the basic modal logic SAT?").
- We say that a problem P is decidable, if we have an algorithm that given any instance of P terminates after a finite number of steps answering correctly correctly YES or NO.
- Once we know that a problem is decidable, we can investigate how expensive it is to solve it. We usually look at two parameters:
  - Time: How many steps takes algorithm A to find a solution?
  - Space: How much memory uses algorithm A to find a solution?

Worst-case Complexity

- Time and Space are usually taken as a function of the size of the input. E.g., Polynomial Time (P), Polynomial Space (PSpace), Exponential Time (Exp), Exponential SPACE (ExpSpace).
- We also differentiate whether the algorithm is deterministic or non-deterministic.
- All these choices gives as many complexity classes, like:

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSpace} \subseteq \mathrm{NPSpace} \subseteq \mathrm{Exp} \subseteq \mathrm{NExp} \subseteq \mathrm{ExpSpace} \subseteq \mathrm{NExpSpace}$$

- Classifying a problem P in a complexity class C:
  - A problem P is in C if all instances of P can be solved using the resources allowed by C.
  - A problem P is hard for C if all problems in C can be reduced to some problem in P.
  - A problem is C-complete if it is in C and it is hard for C.

Enumerating Models

- The proof that the satisfiability problem for PL is decidable is very simple:
  - Suppose that you are given a formula φ and you are looking for a model of φ.
  - First note that propositional symbols that do not appear in φ are irrelevant.
  - Then note that models for PL have only one point.
  - Hence, we only need to list all possible ways of labelling that single point with propositional symbols in φ.
- What about the basic modal language?

The Tableaux Method for Relational Structures

- We want to devise a tableau method for the language we introduced to talk about complex relational structures.
- Let's review the tableaux method that we introduced for propositional logic:

$$\frac{s{:}(\varphi \wedge \psi)}{\begin{array}{c} s{:}\varphi \\ s{:}\psi \end{array}} \ (\wedge)$$

$$\frac{s{:}\neg(\varphi \wedge \psi)}{s{:}\neg\varphi \quad\quad s{:}\neg\psi} \ (\neg\wedge)$$

$$\frac{s{:}\neg\neg\varphi}{s{:}\varphi} \ (\neg\neg)$$

- Pretty neat: 3 rules for an NP-complete problem!
- But now we want to deal with more than a single point.
- The solution is: labels!
- They will help us keep track of what is going on in each point in our model.

Now Lines!

- We have dealt in the previous slide with multiple points. What about lines?
- Remember that the operator we introduced to talk about lines in our language was $\Diamond\varphi$ and we said that

$$\mathcal{M}, w \models \Diamond\varphi \text{ iff there is } w' \text{ s.t. } wRw' \text{ and } \mathcal{M}, w' \models \varphi.$$

- Start with the labelled formula $s{:}\Diamond\varphi$. If this formula is satisfiable, it is because there is an R-sucessor t where φ holds.

$$\frac{s{:}\Diamond\varphi}{\begin{array}{c} sRt \\ t{:}\varphi \end{array}} \ (\Diamond)$$

for t a new label

- Start with the labelled formula $s{:}\neg\Diamond\varphi$. If there is an R-successor t, then φ should not hold at t.

$$\frac{\begin{array}{c} s{:}\neg\Diamond\varphi \\ sRt \end{array}}{t{:}\neg\varphi} \ (\neg\Diamond)$$

Tableaux for Modal Logics

- Clash: $s{:}p$ and $s{:}\neg p$ in a branch.
- To check satisfiability of φ, start the tableaux with the labelled formula $s{:}\varphi$.
- The previous 5 rules provide a sound and complete calculus for the basic modal logic
- It is actually terminating, and by imposing some restrictions on applications it can run in PSPACE, so it is optimal.
- To think: what is the status of labeled formulas $s{:}\varphi$ and accessibility statements sRt?

The Complete Cast, plus an Example

$$\frac{s{:}(\varphi \wedge \psi)}{\begin{array}{c} s{:}\varphi \\ s{:}\psi \end{array}} \qquad \frac{s{:}\Diamond\varphi}{\begin{array}{c} sRt \\ t{:}\varphi \end{array}}$$

for t a new label

$$\frac{s{:}\neg(\varphi \wedge \psi)}{s{:}\neg\varphi \quad\quad s{:}\neg\psi} \qquad \frac{s{:}\neg\neg\varphi}{s{:}\varphi} \qquad \frac{\begin{array}{c} s{:}\neg\Diamond\varphi \\ sRt \end{array}}{t{:}\neg\varphi}$$

$$\begin{array}{c}
s{:}(\Diamond p \wedge (\neg\Diamond\neg q \wedge \neg\Diamond(p \wedge q))) \\
s{:}\Diamond p \\
s{:}\neg\Diamond\neg q \wedge \neg\Diamond(p \wedge q) \\
s{:}\neg\Diamond\neg q \\
s{:}\neg\Diamond(p \wedge q) \\
sRt \\
t{:}p \\
t{:}\neg\neg q \\
t{:}q \\
t{:}\neg(p \wedge q) \\
t{:}\neg p \qquad\qquad t{:}\neg q \\
\text{closed} \qquad\qquad \text{closed}
\end{array}$$

- Which are the similarities/differences with tableaux for PL?
- How do we know that we got it right?
- What can we learn from the calculus?

A Closer Look

- Which similarities / differences with tableaux for PL?
  - What are labels? What are they doing? Can we use them?
  - Is this an algorithm?
  - Is it a good algorithm?
  - Does it terminate?
- Did we get it right?
  - Did we get it right in the PL case, to start with?! Consider the rule:
  - We should prove soundness and completeness
- What can we learn from the calculus?
  - Something about models!

$$\frac{s{:}(\varphi \wedge \psi)}{\begin{array}{c} s{:}\varphi \\ s{:}\psi \end{array}} \qquad \frac{s{:}\Diamond\varphi}{\begin{array}{c} sRt \\ t{:}\varphi \end{array}} \text{ for } t \text{ a new label}$$

$$\frac{s{:}\neg(\varphi \wedge \psi)}{s{:}\neg\varphi \quad\quad s{:}\neg\psi} \qquad \frac{\begin{array}{c} s{:}\neg\Diamond\varphi \\ sRt \end{array}}{t{:}\neg\varphi}$$

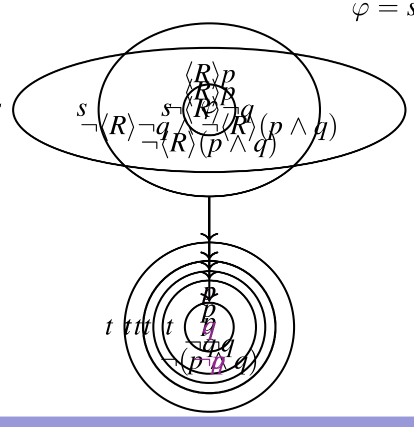
$$\frac{s{:}\neg\neg\varphi}{s{:}\varphi}$$

$$\frac{s{:}\neg(\varphi \wedge \psi)}{s{:}\neg\varphi \quad\quad \begin{array}{c} s{:}\varphi \\ s{:}\neg\psi \end{array}}$$

Tree Models

- Let us see the tableux proof we did before again, for the formula

$\varphi = s{:}(\langle R\rangle p \wedge (\neg\langle R\rangle\neg q \wedge \neg\langle R\rangle(p \wedge q)))$
$s{:}\langle R\rangle p$
$s{:}\neg\langle R\rangle\neg q \wedge \neg\langle R\rangle(p \wedge q)$
$s{:}\neg\langle R\rangle\neg q$
$s{:}\neg\langle R\rangle(p \wedge q)$
sRt
$t{:}p$
$t{:}\neg\neg q$
$t{:}q$
$t{:}\neg(p \wedge q)$
$t{:}\neg p \qquad t{:}\neg q$



Tree and Finite Model Properties

- Using the rules of the tableaux calculus we only explore finite, tree models.
- Let's assume that the calculus is correct.
- Then the $\langle R\rangle$-language
  - cannot say infinite,
  - cannot say non-tree.

$$\frac{s{:}(\varphi \wedge \psi)}{\begin{array}{c} s{:}\varphi \\ s{:}\psi \end{array}} \qquad \frac{s{:}\Diamond\varphi}{\begin{array}{c} sRt \\ t{:}\varphi \end{array}} \text{ for } t \text{ a new label}$$

$$\frac{s{:}\neg(\varphi \wedge \psi)}{s{:}\neg\varphi \quad\quad s{:}\neg\psi} \qquad \frac{\begin{array}{c} s{:}\neg\Diamond\varphi \\ sRt \end{array}}{t{:}\neg\varphi}$$

$$\frac{s{:}\neg\neg\varphi}{s{:}\varphi}$$

Theorem: A formula in the $\langle R\rangle$-language is satisfialle if and only if it is satisfiable in a finite, tree relational structure.

Soundness and Completeness

- We would like to verify that:
  - if the tableau for φ is closed (all tranches contains a class) then φ is UNSAT [Soundness].
  - if the tableau for φ cannot be further extended and it has an open branch then φ is SAT [Completeness].
- Soundness is usually easy to establish. Prove, for each rule of the tableaux, that if the antecedent has a model, then at least one of the generated branches has a model.
- To show completeness we need to build a model from a saturated, open branch.

Completeness

Theorem. If Γ is a saturated open branch from a tableaux for φ, then φ is SAT.

Proof. Given Γ we define the model $\mathcal{M}_\Gamma = \langle W_\Gamma, R_\Gamma, V_\Gamma \rangle$ where

$$\begin{array}{rcl} W_\Gamma & = & \{w \mid w{:}\varphi \in \Gamma\} \\ R_\Gamma & = & \{(w,v) \mid wRv \in \Gamma\} \\ V_\Gamma(p) & = & \{w \mid w{:}p \in \Gamma\} \end{array}$$

Let ψ be the smallest formula such that $w{:}\psi \in \Gamma$ and $\mathcal{M}_\Gamma, w \not\models \psi$.

- $\psi \neq p$ (otherwise $w \in V_\Gamma(p)$) and $\psi \neq \neg p$ (the branch would be closed).
- $\psi \neq \psi_1 \wedge \psi_2$ otherwise as both $\psi_i \in \Gamma$, ψ won't be minimal.
- $\psi \neq \neg(\psi_1 \wedge \psi_2)$ for similar reasons.
- $\psi \neq \langle R \rangle \xi$ as we would have wRv and $v{:}\xi \in \Gamma$ and ψ won't be minimal.
- Similarly for $\neg R\xi$.

Hence, $w{:}\varphi \in \Gamma$ implies $\mathcal{M}_\Gamma, w \models \varphi$.

But what about Hybrid Logics?

- We devised a sound and complete (and terminating) tableau for the basic modal logic. But this is a lecture about Hybrid Logics!!! (I wan't my money back).
- We have been using hybrid logics all the time
  - Write $@_s\varphi$ instead of $s{:}\varphi$
  - Write $@_s\langle R \rangle t$ instead of sRt
- But the set of rules is not complete for the whole basic hybrid language

$$\frac{@_s(\varphi \wedge \psi)}{\begin{array}{c}@_s\varphi \\ @_s\psi\end{array}} \qquad \frac{@_s\Diamond\varphi}{\begin{array}{c}@_s\langle R \rangle t \\ @_t\varphi\end{array}} \text{ for } t \text{ a new label}$$

$$\frac{@_s\neg(\varphi \wedge \psi)}{@_s\neg\varphi \quad\quad @_s\neg\psi} \qquad \frac{\begin{array}{c}@_s\neg\Diamond\varphi \\ @_s\langle R \rangle t\end{array}}{@_t\neg\varphi}$$

$$\frac{@_s\neg\neg\varphi}{@_s\varphi}$$

@-Formulas

- Notation: For simplicity and to keep the notation we were using we will keep writing $i{:}\varphi$ instead of $@_i\varphi$.
- Fact 1: φ is satisfiable iff $i{:}\varphi$ is satisfiable, for i not in φ. I.e., to check the satisfiability of φ we start the tableax with $i{:}\varphi$.
- Fact 2: : is self-dual, hence $\neg i{:}\varphi$ is equivalent to $i{:}\neg\varphi$.

Sound and Complete Tableaux for $\mathcal{H}(@)$

$\neg$ rules: $(\neg)\ \dfrac{i{:}\neg\neg\varphi}{i{:}\varphi}$

$\wedge$ rules: $(\wedge)\ \dfrac{i{:}(\varphi \wedge \psi)}{\begin{array}{c}i{:}\varphi \\ i{:}\psi\end{array}}$ $\qquad (\neg\wedge)\ \dfrac{i{:}\neg(\varphi \wedge \psi)}{i{:}\neg\varphi \quad\quad s{:}\neg\psi}$

$\langle r \rangle$ rules: $(\langle r \rangle)\ \dfrac{i{:}\langle r \rangle\varphi}{\begin{array}{c}i{:}\langle r \rangle j \\ j{:}\varphi\end{array}}$ for j new in branch $\qquad (\neg\langle r \rangle)\ \dfrac{\begin{array}{c}i{:}\neg\langle r \rangle\varphi \\ i{:}\langle r \rangle j\end{array}}{j{:}\neg\varphi}$

@ rules: $(@)\ \dfrac{i{:}j{:}\varphi}{j{:}\varphi}$ $\qquad (\neg@)\ \dfrac{i{:}\neg j{:}\varphi}{j{:}\neg\varphi}$

Equality: (Ref) $\dfrac{[i \text{ on branch}]}{i{:}i}$ $\quad$ (Sym) $\dfrac{i{:}j}{j{:}i}$ $\quad$ (Cong) $\dfrac{i{:}k \quad j{:}k \quad i{:}\varphi}{j{:}\varphi}$

Hybrid Termination

- Once nominals and satisfaction operators are introduced, ensuring termination is more difficult.
- An obvious problem is the *Cong* rule: $\dfrac{i{:}k \quad j{:}k \quad i{:}\varphi}{j{:}\varphi}$
- The solution is to impose a "direction": Only one nominal in the equivalence class is saturated. But equality should still be an equivalence, so an irrestricted version for nominals is introduced.

$$\text{(wCong)}\ \frac{i{:}k \quad j{:}k \quad i{:}\varphi}{j{:}\varphi} \qquad \text{(Nom)}\ \frac{i{:}k \quad j{:}k \quad i{:}n}{j{:}n}$$

j is the earliest introduced nominal making k true

Bolander, T. and Blackburn, P..
Termination for Hybrid Tableaus
Journal of Logic and Computation, 17, 517-554, 2007.

A Tableau Based Prover for Hybrid Logics

- A sound, complete and terminating calculus has been implemented in the prover HTab
- Available at `https://hackage.haskell.org/package/HTab`.
- Implemented in haskell, with a GPL license. I.e., you can get the code and change it!
- Demo