Lógica modal computacional

Carlos Areces

1er Cuatrimestre 2010

Part I

Memory logics

Changing the model

- The Modal Logic book says

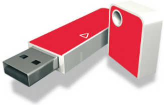
> A modal formula is a little automaton standing at some state in a relational structure, and only permitted to explore the structure by making journeys to neighbouring states.

- What about granting our automaton the additional power to **modify the model** during its exploratory trips?
- There may be many ways to modify a model (changing the domain, the edges, the valuation, . . .)
- We want to restrict our atention to a specific way of modifying a model: **adding a memory** to the model, and **performing changes** on it

Changing the model

- We are going to add a storage structure to standard Kripke models:

$$\mathcal{M} = \langle W, (R_r)_{r \in \mathsf{rel}}, V \rangle \qquad + $$



- There are many possible types of structures: a set, a list, a stack, ...
- We want to start with a very simple structure, so we are going to add a **set** S to the standard Kripke model:

Memory Kripke model

Given a set $S \subseteq W$, a memory Kripke model is

$$\mathcal{M} = \langle W, (R_r)_{r \in \mathsf{rel}}, V, S \rangle$$

Changing the model

We have to add suitable operators to manipulate the memory

- Since we are using a set S as the container, there are two "natural" operators to use:
  - An operator ⓡ to *remember* the current point, storing it in S.
  - An operator ⓚ to check membership of the current point, and find out whether it is *known*

Some notation

Given $\mathcal{M} = \langle W, (R_r)_{r \in \mathsf{rel}}, V, S \rangle$, we define

$$\mathcal{M}[w] = \langle W, (R_r)_{r \in \mathsf{rel}}, V, S \cup \{w\} \rangle$$

Now, more formally

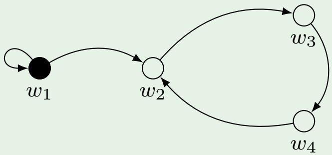
Semantics of ⓡ and ⓚ

$$\begin{array}{lll} \mathcal{M}, w \models ⓡ\varphi & \text{iff} & \mathcal{M}[w], w \models \varphi \\ \mathcal{M}, w \models ⓚ & \text{iff} & w \in S \end{array}$$

Changing the model

Let's see the use of $\textcircled{r}$ and $\textcircled{k}$ with an example. Suppose we start with the following model:

A model with an initially empty memory



- $V(p) = \emptyset$ for all $p \in$ prop
- $S = \emptyset$
- $S = \{w_1\}$

- How can we check whether w_1 has a successor different from itself?

$$\mathcal{M}, w_1 \models \textcircled{r}\Diamond\neg\textcircled{k}$$
$$\Updownarrow$$
$$\mathcal{M}[w_1], w_1 \models \Diamond\neg\textcircled{k}$$
$$\Updownarrow$$
$$\mathcal{M}[w_1], w_2 \models \neg\textcircled{k} \qquad \checkmark$$

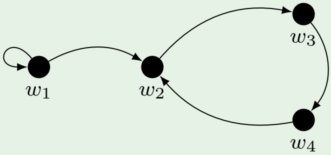
Memory logics

- The idea of using operators that **change** the model is not new
- The family of languages with these characteristics are sometimes called **dynamic logics**
- For example:
  - Dynamic epistemic logics
  - Real time logics
  - Dynamic predicate logic
- Memory logics can be seen as dynamic languages that
  - Incorporate the notion of state from a 'pure' perspective
  - Do not add any domain-specific behaviour in the evolution of the model
  - Analyze dynamic behaviour from a very simple perspective
  - Can be thought of as a 'weak' version of the standard $\downarrow$ modal binder
- Can be combined with other modal and hybrid operators (A, nominals, $@$, etc.)

Other operators

- We can think in other operators, that *delete* elements from the memory.
- In the previous example, the memory was initially empty, which was quite convenient

A model where every point is memorized



- How can we check whether w_1 has a successor different from itself?
- There doesn't seem to be a way. . .

Other operators

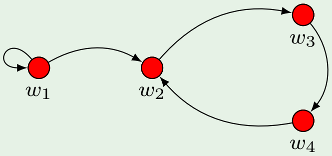
We can define an operator $\textcircled{e}$ (for ‘erase’) that completely wipes out the memory

Semantics of $\textcircled{e}$

$$\langle M, (R_r)_{r\in\mathsf{rel}}, V, S\rangle, w \models \textcircled{e}\varphi \quad \text{iff} \quad \langle M, (R_r)_{r\in\mathsf{rel}}, V, \emptyset\rangle, w \models \varphi$$

So now, in order to check in $\mathcal{M}$ whether w_1 has a successor different from itself

A model $\mathcal{M}$, where every point is memorized



we can evaluate

$$\mathcal{M}, w_1 \models \textcircled{e}\textcircled{e}\textcircled{r}\Diamond\neg\textcircled{k}\textcircled{r}\Diamond\neg\textcircled{k}$$

This formula works independently of the initial state of the memory

Other ingredients

There are other “dimensions” we can take into consideration:

- Class of models: for example, it is quite natural to consider the class of models whose memory is initially empty
- Memorizing policies: we can try to impose some restrictions on the interplay between memory and modal operators
  - These restrictions are going to help us find decidable fragments
- Other memory operators and containers: are there other memory operators? What happens if we change a set by other type of structure?
  - We can define f, a local version of e
  - We can try using a stack instead of a set as the memory container

Notation

We are going to work with several memory logic fragments

Notational convention

- We call $\mathcal{ML}$ the basic modal logic, and $\mathcal{HL}$ the extension of $\mathcal{ML}$ with nominals
- When we add a set S and the operators ⓡ and ⓚ we add m as a superscript, e.g. $\mathcal{ML}^m(\ldots$
- We add $\emptyset$ as a subscript when we work with $\mathcal{C}_\emptyset$ (otherwise is the class of all models), e.g. $\mathcal{ML}^m_\emptyset(\ldots$
- Then we list the additional operators

For example

- $\mathcal{ML}^m_\emptyset(\langle r\rangle, ⓔ)$: the modal memory logic with ⓡ, ⓚ, ⓔ and the usual diamond $\langle r\rangle$ over the class $\mathcal{C}_\emptyset$
- $\mathcal{HL}^m(@, \langle r\rangle)$: the hybrid memory logic with ⓡ, ⓚ, $\langle r\rangle$, $@$ over the class of all models

Getting to know a logic

This is a new family of logics, and there are characteristics that are worth investigating

- Expressivity: What can we say with memory logics? Which is the relation between them and other well-known logics?
- Decidability: Which is the computational complexity of the different fragments? How much are memory operators adding to the basic modal logic?
- Interpolation: How they behave in term of Craig interpolation and Beth definability?
- Axiomatization: Do they have sound and complete axiomatic systems?
- Tableau systems: Can we adapt known tableau techniques to produce sound and complete tableau systems? Can we find terminating tableaux for the decidable memory fragments?

Disclaimer: we are not going to see all these topics during this talk

Expressivity results

We compare the expressive power of the different fragments via the existence of *equivalence preserving translations*

$\mathcal{L}'$ is as least as expressive as $\mathcal{L}$ ($\mathcal{L} \leq \mathcal{L}'$) if there is a Tr such that

$$\mathcal{M}, w \models_{\mathcal{L}} \varphi \text{ iff } \mathcal{M}, w \models_{\mathcal{L}'} \mathsf{Tr}(\varphi)$$

Theorem

$\mathcal{ML}^m_\emptyset(\langle r \rangle) < \mathcal{HL}(\downarrow)$.

To see that $\mathcal{ML}^m_\emptyset(\langle r \rangle) \leq \mathcal{HL}(\downarrow)$ we define a translation Tr that maps formulas of $\mathcal{ML}^m_\emptyset(\langle r \rangle)$ into sentences of $\mathcal{HL}(\downarrow)$.

- We use $\downarrow$ to simulate ⓡ.
- We use a finite set N to simulate that ⓚ does not distinguish between different memorized states.

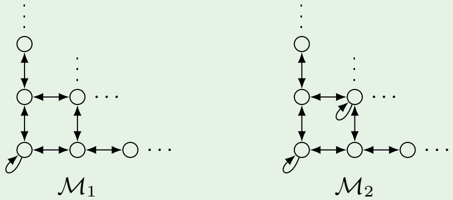
$$\begin{array}{rcll} \mathsf{Tr}_N(ⓡ\varphi) & = & \downarrow i.\mathsf{Tr}_{N\cup\{i\}}(\varphi) & (\text{for } i \text{ a new nominal}) \\ \mathsf{Tr}_N(ⓚ) & = & \bigvee_{i \in N} i & \end{array}$$

Expressivity results

How can we see that $\mathcal{ML}^m_\emptyset(\langle r\rangle) \neq \mathcal{HL}(\downarrow)$? We need to show that there is *no possible* translation from $\mathcal{HL}(\downarrow)$ to $\mathcal{ML}^m_\emptyset(\langle r\rangle)$...

- We developed a notion of *bisimulation* for each fragment. Intuitively, two models are bisimilar for a logic $\mathcal{L}$ when they cannot be distinguished by $\mathcal{L}$-formulas

$\mathcal{M}_1$ and $\mathcal{M}_2$ are $\mathcal{ML}^m_\emptyset(\langle r\rangle)$-bisimilar



But there is a formula $\varphi \in \mathcal{HL}(\downarrow)$ such that

$$\mathcal{M}_1, w \models_{\mathcal{HL}(\downarrow)} \varphi \text{ and } \mathcal{M}_2, v \not\models_{\mathcal{HL}(\downarrow)} \varphi$$

So a translation from $\mathcal{HL}(\downarrow)$ to $\mathcal{ML}^m_\emptyset(\langle r\rangle)$ cannot exist

Expressivity results

We establish in this way an "expressivity map" for many memory logic fragments:

$$
\begin{tikzcd}
 & & & \mathcal{ML}^m_\emptyset(\langle r\rangle, \mathtt{f}) \arrow[dd, "\not\leq"'] \arrow[dr, dashed, "<"'] \arrow[dr, bend left, "\leq"] & & \\
\mathcal{ML} \arrow[r] & \mathcal{ML}^m_\emptyset(\langle\langle r\rangle\rangle) \arrow[r] & \mathcal{ML}^m_\emptyset(\langle r\rangle) \arrow[ur, bend left] \arrow[dr, bend right] & & \mathcal{ML}^m_\emptyset(\langle r\rangle, \mathtt{e}, \mathtt{f}) \arrow[r, bend left, "\leq"] \arrow[r, bend right, dashed, "<"'] & \mathcal{HL}(\downarrow) \\
 & & & \mathcal{ML}^m_\emptyset(\langle r\rangle, \mathtt{e}) \arrow[uu, dashed, "\not\leq"'] \arrow[ur, bend right] & &
\end{tikzcd}
$$

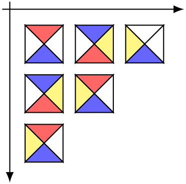
$\mathcal{L} \longrightarrow \mathcal{L}' = \mathcal{L} < \mathcal{L}'$

- All the memory logic fragments are **strictly** between the basic modal logic and the logic $\mathcal{HL}(\downarrow)$ (and therefore below first order logic)

Decidability results

- Expressive power and computational complexity are usually at opposite sides of the scales
- We use the *tiling technique* to prove **undecidability** for many memory fragments
- Given a finite set of *tile types* $\mathcal{T}$

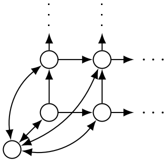
The *tiling problem*: Is it possible to arrange tiles of type $\mathcal{T}$ in $\mathbb{N} \times \mathbb{N}$ such that every pair of adjacent tiles has the same color?



- The $\mathbb{N} \times \mathbb{N}$ tiling problem is known to be undecidable
- Given a set of tile types $\mathcal{T}$, the idea is to build a formula $\varphi_{\mathcal{T}}$ such that $\varphi_{\mathcal{T}}$ is satisfiable if and only if there is a tiling for $\mathcal{T}$

Decidability results

- We have encoded the tiling problem for several memory fragments using a *spy point*: a point that sees every other point in the model



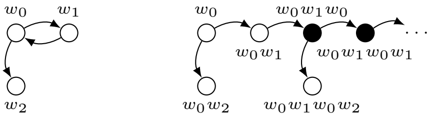
- Most of the memory logic fragments turned out to be undecidable
- We found decidable fragments restricting the interplay between $\langle r \rangle$ and $\mathtt{r}$: we force them to act at the same time

$\langle r \rangle$ and $\mathtt{r}$ working together

$$\mathcal{M}, w \models \langle\!\langle r \rangle\!\rangle \varphi \quad \text{iff} \quad \exists w' \in W, R_r(w, w') \text{ and } \mathcal{M}[w], w' \models \varphi.$$

Decidability results

- We proved that some fragments are PSPACE-complete showing that they enjoy the bounded tree-model property: every satisfiable formula can be satisfied in a bounded tree
- We showed that there is a procedure to transform an arbitrary model into a tree-like model, preserving equivalence



- We also built a “decidability map” for the different memory fragments

PSPACE-complete	Undecidable
$\mathcal{ML}^m(\langle\!\langle r \rangle\!\rangle)$	$\mathcal{ML}^m_\emptyset(\langle\!\langle r \rangle\!\rangle)$, $\mathcal{ML}^m(\langle\!\langle r \rangle\!\rangle) + i$
$\mathcal{ML}^m(\langle\!\langle r \rangle\!\rangle, \text{ⓕ})$	$\mathcal{ML}^m(\langle r \rangle)$, ...

Axiomatizations

- We characterized many memory logics fragments in terms of axiomatic systems *à la Hilbert*
- **Nominals** proved to be a very useful device to find sound and complete axiomatizations

Axiomatization for $\mathcal{HL}^m(@, \langle r \rangle)$

$$\text{All axioms and rules for } \mathcal{HL}(@)$$
$$+$$
$$\vdash @_i(\textcircled{r}\varphi \leftrightarrow \varphi[\textcircled{k}/(\textcircled{k} \vee i)])$$

- We found sound and complete axiomatizations for all the *hybrid* memory fragments (and establish automatic completeness for pure extensions)
- We could provide axiomatizations for some cases even in the absence of nominals (i.e., $\mathcal{ML}^m(\langle\!\langle r \rangle\!\rangle)$ and $\mathcal{ML}^m(\langle\!\langle r \rangle\!\rangle, \textcircled{f})$)
- The tree-model property was a key feature to use when nominals were not present

Tableau systems

- We presented a sound and complete tableau system for $\mathcal{ML}^m(\langle r \rangle, ⓔ, ⓕ)$, $\mathcal{ML}^m_\emptyset(\langle r \rangle, ⓔ, ⓕ)$, and its sublanguages
- It is a *prefixed* tableau where we use prefixed formulas with the shape

$$\langle w, R, F \rangle^{\mathcal{C}} : \varphi$$

- w: point of evaluation
- R: set of memorized labels
- F: set of forgotten labels
- $\mathcal{C}$: either $\mathcal{C}_\emptyset$ or the class of all models
- φ: current formula

- The rules for propositional and modal operators are standard

Tableau systems

- For example, the rule for $\textcircled{\texttt{r}}$ is quite straightforward

$$(\textcircled{\texttt{r}}) \qquad \frac{\langle w, R, F\rangle^C{:}\textcircled{\texttt{r}}\varphi}{\langle w, R \cup \{w\}, F - \{w\}\rangle^C{:}\varphi}$$

- The rule for $\textcircled{\texttt{k}}$ (and for $\neg\textcircled{\texttt{k}}$) introduces an equivalence class

$$(\textcircled{\texttt{k}}) \qquad \frac{\langle w, \{v_1, \ldots v_k\}, F\rangle^C{:}\textcircled{\texttt{k}}}{w \approx v_1 \mid \cdots \mid w \approx v_k \mid \langle w, \emptyset, \emptyset\rangle^C{:}\textcircled{\texttt{k}}}$$

$$(\mathsf{repl}) \qquad \frac{\begin{array}{c}\langle w, R, F\rangle^C{:}\varphi \\ w \approx^* w'\end{array}}{\langle w', R[w \mapsto w'], F[w \mapsto w']\rangle^C{:}\varphi}$$

- Since this fragment in undecidable, the tableau is non-terminating
- We also provided a sound, complete and terminating tableau for the decidable fragments

Open questions

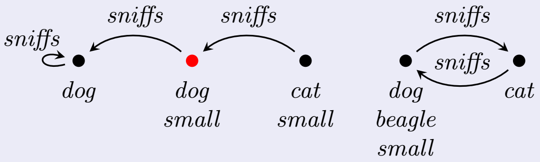
- We left some missing links in the expressivity map. We would like to complete it.
- The decidable fragments we found are strictly more expressive than $\mathcal{ML}$, but still really close to it. Can we find more expressive but still decidable fragments? We have some ideas
  - Concrete domains: storing values, not points
  - Restricted classes of models
  - Weaker containers (or syntactic restrictions)
- Beth definability needs further research, we would like some general result
- We want to explore the relation between memory logics and other dynamic logics (DEL is a good candidate). This could also lead to decidable fragments
- Can we find suitable axiomatizations in the absence of nominals. We still don't have one for $\mathcal{ML}^m(\langle r \rangle)$!

Part II

Logical methods in the generation de referring expressions

Logics in the Generation of Referring Expressions

The *content determination problem* as a logical task



content determination
(e.g., as a FOL formula)

$$dog(x) \wedge \exists y.(x \not\approx y \wedge dog(y) \wedge sniffs(x, y))$$

surface realization

“the dog sniffing another dog”

expressivity vs. linguistic adequacy vs. complexity vs. . . .

Towards *incremental* content determination

Use of model minimization algorithms

- Minimize the model using the *right* notion of equivalence:
  - basic modal logic $\rightsquigarrow$ bisimulation
  - positive existential modal logic $\rightsquigarrow$ simulation
  - positive existential FOL $\rightsquigarrow$ subgraph isomorphism
  - ...
- A witness formula for each equivalence class is computed.
- Conceptually simple, but with little "linguistic control".

Relativization of known algorithms

1. Make explicit the underlying notion of expressiveness.
2. Try to make the algorithm "parametric" in this notion.

Return to theory-land…

What is the *complexity* of content-determination?

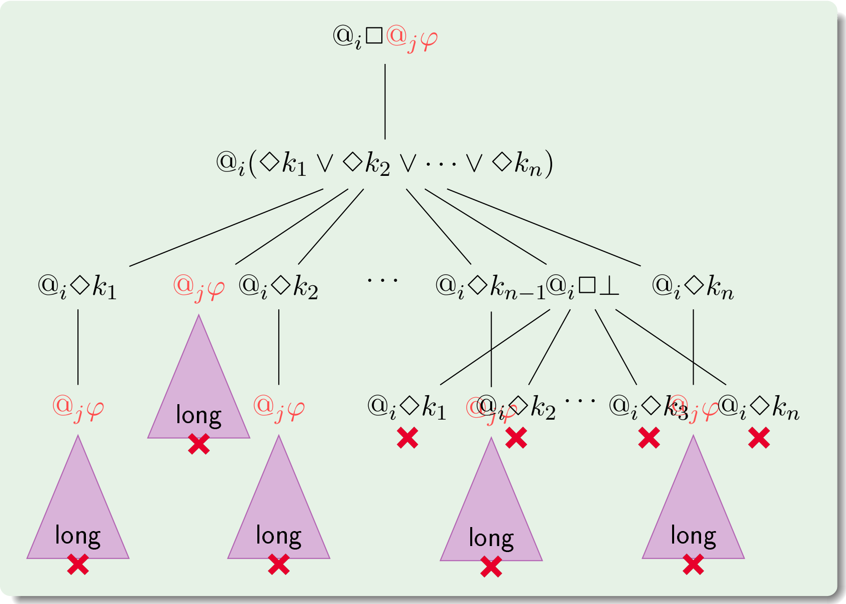
- Ie, what is the complexity of the "fastest" algorithm for it?
- Of course, the answer varies with the logical language.
- But for modal languages, we were able to show that:
  - No polynomial bounds the size of the generated formula.
  - Therefore, no polynomial time algorithm can exist!*

* Technically, none that outputs the formula as a tree.

Part IV

Coinduction, extractability, normal forms

Global modalities should be "extracted"



Globality $\sim$ extractability?

Global modalities are extractable from other modalities...

$$[r]@_i\varphi \equiv [r]\bot \vee @_i\varphi \qquad [r]\mathsf{A}\varphi \equiv [r]\bot \vee \mathsf{A}\varphi$$

$$@_j@_i\varphi \equiv @_j\bot \vee @_i\varphi \qquad @_j\mathsf{A}\varphi \equiv @_j\bot \vee \mathsf{A}\varphi$$

$$\mathsf{A}@_i\varphi \equiv \mathsf{A}\bot \vee @_i\varphi \qquad \mathsf{A}\mathsf{A}\varphi \equiv \mathsf{A}\bot \vee \mathsf{A}\varphi$$

$$\vdots \qquad \vdots$$

...but some modalities are more equal than others

$$\downarrow i.@_i\varphi \not\equiv \downarrow i.\bot \vee @_i\varphi$$

$$\text{(r)}\mathsf{A}\varphi \not\equiv \text{(r)}\bot \vee \mathsf{A}\varphi$$

Coinductive models – a unifying framework

The class of all (rooted) Kripke models with domain W

- $\mathrm{Kripke}_W \stackrel{def}{=}$ all the tuples $\langle W, w_0, V, R\rangle$ such that
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq W$
- $\langle W, w, V, R\rangle \models [r]\varphi$ iff $\langle W, v, V, R\rangle \models \varphi$, $\forall v \in R(r, w)$
- Many modal operators can be defined as classes of models

The class of all *coinductive models* with domain W

- $\mathrm{Mods}_W \stackrel{def}{=}$ all the tuples $\langle W, w_0, V, R\rangle$ such that
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq \mathrm{Mods}_W \Longleftarrow$ **coinductive definition!**
- $\langle W, w, V, R\rangle \models [r]\varphi$ iff $\mathcal{M} \models \varphi$, $\forall \mathcal{M} \in R(r, w)$
- More modal operators can be defined as classes of models

Some initial results using the coinductive framework

- The basic modal logic is complete wrt coinductive models
- *Bisimulations*: one size fits all
- General conditions that guarantee extractability
- Extractability is preserved when new operators are added