Lógicas modales

Carlos Areces & Raul Fervari

1er Cuatrimestre 2017
Córdoba, Argentina

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.

Changing the model

- The Modal Logic book says

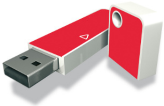
> A modal formula is a little automaton standing at some state in a relational structure, and only permitted to explore the structure by making journeys to neighbouring states.

- 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:

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



Changing the model

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

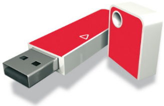
$$\mathcal{M} = \langle W, (R_r)_{r \in \mathsf{rel}}, V \rangle \quad + $$

- There are many possible types of structures: a set, a list, a stack, ...

Changing the model

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

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



- 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:

Changing the model

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

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

- 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 $\texttt{r}$ to *remember* the current point, storing it in S.
  - An operator $\texttt{k}$ to check membership of the current point, and find out whether it is *known*

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

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

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

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

Now, more formally

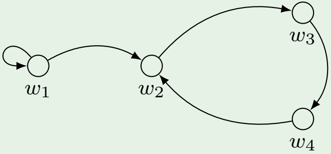
Semantics of ⓡ and ⓚ

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

Changing the model

Let's see the use of ⓡ and ⓚ 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
- $S = \emptyset$

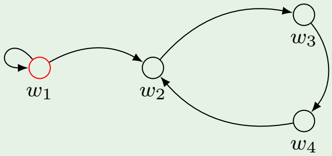
- How can we check whether w_1 has a successor different from itself?

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

Changing the model

Let's see the use of ⓡ and ⓚ 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
- $S = \emptyset$

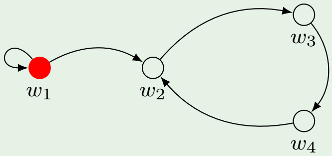
- How can we check whether w_1 has a successor different from itself?

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

Changing the model

Let's see the use of ⓡ and ⓚ 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
- $S = \{w_1\}$

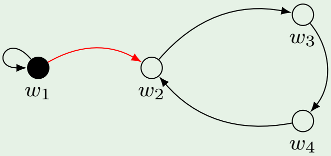
- How can we check whether w_1 has a successor different from itself?

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

Changing the model

Let's see the use of ⓡ and ⓚ 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
- $S = \{w_1\}$

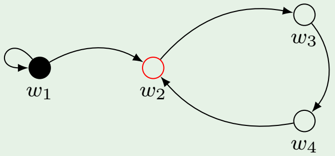
- How can we check whether w_1 has a successor different from itself?

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

Changing the model

Let's see the use of ⓡ and ⓚ 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
- $S = \{w_1\}$

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

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

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

- 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
  - 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.)

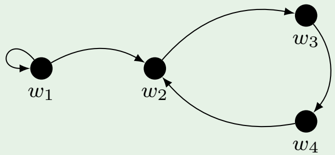
Other operators

- 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

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

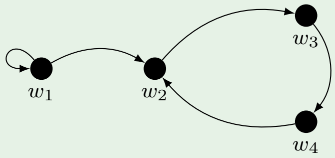


- How can we check whether w_1 has a successor different from itself?

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 ⓔ (for ‘erase’) that completely wipes out the memory

Other operators

We can define an operator ⓔ (for 'erase') that completely wipes out the memory

Semantics of ⓔ

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

Other operators

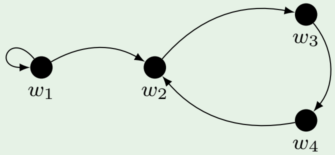
We can define an operator ⓔ (for 'erase') that completely wipes out the memory

Semantics of ⓔ

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

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



Other operators

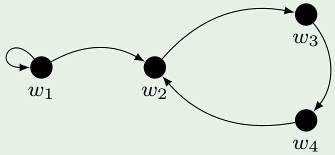
We can define an operator ⓔ (for 'erase') that completely wipes out the memory

Semantics of ⓔ

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

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

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

Other operators

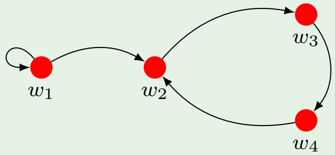
We can define an operator ⓔ (for 'erase') that completely wipes out the memory

Semantics of ⓔ

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

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

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

Other operators

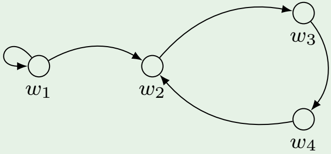
We can define an operator ⓔ (for 'erase') that completely wipes out the memory

Semantics of ⓔ

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

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

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

Other operators

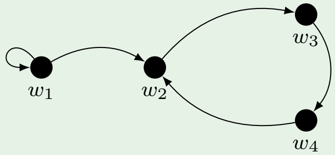
We can define an operator ⓔ (for 'erase') that completely wipes out the memory

Semantics of ⓔ

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

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

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

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

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 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 ⓕ, a local version of ⓔ
  - We can try using a stack instead of a set as the memory container

Other operators

- We can also think in a ‘local’ version of ⓔ, that only deletes the current point of evaluation.
- Let’s consider then the operator ⓕ (for ‘forget’)

Other operators

- We can also think in a 'local' version of ⓔ, that only deletes the current point of evaluation.
- Let's consider then the operator ⓕ (for 'forget')

Semantics of ⓕ

$$\langle M, (R_r)_{r\in\mathsf{rel}}, S\rangle, w \models ⓕ\varphi \quad \text{iff} \quad \langle M, (R_r)_{r\in\mathsf{rel}}, S \setminus \{w\}\rangle, w \models \varphi$$

Other operators

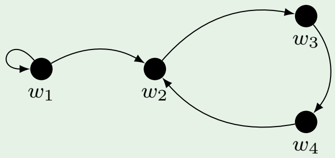
- We can also think in a 'local' version of ⓔ, that only deletes the current point of evaluation.
- Let's consider then the operator ⓕ (for 'forget')

Semantics of ⓕ

$$\langle M, (R_r)_{r \in \mathsf{rel}}, S\rangle, w \models ⓕ\varphi \quad \text{iff} \quad \langle M, (R_r)_{r \in \mathsf{rel}}, S \setminus \{w\}\rangle, w \models \varphi$$

Again, if we want 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

$$\mathcal{M}, w_1 \models ⓕⓡ\Diamond ⓚ$$

Other ingredients: classes of models

Observe that when the memory of $\mathcal{M}$ is initially empty,

$$\mathcal{M}, w \models \text{ⓡ}\langle r \rangle\text{ⓚ} \quad \text{iff} \quad wR_r w$$

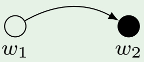
Other ingredients: classes of models

Observe that when the memory of $\mathcal{M}$ is initially empty,

$$\mathcal{M}, w \models \texttt{r}\langle r\rangle\texttt{k} \quad \text{iff} \quad wR_r w$$

But this formula is also true at

A model with a non-empty memory



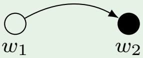
Other ingredients: classes of models

Observe that when the memory of $\mathcal{M}$ is initially empty,

$$\mathcal{M}, w \models ⓡ\langle r \rangle ⓚ \quad \text{iff} \quad wR_r w$$

But this formula is also true at

A model with a non-empty memory



Taking this into consideration, it is natural to consider memory logics restricted to

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

the class of models with an empty memory.

Other ingredients: memorizing policies

- Until now memory and modal operators were working ‘in parallel’
- Restricting expressivity sometimes can be helpful to reduce computational cost
- We can try to impose some restrictions in the interplay between memory and modal operators

Let's define an operator where $\langle r \rangle$ and $\textcircled{\texttt{r}}$ act **at the same time**

Other ingredients: memorizing policies

- Until now memory and modal operators were working 'in parallel'
- Restricting expressivity sometimes can be helpful to reduce computational cost
- We can try to impose some restrictions in the interplay between memory and modal operators

Let's define an operator where $\langle r \rangle$ and ⓡ act **at the same time**

$\langle r \rangle$ and ⓡ working together

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

We are going to see later that this operator helps us to find **decidable** memory fragments

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

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?

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?

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?

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?

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?

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

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

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

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

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.

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

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

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

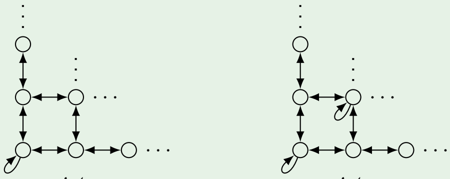
- We developed a notion of *bisimulation* for each fragment. Intuitively, two models are bisimilar for a logic $\mathcal{L}$ when they cannot be distinguished by $\mathcal{L}$-formulas

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



$\mathcal{M}_1$ $\quad$ $\mathcal{M}_2$

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:

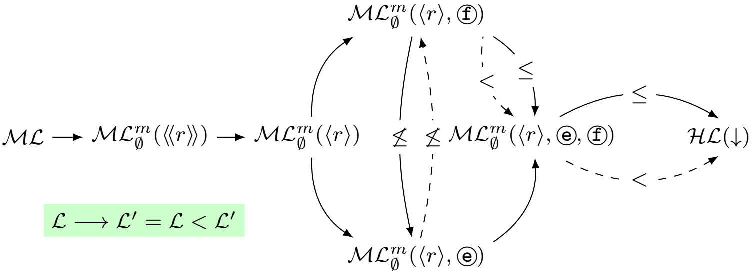
$$
\begin{tikzcd}
 & & & \mathcal{ML}^m_\emptyset(\langle r\rangle, \textcircled{f}) \arrow[dd, "\not\leq"] \arrow[dr, "\leq"] & & \\
\mathcal{ML} \arrow[r] & \mathcal{ML}^m_\emptyset(\langle\langle r\rangle\rangle) \arrow[r] & \mathcal{ML}^m_\emptyset(\langle r\rangle) \arrow[ur, bend left] \arrow[dr,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

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

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

Expressivity results

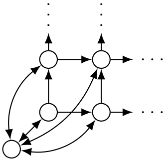
We establish in this way an “expressivity map” for many memory logic fragments:



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

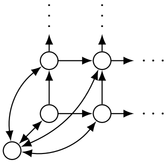
Decidability results

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



Decidability results

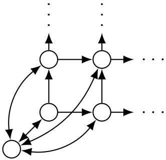
- We have encoded the tiling problem for several memory fragments using a *spy point*: a point that sees every other point in the model



- Most of the memory logic fragments turned out to be undecidable

Decidability results

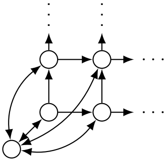
- We have encoded the tiling problem for several memory fragments using a *spy point*: a point that sees every other point in the model



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

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



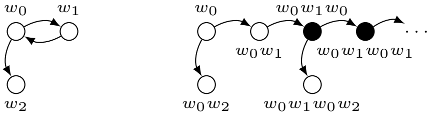
- Most of the memory logic fragments turned out to be undecidable
- We found decidable fragments restricting the interplay between $\langle r \rangle$ and ⓡ: we force them to act at the same time

$\langle r \rangle$ and ⓡ working together

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

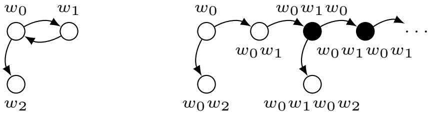
Decidability results

- 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



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

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

All axioms and rules for $\mathcal{HL}(@)$

$$+$$

$$\vdash @_i(\textcircled{\texttt{r}}\varphi \leftrightarrow \varphi[\textcircled{\texttt{k}}/(\textcircled{\texttt{k}} \vee i)])$$

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

All axioms and rules for $\mathcal{HL}(@)$

$$+$$

$$\vdash @_i(ⓡ\varphi \leftrightarrow \varphi[ⓚ/(ⓚ \vee i)])$$

- 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 $\mathcal{ML}^m(\langle\langle r \rangle\rangle, ⓕ)$)
- 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 ⓡ is quite straightforward

Tableau systems

- For example, the rule for ⓡ is quite straightforward

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

Tableau systems

- For example, the rule for ⓡ is quite straightforward

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

- The rule for ⓚ (and for $\neg$ⓚ) introduces an equivalence class

Tableau systems

- For example, the rule for ⓡ is quite straightforward

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

- The rule for ⓚ (and for $\neg$ⓚ) introduces an equivalence class

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

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

Tableau systems

- For example, the rule for ⓡ is quite straightforward

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

- The rule for ⓚ (and for $\neg$ⓚ) introduces an equivalence class

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

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

- Since this fragment in undecidable, the tableau is non-terminating

Tableau systems

- For example, the rule for ⓡ is quite straightforward

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

- The rule for ⓚ (and for $\neg$ⓚ) introduces an equivalence class

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

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

- 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.

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)

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

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

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 $\mathcal{ML}^m(\langle r \rangle)$!

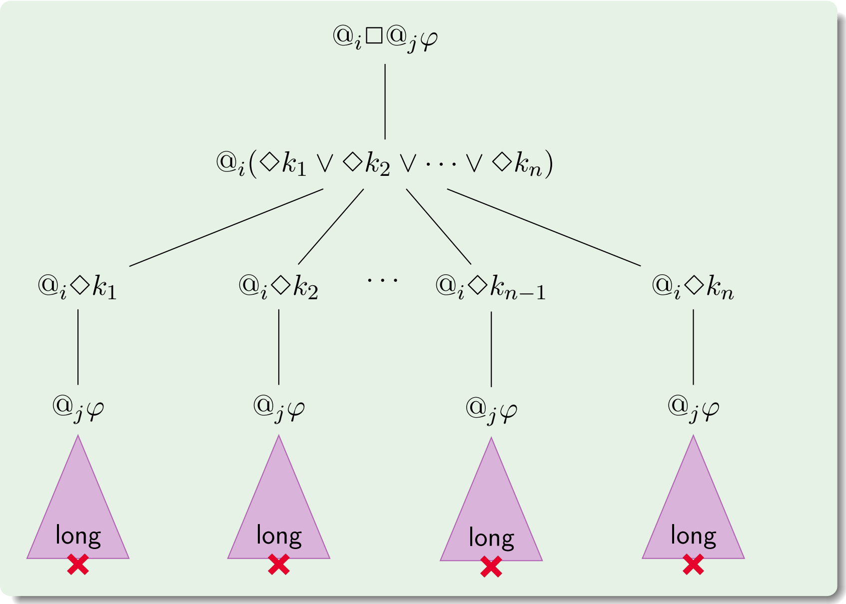
References

- Areces, C., Fervari, R., and Hoffmann, G.. Moving Arrows and Four Model Checking Results. In Proceedings of WoLLIC 2012, Buenos Aires, Argentina, September 2012.
- Areces, C., Figueira, S., and Mera, S.. Completeness results for memory logics. Annals of Pure and Applied Logic, 163(7):961–972, 2012.
- Areces, C., Figueira, D., Figueira, S., and Mera, S.. The Expressive Power of Memory Logics. Review of Symbolic Logic, 4(2):290–318, Cambridge University Press, 2011.
- Areces, C., Carreiro, F., Figueira, S., and Mera, S.. Basic Model Theory for Memory Logics. In Proceedings WoLLIC 2011, pp. 20–34, Springer, Philadelphia, October 2011.
- Areces, C., Figueira, D., Gorin, D., and Mera, S.. Tableaux and Model Checking for Memory Logics. In Automated Reasoning with Analytic Tableaux and Related Methods, pp. 47–61, Springer Berling / Heidelberg, Oslo, Norway, 2009.

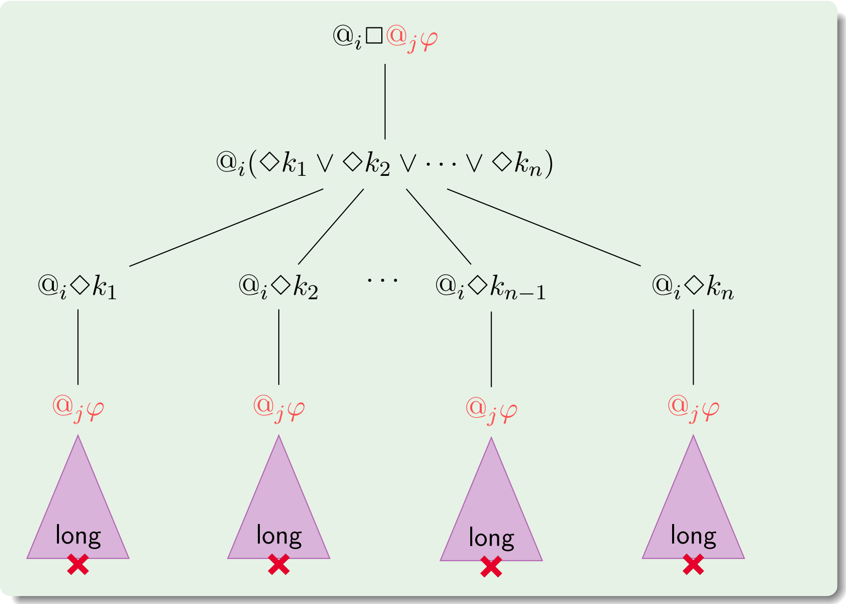
Part III

Coinduction, extractability, normal forms

Global modalities should be "extracted"



Global modalities should be "extracted"

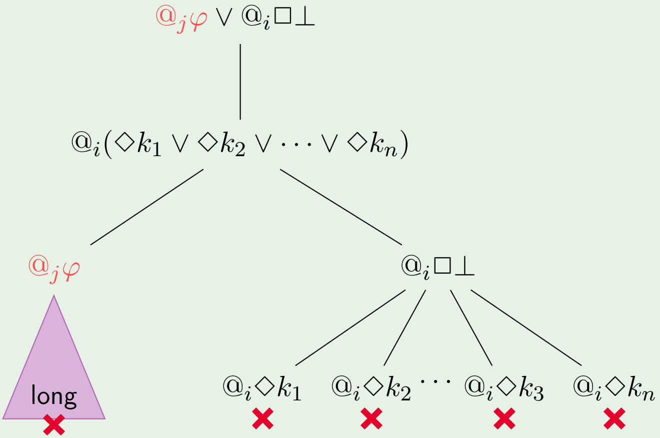


Global modalities should be "extracted"

$$@_j\varphi \vee @_i\Box\bot$$

$$@_i(\Diamond k_1 \vee \Diamond k_2 \vee \cdots \vee \Diamond k_n)$$

Global modalities should be "extracted"



Globality $\sim$ extractability?

Global modalities are extractable from other modalities...

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

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

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

$$\vdots \qquad\qquad \vdots$$

Globality $\sim$ extractability?

Global modalities are extractable from other modalities...

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

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

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

$$\vdots \qquad\qquad \vdots$$

...but some modalities are more equal than others

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

$$\textcircled{r}\mathsf{A}\varphi \not\equiv \textcircled{r}\bot \vee \mathsf{A}\varphi$$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq W$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq \mathrm{Mods}_W$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq W$

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!**

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 $\langle W, v, V, R\rangle \models \varphi$, $\forall v \in R(r, w)$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq \mathrm{Mods}_W$

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 $\langle W, v, V, R\rangle \models \varphi$, $\forall v \in R(r, w)$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq \mathrm{Mods}_W$
- $\langle W, w, V, R\rangle \models [r]\varphi$ iff $\mathcal{M} \models \varphi$, $\forall \mathcal{M} \in R(r, w)$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq \mathrm{Mods}_W$
- $\langle W, w, V, R\rangle \models [r]\varphi$ iff $\mathcal{M} \models \varphi$, $\forall \mathcal{M} \in R(r, w)$

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
  - $w_0 \in W$
  - $V(p) \subseteq W$
  - $R(r, w) \subseteq \mathrm{Mods}_W$
- $\langle W, w, V, R \rangle \models [r]\varphi$ iff $\mathcal{M} \models \varphi$, $\forall \mathcal{M} \in R(r, w)$
- More modal operators can be defined as classes of models

Defining Conditions

Defining condition
$\mathcal{P}_{\mathsf{A}}(\mathcal{M}) \iff R^{\mathcal{M}}(\mathsf{A}, w) = \{\langle v, |\mathcal{M}|, V^{\mathcal{M}}, R^{\mathcal{M}}\rangle \mid v \in |\mathcal{M}|\}$

Defining Conditions

Defining condition
$\mathcal{P}_{\mathsf{A}}(\mathcal{M}) \iff R^{\mathcal{M}}(\mathsf{A}, w) = \{\langle v, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}, R^{\mathcal{M}}\rangle \mid v \in \lvert\mathcal{M}\rvert\}$

Defining condition
$\mathcal{P}_{@_i}(\mathcal{M}) \iff R^{\mathcal{M}}(@_i, w) = \{\langle v, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}, R^{\mathcal{M}}\rangle \mid v \in V(i)\}, i \in \mathsf{Nom}$
$\mathcal{P}_{\downarrow i}(\mathcal{M}) \iff R^{\mathcal{M}}(\downarrow i, w) = \{\langle w, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}[i \mapsto \{w\}], R^{\mathcal{M}}\rangle\}, i \in \mathsf{Nom}$
$\mathcal{P}_{\mathsf{Nom}}(\mathcal{M}) \iff V^{\mathcal{M}}(i)$ is a singleton, $\forall i \in \mathsf{Nom}$

Defining Conditions

Defining condition
$\mathcal{P}_{\mathsf{A}}(\mathcal{M}) \iff R^{\mathcal{M}}(\mathsf{A}, w) = \{\langle v, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}, R^{\mathcal{M}}\rangle \mid v \in \lvert\mathcal{M}\rvert\}$

Defining condition
$\mathcal{P}_{@_i}(\mathcal{M}) \iff R^{\mathcal{M}}(@_i, w) = \{\langle v, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}, R^{\mathcal{M}}\rangle \mid v \in V(i)\}, i \in \mathsf{Nom}$
$\mathcal{P}_{\downarrow i}(\mathcal{M}) \iff R^{\mathcal{M}}(\downarrow i, w) = \{\langle w, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}[i \mapsto \{w\}], R^{\mathcal{M}}\rangle\}, i \in \mathsf{Nom}$
$\mathcal{P}_{\mathsf{Nom}}(\mathcal{M}) \iff V^{\mathcal{M}}(i)$ is a singleton, $\forall i \in \mathsf{Nom}$

Defining condition
$\mathcal{P}_{\texttt{r}}(\mathcal{M}) \iff R^{\mathcal{M}}(\texttt{r}, w) = \{\langle w, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}[\texttt{k} \mapsto V^{\mathcal{M}}(\texttt{k}) \cup \{w\}], R^{\mathcal{M}}\rangle\}$
$\mathcal{P}_{\texttt{f}}(\mathcal{M}) \iff R^{\mathcal{M}}(\texttt{f}, w) = \{\langle w, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}[\texttt{k} \mapsto V^{\mathcal{M}}(\texttt{k}) \setminus \{w\}], R^{\mathcal{M}}\rangle\}$
$\mathcal{P}_{\texttt{f}}(\mathcal{M}) \iff R^{\mathcal{M}}(\texttt{e}, w) = \{\langle w, \lvert\mathcal{M}\rvert, V^{\mathcal{M}}[\texttt{k} \mapsto \emptyset], R^{\mathcal{M}}\rangle\}$

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

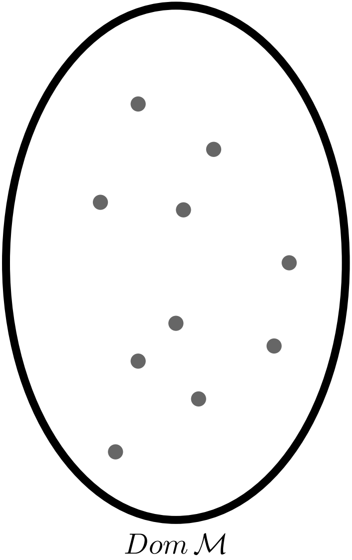
References

- Areces, C. and Gorín, D.. Coinductive models and normal forms for modal logics (or how we learned to stop worrying and love coinduction). Journal of Applied Logic, 8(4):305–318, Elsevier, 2010.

Part IV

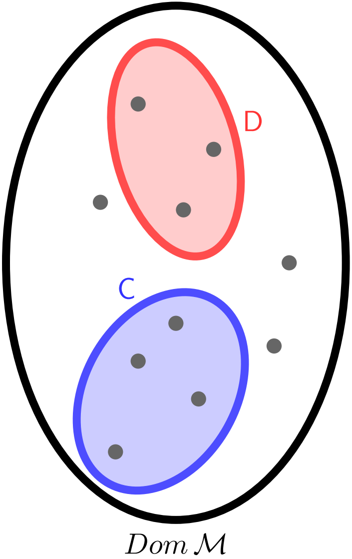
Logical methods in the generation de referring expressions

Separations and descriptions



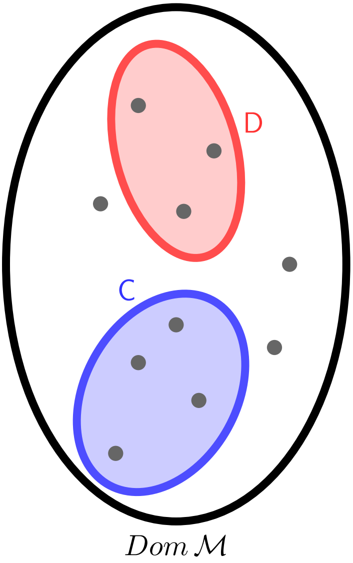
- Let $\mathcal{M}$ be a Kripke model

Separations and descriptions



- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq$ C, D $\subset Dom\, \mathcal{M}$

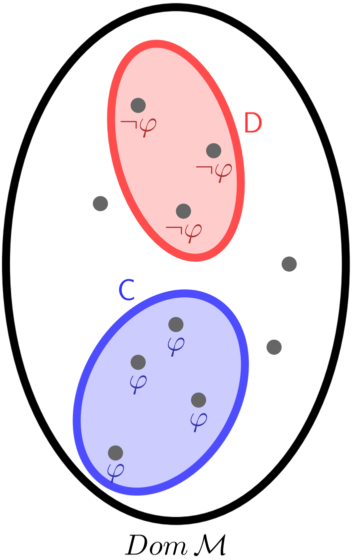
Separations and descriptions



- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq$ C, D $\subset Dom\,\mathcal{M}$
- For any formula φ, we say that:
  - φ *separates* C and D in $\mathcal{M}$ iff

    $\mathcal{M}$, C $\models \varphi$ and $\mathcal{M}$, D $\not\models \varphi$

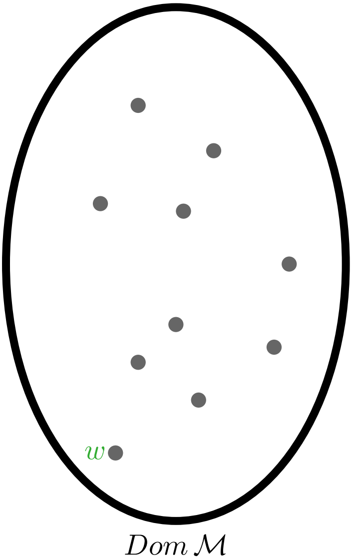
Separations and descriptions



- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq \mathsf{C}, \mathsf{D} \subset Dom\,\mathcal{M}$
- For any formula φ, we say that:
  - φ *separates* C and D in $\mathcal{M}$ iff

$$\mathcal{M}, \mathsf{C} \models \varphi \text{ and } \mathcal{M}, \mathsf{D} \not\models \varphi$$

Separations and descriptions



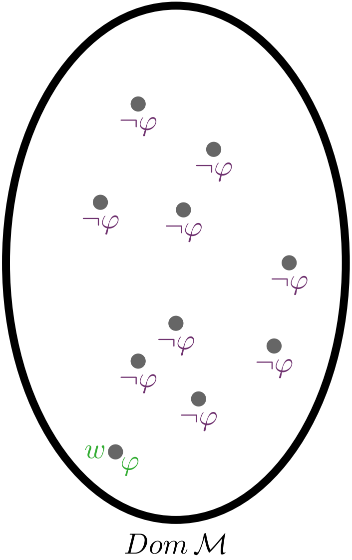
- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq \mathsf{C}, \mathsf{D} \subset Dom\,\mathcal{M}$
- For any formula φ, we say that:
  - φ *separates* C and D in $\mathcal{M}$ iff

    $\mathcal{M}, \mathsf{C} \models \varphi$ and $\mathcal{M}, \mathsf{D} \not\models \varphi$

- Similarly, for $w \in Dom\,\mathcal{M}$ we say:
  - φ *describes* w in $\mathcal{M}$ iff

    $\mathcal{M}, w \models \varphi$ and $\mathcal{M}, v \not\models \varphi \forall v \neq w$

Separations and descriptions



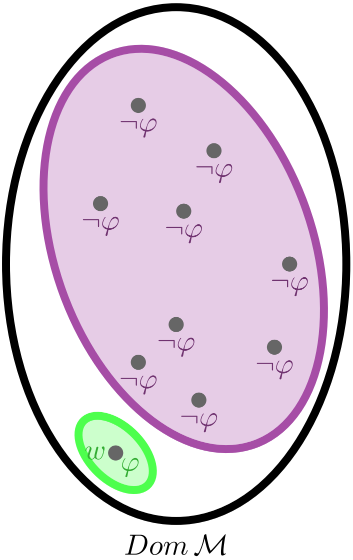
- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq \mathsf{C}, \mathsf{D} \subset Dom\,\mathcal{M}$
- For any formula φ, we say that:
  - φ *separates* C and D in $\mathcal{M}$ iff

$$\mathcal{M}, \mathsf{C} \models \varphi \text{ and } \mathcal{M}, \mathsf{D} \not\models \varphi$$

- Similarly, for $w \in Dom\,\mathcal{M}$ we say:
  - φ *describes* w in $\mathcal{M}$ iff

$$\mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, v \not\models \varphi \forall v \neq w$$

Separations and descriptions



- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq \mathsf{C}, \mathsf{D} \subset Dom\,\mathcal{M}$
- For any formula φ, we say that:
  - φ *separates* C and D in $\mathcal{M}$ iff

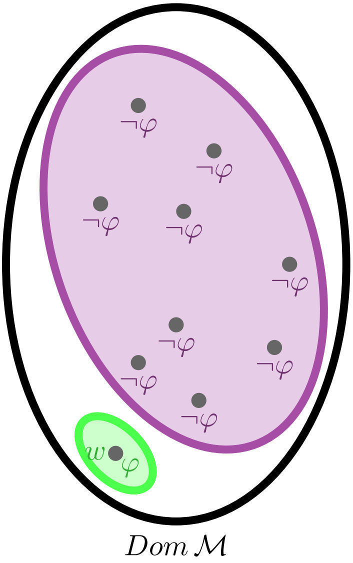
$$\mathcal{M}, \mathsf{C} \models \varphi \text{ and } \mathcal{M}, \mathsf{D} \not\models \varphi$$

- Similarly, for $w \in Dom\,\mathcal{M}$ we say:
  - φ *describes* w in $\mathcal{M}$ iff

$$\mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, v \not\models \varphi \forall v \neq w$$

- Description is a form of separation

Separations and descriptions



- Let $\mathcal{M}$ be a Kripke model
- And let $\emptyset \neq \mathsf{C}, \mathsf{D} \subset Dom\,\mathcal{M}$
- For any formula φ, we say that:
  - φ *separates* C and D in $\mathcal{M}$ iff

    $\mathcal{M}, \mathsf{C} \models \varphi$ and $\mathcal{M}, \mathsf{D} \not\models \varphi$

- Similarly, for $w \in Dom\,\mathcal{M}$ we say:
  - φ *describes* w in $\mathcal{M}$ iff

    $\mathcal{M}, w \models \varphi$ and $\mathcal{M}, v \not\models \varphi \forall v \neq w$

- Description is a form of separation
- φ could be of any suitable logic

Separation and description problems

The separation problem

Given a finite model $\mathcal{M}$ and sets $\mathsf{C}, \mathsf{D} \subset Dom\,\mathcal{M}$, find a φ that separates C and D, if possible.

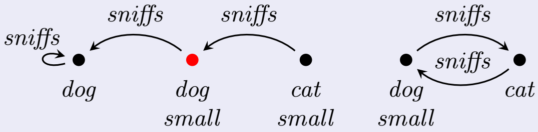
The description problem

Given a finite model $\mathcal{M}$ and a world $w \in Dom\,\mathcal{M}$, find a φ that describes w, if possible.

- They can be seen as another kind of inference task
- But they didn't receive much attention so far
- We are interested in their computational properties

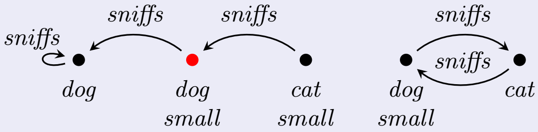
Motivation: Generation of Referring Expressions

An application of logics in Natural Language Generation



Motivation: Generation of Referring Expressions

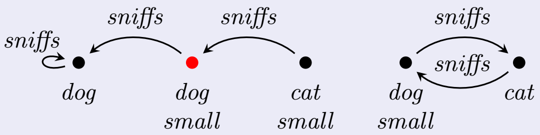
An application of logics in Natural Language Generation



- "the dog that is sniffing another dog"

Motivation: Generation of Referring Expressions

An application of logics in Natural Language Generation



content determination

(e.g., as a FOL formula)

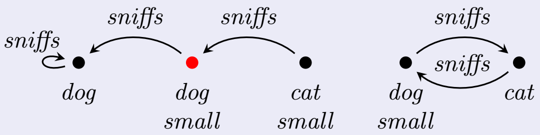
- $dog(x) \wedge \exists y.(x \not\approx y \wedge dog(y) \wedge sniffs(x, y))$

surface realization

- "the dog that is sniffing another dog"

Motivation: Generation of Referring Expressions

An application of logics in Natural Language Generation



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

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

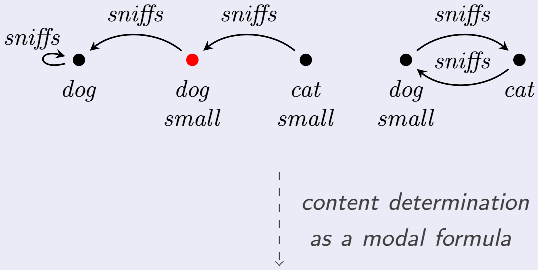
surface realization

- "the dog that is sniffing another dog"

(logical) content determination $\approx$ description problem

Motivation: Generation of Referring Expressions

An application of logics in Natural Language Generation

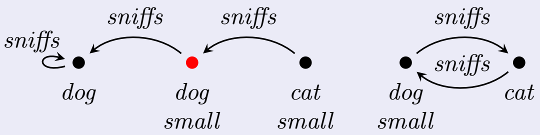


$$dog \wedge small \wedge \langle sniffs \rangle dog$$

(logical) content determination $\approx$ description problem

Motivation: Generation of Referring Expressions

An application of logics in Natural Language Generation



content determination as a modal formula

• $dog \wedge small \wedge \langle sniffs \rangle dog$

surface realization

• "the small dog that is sniffing a dog"

(logical) content determination $\approx$ description problem

Motivation: Modal logics in the GRE

Areces, Koller & Striegnitz (2008)

- We propose modal logics for content determination:

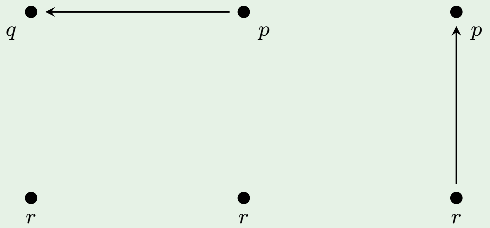
  $\mathcal{ML}$ – the basic modal language $(\neg, \wedge, \Diamond)$

  $\mathcal{EL}$ – the existential positive fragment of $\mathcal{ML}$ $(\wedge, \Diamond)$
- *Rationale*:
  - Good expressive power
  - Simple surface realization algorithms
  - Relatively low computational complexity for inference tasks
- In particular, we show that:

  "The modal description problem needs polynomial time"

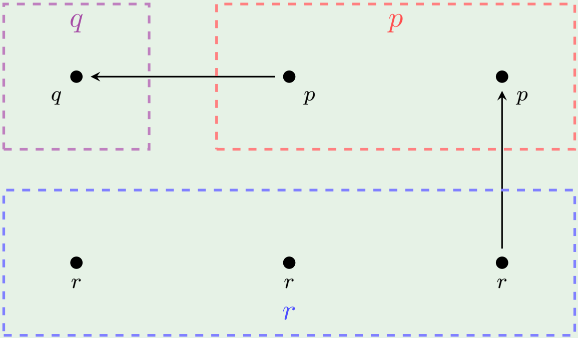
The modal description problem in polynomial time

A variation of Tarjan's bisimulation contraction algorithm



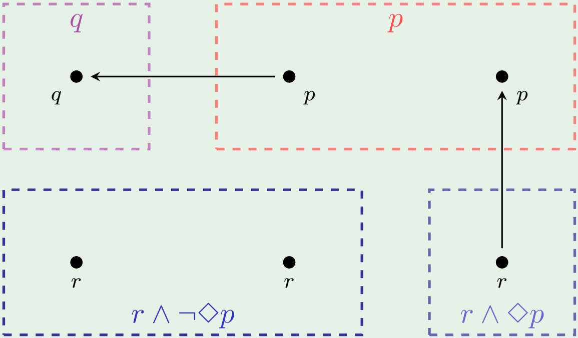
The modal description problem in polynomial time

A variation of Tarjan's bisimulation contraction algorithm



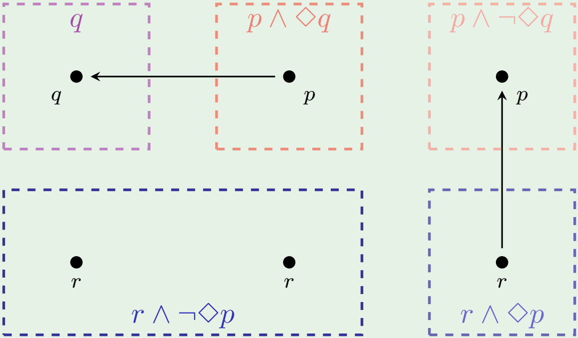
The modal description problem in polynomial time

A variation of Tarjan's bisimulation contraction algorithm



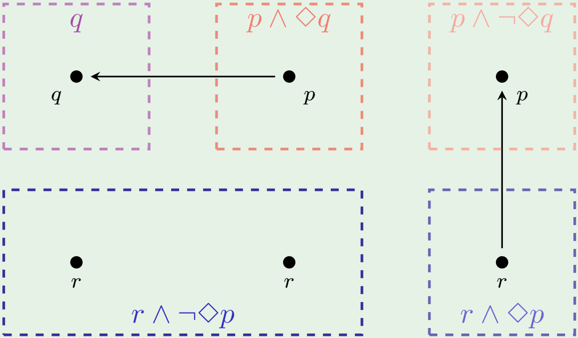
The modal description problem in polynomial time

A variation of Tarjan's bisimulation contraction algorithm



The modal description problem in polynomial time

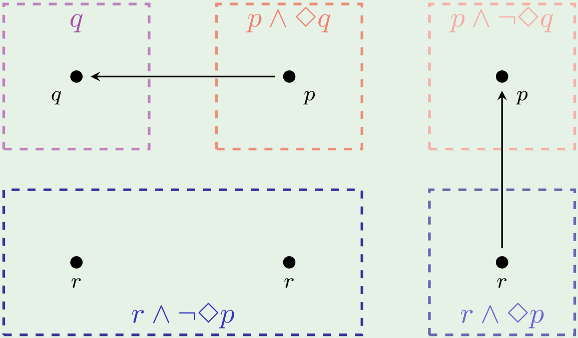
A variation of Tarjan's bisimulation contraction algorithm



- Tarjan's algorithm runs in polynomial time
- Hence, the modal description problem is polynomial

The modal description problem in polynomial time

A variation of Tarjan's bisimulation contraction algorithm



- Tarjan's algorithm runs in polynomial time
- Hence, the modal description problem is polynomial
- But this is assuming that $\wedge$ takes constant time!

The modal description problem in polynomial time

for DAG representation!

- This algorithm produces a formula represented as a DAG

The modal description problem in polynomial time

for DAG representation!

- This algorithm produces a formula represented as a DAG
- The size of the DAG is polynomial in the size of the model

The modal description problem in polynomial time

for DAG representation!

- This algorithm produces a formula represented as a DAG
- The size of the DAG is polynomial in the size of the model
- Surface realization step doesn't exploit DAG representation

The modal description problem in polynomial time

for DAG representation!

- This algorithm produces a formula represented as a DAG
- The size of the DAG is polynomial in the size of the model
- Surface realization step doesn't exploit DAG representation
  - Most probably can't be done anyway

The modal description problem in polynomial time

for DAG representation!

- This algorithm produces a formula represented as a DAG
- The size of the DAG is polynomial in the size of the model
- Surface realization step doesn't exploit DAG representation
  - Most probably can't be done anyway
- Is the *tree* representation of this formula also polynomial?

The modal description problem in polynomial time

for DAG representation!

- This algorithm produces a formula represented as a DAG
- The size of the DAG is polynomial in the size of the model
- Surface realization step doesn't exploit DAG representation
  - Most probably can't be done anyway
- Is the *tree* representation of this formula also polynomial?
- If not, "modal content determination" can't be said to take polynomial time

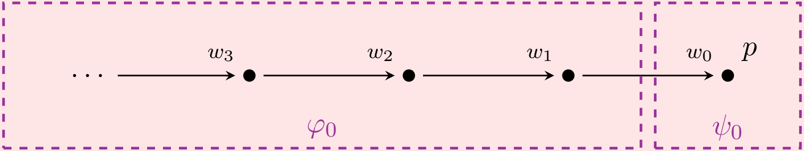
The modal description problem in polynomial time

also for tree representation?

$$\cdots \longrightarrow w_3 \longrightarrow w_2 \longrightarrow w_1 \longrightarrow w_0 \quad p$$

The modal description problem in polynomial time

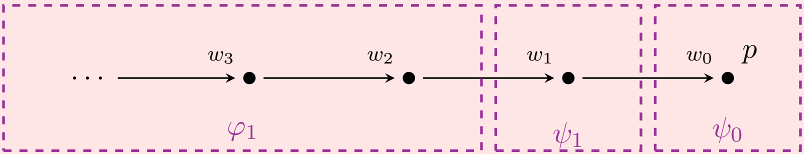
also for tree representation?



$\varphi_0 = \neg p$ $\psi_0 = p$

The modal description problem in polynomial time

also for tree representation?



$\varphi_0 = \neg p$

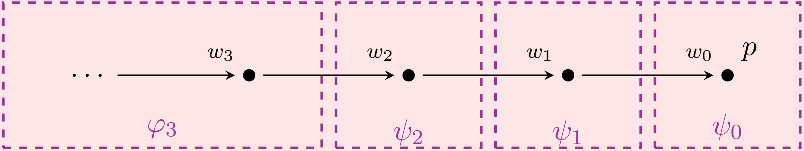
$\varphi_1 = \varphi_0 \wedge \neg\Diamond\psi_0$

$\psi_0 = p$

$\psi_1 = \varphi_0 \wedge \Diamond\psi_0$

The modal description problem in polynomial time

also for tree representation?



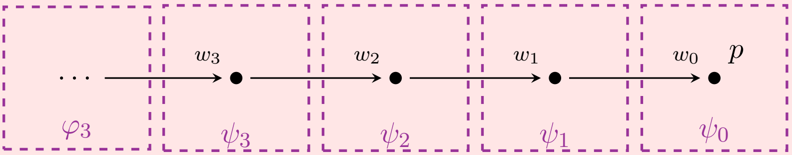
$$\varphi_0 = \neg p \qquad \psi_0 = p$$

$$\varphi_1 = \varphi_0 \wedge \neg\Diamond\psi_0 \qquad \psi_1 = \varphi_0 \wedge \Diamond\psi_0$$

$$\varphi_2 = \varphi_1 \wedge \neg\Diamond\psi_1 \qquad \psi_2 = \varphi_1 \wedge \Diamond\psi_1$$

The modal description problem in polynomial time

also for tree representation?



$$\varphi_0 = \neg p \qquad \psi_0 = p$$

$$\varphi_1 = \varphi_0 \wedge \neg\Diamond\psi_0 \qquad \psi_1 = \varphi_0 \wedge \Diamond\psi_0$$

$$\varphi_2 = \varphi_1 \wedge \neg\Diamond\psi_1 \qquad \psi_2 = \varphi_1 \wedge \Diamond\psi_1$$

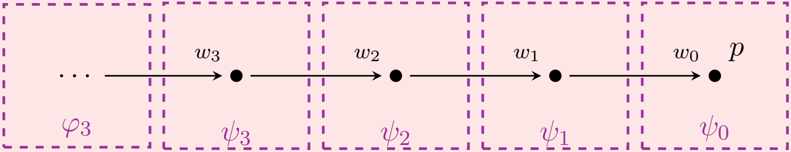
$$\vdots \qquad \vdots$$

$$\varphi_{i+1} = \varphi_i \wedge \neg\Diamond\psi_i \qquad \psi_{i+1} = \varphi_i \wedge \Diamond\psi_i$$

- Each ψ_i is description for w_i with size exponential in i

The modal description problem in polynomial time

also for tree representation?



$$
\begin{aligned}
\varphi_0 &= \neg p & \psi_0 &= p \\
\varphi_1 &= \varphi_0 \wedge \neg\Diamond\psi_0 & \psi_1 &= \varphi_0 \wedge \Diamond\psi_0 \\
\varphi_2 &= \varphi_1 \wedge \neg\Diamond\psi_1 & \psi_2 &= \varphi_1 \wedge \Diamond\psi_1 \\
&\vdots & &\vdots \\
\varphi_{i+1} &= \varphi_i \wedge \neg\Diamond\psi_i & \psi_{i+1} &= \varphi_i \wedge \Diamond\psi_i
\end{aligned}
$$

- Each ψ_i is description for w_i with size exponential in i
- Observe that w_i admits a linear description: $\underbrace{\Diamond\Diamond\ldots\Diamond}_{i \text{ times}}p$

Where do we go from here?

- The example shows that this algorithm is not polynomial
- Can we *fix* it?
- Can we find *another* one that is indeed polynomial?

Where do we go from here?

- The example shows that this algorithm is not polynomial
- Can we *fix* it?
- Can we find *another* one that is indeed polynomial?
- We show that no such algorithm exists!

Bounds for the separation / description problems

Basic modal language $\mathcal{ML}$

Theorem (Lower bound)

Any upper bound for the size of a solution for the separation or description problem for $\mathcal{ML}$ *is at least exponential.*

Corollary

No polynomial time algorithm exists that solves the description or separation problem returning the formula as a tree.

Bounds for the separation / description problems

Basic modal language $\mathcal{ML}$

Theorem (Lower bound)

Any upper bound for the size of a solution for the separation or description problem for $\mathcal{ML}$ is at least exponential.

Corollary

No polynomial time algorithm exists that solves the description or separation problem returning the formula as a tree.

Theorem (Upper bound)

If $\varphi \in \mathcal{ML}$ is a minimum description for v in $\mathcal{M} = \langle W, R, V \rangle$, then $|\varphi| \in O(2^{\frac{1}{2}|W|^2} \cdot |V|)$.