Relation-Changing Modal Logics

Raul Fervari

Grupo LIIS (Logics, Interaction and Intelligent Systems)
FaMAF, Universidad Nacional de Córdoba, Argentina

Tesis presentada para optar al título de Doctor en Ciencias de la Computación
5 de Marzo de 2014

Acerca de la tesis

- Lógicas Modales.

Acerca de la tesis

- Lógicas Modales.
- Lógicas Modales Dinámicas.
  - Ejemplos.

Acerca de la tesis

- Lógicas Modales.
- Lógicas Modales Dinámicas.
  - Ejemplos.
- *"Relation-Changing Modal Logics"*.
  - Conociendo las nuevas primitivas.
  - Bisimulaciones y poder expresivo.
  - Comportamiento computacional.

Acerca de la tesis

- Lógicas Modales.
- Lógicas Modales Dinámicas.
  - Ejemplos.
- *"Relation-Changing Modal Logics"*.
  - Conociendo las nuevas primitivas.
  - Bisimulaciones y poder expresivo.
  - Comportamiento computacional.
- Un nuevo enfoque: Lógicas Dinámicas Epistémicas.

Lógicas Modales

Lógicas Modales

- Los lenguajes de la familia de las lógicas modales pueden usarse en áreas muy diversas:

Lógicas Modales

- Los lenguajes de la familia de las lógicas modales pueden usarse en áreas muy diversas:
  - Verificación de Software y Hardware.
  - Representación de Conocimientos.
  - Criptografía.
  - Inteligencia Artificial.
  - Filosofía, Epistemología.
  - Lingüística Computacional.
- Por qué?

Lógicas Modales

- Los lenguajes de la familia de las lógicas modales pueden usarse en áreas muy diversas:
  - Verificación de Software y Hardware.
  - Representación de Conocimientos.
  - Criptografía.
  - Inteligencia Artificial.
  - Filosofía, Epistemología.
  - Lingüística Computacional.
- Por qué? Muchas cosas pueden ser representadas como grafos!!

Lógicas Modales

- Los lenguajes de la familia de las lógicas modales pueden usarse en áreas muy diversas:
  - Verificación de Software y Hardware.
  - Representación de Conocimientos.
  - Criptografía.
  - Inteligencia Artificial.
  - Filosofía, Epistemología.
  - Lingüística Computacional.
- Por qué? Muchas cosas pueden ser representadas como grafos!! Los lenguajes modales están diseñados especialmente para razonar y describir propiedades de grafos.

Lógicas Modales

- Los lenguajes de la familia de las lógicas modales pueden usarse en áreas muy diversas:
  - Verificación de Software y Hardware.
  - Representación de Conocimientos.
  - Criptografía.
  - Inteligencia Artificial.
  - Filosofía, Epistemología.
  - Lingüística Computacional.
- Por qué? Muchas cosas pueden ser representadas como grafos!! Los lenguajes modales están diseñados especialmente para razonar y describir propiedades de grafos.
- Hablamos de *lógicas* (en plural):
  - $\Diamond\varphi$, $\Diamond^{-}\varphi$, $\mathsf{E}\varphi$, $\Diamond_{\geq n}\varphi$, $\Diamond^{*}\varphi$.

$\mathcal{ML}$ - Sintaxis

$\mathcal{ML}$ - Sintaxis

Las fórmulas del lenguaje modal básico se contruyen a partir de:

- El lenguaje proposicional: p, q,..., $\wedge$, $\neg$, etc.
- Los operadores modales: $\Diamond, \Box$.

$\mathcal{ML}$ - Sintaxis

Las fórmulas del lenguaje modal básico se contruyen a partir de:

- El lenguaje proposicional: p, q,..., $\wedge$, $\neg$, etc.
- Los operadores modales: $\Diamond, \Box$.

Formalmente:

$$\text{FORM} ::= \bot \mid p \mid \neg\varphi \mid \varphi \wedge \varphi' \mid \Diamond\varphi,$$

where $p \in \text{PROP}$, $\varphi, \varphi' \in \text{FORM}$.

$\mathcal{ML}$ - Modelos

Las fórmulas son interpretadas sobre modelos relacionales:

$$\mathcal{M} = \langle W, R, V \rangle$$

- W es un conjunto no vacío de elementos.
- R es una relación binaria sobre W.
- $V : \text{PROP} \to \mathcal{P}(W)$ el conjunto de estados donde vale cada símbolo proposicional.

$\mathcal{ML}$ - Modelos

Las fórmulas son interpretadas sobre modelos relacionales:

$$\mathcal{M} = \langle W, R, V \rangle$$

- W es un conjunto no vacío de elementos.
- R es una relación binaria sobre W.
- $V : \text{PROP} \to \mathcal{P}(W)$ el conjunto de estados donde vale cada símbolo proposicional.

Intuitivamente, un modelo es un grafo dirigido de nodos etiquetados.

$\mathcal{ML}$ - Semántica

$\mathcal{ML}$ - Semántica

Dado un modelo $\mathcal{M}$, un estado w en $\mathcal{M}$ y una fórmula φ escribimos $\mathcal{M}, w \models \varphi$ para decir que w satisface φ en $\mathcal{M}$.

$\mathcal{ML}$ - Semántica

Dado un modelo $\mathcal{M}$, un estado w en $\mathcal{M}$ y una fórmula φ escribimos $\mathcal{M}, w \models \varphi$ para decir que w satisface φ en $\mathcal{M}$.

- Los operadores proposicionales se interpretan como siempre:
  - $\mathcal{M}, w \models p$ sii $w \in V(p)$.

$\mathcal{ML}$ - Semántica

Dado un modelo $\mathcal{M}$, un estado w en $\mathcal{M}$ y una fórmula φ escribimos $\mathcal{M}, w \models \varphi$ para decir que w satisface φ en $\mathcal{M}$.

- Los operadores proposicionales se interpretan como siempre:
  - $\mathcal{M}, w \models p$ sii $w \in V(p)$.
  - $\mathcal{M}, w \models \neg\varphi$ sii w no satisface φ.

$\mathcal{ML}$ - Semántica

Dado un modelo $\mathcal{M}$, un estado w en $\mathcal{M}$ y una fórmula φ escribimos $\mathcal{M}, w \models \varphi$ para decir que w satisface φ en $\mathcal{M}$.

- Los operadores proposicionales se interpretan como siempre:
  - $\mathcal{M}, w \models p$ sii $w \in V(p)$.
  - $\mathcal{M}, w \models \neg\varphi$ sii w no satisface φ.
  - $\mathcal{M}, w \models \varphi \wedge \varphi'$ sii w satisface ambas φ y φ'.

$\mathcal{ML}$ - Semántica

Dado un modelo $\mathcal{M}$, un estado w en $\mathcal{M}$ y una fórmula φ escribimos $\mathcal{M}, w \models \varphi$ para decir que w satisface φ en $\mathcal{M}$.

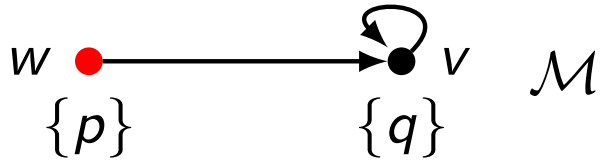
- Los operadores proposicionales se interpretan como siempre:
  - $\mathcal{M}, w \models p$ sii $w \in V(p)$.
  - $\mathcal{M}, w \models \neg\varphi$ sii w no satisface φ.
  - $\mathcal{M}, w \models \varphi \wedge \varphi'$ sii w satisface ambas φ y φ'.
- $\mathcal{M}, w \models \Diamond\varphi$ sii φ se satisface en algún sucesor de w.

$\mathcal{ML}$ - Semántica

Dado un modelo $\mathcal{M}$, un estado w en $\mathcal{M}$ y una fórmula φ escribimos $\mathcal{M}, w \models \varphi$ para decir que w satisface φ en $\mathcal{M}$.

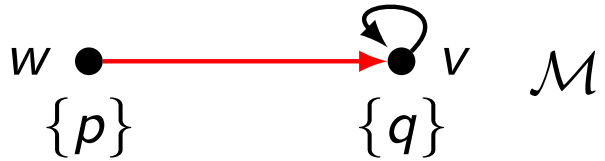
- Los operadores proposicionales se interpretan como siempre:
  - $\mathcal{M}, w \models p$ sii $w \in V(p)$.
  - $\mathcal{M}, w \models \neg\varphi$ sii w no satisface φ.
  - $\mathcal{M}, w \models \varphi \wedge \varphi'$ sii w satisface ambas φ y φ'.
- $\mathcal{M}, w \models \Diamond\varphi$ sii φ se satisface en algún sucesor de w.
- $\mathcal{M}, w \models \Box\varphi$ sii φ se satisface en todos los sucesores de w.

$\mathcal{ML}$ - Ejemplo



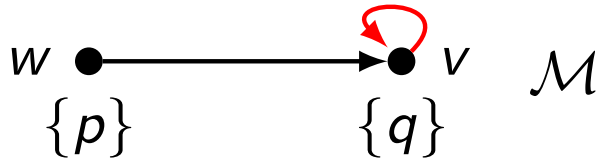
$$\mathcal{M}, w \models \Diamond\Diamond p?$$

$\mathcal{ML}$ - Ejemplo



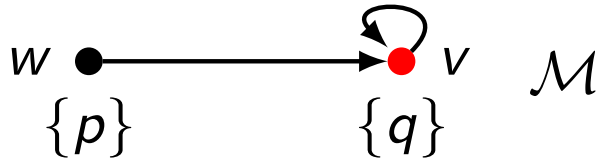
$$\mathcal{M}, w \models \Diamond\Diamond p?$$

$\mathcal{ML}$ - Ejemplo



$$\mathcal{M}, v \not\models \Diamond p$$

$\mathcal{ML}$ - Ejemplo



$$\mathcal{M}, v \not\models p$$

$\mathcal{ML}$ - Bisimulaciones

Sean $\mathcal{M} = \langle W, R, V \rangle$, $\mathcal{M}' = \langle W', R', V' \rangle$ dos modelos. Una relación no vacía $Z \subseteq W \times W'$ es una $\mathcal{ML}$-bisimulación si satisface las siguientes condiciones. Si wZw' entonces

(atomic harmony) para todo $p \in \text{PROP}$, $w \in V(p)$ sii $w' \in V'(p)$;

$\mathcal{ML}$ - Bisimulaciones

Sean $\mathcal{M} = \langle W, R, V \rangle$, $\mathcal{M}' = \langle W', R', V' \rangle$ dos modelos. Una relación no vacía $Z \subseteq W \times W'$ es una $\mathcal{ML}$-bisimulación si satisface las siguientes condiciones. Si wZw' entonces

(atomic harmony) para todo $p \in \text{PROP}$, $w \in V(p)$ sii $w' \in V'(p)$;

(zig) si $(w, v) \in R$ entonces existe v', $(w', v') \in R'$ y vZv';

$\mathcal{ML}$ - Bisimulaciones

Sean $\mathcal{M} = \langle W, R, V \rangle$, $\mathcal{M}' = \langle W', R', V' \rangle$ dos modelos. Una relación no vacía $Z \subseteq W \times W'$ es una $\mathcal{ML}$-bisimulación si satisface las siguientes condiciones. Si wZw' entonces

(atomic harmony) para todo $p \in \text{PROP}$, $w \in V(p)$ sii $w' \in V'(p)$;

(zig) si $(w, v) \in R$ entonces existe v', $(w', v') \in R'$ y vZv';

(zag) si $(w', v') \in R'$ entonces existe v, $(w, v) \in R$ y vZv'.

$\mathcal{ML}$ - Bisimulaciones

Sean $\mathcal{M} = \langle W, R, V \rangle$, $\mathcal{M}' = \langle W', R', V' \rangle$ dos modelos. Una relación no vacía $Z \subseteq W \times W'$ es una $\mathcal{ML}$-bisimulación si satisface las siguientes condiciones. Si wZw' entonces

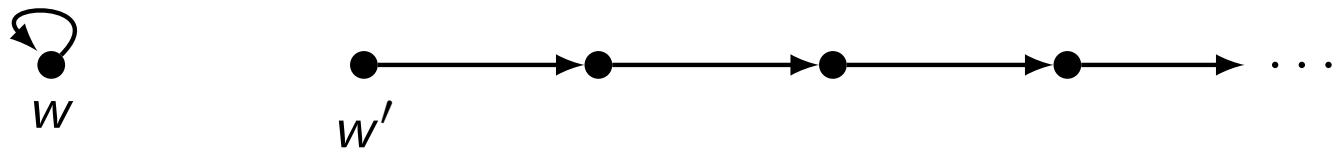
(atomic harmony) para todo $p \in \text{PROP}$, $w \in V(p)$ sii $w' \in V'(p)$;

(zig) si $(w, v) \in R$ entonces existe v', $(w', v') \in R'$ y vZv';

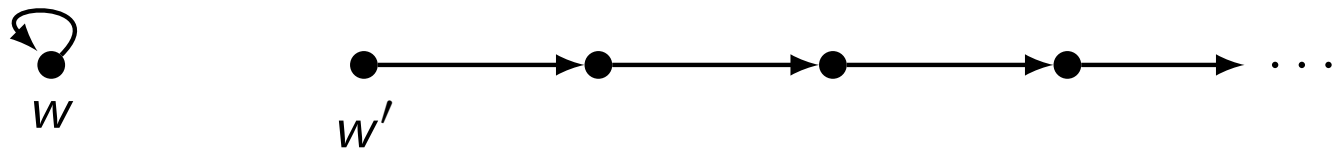
(zag) si $(w', v') \in R'$ entonces existe v, $(w, v) \in R$ y vZv'.

Decimos que $\mathcal{M}, w$ y $\mathcal{M}', w'$ son bisimilares si existe una bisimulación Z tal que wZw' ($\mathcal{M}, w \leftrightarroweq_{\mathcal{ML}} \mathcal{M}', w'$).

$\mathcal{ML}$ - Bisimulaciones (Ejemplo)

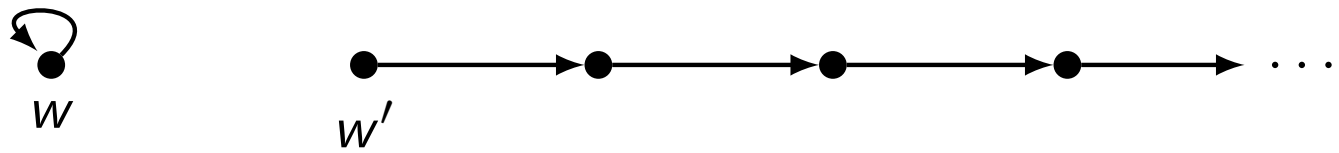


$\mathcal{ML}$ - Bisimulaciones (Ejemplo)

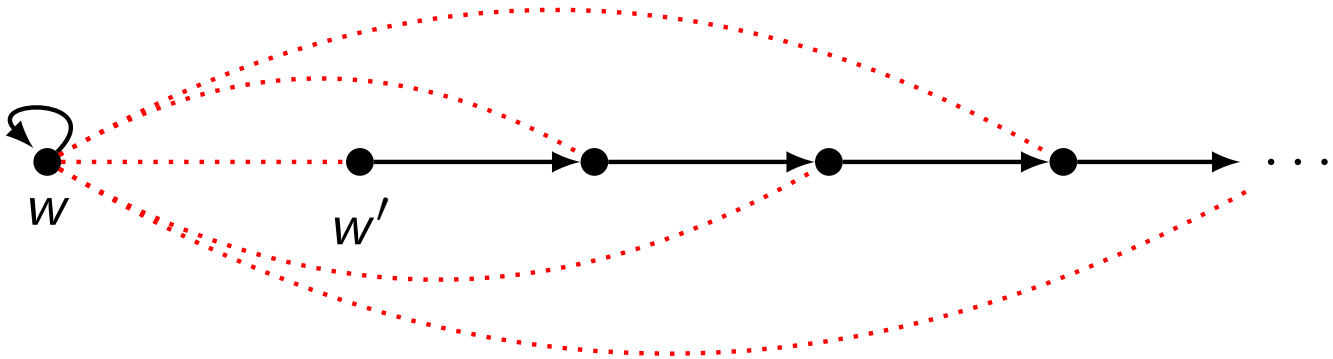


$\exists x, y.(x \neq y)$

$\mathcal{ML}$ - Bisimulaciones (Ejemplo)



$\mathcal{ML}$ - Bisimulaciones (Ejemplo)



$\mathcal{ML}$ - Teorema de Invarianza

Teorema

Sean $\mathcal{M} = \langle W, R, V \rangle$, $\mathcal{M}' = \langle W', R', V' \rangle$ *dos modelos,* $w \in W$ *ya* $w' \in W'$.
$\mathcal{M}, w \leftrightarrows_{\mathcal{ML}} \mathcal{M}', w'$ *implica* $\mathcal{M}, w \equiv_{\mathcal{ML}} \mathcal{M}', w'$.

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.
+ Preservación de propiedades via bisimulación.

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.
+ Preservación de propiedades via bisimulación.
+ Tree/Finite-Model Property.

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.
+ Preservación de propiedades via bisimulación.
+ Tree/Finite-Model Property.
+ Buen comportamiento computacional (SAT PSPACE-completo, model checking en P).

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.
+ Preservación de propiedades via bisimulación.
+ Tree/Finite-Model Property.
+ Buen comportamiento computacional (SAT PSPACE-completo, model checking en P).
- El poder expresivo puede ser insuficiente.

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.
+ Preservación de propiedades via bisimulación.
+ Tree/Finite-Model Property.
+ Buen comportamiento computacional (SAT PSPACE-completo, model checking en P).
- El poder expresivo puede ser insuficiente.
- Lenguaje completamente estático: puede resultar complicado expresar cambios sobre estructuras.

$\mathcal{ML}$ - Pros y Contras

+ Apropiado para describir propiedades de estructuras relacionales.
+ Preservación de propiedades via bisimulación.
+ Tree/Finite-Model Property.
+ Buen comportamiento computacional (SAT PSPACE-completo, model checking en P).
- El poder expresivo puede ser insuficiente.
- Lenguaje completamente estático: puede resultar complicado expresar cambios sobre estructuras.

Qué pasa si necesitamos modelar situaciones en las cuales el grafo puede cambiar después de la aplicación de ciertas operaciones?

PDL - Propositional Dynamic Logic

Podemos expresar algunas propiedades interesantes con fórmulas **PDL**:

$$\langle(\varphi?; a)^*; (\neg\varphi)?\rangle\psi$$

PDL - Propositional Dynamic Logic

Podemos expresar algunas propiedades interesantes con fórmulas **PDL**:

$$\langle(\varphi?; a)^*; (\neg\varphi)?\rangle\psi$$

representa el programa

"**while** φ **do** a" termina en un estado que satisface ψ.

PDL - Propositional Dynamic Logic

Podemos expresar algunas propiedades interesantes con fórmulas **PDL**:

$$\langle(\varphi?; a)^*; (\neg\varphi)?\rangle\psi$$

representa el programa

"**while** φ **do** a" termina en un estado que satisface ψ.

+ **PDL** nos permite razonar acerca de cambios de estado.

PDL - Propositional Dynamic Logic

Podemos expresar algunas propiedades interesantes con fórmulas **PDL**:

$$\langle(\varphi?; a)^*; (\neg\varphi)?\rangle\psi$$

representa el programa

"**while** φ **do** a" termina en un estado que satisface ψ.

+ **PDL** nos permite razonar acerca de cambios de estado.
- Ésta es una noción demasiado débil de comportamiento dinámico.

PDL - Propositional Dynamic Logic

Podemos expresar algunas propiedades interesantes con fórmulas **PDL**:

$$\langle(\varphi?; a)^*; (\neg\varphi)?\rangle\psi$$

representa el programa

"**while** φ **do** a" termina en un estado que satisface ψ.

+ **PDL** nos permite razonar acerca de cambios de estado.
  - Ésta es una noción demasiado débil de comportamiento dinámico.
  - No nos permite modificar el modelo: sólo podemos hablar de ejecuciones de programas.

Lógicas Dinámicas - Ejemplos

- Hybrid Logics ($\mathcal{HL}(@, \downarrow)$): $\downarrow$ cambia la etiqueta de los estados en el modelo, y @ permite saltar a un estado nombrado.

Lógicas Dinámicas - Ejemplos

- Hybrid Logics ($\mathcal{HL}(@, \downarrow)$): $\downarrow$ cambia la etiqueta de los estados en el modelo, y @ permite saltar a un estado nombrado.
- Memory Logics ($\mathcal{ML}(\textcircled{r}, \textcircled{k})$): $\textcircled{r}$ almacena el estado actual en la memoria y $\textcircled{k}$ permite consultar si éste ya está en la memoria.

Lógicas Dinámicas - Ejemplos

- Hybrid Logics ($\mathcal{HL}(@, \downarrow)$): $\downarrow$ cambia la etiqueta de los estados en el modelo, y @ permite saltar a un estado nombrado.
- Memory Logics ($\mathcal{ML}(ⓡ, ⓚ)$): ⓡ almacena el estado actual en la memoria y ⓚ permite consultar si éste ya está en la memoria.
- Arrow Updates (AUL): preserva solo las aristas del modelo que satisfacen una pre y una post-condición.

Lógicas Dinámicas - Ejemplos

- Hybrid Logics ($\mathcal{HL}(@, \downarrow)$): $\downarrow$ cambia la etiqueta de los estados en el modelo, y @ permite saltar a un estado nombrado.
- Memory Logics ($\mathcal{ML}(ⓡ, ⓚ)$): ⓡ almacena el estado actual en la memoria y ⓚ permite consultar si éste ya está en la memoria.
- Arrow Updates (AUL): preserva solo las aristas del modelo que satisfacen una pre y una post-condición.
- Sabotage Logic (*SML*): elimina estados arbitrariamente en el modelo.

Lógicas Dinámicas - Ejemplos

- Hybrid Logics ($\mathcal{HL}(@, \downarrow)$): $\downarrow$ cambia la etiqueta de los estados en el modelo, y @ permite saltar a un estado nombrado.
- Memory Logics ($\mathcal{ML}(ⓡ, ⓚ)$): ⓡ almacena el estado actual en la memoria y ⓚ permite consultar si éste ya está en la memoria.
- Arrow Updates (AUL): preserva solo las aristas del modelo que satisfacen una pre y una post-condición.
- Sabotage Logic (SML): elimina estados arbitrariamente en el modelo.
- Public Announcement Logic ($\mathcal{PAL}$): elimina todos los estados que no satisfacen cierto anuncio público.

Relation-Changing Modal Logics

Definimos nuestros propios operadores dinámicos. Intuitivamente:

- $\langle \mathsf{sw} \rangle \varphi$: atravesar alguna arista, invertirla, luego evaluar φ.

Relation-Changing Modal Logics

Definimos nuestros propios operadores dinámicos. Intuitivamente:

- $\langle \text{sw} \rangle \varphi$: atravesar alguna arista, invertirla, luego evaluar φ.
- $\langle \text{gsw} \rangle \varphi$: invertir alguna arista, luego evaluar φ.

Relation-Changing Modal Logics

Definimos nuestros propios operadores dinámicos. Intuitivamente:

- $\langle \text{sw} \rangle \varphi$: atravesar alguna arista, invertirla, luego evaluar φ.
- $\langle \text{gsw} \rangle \varphi$: invertir alguna arista, luego evaluar φ.
- $\langle \text{sb} \rangle \varphi$: atravesar alguna arista, borrarla, luego evaluar φ.

Relation-Changing Modal Logics

Definimos nuestros propios operadores dinámicos. Intuitivamente:

- $\langle \text{sw} \rangle \varphi$: atravesar alguna arista, invertirla, luego evaluar φ.
- $\langle \text{gsw} \rangle \varphi$: invertir alguna arista, luego evaluar φ.
- $\langle \text{sb} \rangle \varphi$: atravesar alguna arista, borrarla, luego evaluar φ.
- $\langle \text{gsb} \rangle \varphi$: borrar cualquier arista, luego evaluar φ.

Relation-Changing Modal Logics

Definimos nuestros propios operadores dinámicos. Intuitivamente:

- $\langle \text{sw} \rangle \varphi$: atravesar alguna arista, invertirla, luego evaluar φ.
- $\langle \text{gsw} \rangle \varphi$: invertir alguna arista, luego evaluar φ.
- $\langle \text{sb} \rangle \varphi$: atravesar alguna arista, borrarla, luego evaluar φ.
- $\langle \text{gsb} \rangle \varphi$: borrar cualquier arista, luego evaluar φ.
- $\langle \text{br} \rangle \varphi$: agregar una nueva arista, atravesarla, luego evaluar φ.

Relation-Changing Modal Logics

Definimos nuestros propios operadores dinámicos. Intuitivamente:

- $\langle \text{sw} \rangle \varphi$: atravesar alguna arista, invertirla, luego evaluar φ.
- $\langle \text{gsw} \rangle \varphi$: invertir alguna arista, luego evaluar φ.
- $\langle \text{sb} \rangle \varphi$: atravesar alguna arista, borrarla, luego evaluar φ.
- $\langle \text{gsb} \rangle \varphi$: borrar cualquier arista, luego evaluar φ.
- $\langle \text{br} \rangle \varphi$: agregar una nueva arista, atravesarla, luego evaluar φ.
- $\langle \text{gbr} \rangle \varphi$: agregar una nueva arista, luego evaluar φ.

Relation-Changing Modal Logics

Dado un modelo $\mathcal{M} = \langle W, R, V \rangle$, definimos las siguientes notaciones:

(sabotaging) $\mathcal{M}^-_{wv} = \langle W, R^-_{wv}, V \rangle$, with $R^-_{wv} = R \backslash \{(w, v)\}$.

Relation-Changing Modal Logics

Dado un modelo $\mathcal{M} = \langle W, R, V \rangle$, definimos las siguientes notaciones:

(sabotaging) $\mathcal{M}^-_{wv} = \langle W, R^-_{wv}, V \rangle$, with $R^-_{wv} = R \backslash \{(w, v)\}$.

(swapping) $\mathcal{M}^*_{wv} = \langle W, R^*_{wv}, V \rangle$, with $R^*_{wv} = (R \backslash \{(v, w)\}) \cup \{(w, v)\}$.

Relation-Changing Modal Logics

Dado un modelo $\mathcal{M} = \langle W, R, V \rangle$, definimos las siguientes notaciones:

(sabotaging) $\mathcal{M}^-_{wv} = \langle W, R^-_{wv}, V \rangle$, with $R^-_{wv} = R \backslash \{(w, v)\}$.

(swapping) $\mathcal{M}^*_{wv} = \langle W, R^*_{wv}, V \rangle$, with $R^*_{wv} = (R \backslash \{(v, w)\}) \cup \{(w, v)\}$.

(bridging) $\mathcal{M}^+_{wv} = \langle W, R^+_{wv}, V \rangle$, with $R^+_{wv} = R \cup \{(w, v)\}$.

Relation-Changing Modal Logics

Formalmente:

$$\mathcal{M}, w \models \Diamond\varphi \qquad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \in R,\ \mathcal{M}, v \models \varphi$$

Relation-Changing Modal Logics

Formalmente:

$$\mathcal{M}, w \models \Diamond\varphi \quad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \in R,\ \mathcal{M}, v \models \varphi$$

$$\mathcal{M}, w \models \langle \mathsf{sb} \rangle\varphi \quad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \in R,\ \mathcal{M}^{-}_{wv}, v \models \varphi$$

Relation-Changing Modal Logics

Formalmente:

$$\mathcal{M}, w \models \Diamond\varphi \quad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \in R,\ \mathcal{M}, v \models \varphi$$

$$\mathcal{M}, w \models \langle \text{sb} \rangle \varphi \quad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \in R,\ \mathcal{M}^{-}_{wv}, v \models \varphi$$

$$\mathcal{M}, w \models \langle \text{gsb} \rangle \varphi \quad \text{sii} \quad \exists v, u \in W, \text{ t.q. } (v, u) \in R,\ \mathcal{M}^{-}_{vu}, w \models \varphi$$

Relation-Changing Modal Logics

Formalmente:

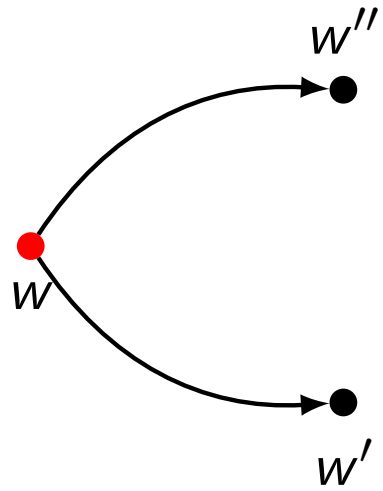
$$\mathcal{M}, w \models \langle \mathsf{sw} \rangle \varphi \quad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \in R,\ \mathcal{M}^*_{vw}, v \models \varphi$$

$$\mathcal{M}, w \models \langle \mathsf{gsw} \rangle \varphi \quad \text{sii} \quad \exists v, u \in W,\ \text{t.q. } (v, u) \in R,\ \mathcal{M}^*_{uv}, w \models \varphi$$

$$\mathcal{M}, w \models \langle \mathsf{br} \rangle \varphi \quad \text{sii} \quad \exists v \in W \text{ t.q. } (w, v) \notin R,\ \mathcal{M}^+_{wv}, v \models \varphi$$

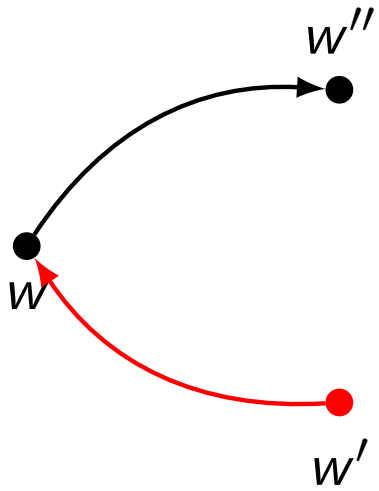
$$\mathcal{M}, w \models \langle \mathsf{gbr} \rangle \varphi \quad \text{sii} \quad \exists v, u \in W,\ \text{t.q. } (v, u) \notin R,\ \mathcal{M}^+_{vu}, w \models \varphi.$$

Relation-Changing Modal Logics - Ejemplos



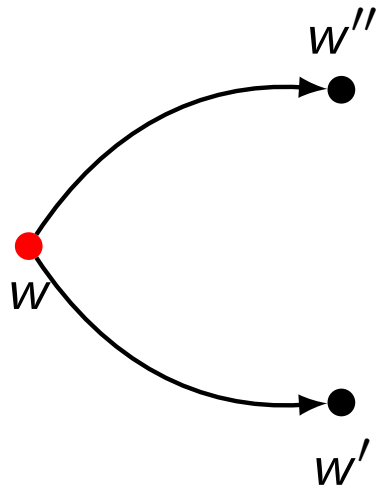
$$\mathcal{M}, w \models \langle \mathsf{sw} \rangle \varphi$$

Relation-Changing Modal Logics - Ejemplos



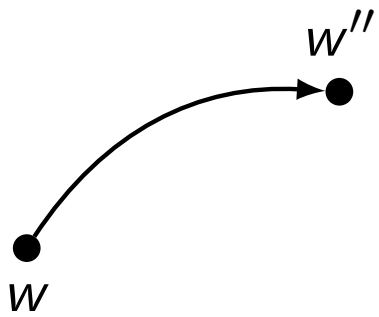
$$\mathcal{M}^*_{w'w}, w' \models \varphi$$

Relation-Changing Modal Logics - Ejemplos



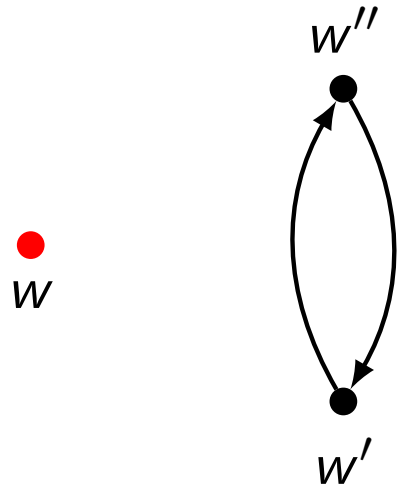
$$\mathcal{M}, w \models \langle \mathsf{sb} \rangle \varphi$$

Relation-Changing Modal Logics - Ejemplos



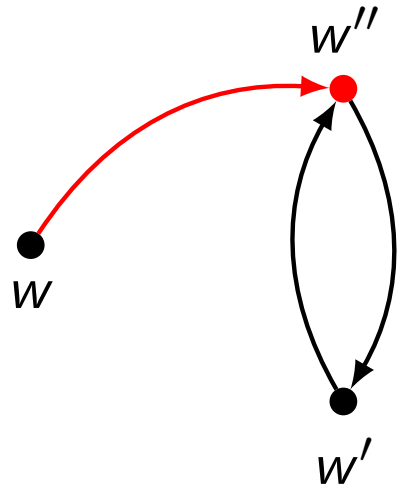
$$\mathcal{M}^{-}_{ww'}, w' \models \varphi$$

Relation-Changing Modal Logics - Ejemplos



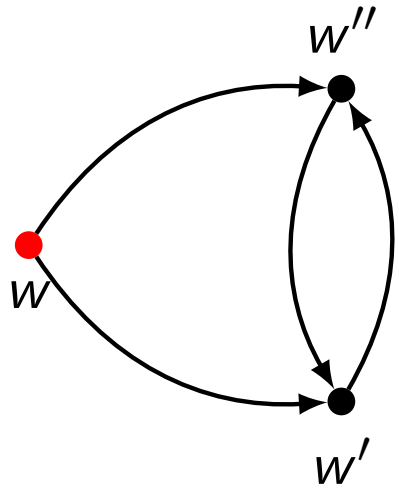
$$\mathcal{M}, w \models \langle \mathsf{br} \rangle \varphi$$

Relation-Changing Modal Logics - Ejemplos



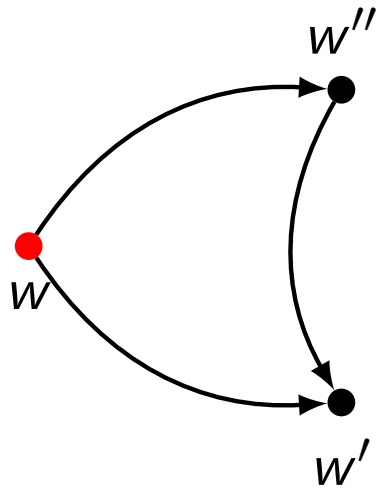
$$\mathcal{M}^{+}_{ww''}, w'' \models \varphi$$

Relation-Changing Modal Logics - Ejemplos



$$\mathcal{M}, w \models \langle \mathsf{gsb} \rangle \varphi$$

Relation-Changing Modal Logics - Ejemplos



$$\mathcal{M}^{-}_{w'w''}, w \models \varphi$$

$\mathcal{ML}(\blacklozenge)$-bisimulaciones 1/2

always	(nontriv)	$Z \neq \emptyset$
always	(agree)	si $(w,S)Z(w',S')$, w y w' satisfacen los mismos símbolos proposicionales.
$\Diamond$	(zig)	si wSv, existe $v' \in W'$ t.q. $w'S'v'$ y $(v,S)Z(v',S')$
	(zag)	si $w'S'v'$, existe $v \in W$ t.q. wSv y $(v,S)Z(v',S')$
$\langle$sw$\rangle$	(zig)	si wSv, existe $v' \in W'$ t.q. $w'S'v'$ y $(v,S^*_{vw})Z(v',S'^*_{v'w'})$
	(zag)	si $w'S'v'$, existe $v \in W$ t.q. wSv y $(v,S^*_{vw})Z(v',S'^*_{v'w'})$
$\langle$gsw$\rangle$	(zig)	si vSu, existe $v',u' \in W'$ t.q. $v'S'u'$ y $(w,S^*_{uv})Z(w',S'^*_{u'v'})$
	(zag)	si $v'S'u'$, existe $v,u \in W$ t.q. vSu y $(w,S^*_{uv})Z(w',S'^*_{u'v'})$
$\langle$sb$\rangle$	(zig)	si wSv, existe $v' \in W'$ t.q. $w'S'v'$ y $(v,S^-_{wv})Z(v',S'^-_{w'v'})$
	(zag)	si $w'S'v'$, existe $v \in W$ t.q. wSv y $(v,S^-_{wv})Z(v',S'^-_{w'v'})$

$\mathcal{ML}(\blacklozenge)$-bisimulaciones 2/2

$\langle$gsb$\rangle$	(zig)	si vSu, existe $v', u' \in W'$ t.q. $v'S'u'$ y $(w, S^-_{vu})Z(w', S'^-_{v'u'})$
	(zag)	si $v'S'u'$, existe $v, u \in W$ t.q. vSu y $(w, S^-_{vu})Z(w', S'^-_{v'u'})$
$\langle$br$\rangle$	(zig)	si no wSv, existe $v' \in W'$ t.q. no $w'S'v'$ y $(v, S^+_{wv})Z(v', S'^+_{w'v'})$
	(zag)	si no $w'S'v'$, existe $v \in W$ t.q. no wSv y $(v, S^+_{wv})Z(v', S'^+_{w'v'})$
$\langle$gbr$\rangle$	(zig)	si no vSu, existe $v', u' \in W'$ t.q. no $v'S'u'$ y $(w, S^+_{vu})Z(w', S'^+_{v'u'})$
	(zag)	si no $v'S'u'$, existe $v, u \in W$ t.q. no vSu y $(w, S^+_{vu})Z(w', S'^+_{v'u'})$

$\mathcal{ML}(\blacklozenge)$-bisimulaciones 2/2

$\langle$gsb$\rangle$	(zig)	si vSu, existe $v', u' \in W'$ t.q. $v'S'u'$ y $(w, S^{-}_{vu})Z(w', S'^{-}_{v'u'})$
	(zag)	si $v'S'u'$, existe $v, u \in W$ t.q. vSu y $(w, S^{-}_{vu})Z(w', S'^{-}_{v'u'})$
$\langle$br$\rangle$	(zig)	si no wSv, existe $v' \in W'$ t.q. no $w'S'v'$ y $(v, S^{+}_{wv})Z(v', S'^{+}_{w'v'})$
	(zag)	si no $w'S'v'$, existe $v \in W$ t.q. no wSv y $(v, S^{+}_{wv})Z(v', S'^{+}_{w'v'})$
$\langle$gbr$\rangle$	(zig)	si no vSu, existe $v', u' \in W'$ t.q. no $v'S'u'$ y $(w, S^{+}_{vu})Z(w', S'^{+}_{v'u'})$
	(zag)	si no $v'S'u'$, existe $v, u \in W$ t.q. no vSu y $(w, S^{+}_{vu})Z(w', S'^{+}_{v'u'})$

Teorema

Para $\mathcal{ML}(\blacklozenge)$, $\blacklozenge \in \{\langle \text{sw} \rangle, \langle \text{gsw} \rangle, \langle \text{sb} \rangle, \langle \text{gsb} \rangle, \langle \text{br} \rangle, \langle \text{gbr} \rangle\}$,
$\mathcal{M}, w \leftrightarroweq_{\mathcal{ML}(\blacklozenge)} \mathcal{M}', w'$ *implica* $\mathcal{M}, w \equiv_{\mathcal{ML}(\blacklozenge)} \mathcal{M}', w'$.

Comparando poder expresivo

Qué necesitamos si queremos demostrar que nuestras lógicas son incomparables?

- Encontrar dos modelos $\blacklozenge_1$-bisimilares distinguibles por $\mathcal{ML}(\blacklozenge_2)$.
- Encontrar dos modelos $\blacklozenge_2$-bisimilares distinguibles por $\mathcal{ML}(\blacklozenge_1)$.

Entonces $\mathcal{ML}(\blacklozenge_1)$ y $\mathcal{ML}(\blacklozenge_2)$ son incomparables.

[1] Excepto el caso entre $\mathcal{ML}(\langle \text{sw} \rangle)$ y $\mathcal{ML}(\langle \text{gsw} \rangle)$, con una dirección abierta.

Comparando poder expresivo

Qué necesitamos si queremos demostrar que nuestras lógicas son incomparables?

- Encontrar dos modelos $\blacklozenge_1$-bisimilares distinguibles por $\mathcal{ML}(\blacklozenge_2)$.
- Encontrar dos modelos $\blacklozenge_2$-bisimilares distinguibles por $\mathcal{ML}(\blacklozenge_1)$.

Entonces $\mathcal{ML}(\blacklozenge_1)$ y $\mathcal{ML}(\blacklozenge_2)$ son incomparables.

Teorema

Para todo $\blacklozenge_1, \blacklozenge_2$ tal que $\blacklozenge_1 \neq \blacklozenge_2$, $\mathcal{ML}(\blacklozenge_1)$ y $\mathcal{ML}(\blacklozenge_2)$ son incomparables[1].

[1] Excepto el caso entre $\mathcal{ML}(\langle \mathrm{sw} \rangle)$ y $\mathcal{ML}(\langle \mathrm{gsw} \rangle)$, con una dirección abierta.

Satisfactibilidad

- Las lógicas que introdujimos no poseen tree/finite-model property.

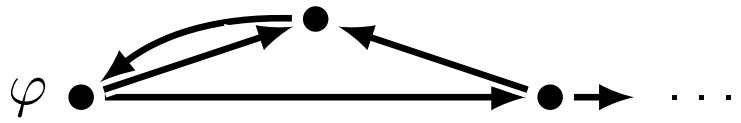
Satisfactibilidad

- Las lógicas que introdujimos no poseen tree/finite-model property.
- Mayor poder expresivo con respecto a $\mathcal{ML}$ + indecidibilidad para otras lógicas dinámicas ($\mathcal{HL}(@, \downarrow)$, $\mathcal{ML}(ⓡ, ⓚ)$), "sugieren" indecidibilidad.

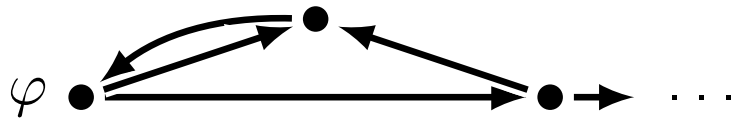
Satisfactibilidad

- Las lógicas que introdujimos no poseen tree/finite-model property.
- Mayor poder expresivo con respecto a $\mathcal{ML}$ + indecidibilidad para otras lógicas dinámicas ($\mathcal{HL}(@, \downarrow)$, $\mathcal{ML}(ⓡ, ⓚ)$), "sugieren" indecidibilidad.
- Usamos una técnica de spy point + reducción de $\mathcal{ML}(ⓡ, ⓚ)$ para demostrarlo.

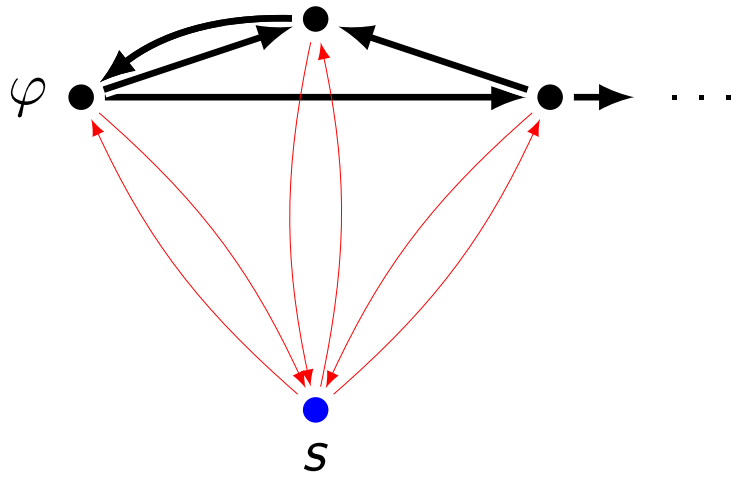
Indecidibilidad de $\mathcal{ML}(\langle \mathrm{sb} \rangle)$



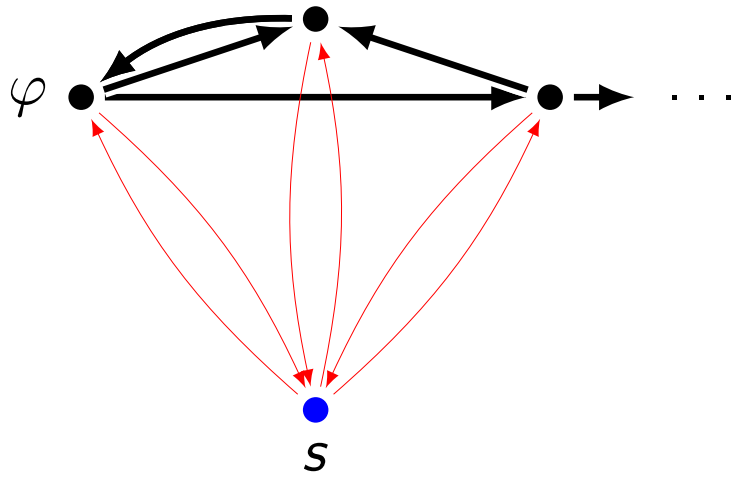
Indecidibilidad de $\mathcal{ML}(\langle \mathrm{sb} \rangle)$



Indecidibilidad de $\mathcal{ML}(\langle \text{sb} \rangle)$

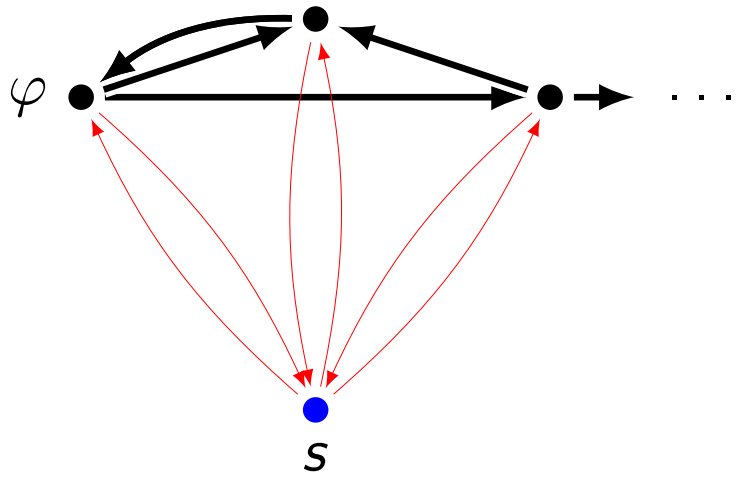


Indecidibilidad de $\mathcal{ML}(\langle \text{sb} \rangle)$



- En la tesis usamos la técnica de spy point para los 3 casos locales.

Indecidibilidad de $\mathcal{ML}(\langle \text{sb} \rangle)$



- En la tesis usamos la técnica de spy point para los 3 casos locales.
- Queda pendiente completar los casos globales, usando ideas similares.

Model Checking

Model Checking

Quantified Boolean Formulas (QBF):

- Sintaxis:

$$\alpha ::= x \mid \neg\alpha \mid \alpha \wedge \alpha \mid \exists x.\alpha$$

- Semántica:

$$\begin{array}{lcl} v \models x & \Leftrightarrow & v(x) = 1 \\ v \models \neg\alpha & \Leftrightarrow & v \not\models \alpha \\ v \models \alpha \wedge \alpha' & \Leftrightarrow & v \models \alpha \text{ y } v \models \alpha' \\ v \models \exists x.\alpha & \Leftrightarrow & v[x \to 1] \models \alpha \text{ ó } v[x \to 0] \models \alpha \end{array}$$

Model Checking

Quantified Boolean Formulas (QBF):

- Sintaxis:

$$\alpha ::= x \mid \neg\alpha \mid \alpha \wedge \alpha \mid \exists x.\alpha$$

- Semántica:

$$\begin{array}{lcl} v \models x & \Leftrightarrow & v(x) = 1 \\ v \models \neg\alpha & \Leftrightarrow & v \not\models \alpha \\ v \models \alpha \wedge \alpha' & \Leftrightarrow & v \models \alpha \text{ y } v \models \alpha' \\ v \models \exists x.\alpha & \Leftrightarrow & v[x \to 1] \models \alpha \text{ ó } v[x \to 0] \models \alpha \end{array}$$

Teorema

Decidir validez de una QBF es un problema PSPACE*-completo.*

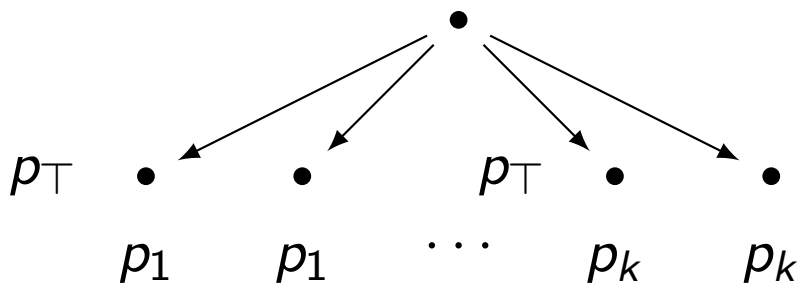
Model checking $\mathcal{ML}(\langle \text{sw} \rangle)$ es PSPACE-hard

Sea α una QBF con k variables:

Model checking $\mathcal{ML}(\langle \text{sw} \rangle)$ es PSPACE-hard

Sea α una QBF con k variables:

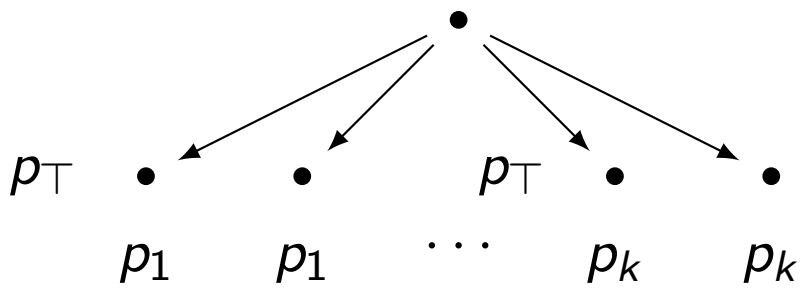
1. Contruimos $\mathcal{M}_k$ de la siguiente manera:



Model checking $\mathcal{ML}(\langle \mathsf{sw} \rangle)$ es PSPACE-hard

Sea α una QBF con k variables:

1. Contruimos $\mathcal{M}_k$ de la siguiente manera:



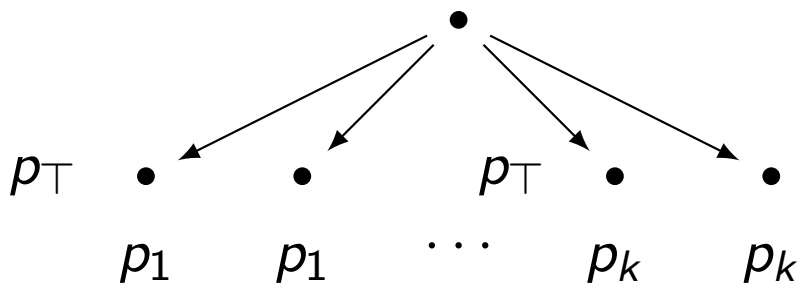
2. Construimos una $\mathcal{ML}(\langle \mathsf{sw} \rangle)$ fórmula a partir de una QBF:

$$
\begin{aligned}
(\exists x_i.\alpha)' &= \langle \mathsf{sw} \rangle (p_i \wedge \Diamond(\alpha)') \\
(x_i)' &= \neg\Diamond(p_i \wedge p_\top) \\
(\neg\alpha)' &= \neg(\alpha)' \\
(\alpha \wedge \beta)' &= (\alpha)' \wedge (\beta)'
\end{aligned}
$$

Model checking $\mathcal{ML}(\langle \mathsf{sw} \rangle)$ es PSPACE-hard

Sea α una QBF con k variables:

1. Contruimos $\mathcal{M}_k$ de la siguiente manera:



2. Construimos una $\mathcal{ML}(\langle \mathsf{sw} \rangle)$ fórmula a partir de una QBF:

$$
\begin{aligned}
(\exists x_i.\alpha)' &= \langle \mathsf{sw} \rangle (p_i \wedge \Diamond(\alpha)') \\
(x_i)' &= \neg\Diamond(p_i \wedge p_\top) \\
(\neg\alpha)' &= \neg(\alpha)' \\
(\alpha \wedge \beta)' &= (\alpha)' \wedge (\beta)'
\end{aligned}
$$

3. α es verdadera sii $\mathcal{M}_k, w \models (\alpha)'$

Model checking es PSPACE-completo

En la tesis se introducen traducciones similares para el resto de las lógicas que introdujimos.

Model checking es PSPACE-completo

En la tesis se introducen traducciones similares para el resto de las lógicas que introdujimos.

Para mostrar que model checking está en PSPACE se puede dar un algoritmo depth-first que sigue la definición de $\models$.

Model checking es PSPACE-completo

En la tesis se introducen traducciones similares para el resto de las lógicas que introdujimos.

Para mostrar que model checking está en PSPACE se puede dar un algoritmo depth-first que sigue la definición de $\models$.

Teorema

Sea $\blacklozenge \in \{\langle \mathsf{sw} \rangle, \langle \mathsf{gsw} \rangle, \langle \mathsf{sb} \rangle, \langle \mathsf{gsb} \rangle, \langle \mathsf{br} \rangle, \langle \mathsf{gbr} \rangle\}$, *model checking para cualquiera de las lógicas* $\mathcal{ML}(\blacklozenge)$ *es PSPACE-completo.*

Tableaux

Tableaux

- Los tableaux modales usualmente usan fórmulas con constantes como prefijos, los cuales indican en que parte del modelo las fórmulas vale.

Tableaux

- Los tableaux modales usualmente usan fórmulas con constantes como prefijos, los cuales indican en que parte del modelo las fórmulas vale.
- En nuestros tableaux, los prefijos están compuestos por una constante y un conjunto de "nombres de aristas" (pares de constantes).

Tableaux

- Los tableaux modales usualmente usan fórmulas con constantes como prefijos, los cuales indican en que parte del modelo las fórmulas vale.
- En nuestros tableaux, los prefijos están compuestos por una constante y un conjunto de "nombres de aristas" (pares de constantes).
- Ejemplo de fórmula con prefijo: $(s, S) : \varphi$.

Tableaux

- Los tableaux modales usualmente usan fórmulas con constantes como prefijos, los cuales indican en que parte del modelo las fórmulas vale.
- En nuestros tableaux, los prefijos están compuestos por una constante y un conjunto de "nombres de aristas" (pares de constantes).
- Ejemplo de fórmula con prefijo: $(s, S) : \varphi$.
- "φ se satisface en el estado nombrado por s en el modelo modificado descripto por el conjunto S de aristas borradas/invertidas/agregadas".

Tableaux

- Los tableaux modales usualmente usan fórmulas con constantes como prefijos, los cuales indican en que parte del modelo las fórmulas vale.
- En nuestros tableaux, los prefijos están compuestos por una constante y un conjunto de "nombres de aristas" (pares de constantes).
- Ejemplo de fórmula con prefijo: $(s, S) : \varphi$.
- "φ se satisface en el estado nombrado por s en el modelo modificado descripto por el conjunto S de aristas borradas/invertidas/agregadas".

Ejemplo

$$\frac{(n, S) : \langle \mathrm{sb} \rangle \varphi}{\begin{array}{c} \dot{R} nm \\ nm \dot{\notin} S \\ (m, S \cup nm) : \varphi \end{array}} (\langle \mathrm{sb} \rangle)$$

Tableaux - Comentarios

+ Cálculo sound y completo para las seis lógicas.

Tableaux - Comentarios

+ Cálculo sound y completo para las seis lógicas.
- Poca uniformidad entre diferentes tipos de modificadores.

Tableaux - Comentarios

+ Cálculo sound y completo para las seis lógicas.
- Poca uniformidad entre diferentes tipos de modificadores.
- Reglas para swap más complicadas: mantener información de pares invertidos una vez, y dos veces.

Tableaux - Comentarios

+ Cálculo sound y completo para las seis lógicas.
- Poca uniformidad entre diferentes tipos de modificadores.
- Reglas para swap más complicadas: mantener información de pares invertidos una vez, y dos veces.
* Tableaux + Model Checking: procedimento efectivo para chequear satisfactibilidad de una fórmula (pero incompleto!).

Relation-Changing & Dynamic Epistemic Logics

- En la primer parte de la tesis, estudiamos operadores dinámicos desde un punto de vista abstracto.

Relation-Changing & Dynamic Epistemic Logics

- En la primer parte de la tesis, estudiamos operadores dinámicos desde un punto de vista abstracto.
- En la segunda parte, nos dedicamos a investigar los campos de aplicación de estas lógicas.

Relation-Changing & Dynamic Epistemic Logics

- En la primer parte de la tesis, estudiamos operadores dinámicos desde un punto de vista abstracto.
- En la segunda parte, nos dedicamos a investigar los campos de aplicación de estas lógicas.
- $\mathcal{DEL}$ es un caso particular de lógicas de cambio de modelo.

Relation-Changing & Dynamic Epistemic Logics

- En la primer parte de la tesis, estudiamos operadores dinámicos desde un punto de vista abstracto.
- En la segunda parte, nos dedicamos a investigar los campos de aplicación de estas lógicas.
- $\mathcal{DEL}$ es un caso particular de lógicas de cambio de modelo.
- Definimos la lógica $\mathcal{ML}(\mathsf{cp}, [\![\ ]\!])$, que codifica algunas $\mathcal{DEL}$s.

Principales Contribuciones de la Tesis

- Completamos el espectro de lógicas dinámicas.

Principales Contribuciones de la Tesis

- Completamos el espectro de lógicas dinámicas.
- Definimos bisimulaciones apropiadamente y estudiamos el poder expresivo. [Cap. 4]

Principales Contribuciones de la Tesis

- Completamos el espectro de lógicas dinámicas.
- Definimos bisimulaciones apropiadamente y estudiamos el poder expresivo. [Cap. 4]
- Probamos indecidibilidad de los casos locales. [Cap. 5]

Principales Contribuciones de la Tesis

- Completamos el espectro de lógicas dinámicas.
- Definimos bisimulaciones apropiadamente y estudiamos el poder expresivo. [Cap. 4]
- Probamos indecidibilidad de los casos locales. [Cap. 5]
- Demostramos que model checking es PSPACE-completo para las 6 lógicas. [Cap. 6]

Principales Contribuciones de la Tesis

- Completamos el espectro de lógicas dinámicas.
- Definimos bisimulaciones apropiadamente y estudiamos el poder expresivo. [Cap. 4]
- Probamos indecidibilidad de los casos locales. [Cap. 5]
- Demostramos que model checking es PSPACE-completo para las 6 lógicas. [Cap. 6]
- Definimos un cálculo de tableau correcto y completo. [Cap. 7]

Principales Contribuciones de la Tesis

- Completamos el espectro de lógicas dinámicas.
- Definimos bisimulaciones apropiadamente y estudiamos el poder expresivo. [Cap. 4]
- Probamos indecidibilidad de los casos locales. [Cap. 5]
- Demostramos que model checking es PSPACE-completo para las 6 lógicas. [Cap. 6]
- Definimos un cálculo de tableau correcto y completo. [Cap. 7]
- Encontramos un campo de aplicación para esta familia de lógicas, y comenzamos a investigar esta nueva línea. [Cap. 9]

Algunas líneas de trabajo futuro

- Caracterización de van Benthem.

Algunas líneas de trabajo futuro

- Caracterización de van Benthem.
- Estudiar fragmentos decidibles.

Algunas líneas de trabajo futuro

- Caracterización de van Benthem.
- Estudiar fragmentos decidibles.
- Usar las definiciones de Tableaux para traducir Relation-Changing Modal Logics en Hybrid Logics.

Algunas líneas de trabajo futuro

- Caracterización de van Benthem.
- Estudiar fragmentos decidibles.
- Usar las definiciones de Tableaux para traducir Relation-Changing Modal Logics en Hybrid Logics.
- Continuar el trabajo sobre $\mathcal{DEL}$.

Algunas líneas de trabajo futuro

- Caracterización de van Benthem.
- Estudiar fragmentos decidibles.
- Usar las definiciones de Tableaux para traducir Relation-Changing Modal Logics en Hybrid Logics.
- Continuar el trabajo sobre $\mathcal{DEL}$.
- Investigar operadores de cambio en Álgebras Relacionales.

Preguntas?