# Lógicas Modales

## Decidibilidad y complejidad de lógicas modales (i)

Carlos Areces & Raul Fervari

1er cuatrimestre de 2017
Córdoba, Argentina



## Repaso

### En el episodio anterior...

- Repasamos las principales clases de complejidad
- Vimos una forma de dar cotas de complejidad:
  - Si muestro que puedo adivinar una solución en $f(n)$ pasos
  - Y que puedo chequear si es correcta en a lo sumo $f(n)$ pasos
  - Entonces el problema seguro está en $\mathsf{NTIME}(f(n))$
- Usando esto vimos que satisfacibilidad de la lógica modal básica está en NEXPTIME

### Bibliografía Relevante:

Capítulo 6 de Modal Logic, Blackburn et al.



## Modelos "más chicos" vía una función de selección

Dados $\varphi$ y un modelo $\mathcal{M} = \langle W, R, V\rangle$, definimos:

$$s^{\mathcal{M}}(p, w) = \{w\} \qquad s^{\mathcal{M}}(\varphi \wedge \psi, w) = s^{\mathcal{M}}(\varphi, w) \cup s^{\mathcal{M}}(\psi, w)$$
$$s^{\mathcal{M}}(\neg\varphi, w) = s^{\mathcal{M}}(\varphi, w) \qquad s^{\mathcal{M}}(\Diamond\psi, w) = \{w\} \cup \bigcup_{\{v \mid wRv\}} s^{\mathcal{M}}(\psi, v)$$

### Teorema

Para todo $\mathcal{M}$, $w$ y $\varphi$, $\mathcal{M}, w \models \varphi$ sii $\mathcal{M} \restriction s^{\mathcal{M}}(\varphi, w), w \models \varphi$

### Demostración (idea)

- Sea $k$ el modal depth de $\varphi$
- Se puede ver que $\mathcal{M} \restriction s^{\mathcal{M}}(\varphi, w)$ es $k$-bisimilar a $\mathcal{M}$
- De donde se sigue el resultado buscado



## Satisfacibilidad de $\mathbf{KAlt}_1$ está en NP

Vía una función de selección

### $\mathbf{KAlt}_1$

Es la lógica modal básica restringida a la clase de modelos $\mathcal{C}_{\mathbf{Alt}_1}$ donde $R$ es una función parcial.

### Observación 1

Si $\mathcal{M} \in \mathcal{C}_{\mathbf{Alt}_1}$, entonces $\mathcal{M} \restriction s^{\mathcal{M}}(\varphi, w) \in \mathcal{C}_{\mathbf{Alt}_1}$ y $|\mathcal{M} \restriction s^{\mathcal{M}}(\varphi, w)| \leq |\varphi|$.

### Observación 2

Dado $\mathcal{M}$ finito, se puede decidir si $\mathcal{M} \in \mathcal{C}_{\mathbf{Alt}_1}$ en tiempo polinomial.

### Algoritmo NP para satisfacibilidad de $\mathbf{KAlt}_1$

Dado $\varphi$, adivinar un modelo de tamaño a lo sumo $|\varphi|$ y chequear polinomialmente que esté en $\mathcal{C}_{\mathbf{Alt}_1}$ y que satisfaga $\varphi$.

## Satisfacibilidad de $\mathbf{KAlt}_1$ es NP-completo

### Demostración

- Ya probamos que satisfacibilidad de $\mathbf{KAlt}_1$ está en NP.
- Sólo necesitamos reducir (polinomialmente) un problema que se sepa NP-completo.
- Satisfacibilidad proposicional es NP-completo.
- Y podemos resolver SAT proposicional con SAT para $\mathbf{KAlt}_1$.



## Lógicas modales NP-completas

- Usamos funciones de selección para mostrar que $\mathbf{KAlt}_1$ tiene la "propiedad de modelos polinomiales".
- Como sus modelos son reconocibles en tiempo polinomial, y podemos codificar (trivialmente) SAT proposicional, concluimos que $\mathbf{KAlt}_1$-SAT es NP-completo.

### De manera similar se puede ver que son NP-completas:

- **S5**: La LMB sobre modelos con $R$ relación de equivalencia.
- **S**4,3: La LMB sobre modelos donde $R$ es transitiva y conexa $(\forall xy.(Rxy \vee Ryx))$ y existe un nodo que es la raíz (sin predecesor y todo otro nodo es accesible desde él).
- Toda lógica que extienda **S4**,**3**.

¿Tendrá K la propiedad de modelos polinomiales?



## K no tiene la propiedad de modelos polinomiales

### Veremos que:

Para todo natural $k$, existe una $\varphi_k$ satisfacible, tal que:

I. el tamaño de $\varphi_k$ es polinomial en $k$,
II. todo modelo de $\varphi_k$ tiene al menos $2^k$ nodos.

### De donde se sigue que:

- Ningún polinomio acota el tamaño de un modelo mínimo para una fórmula en función de su tamaño.
- Luego, K no tiene la propiedad de modelos polinomiales.



## K no tiene la propiedad de modelos polinomiales

Estrategia de la demostración

- En cada $\varphi_k$ usamos proposiciones $p_1, \ldots, p_k$ y $l_0, \ldots, l_k$.
- $\varphi_k$ exige un nodo por cada asignación posible de $p_1 \ldots p_k$.
- Serán nodos a profundidad $k$ en un arbol binario completo.
- Usamos $l_i$ para marcar aquellos nodos a profundidad $i$.

## K no tiene la propiedad de modelos polinomiales

### Ladrillos para armar cada $\varphi_k$

- $B_i$ fuerza dos sucesores, uno para cada valor de $p_i$:

$$B_i := \Diamond p_{i+1} \wedge \Diamond \neg p_{i+1}$$

- $S_i$ propaga los valores de $p_i$ y $\neg p_i$ al siguiente nivel:

$$S_i := (p_i \rightarrow \Box p_i) \wedge (\neg p_i \rightarrow \Box \neg p_i)$$

- $L_{ki}$ asegura que un nodo esté en el nivel $i$ y sólo en ese:

$$L_{ki} \;:=\; \bigwedge_{j \in \{0 \ldots k\} \setminus \{i\}} \neg l_j \;\wedge\; l_i$$



## K no tiene la propiedad de modelos polinomiales

$\varphi_k$ es la conjunción de:

$$\begin{array}{cccccccccc}
L_{k0} & \wedge & \Box L_{k1} & \wedge & \Box^2 L_{k2} & \wedge & \Box^3 L_{k3} & \wedge & \ldots & \wedge & \Box^{k-1} L_{kk-1} \wedge \Box^k L_{kk} \\
B_0 & \wedge & \Box B_1 & \wedge & \Box^2 B_2 & \wedge & \Box^3 B_3 & \wedge & \ldots & \wedge & \Box^{k-1} B_{k-1} \\
 & & \Box S_1 & \wedge & \Box^2 S_1 & \wedge & \Box^3 S_1 & \wedge & \ldots & \wedge & \Box^{k-1} S_1 \\
 & & & & \Box^2 S_2 & \wedge & \Box^3 S_2 & \wedge & \ldots & \wedge & \Box^{k-1} S_2 \\
 & & & & & \wedge & \Box^3 S_3 & \wedge & \ldots & \wedge & \Box^{k-1} S_3 \\
 & & & & & & & & & & \vdots \\
 & & & & & & & & & \wedge & \Box^{k-1} S_{k-1}
\end{array}$$

$\varphi_k$ crece "poco" a medida que aumentamos $k$

I. Notar que $|\Box^k L_{ki}|$, $|\Box^k B_i|$ y $|\Box^k S_i|$ son $O(k)$.

II. Viendo la matriz, acotamos "a lo bruto": $|\varphi_k| \in O(k^3)$.

¡Pero todo modelo para $\varphi_k$ tiene al menos $2^k$ nodos!



## Qué podemos concluir (y qué no)

### Concluimos que...

- No es verdad que en K las fórmulas satisfacibles tengan modelos polinomiales.
- No podremos usar la técnica de "adivinar modelos" para probar que satisfacibilidad de K está en NP.

### No podemos concluir que...

- No sea el caso que satisfacibilidad de K esté en NP
- (aunque parece poco probable)



## Satisfacibilidad de K está en PSPACE

### Intuición

### Idea general

- No tenemos espacio para adivinar un modelo entero
- Pero podemos ir adivinando de a una "rama" por vez
- Y, sobre la marcha, ir verificando si satisface la fórmula
- ¡Las ramas podemos asumirlas lineales en la fórmula!

### Detalles escabrosos

- Necesitamos garantizar que todo diamante sea verificado
- Podemos usar no-determinismo! (PSPACE $=$ NPSPACE)
- Formalizaremos la idea usando "Hintikka sets"

## Satisfacibilidad de K está en PSPACE

### Negation Normal Form (NNF):

- Por simplicidad, y sin perder generarlidad, asumamos NNF:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Diamond\varphi \mid \Box\varphi$$

- $\overline{\varphi}$ es la "negación en NNF" de $\varphi$ (e.g., $\overline{\neg p} = p$, $\overline{\varphi \wedge \psi} = \overline{\varphi} \vee \overline{\psi}$, etc.)

### Clausura de un conjunto de fórmulas $\Sigma$ ($\mathrm{Cl}(\Sigma)$):

$\mathrm{Cl}(\Sigma) = \{\varphi \mid \varphi \text{ ocurre en } \Sigma\} \cup \{\overline{\varphi} \mid \varphi \text{ ocurre en } \Sigma\}$

### Intuición

$\mathrm{Cl}(\Sigma)$ es el conjunto de "fórmulas relevantes" de $\Sigma$.



## Satisfacibilidad de K está en PSPACE

Hintikka sets

### Hintikka sets

Decimos que $H$ es un *Hintikka set para* $\Sigma$ si cumple:

I. $H \subseteq \mathrm{Cl}(\Sigma)$

II. $\varphi \in \mathrm{Cl}(\Sigma) \Rightarrow \varphi \in H$ sii $\overline{\varphi} \notin H$

III. $\varphi \wedge \psi \in \mathrm{Cl}(\Sigma) \Rightarrow \varphi \wedge \psi \in H$ sii $\varphi \in H$ y $\psi \in H$

IV. $\varphi \vee \psi \in \mathrm{Cl}(\Sigma) \Rightarrow \varphi \vee \psi \in H$ sii $\varphi \in H$ ó $\psi \in H$

### Intuición

Un Hintikka set para $\Sigma$ es un conjunto "suficientemente grande" de subfórmulas de $\Sigma$ que alcanza para verificar la información local de $\Sigma$ en un estado de un modelo.



## Satisfacibilidad de K está en PSPACE

Hintikka sets – ¿para qué?

### Teorema

Son equivalentes:

1. $\Sigma$ es satisfacible.
2. Existe un Hintikka set $H$ para $\Sigma$ que es satisfacible e incluye a $\Sigma$.

### Demostración

$\Leftarrow$) Directo dado que $\Sigma \subseteq H$.

$\Rightarrow$)
- Dado $\mathcal{M}, w \models \Sigma$, sea $H = \{\varphi \mid \mathcal{M}, w \models \varphi \text{ y } \varphi \in \mathrm{Cl}(\Sigma)\}$
- Es fácil ver que $H$ es un Hintikka set para $\Sigma$ y $\mathcal{M}, w \models H$.



## Satisfacibilidad de K está en PSPACE

Hintikka sets – ¿para qué?

### Teorema

Sea $H$ un Hintikka set para $\Sigma$. Son equivalentes:

I. $H$ es satisfacible.

II. Para todo $\Diamond\varphi_i \in H$, $H_i = \{\varphi_i\} \cup \Box(H)$ es satisfacible.

*Notación:* $\Box(H) = \{\varphi \mid \Box\varphi \in H\}$

### Demostración

$\Rightarrow$) Si $\mathcal{M}, w \models H$ y $\Diamond\varphi_i \in H$, $\exists v\ \mathcal{M}, v \models \varphi_i$ y $\mathcal{M}, v \models \Box(H)$.

$\Leftarrow$) Para cada $\Diamond\varphi_i \in H$, sea $\mathcal{M}_i = \langle W_i, R_i, V_i\rangle$ tq $\mathcal{M}_i, w_i \models H_i$.
Sea $\mathcal{M}$ el modelo $\biguplus \mathcal{M}_i$ más un nuevo $w$ tal que $wRw_i$ para todo $w_i$ y $w \in V(p)$ sii $p \in H$.
$\mathcal{M}, w_i \underline{\leftrightarrow} \mathcal{M}_i, w_i$, con lo cual $\mathcal{M}, w_i \models H_i$.
Es fácil ver que, $\mathcal{M}, w \models H$.

## Satisfacibilidad de K está en PSPACE

### Un algoritmo no-determinístico basado en Hintikka sets

```
EsSat(Σ)
H ← adivinar un subconjunto de Cl(Σ)
si H no es un Hintikka set sobre Σ
  devolver 0
para todo ◇φ ∈ H
  si EsSat({φ} ∪ □(H)) = 0
    devolver 0
devolver 1
```

**Observaciones**

- `EsSat(Σ)` computa K-satisfacibilidad de $\Sigma$ (para $\Sigma$ finito)
- Recursion depth de `EsSat(Σ)` $\leq$ modal depth de $\Sigma$
- En cada paso se necesita espacio polinomial en $\Sigma$



## Satisfacibilidad de K está en PSPACE

### Recapitulando

`EsSat(Σ)`

- Algoritmo no-determinístico para la satisfacibilidad de K.
- Requiere espacio polinomial para su ejecución.
- Prueba que este problema está en NPSPACE.
- Por el T. de Savitch prueba también que está en PSPACE.

¿Será además completo para PSPACE?