

# 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  $\text{NTIME}(f(n))$
- Usando esto vimos que satisfacibilidad de la lógica modal básica está en  $\text{NEXPTIME}$

# 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  $\text{NTIME}(f(n))$
- Usando esto vimos que satisfacibilidad de la lógica modal básica está en  $\text{NEXPTIME}$

## Para leer más...

Lo que veamos hoy y la próxima lo pueden encontrar bien explicado en el Modal Logic (Blackburn et al), capítulo 6.

# Modelos “más chicos” vía una función de selección

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

$$\begin{aligned} s(p, w) &= \{w\} & s(\varphi \wedge \psi, w) &= s(\varphi, w) \cup s(\psi, w) \\ s(\neg\varphi, w) &= s(\varphi, w) & s(\Diamond\psi, w) &= \{w\} \cup \bigcup_{\{v \mid wRv\}} s(\psi, v) \end{aligned}$$

# Modelos “más chicos” vía una función de selección

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

$$\begin{aligned}s(p, w) &= \{w\} & s(\varphi \wedge \psi, w) &= s(\varphi, w) \cup s(\psi, w) \\ s(\neg\varphi, w) &= s(\varphi, w) & s(\Diamond\psi, w) &= \{w\} \cup \bigcup_{\{v \mid wRv\}} s(\psi, v)\end{aligned}$$

## Teorema

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

# Modelos “más chicos” vía una función de selección

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

$$\begin{aligned} s(p, w) &= \{w\} & s(\varphi \wedge \psi, w) &= s(\varphi, w) \cup s(\psi, w) \\ s(\neg\varphi, w) &= s(\varphi, w) & s(\Diamond\psi, w) &= \{w\} \cup \bigcup_{\{v \mid wRv\}} s(\psi, v) \end{aligned}$$

## Teorema

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

## Demostración (idea)

- Sea  $k$  el modal depth de  $\varphi$
- Se puede ver que  $\mathcal{M} \upharpoonright s(\varphi, w)$  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.

# 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  $\mathcal{M} \upharpoonright s(\varphi, w) \in \mathcal{C}_{\mathbf{Alt}_1}$  y  $|\mathcal{M} \upharpoonright s(\varphi, w)| \leq |\varphi|$ .



# 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  $\mathcal{M} \upharpoonright s(\varphi, w) \in \mathcal{C}_{\mathbf{Alt}_1}$  y  $|\mathcal{M} \upharpoonright s(\varphi, w)| \leq |\varphi|$ .

## Observación 2

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

# 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  $\mathcal{M} \upharpoonright s(\varphi, w) \in \mathcal{C}_{\mathbf{Alt}_1}$  y  $|\mathcal{M} \upharpoonright s(\varphi, w)| \leq |\varphi|$ .

## 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

# 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  $\mathbf{KAlt}_1$ .

# 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  $\mathbf{KAlt}_1$ .

OJO

¡Las reducciones no siempre son tan triviales!

# Lógicas modales NP-completas

## Recapitulando

- Usamos funciones de selección para mostrar que  $\mathbf{KAlt}_1$  tiene la “propiedad de modelos polinomiales”.
- Como sus modelos son reconocibles en tiempo polinomial,
- Concluimos que es NP-completa (para satisfacibilidad)

# Lógicas modales NP-completas

## Resumiendo

- Usamos funciones de selección para mostrar que  $\mathbf{KAlt}_1$  tiene la “propiedad de modelos polinomiales”.
- Como sus modelos son reconocibles en tiempo polinomial,
- Concluimos que es NP-completa (para satisfacibilidad)

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

- **S5**: La LMB sobre modelos con  $R$  relación de equivalencia.
- **S4.3**: 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**.

# Lógicas modales NP-completas

## Recapitulando

- Usamos funciones de selección para mostrar que  $\mathbf{KAlt}_1$  tiene la “propiedad de modelos polinomiales”.
- Como sus modelos son reconocibles en tiempo polinomial,
- Concluimos que es NP-completa (para satisfacibilidad)

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

- **S5**: La LMB sobre modelos con  $R$  relación de equivalencia.
- **S4.3**: 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**.

## El caso de $\mathbf{K}$ , la LMB sobre modelos arbitrarios

¿Tendrá  $\mathbf{K}$  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.

# 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 desprende trivialmente 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  $p_1, \dots, p_k$  y  $l_0, \dots, l_k$ .
- $\varphi_k$  exige un nodo por cada asignación posible de  $p_1 \dots p_k$ .
- Serán nodos a profundidad  $k$  en un árbol 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:

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

# K no tiene la propiedad de modelos polinomiales

Finalmente,  $\varphi_k$

$\varphi_k$  es la conjunción de:

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

# K no tiene la propiedad de modelos polinomiales

Finalmente,  $\varphi_k$

$\varphi_k$  es la conjunción de:

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

$\varphi_k$  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:  $|\varphi_k| \in O(k^3)$ .

# K no tiene la propiedad de modelos polinomiales

Finalmente,  $\varphi_k$

$\varphi_k$  es la conjunción de:

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

$\varphi_k$  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!

# 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

Hintikka sets – preliminares

## Negation Normal Form (NNF)

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

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

- $\overline{\varphi}$  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 $\Sigma$ ( $\text{Cl}(\Sigma)$ )

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

## Intuición

$\text{Cl}(\Sigma)$  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.  $H \subseteq \text{Cl}(\Sigma)$
- II.  $\varphi \in \text{Cl}(\Sigma) \Rightarrow \varphi \in H$  sii  $\bar{\varphi} \notin H$
- III.  $\varphi \wedge \psi \in \text{Cl}(\Sigma) \Rightarrow \varphi \wedge \psi \in H$  sii  $\varphi \in H$  y  $\psi \in H$
- IV.  $\varphi \vee \psi \in \text{Cl}(\Sigma) \Rightarrow \varphi \vee \psi \in H$  sii  $\varphi \in H$  ó  $\psi \in H$

# Satisfacibilidad de K está en PSPACE

## Hintikka sets

### Hintikka sets

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

- I.  $H \subseteq \text{Cl}(\Sigma)$
- II.  $\varphi \in \text{Cl}(\Sigma) \Rightarrow \varphi \in H$  sii  $\bar{\varphi} \notin H$
- III.  $\varphi \wedge \psi \in \text{Cl}(\Sigma) \Rightarrow \varphi \wedge \psi \in H$  sii  $\varphi \in H$  y  $\psi \in H$
- IV.  $\varphi \vee \psi \in \text{Cl}(\Sigma) \Rightarrow \varphi \vee \psi \in H$  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 si  $\Sigma$  es verdadero en un mundo.

# Satisfacibilidad de K está en PSPACE

Hintikka sets – ¿para qué?

## Teorema

$\Sigma$  es satisfacible sii existe un Hintikka set para  $\Sigma, H$ , que es satisfacible, e incluye a  $\Sigma$ .

# Satisfacibilidad de K está en PSPACE

Hintikka sets – ¿para qué?

## Teorema

$\Sigma$  es satisfacible sii existe un Hintikka set para  $\Sigma$ ,  $H$ , que es satisfacible, e incluye a  $\Sigma$ .

## Demostración

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

$\Rightarrow$ )

- Dado  $\mathcal{M}, w \models \Sigma$ , sea  $H = \{\varphi \mid \mathcal{M}, w \models \varphi \text{ y } \varphi \in \text{Cl}(\Sigma)\}$
- 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:  $\Box(H) = \{\varphi \mid \Box\varphi \in H\}$



# 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  $\mathcal{M}, v \models \Box(H)$ .

# 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  $\mathcal{M}_i = \langle W_i, R_i, V_i \rangle$  tal que  $\mathcal{M}_i, w_i \models H_i$ .
  - Sea  $\mathcal{M}$  la unión disjunta de los  $\mathcal{M}_i$ , con el agregado de un nuevo  $w$  tal que  $wRw_i$  para todo  $w_i$  y  $w \in V(p)$  sii  $p \in H$ .
  - Claramente,  $\mathcal{M}, w_i \leftrightarrow \mathcal{M}_i, w_i$ , con lo cual  $\mathcal{M}, w_i \models H_i$ .
  - Por las clausuras de  $H$ , es fácil ver que,  $\mathcal{M}, w \models H$ .

# Satisfacibilidad de K está en PSPACE

Un algoritmo no-determinístico basado en Hintikka sets

EsSat ( $\Sigma$ )

$\text{subfs}_{\Sigma} \leftarrow \text{Cl}(\Sigma)$

$H \leftarrow$  adivinar un subconjunto de  $\text{subfs}_{\Sigma}$

si  $H$  no es un *Hintikka set* sobre  $\Sigma$

    devolver 0

para todo  $\Diamond\varphi \in H$

    si  $\text{EsSat}(\{\varphi\} \cup \Box(H)) = 0$

        devolver 0

devolver 1

# Satisfacibilidad de $K$ está en PSPACE

Un algoritmo no-determinístico basado en Hintikka sets

$\text{EsSat}(\Sigma)$

$\text{subfs}_{\Sigma} \leftarrow \text{Cl}(\Sigma)$

$H \leftarrow$  adivinar un subconjunto de  $\text{subfs}_{\Sigma}$

si  $H$  no es un *Hintikka set sobre  $\Sigma$*

devolver 0

para todo  $\Diamond\varphi \in H$

si  $\text{EsSat}(\{\varphi\} \cup \Box(H)) = 0$

devolver 0

devolver 1

## Observaciones

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

# Satisfacibilidad de K está en PSPACE

Recapitulando

EsSat( $\Sigma$ )

- 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.

# Satisfacibilidad de K está en PSPACE

Recapitulando

$\text{EsSat}(\Sigma)$

- 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?