# Un algoritmo de Verificación para CTL

Dante Zanarini

LCC

November 11, 2015

# Preguntas

## ¿Qué queremos?

Un algoritmo que, tomando como entrada una fórmula $\phi \in$ **CTL** y un sistema de transiciones $\mathcal{M}$, decida si $\mathcal{M} \models \phi$

## ¿Cuándo lo queremos?

Para las 12 y media.

## ¿Podemos lograrlo?

Y..., no sé. Depende de si **CTL** es **decidible**

# Hoja de ruta

1. Primero, transformaremos $\phi$ en una nueva fórmula $\psi \in \mathbf{CTL}$ tal que $\psi \equiv \phi$, pero $\psi$ utiliza sólo los conectivos temporales $\exists\bigcirc$, $\exists\,\mathbf{U}$ y $\forall\Diamond$
2. Luego, calcularemos el conjunto de estados

$$\mathbf{Sat}(\psi) = \{s \in S \mid \mathcal{M}, s \models \psi\}$$

3. Si $I \subseteq \mathbf{Sat}(\psi)$, entonces $\mathcal{M} \models \phi$

# Pre-primer paso

- Recordemos la semántica de lo que serán nuestros *operadores básicos*:

  1. $\mathcal{M}, s \models \exists[\phi \,\mathbf{U}\, \psi]$ sii para alguna traza $s = s_0 \to s_1 \to s_2 \to \ldots$, existe $j \in \mathbb{N}$ tal que:
     - $\mathcal{M}, s_j \models \psi$
     - $\mathcal{M}, s_i \models \phi$, para todo $i < j$

  2. $\mathcal{M}, s \models \exists\bigcirc\phi$ sii para algún $s'$ tal que $s \to s'$, $\mathcal{M}, s' \models \phi$

  3. $\mathcal{M}, s \models \forall\Diamond\phi$ sii para toda traza $s = s_0 \to s_1 \to s_2 \to \ldots$, existe $j$ tal que $\mathcal{M}, s_j \models \phi$

# Primer paso

$$
\begin{array}{lcl}
T(p) & = & p \\
T(\bot) & = & \bot \\
T(\phi \wedge \psi) & = & T(\phi) \wedge T(\psi) \\
T(\neg\phi) & = & \neg T(\phi) \\
T(\exists\bigcirc\phi) & = & \exists\bigcirc T(\phi) \\
T(\exists[\phi \,\mathbf{U}\, \psi]) & = & \exists[T(\phi) \,\mathbf{U}\, T(\psi)] \\
T(\forall\Diamond\phi) & = & \forall\Diamond T(\phi)
\end{array}
$$

$$
\begin{array}{lcl}
T(\forall\bigcirc\phi) & = & T(\neg\exists\bigcirc\neg\phi) = \neg\exists\bigcirc\neg T(\phi) \\
T(\forall[\phi \,\mathbf{U}\, \psi]) & = & T(\neg(\exists[\neg\psi \,\mathbf{U}\, (\neg\phi \wedge \neg\psi)] \vee \exists\Box\neg\psi)) \\
 & = & \ldots \\
 & = & \neg T(\exists[\neg\psi \,\mathbf{U}\, (\neg\phi \wedge \neg\psi)]) \wedge \neg T(\neg\forall\Diamond\psi) \\
 & = & \ldots \\
 & = & \text{algo que sólo utiliza los operadores de más arriba} \\
 & & \text{y los argumentos de T son } \phi, \psi
\end{array}
$$

# Segundo paso

- Definimos $\psi = T(\phi)$
- Debemos calcular el conjunto $\mathbf{Sat}(\psi)$
- Lo haremos por recursión en la fórmula, asumiendo que sabemos calcular $\mathbf{Sat}(\psi_1)$, para cualquier subfórmula $\psi_1$ de $\psi$.
- Empecemos por los casos fáciles:

$$\begin{array}{lcl} \mathbf{Sat}(\bot) & = & \emptyset \\ \mathbf{Sat}(p_i) & = & \{s \in S \mid p_i \in L(s)\} \\ \mathbf{Sat}(\neg\psi_1) & = & S - \mathbf{Sat}(\psi_1) \\ \mathbf{Sat}(\psi_1 \wedge \psi_2) & = & \mathbf{Sat}(\psi_1) \cap \mathbf{Sat}(\psi_1) \end{array}$$

# Segundo paso, definiciones auxiliares

- Para trabajar con los operadores temporales, definimos las siguientes funciones sobre conjuntos:

$$\begin{aligned} \mathbf{pre}_{\exists}(Y) &= \{s \in S \mid \text{existe } s' \text{ tal que } s \to s' \text{ y } s' \in Y\} \\ \mathbf{pre}_{\forall}(Y) &= \{s \in S \mid \text{para todo } s' \text{ tal que } s \to s' \text{ se cumple } s' \in Y\} \end{aligned}$$

- Un estado está en $\mathbf{pre}_{\exists}(Y)$ sii tiene algún sucesor en $Y$
- Un estado está en $\mathbf{pre}_{\forall}(Y)$ sii todos sus sucesores están en $Y$

# Segundo paso, operador $\exists\bigcirc$

- Supongamos que tenemos $\mathbf{Sat}(\psi_1)$,
- Calculemos

$$
\begin{aligned}
& s \models \exists\bigcirc\psi_1 \\
\iff & \text{definición de } \models \\
& \text{existe } s' \text{ tal que } s \to s' \text{ y } s' \models \psi_1 \\
\iff & \text{definición de } \mathbf{Sat} \\
& \text{existe } s' \text{ tal que } s \to s' \text{ y } s' \in \mathbf{Sat}(\psi_1) \\
\iff & \text{definición de } \mathbf{pre}_\exists \\
& s \in \mathbf{pre}_\exists(\mathbf{Sat}(\psi_1))
\end{aligned}
$$

- Obtenemos entonces

$$\mathbf{Sat}(\exists\bigcirc\psi) \quad = \quad \mathbf{pre}_\exists(\mathbf{Sat}(\psi))$$

# Segundo paso, operador $\forall\Diamond$

- Conociendo **Sat**$(\psi_1)$, ¿Cómo calculo **Sat**$(\forall\Diamond\psi_1)$?
- Algunas pistas:

(1) Si vale ahora, es inevitable: **Sat**$(\psi_1) \subseteq$ **Sat**$(\forall\Diamond\psi_1)$

(2) Si para todos mis sucesores $\psi_1$ es inevitable, para mi también

(3) Es decir, si todos mis sucesores están en **Sat**$(\psi_1)$, yo también

(4) Por lo tanto, si yo estoy en **pre**$_\forall$(**Sat**$(\psi_1)$), $\psi_1$ es inevitable para mi

(5) Es decir, **Sat**$(\psi_1) \cup$ **pre**$_\forall$(**Sat**$(\psi_1)) \subseteq$ **Sat**$(\forall\Diamond\psi_1)$

(6) Volver a la pista (2)

# Segundo paso, operador $\forall\Diamond$

- Proponemos el siguiente procedimiento para calcular $\mathbf{Sat}(\forall\Diamond\psi_1)$:

$$
\begin{array}{l}
\mathit{inev}(Y)\{ \\
\quad \textbf{while } (Y \neq Y \cup \mathbf{pre}_{\forall}(Y)) \textbf{ do} \\
\quad\quad Y \leftarrow Y \cup \mathbf{pre}_{\forall}(Y)\ ; \\
\quad \textbf{return } Y \\
\}
\end{array}
$$

- Tenemos entonces

$$\mathbf{Sat}(\forall\Diamond\psi_1) = \mathit{inev}(\mathbf{Sat}(\psi_1))$$

- Debemos ver que este programa termina y es correcto

# Segundo paso, operador $\exists\,\mathbf{U}$

- Conociendo $\mathbf{Sat}(\psi_1)$ y $\mathbf{Sat}(\psi_2)$, cómo calculo $Y = \mathbf{Sat}(\exists[\psi_1\,\mathbf{U}\,\psi_2])$?
- Pistas:

1. Si un estado satisface $\psi_2$, entonces está en $Y$
2. Si un estado satisface $\psi_1$, y tiene algún sucesor en $Y$, entonces debería estar en $Y$
3. Por lo tanto, si $s \in \mathbf{Sat}(\psi_1) \cap \mathbf{pre}_\exists(Y)$, entonces debería pertenecer a $Y$

# Segundo paso, operador $\exists\,\mathbf{U}$

- Proponemos el siguiente prodedimiento para calcular $\mathbf{Sat}(\exists[\psi_1\,\mathbf{U}\,\psi_2])$

$ex\text{-}until(X, Y)\{$
$\quad$ **while** $(Y \neq Y \cup (X \cap \mathbf{pre}_\exists(Y)))$ **do**
$\quad\quad Y \leftarrow Y \cup (X \cap \mathbf{pre}_\exists(Y))$ ;
$\quad$ **return** $Y$
$\}$

- Tenemos entonces

$$\mathbf{Sat}(\exists[\psi_1\,\mathbf{U}\,\psi_2]) = ex\text{-}until(\mathbf{Sat}(\psi_1), \mathbf{Sat}(\psi_2))$$

- Nuevamente, dejamos terminación y correctitud para más adelante

# Juntando...

$$
\begin{aligned}
\mathbf{Sat}(\bot) &= \emptyset \\
\mathbf{Sat}(p_i) &= \{s \in S \mid p_i \in L(s)\} \\
\mathbf{Sat}(\neg\psi_1) &= S - \mathbf{Sat}(\psi_1) \\
\mathbf{Sat}(\psi_1 \wedge \psi_2) &= \mathbf{Sat}(\psi_1) \cap \mathbf{Sat}(\psi_1) \\
\mathbf{Sat}(\exists\bigcirc\psi) &= \mathbf{pre}_{\exists}(\mathbf{Sat}(\psi)) \\
\mathbf{Sat}(\forall\Diamond\psi_1) &= \mathit{inev}(\mathbf{Sat}(\psi_1)) \\
\mathbf{Sat}(\exists[\psi_1 \,\mathbf{U}\, \psi_2]) &= \mathit{ex\text{-}until}(\mathbf{Sat}(\psi_1), \mathbf{Sat}(\psi_2))
\end{aligned}
$$