Lógica Computacional y Demostración Automática

Carlos Areces

areces@loria.fr

http://www.loria.fr/~areces

INRIA Nancy Grand Est, France

Diciembre 2008

Lo que hacemos hoy

Lo que hacemos hoy

- Describiendo estructuras relacionales
- Lógicas modales
- Sintaxis y semántica
- Lógicas híbridas
- Model Checking
- Aplicaciones
- El model checker `mcheck`

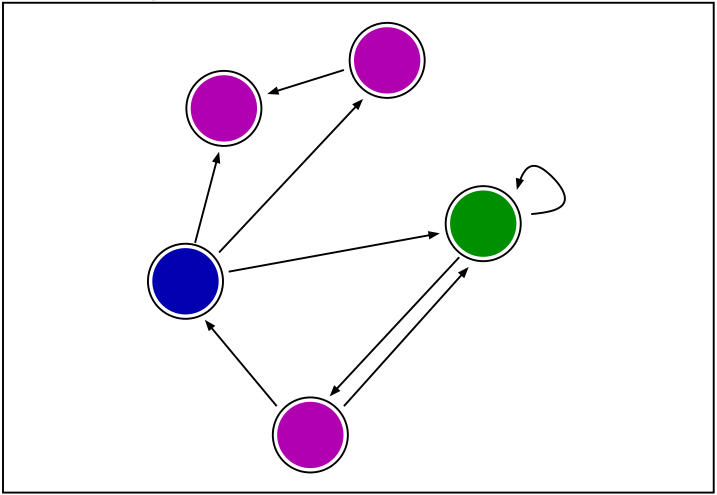
Estructuras Simples / Lenguajes Simples

Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:

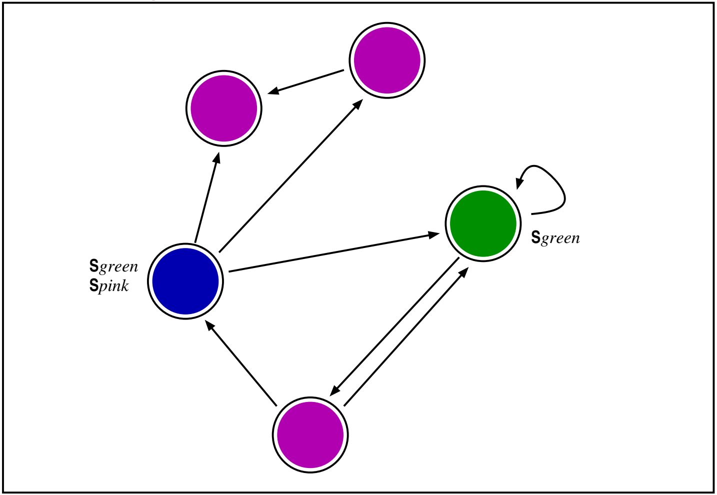
Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



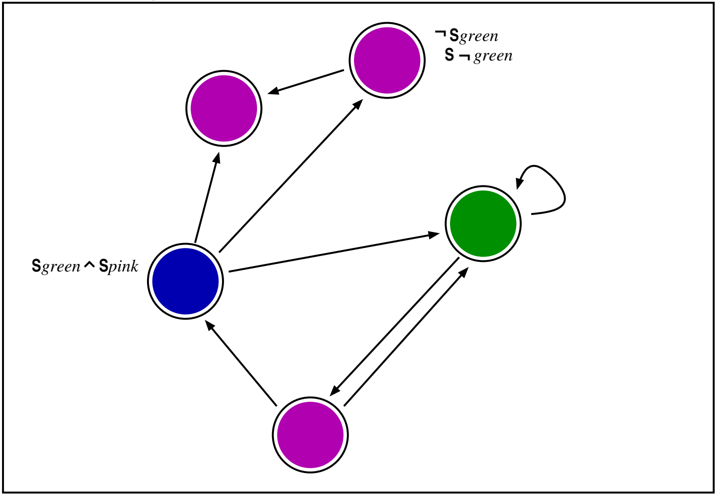
Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



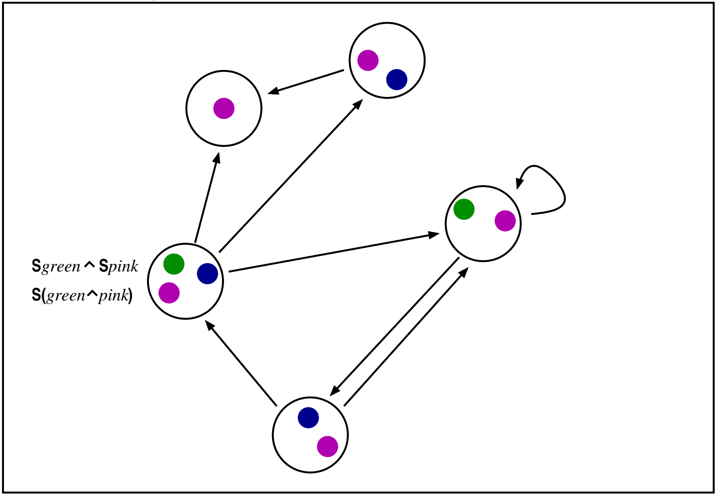
Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



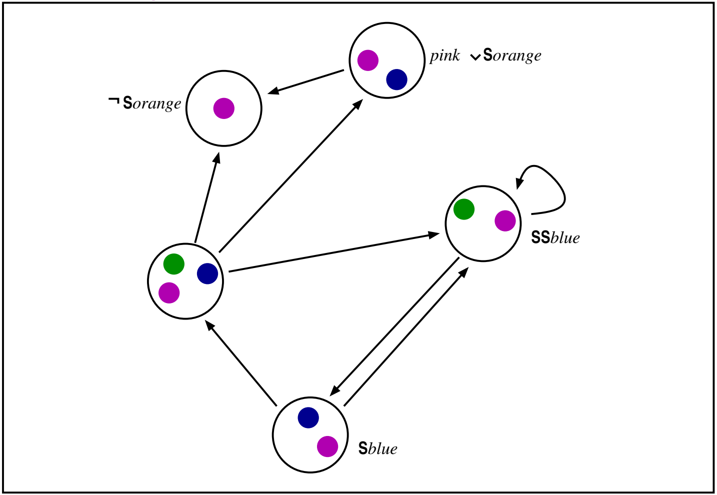
Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



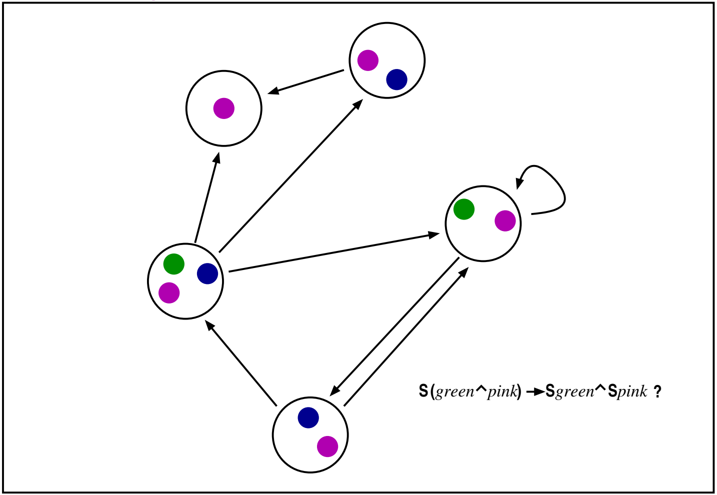
Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



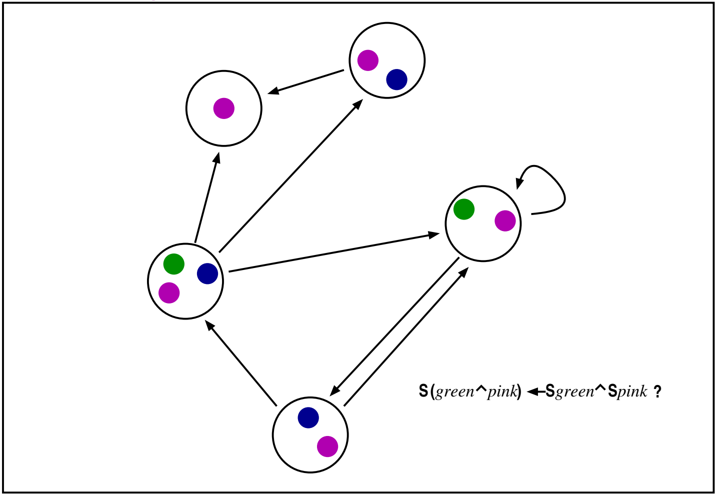
Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



Estructuras Simples / Lenguajes Simples

Pensemos en un grafo coloreado:



Estructuras Simples / Lenguajes Simples

Estructuras Simples / Lenguajes Simples

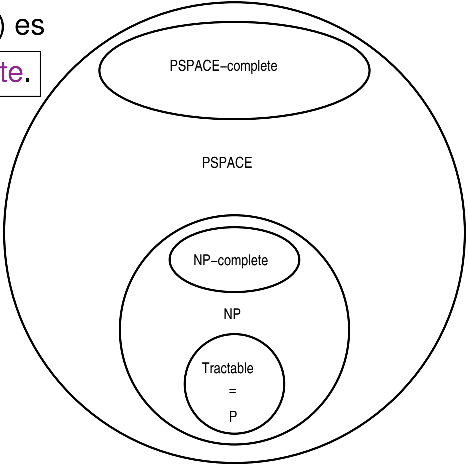
- Aun para este lenguaje (aparentemente) muy simple, decidir si una fórmula es un teorema (i.e., siempre cierta) es

$$\boxed{\text{PSPACE-complete.}}$$

Estructuras Simples / Lenguajes Simples

- Aun para este lenguaje (aparentemente) muy simple, decidir si una fórmula es un teorema (i.e., siempre cierta) es

  PSPACE-complete.



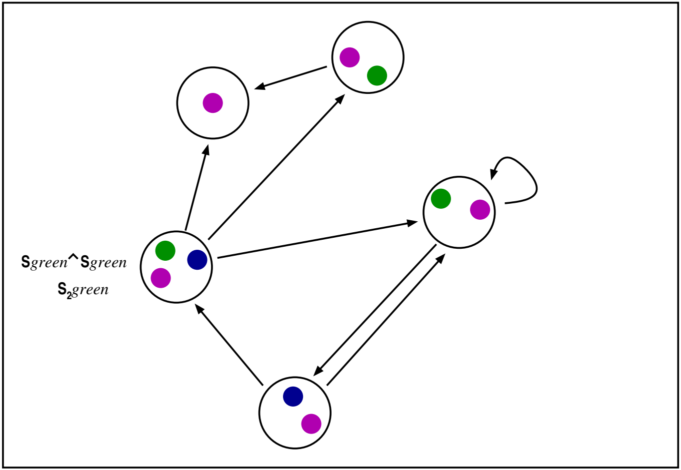
Una Familia de Lenguajes

Una Familia de Lenguajes

- El lenguaje que describimos puede ser inadecuado:

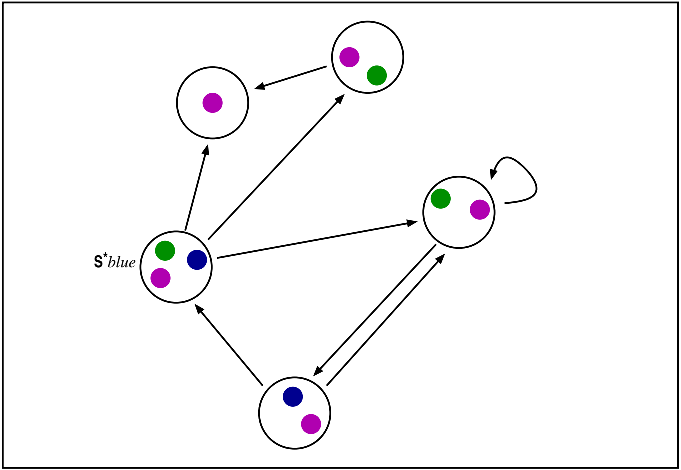
Una Familia de Lenguajes

- El lenguaje que describimos puede ser inadecuado:



Una Familia de Lenguajes

- El lenguaje que describimos puede ser inadecuado:



Una Familia de Lenguajes

- El lenguaje que describimos puede ser inadecuado:
- En área de logica modal, investiga el rango de posibles lenguajes que pueden usarse para describir estructuras relacionales.

Una Familia de Lenguajes

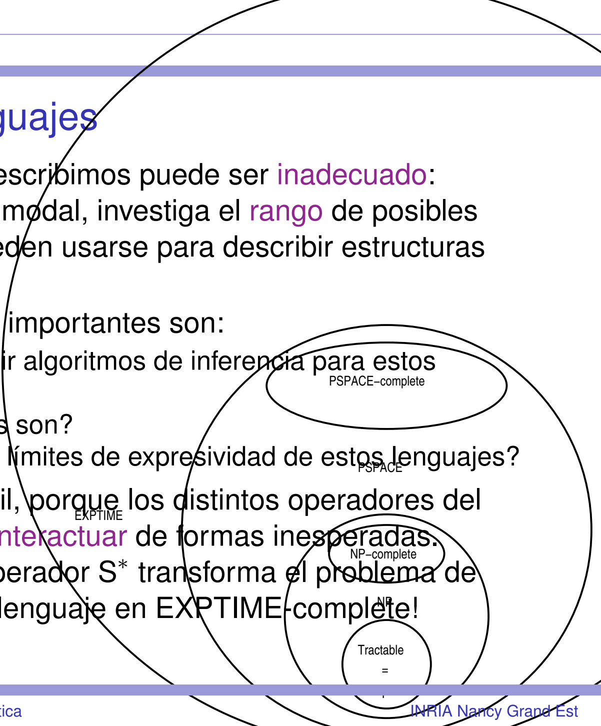
- El lenguaje que describimos puede ser inadecuado:
- En área de logica modal, investiga el rango de posibles lenguajes que pueden usarse para describir estructuras relacionales.
- La preguntas más importantes son:
  - Podemos definir algoritmos de inferencia para estos lenguajes?
  - Cuan eficientes son?
  - Cuáles son los límites de expresividad de estos lenguajes?

Una Familia de Lenguajes

- El lenguaje que describimos puede ser inadecuado:
- En área de logica modal, investiga el rango de posibles lenguajes que pueden usarse para describir estructuras relacionales.
- La preguntas más importantes son:
  - Podemos definir algoritmos de inferencia para estos lenguajes?
  - Cuan eficientes son?
  - Cuáles son los límites de expresividad de estos lenguajes?
- La tarea no es fácil, porque los distintos operadores del lenguaje pueden interactuar de formas inesperadas. E.g., agregar el operador S^* transforma el problema de satisfiabilidad del lenguaje en EXPTIME-complete!

Una Familia de Lenguajes

- El lenguaje que describimos puede ser inadecuado:
- En área de logica modal, investiga el rango de posibles lenguajes que pueden usarse para describir estructuras relacionales.
- La preguntas más importantes son:
  - Podemos definir algoritmos de inferencia para estos lenguajes?
  - Cuan eficientes son?
  - Cuáles son los límites de expresividad de estos lenguajes?
- La tarea no es fácil, porque los distintos operadores del lenguaje pueden interactuar de formas inesperadas. E.g., agregar el operador S^* transforma el problema de satisfiabilidad del lenguaje en EXPTIME-complete!



Posibles Aplicaciones

Posibles Aplicaciones

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

Posibles Aplicaciones

- 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.
  - Linguística Computacional.
  - Epistemología.
  - ...

Posibles Aplicaciones

- 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.
  - Linguística Computacional.
  - Epistemología.
  - ...
- **Porque?**

Posibles Aplicaciones

- 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.
  - Linguística Computacional.
  - Epistemología.
  - ...
- **Porque?** Muchas cosas pueden ser representadas como grafos (i.e., estructuras relacionales).

Posibles Aplicaciones

- 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.
  - Linguística Computacional.
  - Epistemología.
  - ...
- **Porque?** Muchas cosas pueden ser representadas como grafos (i.e., estructuras relacionales).
  Y como vimos, los lenguajes modales fueron desarrollados especialmente para razonar sobre grafos y describir sus propiedades.

Lógicas Híbridas

Lógicas Híbridas

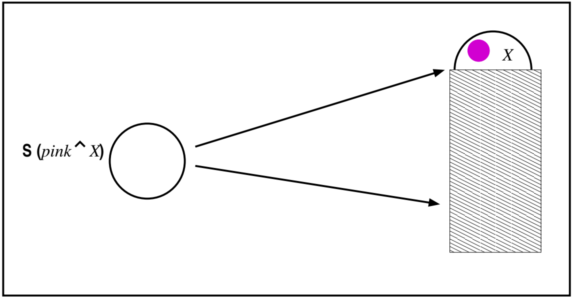
- Vamos a presentar un tipo particular de lenguajes modales llamados lenguajes híbridos.

Lógicas Híbridas

- Vamos a presentar un tipo particular de lenguajes modales llamados lenguajes híbridos.
- Intuitivamente, los lenguajes híbridos son lenguajes modales que incluyen alguna noción de igualdad.

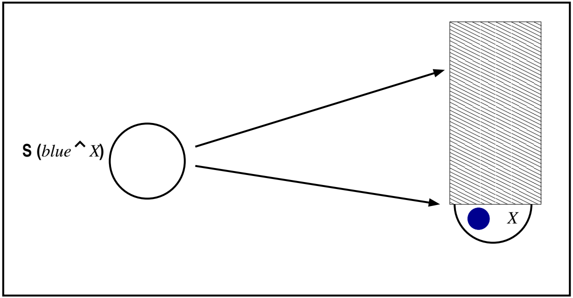
Lógicas Híbridas

- Vamos a presentar un tipo particular de lenguajes modales llamados lenguajes híbridos.
- Intuitivamente, los lenguajes híbridos son lenguajes modales que incluyen alguna noción de igualdad.



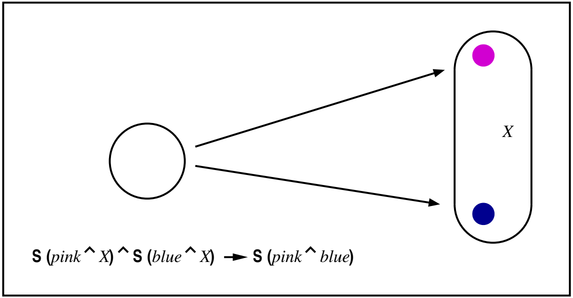
Lógicas Híbridas

- Vamos a presentar un tipo particular de lenguajes modales llamados lenguajes híbridos.
- Intuitivamente, los lenguajes híbridos son lenguajes modales que incluyen alguna noción de igualdad.



Lógicas Híbridas

- Vamos a presentar un tipo particular de lenguajes modales llamados lenguajes híbridos.
- Intuitivamente, los lenguajes híbridos son lenguajes modales que incluyen alguna noción de igualdad.



El Sabor de un Clásico…

- Todo muy moderno, muy superado, pero yo prefiero la lógica clásica.

El Sabor de un Clásico…

- Todo muy moderno, muy superado, pero yo prefiero la lógica clásica.
- Aunque no parezca, estamos haciendo Lógica Clásica!!!

El Sabor de un Clásico…

- Todo muy moderno, muy superado, pero yo prefiero la lógica clásica.
- Aunque no parezca, estamos haciendo Lógica Clásica!!!
- Los lenguajes que estuvimos discutiendo son fragmentos del lenguaje de primer (o segundo) orden. Lo único que hicimos fue elegir ‘el pedacito’ que necesitábamos para una aplicación dada.

El Sabor de un Clásico. . .

- Todo muy moderno, muy superado, pero yo prefiero la lógica clásica.
- Aunque no parezca, estamos haciendo Lógica Clásica!!!
- Los lenguajes que estuvimos discutiendo son fragmentos del lenguaje de primer (o segundo) orden. Lo único que hicimos fue elegir ‘el pedacito’ que necesitábamos para una aplicación dada.
- Esta es exactamente la forma en que vemos hoy por hoy a los lenguajes modales, como una forma de investigar fragmentos particularmente interesantes de los lenguajes clásicos.

El Sabor de un Clásico...

- Todo muy moderno, muy superado, pero yo prefiero la lógica clásica.
- Aunque no parezca, estamos haciendo Lógica Clásica!!!
- Los lenguajes que estuvimos discutiendo son fragmentos del lenguaje de primer (o segundo) orden. Lo único que hicimos fue elegir 'el pedacito' que necesitábamos para una aplicación dada.
- Esta es exactamente la forma en que vemos hoy por hoy a los lenguajes modales, como una forma de investigar fragmentos particularmente interesantes de los lenguajes clásicos.
- Donde "interesantes" significa
  - Decidibles, expresivos, de "baja" complejidad, modulares, ...

Sintaxis

Sintaxis

- El lenguaje hibrido que vamos a usar se define sobre la signatura $S = \langle \mathrm{PROP}, \mathrm{REL}, \mathrm{NOM} \rangle$ donde
  - $\mathrm{PROP} = \{p, q, r, \ldots\}$ es el conjunto de simbolos de proposicion

Sintaxis

- El lenguaje hibrido que vamos a usar se define sobre la signatura $S = \langle \mathrm{PROP}, \mathrm{REL}, \mathrm{NOM} \rangle$ donde
  - $\mathrm{PROP} = \{p, q, r, \ldots\}$ es el conjunto de simbolos de proposicion
  - $\mathrm{REL} = \{r_1, r_2, r_3, \ldots\}$ es el conjunto de simbolos de relacion

Sintaxis

- El lenguaje hibrido que vamos a usar se define sobre la signatura $S = \langle \text{PROP}, \text{REL}, \text{NOM} \rangle$ donde
  - $\text{PROP} = \{p, q, r, \ldots\}$ es el conjunto de simbolos de proposicion
  - $\text{REL} = \{r_1, r_2, r_3, \ldots\}$ es el conjunto de simbolos de relacion
  - $\text{NOM} = \{i, j, k, \ldots\}$ es el conjunto de simbolos de constantes (los llamamos nominales)

Sintaxis

- El lenguaje hibrido que vamos a usar se define sobre la signatura $S = \langle \mathrm{PROP}, \mathrm{REL}, \mathrm{NOM} \rangle$ donde
  - $\mathrm{PROP} = \{p, q, r, \ldots\}$ es el conjunto de simbolos de proposicion
  - $\mathrm{REL} = \{r_1, r_2, r_3, \ldots\}$ es el conjunto de simbolos de relacion
  - $\mathrm{NOM} = \{i, j, k, \ldots\}$ es el conjunto de simbolos de constantes (los llamamos nominales)
  - $\mathrm{VAR} = \{x, y, z, \ldots\}$ es el conjunto de variables

Sintaxis

- El lenguaje hibrido que vamos a usar se define sobre la signatura $S = \langle \text{PROP}, \text{REL}, \text{NOM} \rangle$ donde
  - $\text{PROP} = \{p, q, r, \ldots\}$ es el conjunto de simbolos de proposicion
  - $\text{REL} = \{r_1, r_2, r_3, \ldots\}$ es el conjunto de simbolos de relacion
  - $\text{NOM} = \{i, j, k, \ldots\}$ es el conjunto de simbolos de constantes (los llamamos nominales)
  - $\text{VAR} = \{x, y, z, \ldots\}$ es el conjunto de variables
- Las formulas se definen entonces como

$$\varphi \ := \ a \ | \ \neg\varphi \ | \ \varphi \wedge \varphi' \ | \ \langle r \rangle \varphi \ | \ \langle r \rangle^{-} \varphi \ | \ \mathsf{E}\varphi \ | \ @_i \varphi \ | \ {\downarrow} x.\varphi$$

donde $a \in \text{NOM} \cup \text{PROP} \cup \text{VAR}$, $r \in \text{REL}$, $i \in \text{NOM}$ y $x \in \text{VAR}$.

Sintaxis

- El lenguaje hibrido que vamos a usar se define sobre la signatura $S = \langle \mathrm{PROP}, \mathrm{REL}, \mathrm{NOM} \rangle$ donde
  - $\mathrm{PROP} = \{p, q, r, \ldots\}$ es el conjunto de simbolos de proposicion
  - $\mathrm{REL} = \{r_1, r_2, r_3, \ldots\}$ es el conjunto de simbolos de relacion
  - $\mathrm{NOM} = \{i, j, k, \ldots\}$ es el conjunto de simbolos de constantes (los llamamos nominales)
  - $\mathrm{VAR} = \{x, y, z, \ldots\}$ es el conjunto de variables
- Las formulas se definen entonces como

$$\varphi \ := \ a \ | \ \neg\varphi \ | \ \varphi \wedge \varphi' \ | \ \langle r \rangle \varphi \ | \ \langle r \rangle^{-} \varphi \ | \ \mathrm{E}\varphi \ | \ @_i \varphi \ | \ \downarrow x.\varphi$$

donde $a \in \mathrm{NOM} \cup \mathrm{PROP} \cup \mathrm{VAR}$, $r \in \mathrm{REL}$, $i \in \mathrm{NOM}$ y $x \in \mathrm{VAR}$.(La formula $[r]\varphi$ se define como $\neg\langle r \rangle \neg\varphi$ y $\mathrm{A}\varphi$ se define como $\neg\mathrm{E}\neg\varphi$)

Semantica

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in \text{REL}$,
  - $I(p)$ es un subconjunto de M para $p \in \text{PROP}$,

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \to M$.

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I\rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \to M$.
- Definimos

$$\mathcal{M}, g, m \models i \quad \text{sii} \quad m = I(i),\ i \in \mathrm{NOM}$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I\rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in \mathrm{REL}$,
  - $I(p)$ es un subconjunto de M para $p \in \mathrm{PROP}$,
  - $I(i)$ es un elemento de M para $i \in \mathrm{NOM}$

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \to M$.
- Definimos

$$\begin{array}{rcl} \mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \mathrm{NOM} \\ \mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \mathrm{PROP} \end{array}$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in \mathrm{REL}$,
  - $I(p)$ es un subconjunto de M para $p \in \mathrm{PROP}$,
  - $I(i)$ es un elemento de M para $i \in \mathrm{NOM}$

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \to M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \mathrm{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \mathrm{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi
\end{array}
$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \rightarrow M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \mathrm{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \mathrm{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi \\
\mathcal{M}, g, m \models \varphi \wedge \psi & \text{sii} & \mathcal{M}, g, m \models \varphi \text{ y } \mathcal{M}, g, m \models \psi
\end{array}
$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I\rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \text{VAR} \to M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \text{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \text{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi \\
\mathcal{M}, g, m \models \varphi \wedge \psi & \text{sii} & \mathcal{M}, g, m \models \varphi \text{ y } \mathcal{M}, g, m \models \psi \\
\mathcal{M}, g, m \models \langle r\rangle\varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m, m') \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi
\end{array}
$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \text{VAR} \rightarrow M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \text{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \text{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi \\
\mathcal{M}, g, m \models \varphi \wedge \psi & \text{sii} & \mathcal{M}, g, m \models \varphi \text{ y } \mathcal{M}, g, m \models \psi \\
\mathcal{M}, g, m \models \langle r \rangle \varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m, m') \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \langle r \rangle^{-} \varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m', m) \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi
\end{array}
$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \to M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \mathrm{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \mathrm{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi \\
\mathcal{M}, g, m \models \varphi \wedge \psi & \text{sii} & \mathcal{M}, g, m \models \varphi \text{ y } \mathcal{M}, g, m \models \psi \\
\mathcal{M}, g, m \models \langle r \rangle \varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m, m') \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \langle r \rangle^{-} \varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m', m) \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \mathrm{E}\varphi & \text{sii} & \exists m' \in M \text{ t.q. } \mathcal{M}, g, m' \models \varphi
\end{array}
$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I \rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \text{VAR} \rightarrow M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \text{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \text{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi \\
\mathcal{M}, g, m \models \varphi \wedge \psi & \text{sii} & \mathcal{M}, g, m \models \varphi \text{ y } \mathcal{M}, g, m \models \psi \\
\mathcal{M}, g, m \models \langle r \rangle\varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m, m') \in I(r) \ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \langle r \rangle^{-}\varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m', m) \in I(r) \ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \mathrm{E}\varphi & \text{sii} & \exists m' \in M \text{ t.q. } \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models @_i\varphi & \text{sii} & \mathcal{M}, g, I(i) \models \varphi
\end{array}
$$

Semantica

- Un modelo $\mathcal{M}$ es de la forma $\langle M, I\rangle$ donde M es no vacio, y
  - $I(r)$ es una relacion binaria sobre M para $r \in$ REL,
  - $I(p)$ es un subconjunto de M para $p \in$ PROP,
  - $I(i)$ es un elemento de M para $i \in$ NOM

  Una asignación g para $\mathcal{M}$ es una funcion $g : \mathrm{VAR} \to M$.
- Definimos

$$
\begin{array}{rcl}
\mathcal{M}, g, m \models i & \text{sii} & m = I(i),\ i \in \mathrm{NOM} \\
\mathcal{M}, g, m \models p & \text{sii} & m \in I(p),\ p \in \mathrm{PROP} \\
\mathcal{M}, g, m \models \neg\varphi & \text{sii} & \mathcal{M}, g, m \not\models \varphi \\
\mathcal{M}, g, m \models \varphi \wedge \psi & \text{sii} & \mathcal{M}, g, m \models \varphi \text{ y } \mathcal{M}, g, m \models \psi \\
\mathcal{M}, g, m \models \langle r\rangle\varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m, m') \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \langle r\rangle^{-}\varphi & \text{sii} & \exists m' \in M \text{ t.q. } (m', m) \in I(r)\ \&\ \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models \mathrm{E}\varphi & \text{sii} & \exists m' \in M \text{ t.q. } \mathcal{M}, g, m' \models \varphi \\
\mathcal{M}, g, m \models @_i\varphi & \text{sii} & \mathcal{M}, g, I(i) \models \varphi \\
\mathcal{M}, g, m \models \downarrow x.\varphi & \text{sii} & \mathcal{M}, g[x \leftarrow m], m \models \varphi
\end{array}
$$

Que podemos escribir?

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^{-})$

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^-, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^-)$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.
- Cual es el poder expresivo de $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-})$

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\text{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.
- Cual es el poder expresivo de $\mathcal{L}(\text{NOM}, \langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal hibrido básico
  - SAT es decidible (EXPTIME-complete).
  - El fragmento guarded de LPO^2 con constantes.

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.
- Cual es el poder expresivo de $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal hibrido básico
  - SAT es decidible (EXPTIME-complete).
  - El fragmento guarded de LPO^2 con constantes.
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \downarrow, E)$

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.
- Cual es el poder expresivo de $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal hibrido básico
  - SAT es decidible (EXPTIME-complete).
  - El fragmento guarded de LPO^2 con constantes.
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \downarrow, E)$
  - Es tan expresivo como LPO!!!
  - SAT es indecidible.

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^-, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^-)$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.
- Cual es el poder expresivo de $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^-)$
  - Es el lenguaje temporal hibrido básico
  - SAT es decidible (EXPTIME-complete).
  - El fragmento guarded de LPO^2 con constantes.
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \downarrow, E)$
  - Es tan expresivo como LPO!!!
  - SAT es indecidible.
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \downarrow)$

Que podemos escribir?

- Cuán expresivo es el lenguaje $\mathcal{L}(\mathrm{NOM}, @, \langle r \rangle, \langle r \rangle^{-}, E, \downarrow)$
- Empecemos con menos (y para los que saben...)
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal básico (futuro y pasado)
  - SAT es decidible (PSPACE-complete).
  - El fragmento guarded de LPO^2.
- Cual es el poder expresivo de $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-})$
  - Es el lenguaje temporal hibrido básico
  - SAT es decidible (EXPTIME-complete).
  - El fragmento guarded de LPO^2 con constantes.
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \downarrow, E)$
  - Es tan expresivo como LPO!!!
  - SAT es indecidible.
- Cual es el poder expresivo de $\mathcal{L}(\langle r \rangle, \downarrow)$
  - Es menos expresivo que LPO.
  - SAT es todavia indecidible.

Model Checking

Model Checking

- El problema de model checking global es el siguiente:
  - Dado un modelog $\mathcal{M}$
  - y una formula α,
  - retornar todos los estados de $\mathcal{M}$ en los que α es verdadero.

Model Checking

- El problema de model checking global es el siguiente:
  - Dado un modelog $\mathcal{M}$
  - y una formula α,
  - retornar todos los estados de $\mathcal{M}$ en los que α es verdadero.
- Definimos el truth set de una fórmula α respecto de un modelo $\mathcal{M} = \langle W, R, V \rangle$ como:

$$Truth(\mathcal{M}, \alpha) = \{w \in W \mid \mathcal{M}, w \models \alpha\}$$

Model Checking

- El problema de model checking global es el siguiente:
  - Dado un modelog $\mathcal{M}$
  - y una formula α,
  - retornar todos los estados de $\mathcal{M}$ en los que α es verdadero.
- Definimos el truth set de una fórmula α respecto de un modelo $\mathcal{M} = \langle W, R, V \rangle$ como:

$$\mathit{Truth}(\mathcal{M}, \alpha) = \{w \in W \mid \mathcal{M}, w \models \alpha\}$$

- Un model checker es un programa que dado $\mathcal{M}$ y α retorna $\mathit{Truth}(\mathcal{M}, \alpha)$.

El Model Checker MCLite

El Model Checker MCLite

- MCLite es un model checker para el lenguaje $\mathcal{L}(NOM, \langle r \rangle, \langle r \rangle^{-}, @, E)$.

El Model Checker MCLite

- MCLite es un model checker para el lenguaje $\mathcal{L}(NOM, \langle r \rangle, \langle r \rangle^{-}, @, E)$.
- Por simplicidad, trabajaremos con lógicas monomodales (sólo $\langle r \rangle$) pero la extensión a lógicas multimodales es simple.

El Model Checker MCLite

- MCLite es un model checker para el lenguaje $\mathcal{L}(\mathit{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E)$.
- Por simplicidad, trabajaremos con lógicas monomodales (sólo $\langle r \rangle$) pero la extensión a lógicas multimodales es simple.
- El algoritmo usa una estrategia bottom-up: examina las subfórmulas de la fórmula input α en forma incremental, hasta que finalmente el obtenemos el truth set de α.
- Si una subfórmula β de α es verdadera en un estado w, el model checker marca w con β.

Tipos de Datos y Funcionaes Auxiliares

Tipos de Datos y Funcionaes Auxiliares

- Sea $\mathcal{M} = \langle W, R, V \rangle$ un modelo y α la fórmula que queremos chequear.

Tipos de Datos y Funcionaes Auxiliares

- Sea $\mathcal{M} = \langle W, R, V \rangle$ un modelo y α la fórmula que queremos chequear.
- Sea $sub(\alpha)$ el conjunto de subfórmulas de α.

Tipos de Datos y Funcionaes Auxiliares

- Sea $\mathcal{M} = \langle W, R, V\rangle$ un modelo y α la fórmula que queremos chequear.
- Sea $sub(\alpha)$ el conjunto de subfórmulas de α.
- El model checker mantendrá una bit table L de tamaño $|sub(\alpha)| \times |W|$ tal que, para cada subfórmula β de α y cada $w \in W$,

  $L(\beta, w) = 1$ if $M, w \models \beta$ y $L(\beta, w) = 0$ if $M, w \not\models \beta$

Tipos de Datos y Funcionaes Auxiliares

- Sea $\mathcal{M} = \langle W, R, V \rangle$ un modelo y α la fórmula que queremos chequear.
- Sea $sub(\alpha)$ el conjunto de subfórmulas de α.
- El model checker mantendrá una bit table L de tamaño $|sub(\alpha)| \times |W|$ tal que, para cada subfórmula β de α y cada $w \in W$,

  $$L(\beta, w) = 1 \text{ if } M, w \models \beta \text{ y } L(\beta, w) = 0 \text{ if } M, w \not\models \beta$$
- Ademas, sea $L(\beta) = \{w \in W \mid L(\beta, w) = 1\}$

Tipos de Datos y Funcionaes Auxiliares

- Sea $\mathcal{M} = \langle W, R, V \rangle$ un modelo y α la fórmula que queremos chequear.
- Sea $sub(\alpha)$ el conjunto de subfórmulas de α.
- El model checker mantendrá una bit table L de tamaño $|sub(\alpha)| \times |W|$ tal que, para cada subfórmula β de α y cada $w \in W$,

  $L(\beta, w) = 1$ if $M, w \models \beta$ y $L(\beta, w) = 0$ if $M, w \not\models \beta$
- Ademas, sea $L(\beta) = \{w \in W \mid L(\beta, w) = 1\}$
- Al terminar la corrida tendremos que $Truth(M, \alpha) = L(\alpha)$

Tipos de Datos y Funcionaes Auxiliares

- Sea $\mathcal{M} = \langle W, R, V \rangle$ un modelo y α la fórmula que queremos chequear.
- Sea $sub(\alpha)$ el conjunto de subfórmulas de α.
- El model checker mantendrá una bit table L de tamaño $|sub(\alpha)| \times |W|$ tal que, para cada subfórmula β de α y cada $w \in W$,

  $L(\beta, w) = 1$ if $M, w \models \beta$ y $L(\beta, w) = 0$ if $M, w \not\models \beta$
- Ademas, sea $L(\beta) = \{w \in W \mid L(\beta, w) = 1\}$
- Al terminar la corrida tendremos que $Truth(M, \alpha) = L(\alpha)$
- Finalente dado $w \in W$, sea $R(w)$ el conjunto de R-sucesores de w y $R^{-}(w)$ el conjunto de R-predecesores de w.

MCLite(M, α)

MCLite(M, α)

for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
for $i \leftarrow 1$ **to** $|\alpha|$ **do:**
    for $\beta \in sub(\alpha)$ **such that** $|\beta| = i$ **do:**
        case of β
            • $\beta \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\beta)$ **then** $L(\beta, w) \leftarrow 1$

MCLite(M, α)

for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
for $i \leftarrow 1$ **to** $|\alpha|$ **do:**
  for $\beta \in sub(\alpha)$ **such that** $|\beta| = i$ **do:**
    case of β
      • $\beta \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\beta)$ **then** $L(\beta, w) \leftarrow 1$
      • $\beta = \beta_1 \wedge \beta_2$ **:** $\forall w \in W$**, if** $L(\beta_1, w) = L(\beta_2, w) = 1$ **then** $L(\beta, w) \leftarrow 1$

MCLite(M, α)

for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
for $i \leftarrow 1$ **to** $|\alpha|$ **do:**
  for $\beta \in sub(\alpha)$ **such that** $|\beta| = i$ **do:**
    case of β
      • $\beta \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\beta)$ **then** $L(\beta, w) \leftarrow 1$
      • $\beta = \beta_1 \wedge \beta_2$ **:** $\forall w \in W$**, if** $L(\beta_1, w) = L(\beta_2, w) = 1$ **then** $L(\beta, w) \leftarrow 1$
      • $\beta = \neg\beta_1$ **:** $\forall w \in W$**, if** $L(\beta_1, w) = 0$ **then** $L(\beta, w) \leftarrow 1$

MCLite(M, α)

for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
for $i \leftarrow 1$ **to** $|\alpha|$ **do:**
  for $\beta \in sub(\alpha)$ **such that** $|\beta| = i$ **do:**
    case of β
      • $\beta \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\beta)$ **then** $L(\beta, w) \leftarrow 1$
      • $\beta = \beta_1 \wedge \beta_2$ **:** $\forall w \in W$**, if** $L(\beta_1, w) = L(\beta_2, w) = 1$ **then** $L(\beta, w) \leftarrow 1$
      • $\beta = \neg\beta_1$ **:** $\forall w \in W$**, if** $L(\beta_1, w) = 0$ **then** $L(\beta, w) \leftarrow 1$
      • $\beta = \langle r \rangle \beta_1$ **:** $\mathbf{MC}_{\langle r \rangle}(M, \beta_1)$
      • $\beta = \langle r \rangle^{-} \beta_1$ **:** $\mathbf{MC}_{\langle r \rangle^{-}}(M, \beta_1)$
      • $\beta = E\beta_1$ **:** $\mathbf{MC}_{E}(M, \beta_1)$
      • $\beta = @_i \beta_1$ **:** $\mathbf{MC}_{@}(M, i, \beta_1)$
return $L(\alpha)$

$\mathbf{MC}_{\langle r \rangle}(M, \alpha)$

```
for w ∈ L(α) do
    for v ∈ R⁻¹(w) do
        L(⟨r⟩α, v) ← 1
```

$\mathbf{MC}_{\langle r \rangle}(M, \alpha)$

1: for $w \in L(\alpha)$ **do**
2:   **for** $v \in R^{-1}(w)$ **do**
3: $L(\langle r \rangle \alpha, v) \leftarrow 1$

$\mathbf{MC}_{\langle r \rangle^-}(M, \alpha)$

1: for $w \in L(\alpha)$ **do**
2:   **for** $v \in R(w)$ **do**
3: $L(\langle r \rangle^- \alpha, v) \leftarrow 1$

MC$_{\langle r \rangle}(M, \alpha)$

for $w \in L(\alpha)$ **do**
    for $v \in R^{-1}(w)$ **do**
        $L(\langle r \rangle \alpha, v) \leftarrow 1$

MC$_{\langle r \rangle^-}(M, \alpha)$

for $w \in L(\alpha)$ **do**
    for $v \in R(w)$ **do**
        $L(\langle r \rangle^- \alpha, v) \leftarrow 1$

MC$_E(M, \alpha)$

if $L(\alpha) \neq \emptyset$ **then**
    for $w \in W$ **do**
        $L(E\alpha, w) \leftarrow 1$

$\mathbf{MC}_{\langle r \rangle}(M, \alpha)$

```
for w ∈ L(α) do
    for v ∈ R⁻¹(w) do
        L(⟨r⟩α, v) ← 1
```

$\mathbf{MC}_{\langle r \rangle^-}(M, \alpha)$

```
for w ∈ L(α) do
    for v ∈ R(w) do
        L(⟨r⟩⁻α, v) ← 1
```

$\mathbf{MC}_{E}(M, \alpha)$

```
if L(α) ≠ ∅ then
    for w ∈ W do
        L(Eα, w) ← 1
```

$\mathbf{MC}_{@}(M, i, \alpha)$

```
let {v} = V(i)
if L(α, v) = 1 then
    for w ∈ M do
        L(@ᵢα, w) ← 1
```

Worst-case Complexity de MCLite

Worst-case Complexity de MCLite

- Dado $f, g : N \rightarrow N$, recordemos que

  $$f(n) = O(g(n))$$

  significa que hay una constante $c > 0$ y un número natural $n_0 \geq 1$ tal que

  $$f(n) \leq c \times g(n)$$

  para todo $n \geq n_0$.

Worst-case Complexity de MCLite

- Dado $f, g : N \rightarrow N$, recordemos que

$$f(n) = O(g(n))$$

significa que hay una constante $c > 0$ y un número natural $n_0 \geq 1$ tal que

$$f(n) \leq c \times g(n)$$

para todo $n \geq n_0$.

- Por ejemplo, $2n + 1 = O(n)$.

Contamos…

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;
  - if $\beta = \neg\beta_1$, then the check of β takes $O(n)$;

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;
  - if $\beta = \neg\beta_1$, then the check of β takes $O(n)$;
  - if $\beta = E\beta_1$, then the check of β takes $O(2n) = O(n)$;

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;
  - if $\beta = \neg\beta_1$, then the check of β takes $O(n)$;
  - if $\beta = E\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if $\beta = @_i\beta_1$, then the check of β takes $O(2n) = O(n)$;

Contamos...

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;
  - if $\beta = \neg\beta_1$, then the check of β takes $O(n)$;
  - if $\beta = E\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if $\beta = @_i\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if $\beta = \langle r\rangle\beta_1$, then the check of β takes $O(n + m)$;

Contamos…

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;
  - if $\beta = \neg\beta_1$, then the check of β takes $O(n)$;
  - if $\beta = E\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if $\beta = @_i\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if β = $\langle r\rangle\beta_1$, then the check of β takes $O(n + m)$;
  - if β = $\langle r\rangle^-\beta_1$, then the check of β takes $O(n + m)$

Contamos. . .

- Sea $n = |W|$, $m = |R|$, y k el tamaño de α.
- Notar que $|sub(\alpha)| = O(k)$. Ademas, chequear y cambiar $L(\beta, w)$ es $O(1)$. Igual que $w \in V(p)$ si V es un bit table.
- Entonces, inicializar L toma $O(k \times n)$. El loop principal itera $O(k)$ veces. La complejidad de cada iteracion depende de β.
- In particular,
  - if $\beta \in ATOM$, then the check of β takes $O(n)$;
  - if $\beta = \beta_1 \wedge \beta_2$, then the check of β takes $O(n)$;
  - if $\beta = \neg\beta_1$, then the check of β takes $O(n)$;
  - if $\beta = E\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if $\beta = @_i\beta_1$, then the check of β takes $O(2n) = O(n)$;
  - if $\beta = \langle r\rangle\beta_1$, then the check of β takes $O(n + m)$;
  - if $\beta = \langle r\rangle^-\beta_1$, then the check of β takes $O(n + m)$
- Es decir, la complejidad de MCLite es: $O(k \times (n + m))$.

El problema con $\downarrow$

El problema con $\downarrow$

- Cuando agregamos $\downarrow$ no podemos usar una estrategia bottom-up.

El problema con $\downarrow$

- Cuando agregamos $\downarrow$ no podemos usar una estrategia bottom-up.
- Por que? Consideremos las formulas $\langle r\rangle\alpha$ y $\downarrow x.\langle r\rangle\beta(x)$, donde $\beta(x)$ es una formula donde aparece la variable x.

El problema con $\downarrow$

- Cuando agregamos $\downarrow$ no podemos usar una estrategia bottom-up.
- Por que? Consideremos las formulas $\langle r\rangle\alpha$ y $\downarrow x.\langle r\rangle\beta(x)$, donde $\beta(x)$ es una formula donde aparece la variable x.
  - En el primer caso, podemos chequear α, etiquetar el modelo, y luego chequear $\langle r\rangle\alpha$ como hicimos en MCLite.

El problema con $\downarrow$

- Cuando agregamos $\downarrow$ no podemos usar una estrategia bottom-up.
- Por que? Consideremos las formulas $\langle r\rangle\alpha$ y $\downarrow x.\langle r\rangle\beta(x)$, donde $\beta(x)$ es una formula donde aparece la variable x.
  - En el primer caso, podemos chequear α, etiquetar el modelo, y luego chequear $\langle r\rangle\alpha$ como hicimos en MCLite.
  - En el segundo caso, no podemos chequear primero $\beta(x)$, porque no sabemos a qué elemento esta linqueado x.

MCFull(M, g, α)

MCFull(M, g, α)

1: for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$

MCFull(M, g, α)

1: for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
2: case of α**:**
3: • $\alpha \in ATOM$ **:** $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$

case of α**:**

- $\alpha \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
case of α**:**
• $\alpha \in ATOM$ **:** $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
• $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
• $\alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1); **MCFull**(M, g, α_2)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$

MCFull(M, g, α)

for $\beta \in sub(\alpha), w \in W$ **do:** $L(\beta, w) \leftarrow 0$
case of α**:**
  • $\alpha \in ATOM$ **:** $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
  • $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
  • $\alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1)**; MCFull**(M, g, α_2)
    for $w \in W$ **do:**
      if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
  • $\alpha = \neg\alpha_1$ **: MCFull**(M, g, α_1)
    for $w \in W$ **do:**
      if $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$

case of α**:**

- $\alpha \in ATOM$ **:** $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
- $\alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1)**; MCFull**(M, g, α_2)

  for $w \in W$ **do:**

  if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \neg\alpha_1$ **: MCFull**(M, g, α_1)

  for $w \in W$ **do:**

  if $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \langle r \rangle \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle}(M, \alpha_1)$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
case of α**:**
$\bullet\ \alpha \in ATOM$ **:** $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
$\bullet\ \alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
$\bullet\ \alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1)**; MCFull**(M, g, α_2)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
$\bullet\ \alpha = \neg\alpha_1$ **: MCFull**(M, g, α_1)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$
$\bullet\ \alpha = \langle r \rangle \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle}(M, \alpha_1)$
$\bullet\ \alpha = \langle r \rangle^- \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle^-}(M, \alpha_1)$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
case of α**:**
- $\alpha \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
- $\alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1)**; MCFull**(M, g, α_2)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \neg\alpha_1$ **: MCFull**(M, g, α_1)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \langle r \rangle \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle}(M, \alpha_1)$
- $\alpha = \langle r \rangle^{-} \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle^{-}}(M, \alpha_1)$
- $\alpha = @_t \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{@}(M, t, g, \alpha_1)$ //**almost**

MCFull(M, g, α)

for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
case of α**:**
- $\alpha \in ATOM$: $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha \in VAR$: $L(\alpha, g(\alpha)) \leftarrow 1$
- $\alpha = \alpha_1 \wedge \alpha_2$: **MCFull**(M, g, α_1); **MCFull**(M, g, α_2)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \neg\alpha_1$: **MCFull**(M, g, α_1)
  for $w \in W$ **do:**
    if $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \langle r \rangle \alpha_1$: **MCFull**(M, g, α_1); **MC**$_{\langle r \rangle}(M, \alpha_1)$
- $\alpha = \langle r \rangle^{-} \alpha_1$: **MCFull**(M, g, α_1); **MC**$_{\langle r \rangle^{-}}(M, \alpha_1)$
- $\alpha = @_t \alpha_1$: **MCFull**(M, g, α_1); **MC**$_{@}(M, t, g, \alpha_1)$ //**almost**
- $\alpha = E\alpha_1$: **MCFull**(M, g, α_1); **MC**$_E(M, \alpha_1)$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$**,** $w \in W$ **do:** $L(\beta, w) \leftarrow 0$

case of α**:**

- $\alpha \in ATOM$ **:** $\forall w \in W$**, if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
- $\alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1)**; MCFull**(M, g, α_2)
  - **for** $w \in W$ **do:**
    - **if** $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \neg\alpha_1$ **: MCFull**(M, g, α_1)
  - **for** $w \in W$ **do:**
    - **if** $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$
- $\alpha = \langle r \rangle \alpha_1$ **: MCFull**(M, g, α_1)**;** $\mathbf{MC}_{\langle r \rangle}(M, \alpha_1)$
- $\alpha = \langle r \rangle^{-} \alpha_1$ **: MCFull**(M, g, α_1)**;** $\mathbf{MC}_{\langle r \rangle^-}(M, \alpha_1)$
- $\alpha = @_t \alpha_1$ **: MCFull**(M, g, α_1)**;** $\mathbf{MC}_{@}(M, t, g, \alpha_1)$ **//almost**
- $\alpha = E\alpha_1$ **: MCFull**(M, g, α_1)**;** $\mathbf{MC}_{E}(M, \alpha_1)$
- $\alpha = \downarrow x.\alpha_1$ **:** $\mathbf{Check}_{\downarrow}(M, g, x, \alpha_1)$

MCFull(M, g, α)

for $\beta \in sub(\alpha)$, $w \in W$ **do:** $L(\beta, w) \leftarrow 0$
case of α**:**
  • $\alpha \in ATOM$ **:** $\forall w \in W$, **if** $w \in V(\alpha)$ **then** $L(\alpha, w) \leftarrow 1$
  • $\alpha \in VAR$ **:** $L(\alpha, g(\alpha)) \leftarrow 1$
  • $\alpha = \alpha_1 \wedge \alpha_2$ **: MCFull**(M, g, α_1)**; MCFull**(M, g, α_2)
    for $w \in W$ **do:**
      if $L(\alpha_1, w) = 1 = L(\alpha_2, w) = 1$ **then** $L(\alpha, w) \leftarrow 1$
  • $\alpha = \neg\alpha_1$ **: MCFull**(M, g, α_1)
    for $w \in W$ **do:**
      if $L(\alpha_1, w) = 0$ **then** $L(\alpha, w) \leftarrow 1$
  • $\alpha = \langle r \rangle \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle}(M, \alpha_1)$
  • $\alpha = \langle r \rangle^{-} \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{\langle r \rangle^{-}}(M, \alpha_1)$
  • $\alpha = @_t \alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{@}(M, t, g, \alpha_1)$ **//almost**
  • $\alpha = E\alpha_1$ **: MCFull**(M, g, α_1)**; MC**$_{E}(M, \alpha_1)$
  • $\alpha = \downarrow x.\alpha_1$ **: Check**$_{\downarrow}(M, g, x, \alpha_1)$
return $L(\alpha)$

$\text{Check}_{\downarrow}(M, g, x, \alpha)$

$\text{Check}_{\downarrow}(M, g, x, \alpha)$

for $w \in W$ **do:**
  $g(x) \leftarrow w$
  MCFull(M, g, α)
  if $L(\alpha, w) = 1$ **then**
    Clear(L, x)
    $L(\downarrow x.\alpha, w) \leftarrow 1$
  else
    Clear(L, x)

$\text{Check}_{\downarrow}(M, g, x, \alpha)$

for $w \in W$ **do:**
    $g(x) \leftarrow w$
    MCFull(M, g, α)
    if $L(\alpha, w) = 1$ **then**
        Clear(L, x)
        $L(\downarrow x.\alpha, w) \leftarrow 1$
    else
        Clear(L, x)

La funcion Clear(L, x) pone a 0 el valor de L(α, w) para todo $w \in W$ y toda formula α con x libre.

Complejidad de MCFull

Complejidad de MCFull

- Supongamos que α no contiene el operador $\downarrow$. Sea $\alpha = \tau\alpha_1$, donde τ es el operador principal de α.

Complejidad de MCFull

- Supongamos que α no contiene el operador $\downarrow$. Sea $\alpha = \tau\alpha_1$, donde τ es el operador principal de α.
- Sea $C(\alpha)$ el costo de MCFull en α y sea $C_\tau(\alpha)$ el costo de MC_τ en α. Entonces,

$$C(\tau\alpha_1) = C(\alpha_1) + C_\tau(\alpha)$$

Complejidad de MCFull

- Supongamos que α no contiene el operador $\downarrow$. Sea $\alpha = \tau\alpha_1$, donde τ es el operador principal de α.
- Sea $C(\alpha)$ el costo de MCFull en α y sea $C_\tau(\alpha)$ el costo de MC_τ en α. Entonces,

$$C(\tau\alpha_1) = C(\alpha_1) + C_\tau(\alpha)$$

- Por lo que, en el peor caso, el costo de MCFull en α es $O(k \times (n + m))$ como para MCLite.

Complejidad de MCFull

- Supongamos que α no contiene el operador $\downarrow$. Sea $\alpha = \tau\alpha_1$, donde τ es el operador principal de α.
- Sea $C(\alpha)$ el costo de MCFull en α y sea $C_\tau(\alpha)$ el costo de MC_τ en α. Entonces,

$$C(\tau\alpha_1) = C(\alpha_1) + C_\tau(\alpha)$$

- Por lo que, en el peor caso, el costo de MCFull en α es $O(k \times (n + m))$ como para MCLite.
- Por ejemplo, si $\alpha = \langle r\rangle\langle r\rangle\langle r\rangle p$, entonces

$$C(\alpha) = C(\langle r\rangle\langle r\rangle p) + (n+m) =$$
$$C(\langle r\rangle p) + 2(n+m) =$$
$$C(p) + 3(n+m) = n + 3(n+m)$$

Complejidad de MCFull

Complejidad de MCFull

- Supongamos ahora que α contiene $\downarrow$.

Complejidad de MCFull

- Supongamos ahora que α contiene $\downarrow$.
- Sea d el grado de anidamiento de $\downarrow$ en α. Por ejemplo $\downarrow x.\langle r\rangle x$ tiene grado 1, y $\downarrow x.\langle r\rangle \downarrow y.@_x y$ tiene grado 2.

Complejidad de MCFull

- Supongamos ahora que α contiene $\downarrow$.
- Sea d el grado de anidamiento de $\downarrow$ en α. Por ejemplo $\downarrow x.\langle r\rangle x$ tiene grado 1, y $\downarrow x.\langle r\rangle\downarrow y.@_x y$ tiene grado 2.
- La función $\text{Check}_{\downarrow}(M, g, x, \beta)$ corre en $n \times C(\beta)$.

Complejidad de MCFull

- Supongamos ahora que α contiene $\downarrow$.
- Sea d el grado de anidamiento de $\downarrow$ en α. Por ejemplo $\downarrow x.\langle r \rangle x$ tiene grado 1, y $\downarrow x.\langle r \rangle \downarrow y.@_x y$ tiene grado 2.
- La función $\mathrm{Check}_{\downarrow}(M, g, x, \beta)$ corre en $n \times C(\beta)$.
- Por lo que, en el peor caso, el costo de MCFull en α es $O(k \times (n + m) \times n^d)$.

Complejidad de MCFull

- Supongamos ahora que α contiene $\downarrow$.
- Sea d el grado de anidamiento de $\downarrow$ en α. Por ejemplo $\downarrow x.\langle r\rangle x$ tiene grado 1, y $\downarrow x.\langle r\rangle\downarrow y.@_x y$ tiene grado 2.
- La función $\text{Check}_\downarrow(M, g, x, \beta)$ corre en $n \times C(\beta)$.
- Por lo que, en el peor caso, el costo de MCFull en α es $O(k \times (n+m) \times n^d)$.
- Por ejemplo, si $\alpha = \downarrow x.\langle r\rangle\downarrow y.@_x y$, entonces

$$
\begin{gathered}
C(\alpha) = n \times C(\langle r\rangle\downarrow y.@_x y) = \\
n \times (n + m + C(\downarrow y.@_x y)) = \\
n \times (n + m + n \times C(@_x y)) = \\
n \times (n + m + n \times (n + C(y))) = \\
n \times (n + m + n \times (n + 1)) = O(n^3)
\end{gathered}
$$

Complejidad de MCFull

Complejidad de MCFull

- Es decir, la complejidad temporal de MCFull es exponential en el grado de anidammiento de $\downarrow$ que, en general, es proporcional al tamaño de α.

Complejidad de MCFull

- Es decir, la complejidad temporal de MCFull es exponential en el grado de anidammiento de $\downarrow$ que, en general, es proporcional al tamaño de α.
- Como el tamaño del stack de recursion de MCFull está limitado por el tamaño de α, la complejidad espacial de MCFull es polynomial.

Complejidad de MCFull

- Es decir, la complejidad temporal de MCFull es exponential en el grado de anidammiento de $\downarrow$ que, en general, es proporcional al tamaño de α.
- Como el tamaño del stack de recursion de MCFull está limitado por el tamaño de α, la complejidad espacial de MCFull es polynomial.
- Es decir, model checking para $\mathcal{L}(\text{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$ esté en PSPACE.

Complejidad de MCFull

- Es decir, la complejidad temporal de MCFull es exponential en el grado de anidammiento de $\downarrow$ que, en general, es proporcional al tamaño de α.
- Como el tamaño del stack de recursion de MCFull está limitado por el tamaño de α, la complejidad espacial de MCFull es polynomial.
- Es decir, model checking para $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$ esté en PSPACE.
- Nos podemos preguntar: es este el mejor algoritmo?

Lower bounds

Lower bounds

- Sabemos que $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$ tiene la misma expresividad que LPO.

Lower bounds

- Sabemos que $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$ tiene la misma expresividad que LPO.
- Es conocido que el problema de model checking para LPO es PSPACE-complete, por lo tanto también para $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$.
  Pero podemos demostrar un resultado mas fuerte.

Lower bounds

- Sabemos que $\mathcal{L}(\text{NOM}, \langle r \rangle, \langle r \rangle^-, @, E, \downarrow)$ tiene la misma expresividad que LPO.
- Es conocido que el problema de model checking para LPO es PSPACE-complete, por lo tanto también para $\mathcal{L}(\text{NOM}, \langle r \rangle, \langle r \rangle^-, @, E, \downarrow)$.
  Pero podemos demostrar un resultado mas fuerte.
- Llamemos, fragmento no decorado de L al fragmento de L que no usa ni símbolos de proposición ni nominales

Lower bounds

- Sabemos que $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$ tiene la misma expresividad que LPO.
- Es conocido que el problema de model checking para LPO es PSPACE-complete, por lo tanto también para $\mathcal{L}(\mathrm{NOM}, \langle r \rangle, \langle r \rangle^{-}, @, E, \downarrow)$.
  Pero podemos demostrar un resultado mas fuerte.
- Llamemos, fragmento no decorado de L al fragmento de L que no usa ni símbolos de proposición ni nominales
  Theorem: El problema de model checking para el fragmento no decorado de $\mathcal{L}(\downarrow)$ es PSPACE-complete.

El model checker `mcheck`

- Es un model checker de juguete.
- Warning: no intentar usarlo para nada serio.
- Si estan interesados en model checkers para lógicas hibridas hay otras alternativas pero el input format de `mcheck` es mas simple.