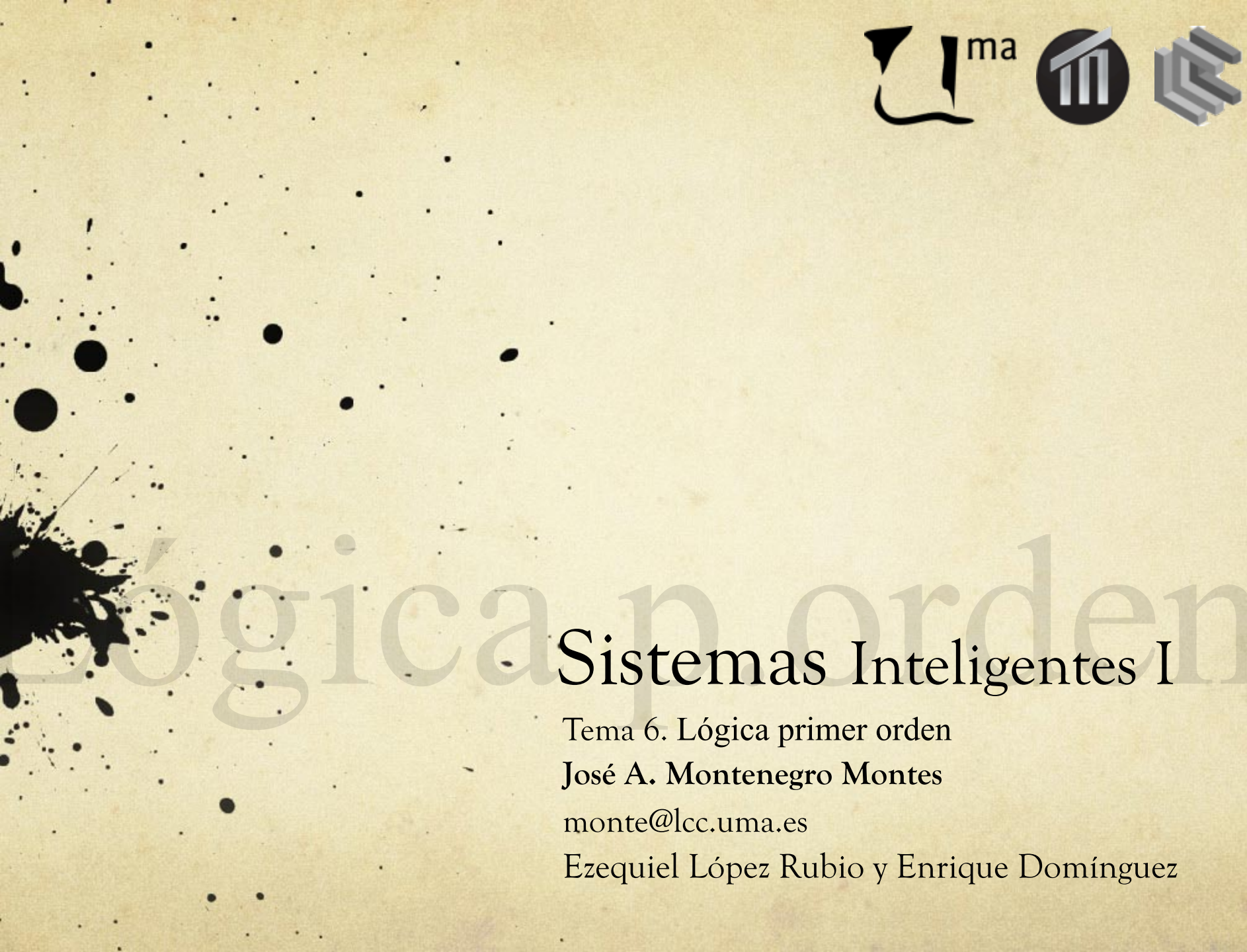


Sistemas Inteligentes I

Tema 6. Lógica primer orden

José A. Montenegro Montes

monte@lcc.uma.es

Ezequiel López Rubio y Enrique Domínguez

Resumen

- Introducción
- Sintaxis y Semántica
- Utilización
- Demostración de Teoremas
- Conclusiones

Introducción

Motivación

- El 14 de septiembre de 2004, los controladores aéreos del aeropuerto internacional de Los Ángeles perdieron la comunicación por voz con los 400 aviones que estaban supervisando
  - El sistema de respaldo falló menos de un minuto después de haber sido activado
  - Dos accidentes aéreos estuvieron a punto de ocurrir
- El ordenador de control aéreo necesitaba ser reiniciado cada 50 días debido a un error del software
  - El técnico no reinició la máquina

Motivación

- El 14 de agosto de 2003, algunas partes del noreste de Estados Unidos y el sureste de Canadá sufrieron apagones generalizados
  - Más de 50 millones de personas se vieron afectadas, y el coste total fue de 13.000 millones de dólares
  - Un error del software provocó que el servidor de control se colgara
  - El servidor de respaldo también se colgó, dado que estaba ejecutando el mismo programa

Generalidades

- La lógica proposicional es demasiado sencilla para representar el conocimiento en entornos complejos
- La lógica de primer orden toma prestadas ideas de los lenguajes naturales a la vez que evita sus ambigüedades
  - Se construye sobre los objetos y las relaciones entre ellos
  - Supone que dichas relaciones o se cumplen o no se cumplen entre los objetos

Sintaxis y Semántica

Modelos

Modelos

- Como en la lógica proposicional, un modelo es un mundo posible que podría considerarse.
- Un modelo de la lógica de primer orden tiene los siguientes componentes:
  - Un dominio, que es el conjunto de objetos o elementos del dominio que contiene.
  - Un conjunto de relaciones entre objetos. Una relación es un conjunto de tuplas de objetos que están relacionados.
  - Un conjunto de funciones, que deben ser totales. Se añade un objeto adicional "invisible" para convertir las funciones parciales en totales.

Ejemplo de modelo

- **Objetos**: El rey Ricardo Corazón de León; su hermano menor, el malvado rey Juan; las piernas izquierdas de Ricardo y Juan; y una corona (en total 5 objetos)
- **Relaciones**:
  - Hermandad={<Ricardo Corazón de León, Rey Juan>, <Rey Juan, Ricardo Corazón de León>}
  - Sobre la cabeza={<La corona, Rey Juan>}
- **Funciones**:
  - Pierna izquierda={ <Ricardo Corazón de León>→La pierna izquierda de Ricardo, <Rey Juan>→La pierna izquierda de Juan }

Logica Primer

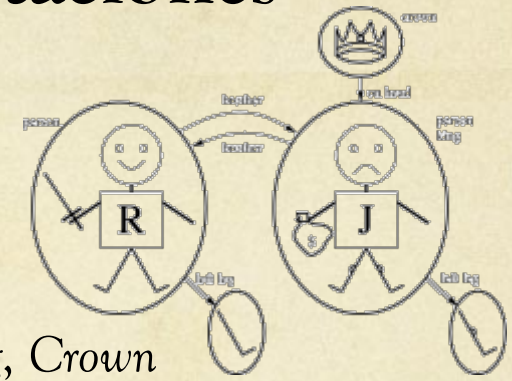
Sintaxis y Semántica

Símbolos e Interpretaciones

Símbolos e interpretaciones

- Hay tres tipos de símbolos:
  - Símbolos de constante, que representan objetos
  - Símbolos de predicado, que representan relaciones
  - Símbolos de función, que representan funciones
- Todos los símbolos empiezan por mayúscula
- Cada símbolo de predicado y de función tiene su aridad que fija su número de argumentos
- Cada modelo incluye una interpretación que dice qué objetos, relaciones y funciones se corresponden con los símbolos de constante, predicado y función

Ejemplo de símbolos e interpretaciones



- **Símbolos de constante**: *Richard*, *John*
- **Símbolos de predicado**: *Brother*, *OnHead*, *Person*, *King*, *Crown*
- **Símbolo de función**: *LeftLeg*
- Una **posible interpretación** (entre otras muchas) :
  - *Richard* se refiere a Ricardo Corazón de León y *John* se refiere al malvado rey Juan
  - *Brother* se refiere a la relación de hermandad; *OnHead* se refiere a la relación "sobre la cabeza"; *Person*, *King* y *Crown* se refieren a los conjuntos de objetos que son personas, reyes y coronas, respectivamente
  - *LeftLeg* se refiere a la función "pierna izquierda"

Sintaxis y Semántica

Términos, Fórmulas Atómicas y Compuestas

Términos

- Un término es una expresión lógica que hace referencia a un objeto
  - Los símbolos de constante son términos
  - Los términos compuestos se forman mediante un símbolo de función seguido de una lista de términos que hacen de argumentos
- Ejemplos de términos:
  - *Richard*,
  - *LeftLeg*(*John*),
  - *LeftLeg*(*LeftLeg*(*Richard*))

Fórmulas atómicas

- Una fórmula atómica (también llamada átomo) es un símbolo de predicado seguido de una lista de términos que hacen de argumentos
- Una sentencia atómica es verdadera en un determinado modelo si la relación a la que se refiere el símbolo de predicado se cumple entre los objetos a los que se refieren los argumentos
- Ejemplos de átomos:
  - *Brother*(*Richard*,*John*),
  - *Person*(*LeftLeg*(*Richard*))

Fórmulas compuestas

- Conectivas lógicas forman fórmulas compuestas a partir de los átomos, con la misma sintaxis y semántica que en la lógica proposicional
- Ejemplos de fórmulas compuestas:
  - $\neg Brother(LeftLeg(Richard), John)$
  - $Brother(Richard, John) \wedge Brother(John, Richard)$
  - $King(Richard) \vee King(John)$
  - $\neg King(Richard) \Rightarrow King(John)$
    - (todas ellas son verdaderas en nuestro ejemplo de modelo bajo la interpretación considerada anteriormente)

Logica Primer

Sintaxis y Semántica

Cuantificadores

17

Cuantificadores

- Los cuantificadores nos permiten expresar propiedades de colecciones de objetos
  - El cuantificador universal ∀ se lee "para todo".
    - Va seguido de una o más variables en minúsculas.
    - La fórmula ∀x P quiere decir que P es verdadero para todo objeto x
  - El cuantificador existencial ∃ se lee "existe".
    - Va seguido de una o más variables en minúsculas.
    - La fórmula ∃x P quiere decir que P es verdadero para al menos un objeto x

Ejemplos de uso de cuantificadores

- *"Todos los reyes son personas"*
  - $\forall x\ King(x) \Rightarrow Person(x)$ *(correcto)*
  - $\forall x\ King(x) \wedge Person(x)$ (error)
    - *"Todos los objetos son reyes y personas"*
- *"El rey Juan tiene una corona sobre su cabeza"*
  - $\exists x\ Crown(x) \wedge OnHead(x,John)$ *(correcto)*
  - $\exists x\ Crown(x) \Rightarrow OnHead(x,John)$ (error)
    - *"Existe un objeto que o no es una corona o está sobre la cabeza del rey Juan"*

Más acerca de los cuantificadores

- Los cuantificadores se pueden anidar.
- Si combinamos existenciales con universales, el orden es muy importante:
  - "Todo el mundo ama a alguien": $\forall x\ \exists y\ Loves(x,y)$
  - "Existe alguien que es amado por todo el mundo": $\exists y\ \forall x\ Loves(x,y)$
- Reglas de De Morgan para fórmulas cuantificadas:
  - $\forall x\ P \equiv \neg\ \exists x\ \neg P$
  - $\exists x\ P \equiv \neg\ \forall x\ \neg P$

Igualdad

- Podemos usar el símbolo de igualdad = para indicar que dos términos se refieren al mismo objeto

  - Si queremos expresar que el objeto al que se refiere *Father*(*John*) y el objeto al que se refiere *Henry* son el mismo, escribimos:

    $$Father(John)=Henry$$

  - "Ricardo tiene al menos dos hermanos":

    $$\exists x,y\ Brother(x,Richard) \wedge Brother(y,Richard) \wedge \neg(x=y)$$

Combinación cuantificadores

Usa el predicado Loves, donde Loves(x,y) quiere decir "x ama a y".

a) "Hay alguien que ama a todo el mundo".

b) "Hay alguien que ama a al menos una persona".

c) "Hay alguien que ama a algún otro".

d) "Todos se aman mutuamente".

e) "Hay alguien que es amado por todos".

f) "Hay alguien a quien todos aman".

g) "Todo el mundo tiene a alguien que lo ama".

Utilización

Asertos y Consultas

Asertos

- Para añadir fórmulas a una base de conocimiento (KB) usamos *TELL*.
  - Tales fórmulas se denominan asertos.
- Por ejemplo, podemos afirmar que Juan es un rey, Ricardo es una persona, y todos los reyes son personas:
  - $TELL(KB, King(John))$
  - $TELL(KB, Person(Richard))$
  - $TELL(KB, \forall x\ King(x) \Rightarrow Person(x))$

Consultas

- Podemos plantear preguntas a la KB mediante *ASK*
- Por ejemplo, las siguientes consultas devuelven *true*:
  - *ASK*(*KB*, *King*(*John*))
  - *ASK*(*KB*, *Person*(*John*))
- Si queremos los valores de una variable que hacen verdadera a una fórmula, empleamos *ASKVARS*
  - *ASKVARS*(*KB*, *Person*(*x*)) devuelve la siguiente sustitución o lista de ligaduras: {*x*/*John*}, {*x*/*Richard*}

Axiomas y Teoremas

Ejemplo: el dominio del parentesco

Ejemplo: el dominio del parentesco (I)

- Los objetos del dominio son personas
- Dos predicados unarios: *Male*, *Female*
- Las relaciones de parentesco se representan mediante predicados binarios: *Parent*, *Sibling*, *Brother*, *Sister*, *Child*, *Daughter*, *Son*, *Spouse*, *Wife*, *Husband*, *Grandparent*, *Grandchild*, *Cousin*, *Aunt*, *Uncle*
- Para simplificar supondremos que cada persona tiene exactamente una madre y un padre, lo cual nos permite usar las funciones *Mother* y *Father* en lugar de predicados

Ejemplo: el dominio del parentesco (II)

- Ahora podemos poner por escrito nuestro conocimiento del dominio:
  - $\forall m,c\ \mathit{Mother}(c)=m \Leftrightarrow \mathit{Female}(m) \wedge \mathit{Parent}(m,c)$
  - $\forall p,c\ \mathit{Parent}(p,c) \Leftrightarrow \mathit{Child}(c,p)$
  - $\forall g,c\ \mathit{GrandParent}(g,c) \Leftrightarrow \exists p\ \mathit{Parent}(g,p) \wedge \mathit{Parent}(p,c)$
  - $\forall x,y\ \mathit{Sibling}(x,y) \Leftrightarrow x \neq y \wedge \exists p\ \mathit{Parent}(p,x) \wedge \mathit{Parent}(p,y)$
  - ...
- Estas fórmulas se llaman axiomas, ya que son las reglas básicas a partir de las cuales se pueden derivar conclusiones útiles

Ejemplo: el dominio del parentesco (III)

- Los **teoremas** son fórmulas que se infieren a partir de los axiomas.
  - Por ejemplo, si preguntamos con *ASK* a la KB la siguiente fórmula, nos responderá *true*, lo cual quiere decir que es un teorema:

$$\forall x,y \; Sibling(x,y) \Leftrightarrow Sibling(y,x)$$

- Podemos tener axiomas que sean "hechos simples".
  - Describen una instancia particular del problema, de tal manera que se puedan contestar preguntas específicas. Por ejemplo:

$$Male(Jim), \; Spouse(Jim,Laura)$$

Demostración de Teoremas

Forma normal conjuntiva

Forma normal conjuntiva

- La resolución se puede extender a la lógica de primer orden
- Como en el caso proposicional, la resolución de primer orden exige que las fórmulas estén en **forma normal conjuntiva (CNF)**
- Una fórmula está en CNF si es una disyunción de literales. Los literales pueden contener variables, que se supone que están cuantificadas universalmente
- Toda fórmula de la lógica de primer orden puede convertirse en una fórmula en CNF inferencialmente equivalente, es decir, una fórmula que es insatisfacible sii la fórmula original es insatisfacible

Conversión a CNF (I)

- Eliminar $\Leftrightarrow$ reemplazando $\alpha \Leftrightarrow \beta$ por $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$
- Eliminar $\Rightarrow$ reemplazando $\alpha \Rightarrow \beta$ por $\neg\alpha \vee \beta$
- Mover $\neg$ hacia dentro aplicando repetidamente:
  - $\neg(\neg\alpha) \equiv \alpha$
  - $\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$
  - $\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$
  - $\neg\forall x\ \alpha \equiv \exists x\ \neg\alpha$
  - $\neg\exists x\ \alpha \equiv \forall x\ \neg\alpha$
- **Estandarizar variables**: si se emplea el mismo nombre de variable en dos cuantificadores, hay que cambiar el nombre de una de las variables

Conversión a CNF (II)

- **Skolemizar**: para cada variable existencialmente cuantificada x ($\exists x$), eliminar el cuantificador existencial y sustituir x por $F(y_1,...,y_n)$, donde F es un símbolo de función nuevo que no se ha usado antes (función de Skolem),
  - Siendo $y_1,...,y_n$ las variables cuantificadas universalmente en cuyo ámbito aparece x

- *Ejemplos:*
  - $\exists x\ Rich(x) \rightarrow Rich(G1)$ *donde G1 es una cte Skolem*
  - $\forall x\ Person(x) \Rightarrow \exists y\ Heart(y) \wedge Has(x,y) \rightarrow \forall x\ Person(x) \Rightarrow Heart(H(x)) \wedge Has(x, H(x))$, *donde H es un nuevo símbolo (Función Skolem)*

Conversión a CNF (III)

- Eliminar los **cuantificadores** universales
- Aplicar la **distributividad** de ∨ respecto a ∧ siempre que sea posible:

$$\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$$

Ejemplo de conversión a CNF (I)

"Todo aquel que ama a todos los animales es amado por alguien"

- $\forall x\ [\forall y\ Animal(y) \Rightarrow Loves(x,y)] \Rightarrow [\exists y\ Loves(y,x)]$
- $\forall x\ [\neg \forall y\ \neg Animal(y) \vee Loves(x,y)] \vee [\exists y\ Loves(y,x)]$
- $\forall x\ [\exists y\ \neg(\neg Animal(y) \vee Loves(x,y))] \vee [\exists y\ Loves(y,x)]$
- $\forall x\ [\exists y\ \neg\ \neg Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y\ Loves(y,x)]$
- $\forall x\ [\exists y\ Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y\ Loves(y,x)]$
- (sigue)...

Ejemplo de conversión a CNF (II)

"Todo aquel que ama a todos los animales es amado por alguien"

- ...(ant) $\forall x\ [\exists y\ Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y\ Loves(y,x)]$
- $\forall x\ [\exists y\ Animal(y) \wedge \neg Loves(x,y)] \vee [\exists z\ Loves(z,x)]$
- $\forall x\ [Animal(F(x)) \wedge \neg Loves(x,\ F(x))] \vee Loves(G(x),x)$
  - *F y G son funciones de Skolem*
- $[Animal(F(x)) \wedge \neg Loves(x,\ F(x))] \vee Loves(G(x),x)$
- $[Animal(F(x)) \vee Loves(G(x),x)\ ] \wedge [\neg Loves(x,F(x)) \vee Loves(G(x),x)\ ]$

Demostración de Teoremas

Unificación

Unificación (I)

- El proceso de encontrar sustituciones que consigan que expresiones lógicas distintas parezcan idénticas se llama unificación
- El algoritmo **Unify** toma dos fórmulas y devuelve un unificador para ellas si es que lo hay:
  - Unify(p,q)=θ donde $SUBST(p,\theta)=SUBST(q,\theta)$
  - Unify(*Knows*(*John*,*x*), *Knows*(*John*,*Jane*))={*x*/*Jane*}
  - Unify(*Knows*(*John*,*x*), *Knows*(*y*,*Bill*))={*x*/*Bill*, *y*/*John*}
  - Unify(*Knows*(*John*,*x*), *Knows*(*x*,*Lisa*))=*fail*

Unificación (II)

- El algoritmo de unificación trabaja comparando las estructuras de las entradas, elemento a elemento
  - Las entradas x e y pueden ser variables, constantes o expresiones compuestas
- Si no se proporciona una sustitución inicial θ, se toma por defecto la sustitución vacía
- A medida que el algoritmo avanza se va modificando θ
- Hay que asegurarse de que las comparaciones que vamos haciendo son consistentes con las ligaduras ya establecidas

Algoritmo de unificación

```
function Unify(x,y,θ)
    if (θ=fail) ∨ (x=y) then return θ
    else if IsVariable(x) then return UnifyVar(x,y,θ)
    else if IsVariable(y) then return UnifyVar(y,x,θ)
    else if IsCompound(x) ∧ IsCompound(y) then
        return Unify(x.Args,y.Args,Unify(x.Op, y.Op,θ))
    else if IsList(x) ∧ IsList(y) then
        return Unify(x.Rest,y.Rest,Unify(x.First, y.First,θ))
else return fail
```

Algoritmo de unificación de variables

function UnifyVar(*var*,*x*,θ)

if $\{var/val\}\in\theta$ **then return** Unify(*val*,*x*,θ)

else if $\{x/val\}\in\theta$ **then return** Unify(*var*,*val*,θ)

else if Occurs(*var*,*x*) **then return** *fail*

else return añadir $\{var/x\}$ a θ

La regla de inferencia de resolución

- A continuación extendemos el procedimiento de resolución para la lógica proposicional a la lógica de primer orden
- Dadas dos cláusulas que no comparten variables, se pueden resolver si contienen literales complementarios
- Dos literales de primer orden l_i y m_j son complementarios si uno unifica con la negación del otro, Unify(l_i,$\neg m_j$)=θ
- Después de aplicar resolución, eliminamos las copias extra de los literales redundantes, es decir, aquellos que son unificables. El unificador debe aplicarse a la cláusula entera

$$\frac{l_1 \vee \ldots \vee l_k, \quad m_1 \vee \ldots \vee m_n}{SUBST(\theta, l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n)}$$

Demostración de Teoremas

Ejemplos

Ejemplo de demostración por resolución (I)

KB:

$\forall x \; P(x) \Rightarrow Q(x)$

$\forall x \; \neg P(x) \Rightarrow R(x)$

$\forall x \; Q(x) \Rightarrow S(x)$

$\forall x \; R(x) \Rightarrow S(x)$

- Probar:

  $S(A)$

- Pasos:

1) convertir KB a CNF

2) Resolver hasta obtener S(x), el cual puede ser utilizado para obtener S(A)

3) Negar queremos probar, $\neg S(A)$

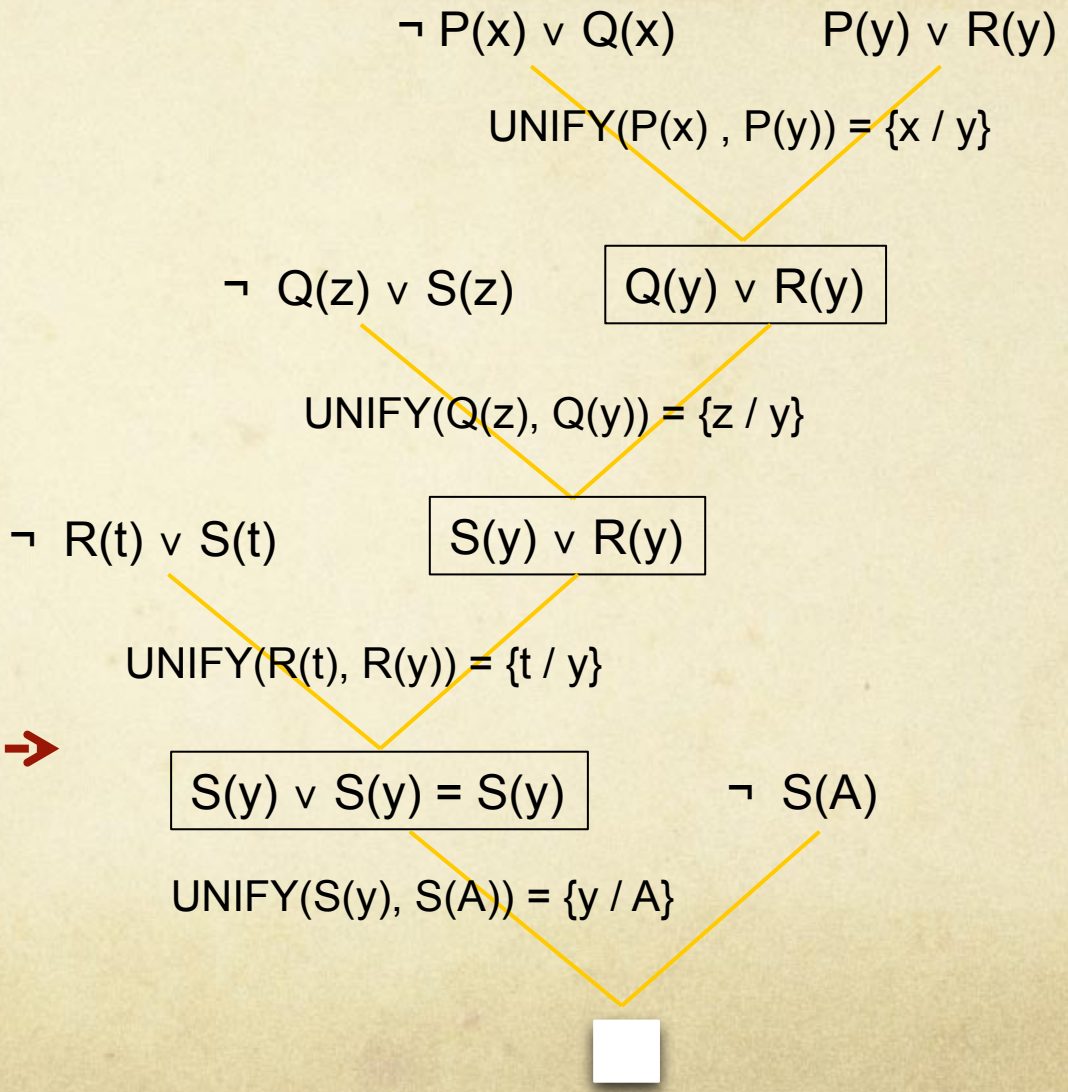
4) Resolver hasta una nueva clausula vacía

Ejemplo de demostración por resolución (I)

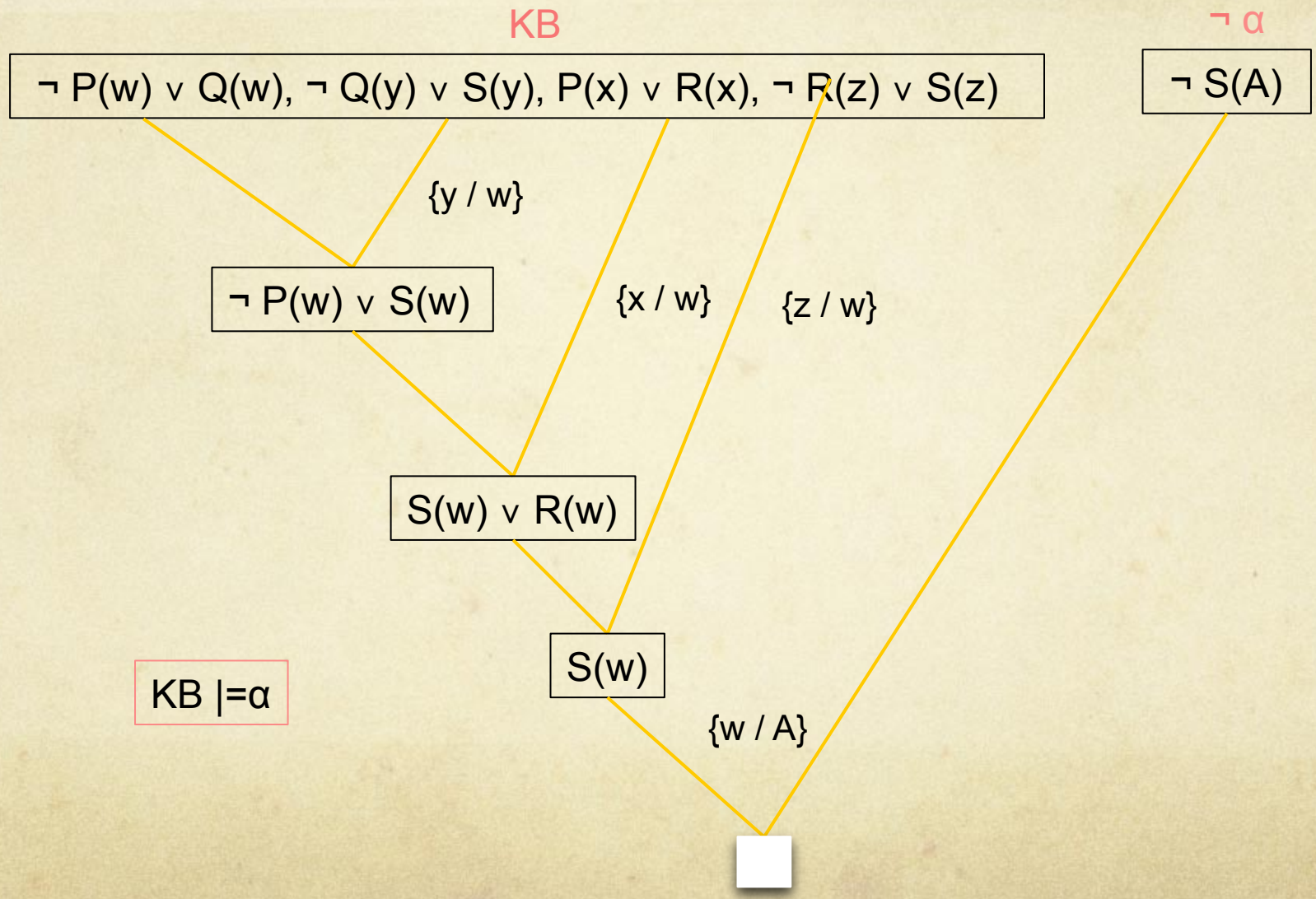
KB en forma CNF

- ¬ P(x) ∨ Q(x)
- P(y) ∨ R(y)
- ¬ Q(z) ∨ S(z)
- ¬ R(t) ∨ S(t)
- ¬ S(A)

Pasos en la Resolución



Ejemplo de demostración por resolución 2 (I)



Ejemplo de demostración por resolución 3 (I)

- El planteamiento del problema es como sigue:
  - Todo aquel que ama a todos los animales es amado por alguien
  - Cualquiera que mate un animal no es amado por nadie
  - Jack ama a todos los animales
  - Jack o Curiosity mataron al gato, que se llama Tuna
  - ¿Mató Curiosity al gato?

Ejemplo de demostración por resolución 3 (II)

- En primer lugar expresamos las frases originales, algo de conocimiento adicional y el objetivo negado G en lógica de primer orden:

A. $\forall x\ [\forall y\ Animal(y) \Rightarrow Loves(x,y)] \Rightarrow [\exists y\ Loves(y,x)]$

B. $\forall x\ [\exists z\ Animal(z) \wedge Kills(x,z)] \Rightarrow [\forall y\ \neg Loves(y,x)]$

C. $\forall x\ Animal(x) \Rightarrow Loves(Jack,x)$

D. $Kills(Jack,Tuna) \vee Kills(Curiosity,Tuna)$

E. $Cat(Tuna)$

F. $\forall x\ Cat(x) \Rightarrow Animal(x)$

¬G. $\neg Kills(Curiosity,Tuna)$

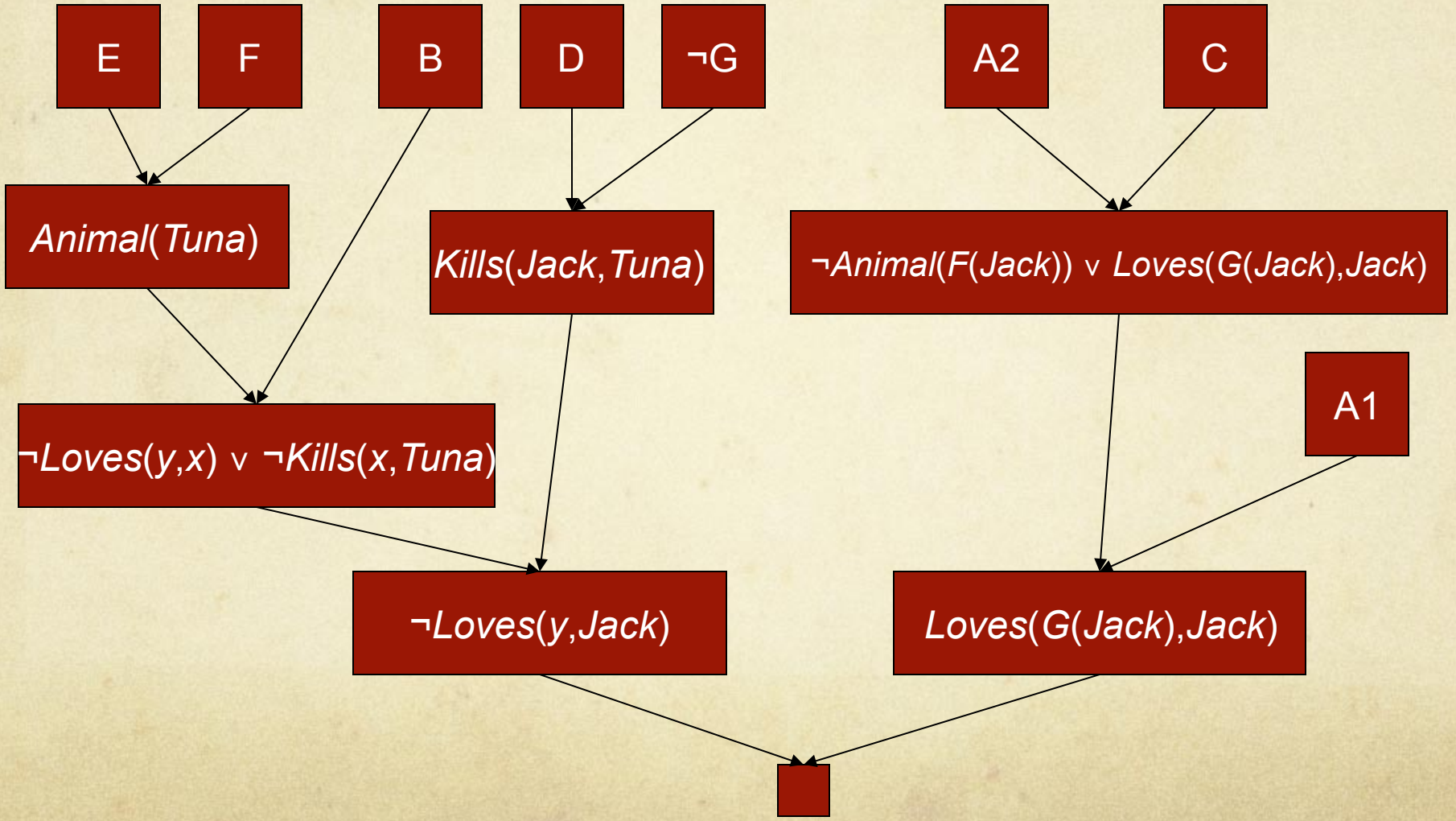
Ejemplo de demostración por resolución 3 (III)

- A continuación convertimos a CNF:
  - A1. $Animal(F(x)) \vee Loves(G(x),x)$
  - A2. $\neg Loves(x,F(x)) \vee Loves(G(x),x)$
  - B. $\neg Loves(y,x) \vee \neg Animal(z) \vee \neg Kills(x,z)$
  - C. $\neg Animal(x) \vee Loves(Jack,x)$
  - D. $Kills(Jack,Tuna) \vee Kills(Curiosity,Tuna)$
  - E. $Cat(Tuna)$
  - F. $\neg Cat(x) \vee Animal(x)$
  - ¬G. $\neg Kills(Curiosity,Tuna)$

Ejemplo de demostración por resolución 3 (IV)

- En la siguiente transparencia se muestra una demostración por resolución de que Curiosity mató al gato
- Nótese que hemos eliminado copias extra de literales redundantes para obtener *Loves*(*G*(*Jack*),*Jack*)
- Asimismo, cuando se resuelven las cláusulas A2 y C, es necesario estandarizar variables antes de la unificación
- Si quisiéramos preguntar "¿Quién mató al gato?", estableceríamos como objetivo $\exists w$ *Kills*(*w*,*Tuna*)
  - Obtendríamos un árbol de demostración parecido, pero con la sustitución {*w*/*Curiosity*}
  - Por desgracia, esta técnica no siempre funciona

Ejemplo de demostración por resolución 3 (V)



Ejemplo de demostración por resolución 3 (VI)

1. $Animal(Tuna)$ *(E y F)*
2. $\neg Loves(y,x) \vee \neg Kills(x,Tuna)$ *(B,1)*
3. $Kills(Jack,Tuna)$ *(D, ¬G)*
4. $\neg Loves(y,Jack)$ *(2,3)*
5. $\neg Animal(F(Jack)) \vee Loves(G(Jack),Jack)$ *(A2,C)*
6. $Loves(G(Jack),Jack)$ *(A1)*
7. [] *(6,4)*

Conclusión

Sumario

- Mientras la lógica proposicional sólo considera la existencia de hechos, la lógica de primer orden considera la existencia de objetos y relaciones
- Un mundo posible, o modelo, de la lógica de primer orden incluye un conjunto de objetos y una interpretación que asigna un significado a los símbolos de constante, predicado y función
- La regla de resolución generalizada proporciona un sistema de demostración completo para la lógica de primer orden, empleando bases de conocimiento en forma normal conjuntiva

Epílogo

- Se está empleando la lógica de primer orden para certificar la seguridad de software crítico
- Las demostraciones encontradas por demostradores automáticos de teoremas garantizan que un programa satisface ciertas propiedades de corrección
- Todavía queda mucho trabajo por hacer

Sistemas Inteligentes

José A. Montenegro Montes

monte@lcc.uma.es