Curso de Inteligencia Artificial:

Representación del Conocimiento

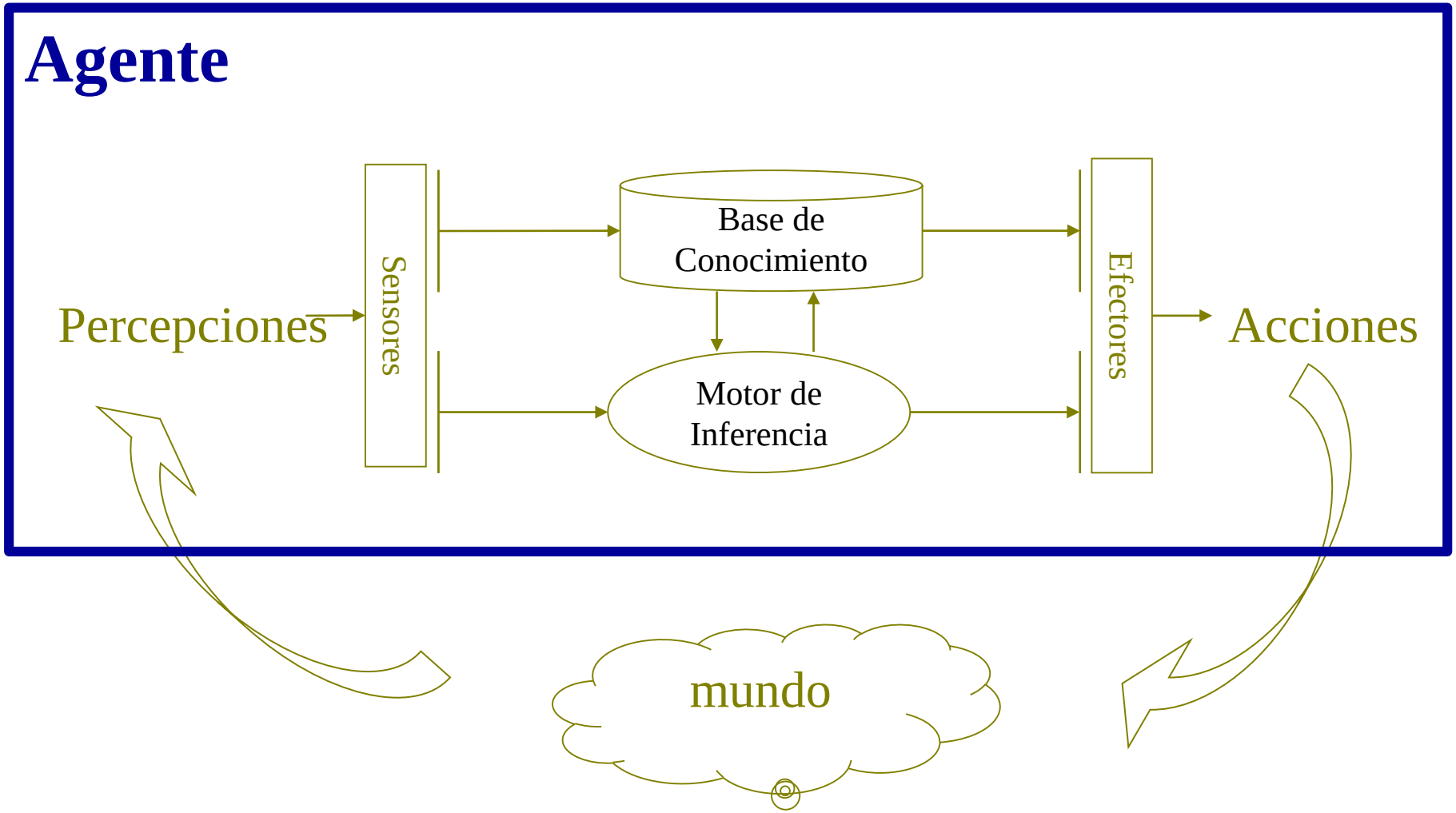
Jose Aguilar
Cemisid, Facultad de Ingeniería
Universidad de los Andes
Mérida, Venezuela
aguilar@ula.ve

Agentes Basado en Conocimiento

- Un Agente Basado en Conocimiento (ABC) posee **conocimiento** de su mundo, y es **capaz de razonar** sobre las posibles acciones que puede tomar para cambiar su mundo.

- El agente posee un conjunto de **sentencias**, representadas mediante un **lenguaje de representación de conocimiento.**

Agentes Basado en Conocimiento



Elementos de un ABC

- Lenguaje de representación de conocimiento.
  - Lenguaje formal de representación: lógica proposicional, lógica de predicados, etc..
  - El conocimiento se representa mediante sentencias.
- Mecanismos de razonamiento: Inferencia.
  - Es la derivación de nuevas sentencias a partir de las sentencias almacenadas y nuevas percepciones.
    - Adición de nuevo conocimiento (aprender)
    - Consultas a la BC (razonar)

Lenguaje + Inferencia = Lógica

Niveles de un ABC

- **Nivel de conocimiento o epistemológico**.
  - Es el nivel abstracto, describe **qué es lo que el agente sabe.** Corresponde al dominio del conocimiento (objeto de conocimiento).

diseñar

- **Nivel lógico.**

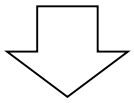
implementar Es donde el **conocimiento se codifica** mediante sentencias.

- **Nivel de implementación**.
  - Es el que opera la arquitectura del sistema.
  - Es **donde se encuentra** las representaciones de **las sentencias** correspondientes al nivel lógico

usar

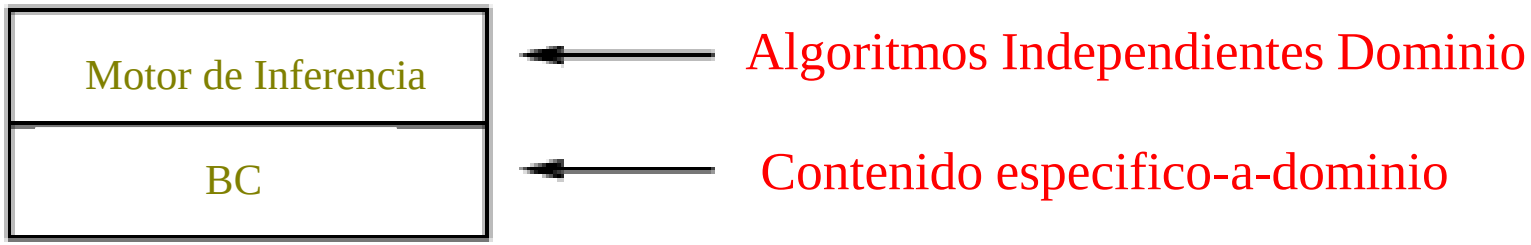
Base de Conocimiento (KB)

- Almacena un conjunto de conocimientos acerca del mundo, usando un lenguaje de representación de conocimiento.
- Cada conocimiento está representado por una sentencia (axioma) o hecho (instancia) del mundo.
- La BC tiene **conocimiento previo**, que corresponde al conocimiento no aprendido.
- Siempre que se ejecuta el ABC, sucede dos cosas:
  - El programa informa a la BC lo que percibe.
  - El programa pregunta a la BC qué hacer.



razonamiento lógico.

Base de Conocimiento



- Ingeniería de Conocimiento
  - Diseño y desarrollo de BC
  - Su objetivo es extraer, articular e informatizar el conocimiento de un experto.
  - Escritura de Sentencias que representan el conocimiento adquirido
- Ontología

Construcción de la Base de Conocimiento: Ingeniería de Conocimiento

1. Sobre que hablar
2. Decidir vocabulario
3. Escoger una lógica
4. Construir sentencias
5. Definir mecanismos de Inferencia
6. Hacer validaciones
7. Inferir nuevos hechos

Base de Conocimiento

1. lista hechos positivas
2. lista hechos negativas
3. lista conclusiones
4. lista premisas

hechos

sentencias

Ejemplos de cada uno:

1. hermano(Luis, Jose) hermano(Carlos, María)
2. no_hermano(Luis, Carlos)
3. hombre(x) hombre(y)
4. hermano(x,y) y hombre(y) => hermano(y,x)

Un Genérico ABC

Indicar(BC, percepción_sentencia(percibido,t))

Acción<- Preguntar(BC,consulta(t))

Indicar(BC, Acción_sentencia (Acción,t))

t+1

Regresa(Acción)

Componentes de la personalidad de un Agente

- Intencionalidad
- Posición (postura)
- Reciprocidad
- Comunicación
- Racionalidad

Intencionalidad

- Se refiere a las intenciones de un Agente definidas *racionalmente* por él.

- Por ejemplo: Si **yo** tengo un deseo de hacer algo o que algo ocurra de determinada manera (en el mundo)

- En los humanos las *creencias*, los *miedos* y los *deseos* influyen en la intencionaldad.

Creencias

ADMITIR COMO VERDAD ALGO SIN SABER EXACTAMENTE EL *GRADO DE VERACIDAD*

- AMBIENTALES
- SOCIALES
- RELACIONALES
    - CAPACIDADES
    - INTENCIONES Y COMPROMISOS
    - PLANES Y METODOS
    - ACTIVIDADES
    - ROLES
- PERSONALES

Intenciones: Creencias, Deseos

- *Consistencia intenciones-creencias*
  - Un agente debe creer que sus objetivos son posibles
  - No debe creer que no alcanzará sus objetivos
- *Completitud intenciones-creencias*
  - Las intenciones deben ser consistentes con las creencias.
- *Consistencia interna*
  - Un agente debe evitar tener intenciones en conflicto, pero puede tener deseos en conflicto

Reciprocidad

Un agente debe responder a las acciones de otros agentes

- Un agente debe ser capaz de razonar sobre las creencias, deseos, intensiones y habilidades de otros agentes para coexistir, cooperar y/o competir con ellos.

Comunicación

- Programas y/o Algoritmos = llamada a un procedimiento o pase de mensajes o sección critica

  ⇒ **conjunto predefinido de mensajes**

- Agentes = lenguaje de comunicación
  - **no existe un conjunto predefinido de mensajes**
  - **Nuevos conceptos a considerar: Conversaciones y Actos de Habla**

Racionalidad

- **Razonamiento:** conjunto de actividades mentales que consisten en conectar unas ideas con otras de acuerdo a ciertas reglas.

- **Inteligencia:** es la capacidad de entender, asimilar, elaborar información y conocimiento, y utilizarlos adecuadamente. Está íntimamente ligada a otras funciones mentales como la percepción (o capacidad de recibir dicha información), y la memoria (o capacidad de almacenarla).

Racionalidad

Es una capacidad humana que permite pensar, evaluar y actuar, de acuerdo a ciertos principios de optimidad y consistencia, para satisfacer algún objetivo o finalidad.

- Usando la razón, el ser humano intenta *elegir* para conseguir los mayores beneficios.
- Cualquier construcción mental llevada a cabo mediante procedimientos racionales tiene por tanto una *estructura lógica* distinguible.
- El ser humano tiene *otras formas para tomar decisiones* o idear comportamientos donde la racionalidad no parece el principal motor mental, adjetivadas a veces como "irracionales".

Racionalidad

- Elementos de la racionalidad:
  - Representación del Conocimiento (lenguaje)
  - Mecanismo de Razonamiento
  - Formas de Planificación
  - Mecanismos de Aprendizaje Automático

- La racionalidad es una conducta compleja:
  - Adaptar la conducta a condiciones cambiantes,
  - Aprender y usar nuevo conocimiento

Racionalidad

- **La racionalidad depende de 4 factores** :
  - De la medida con la que se evalúa el grado de éxito logrado.
  - De todo lo que hasta este momento haya percibido el agente (secuencia de percepciones).
  - Del conocimiento que posea el agente acerca del medio.
  - De las acciones que el agente pueda emprender.

para cada posible percepción, debe escoger la acción que le maximice su rendimiento, en base a la secuencia percibida y a su BC

- Posee:
  - Un hilo propio de control:
    - Su actividad es ordenar *seguir adelante*
    - Maneja su propio destino, tal que le da autonomía, por ejemplo, le permite *detenerse o decir no*

Medida de rendimiento

- Para cada conjunto de percepciones, el agente toma la acción que maximiza su rendimiento, basado en la información de la percepción y su propio conocimiento implícito.
- Medida del desempeño
  - Evalúa el "cómo"
  - ¿qué tan exitoso ha sido un agente?
  - Debe ser objetiva

- La racionalidad NO ES omnisciencia, clarividencia, ni exitosa necesariamente.
- La racionalidad se puede ver como un ***éxito*** *esperado, tomando como base lo que se ha percibido.*

Conocimiento

El **Conocimiento** son los contenidos sabidos o conocidos por un individuo o sociedad.

p. ej., un conocimiento en las sociedades actuales es el hecho de que la Tierra es redonda.

- Por ejemplo, se diría que la existencia de brujas era conocida en la Edad Media, incluso si actualmente decimos que estas creencias son infundadas, y no constituyen propiamente conocimientos.
- Las ciencias constituyen uno de los principales tipos de conocimiento.
- Hay también, no obstante, muchos tipos de conocimiento no científicos:
  - el *saber hacer* como en la artesanía, o el saber nadar, etc.;
  - el conocimiento de la lengua, de las tradiciones, leyendas, costumbres o ideas de *una cultura* particular;
  - el conocimiento que los individuos tienen de su propia *historia*
  - los conocimientos comunes a una sociedad dada, incluso a la humanidad (saber para qué sirve una martillo, saber que el agua extingue el fuego).

Tipos de Conocimiento del Agente

- *Conocimiento del estado actual* (*ej*. sobre el progreso actual de la solución, sobre mi estado interno)
- *Conocimiento de sus capacidades* (*ej*. descripción de las tareas, recetas, *etc. que puede hacer*)
- *Conocimiento evaluativo* (*ej*. tiempo para realizar una tarea, tiempo estimado de terminación, etc.)
- *Conocimiento del dominio* (*ej. ambiente, etc.)*

El ***conocimiento*** puede ser distinguido de las ***creencias*** considerando el conocimiento como una creencia que es verdadera 100%

Inferencia

Según la filosofía, existen tres modos básicos de razonamiento

- Deducción. inferencia desde las causas hacia los efectos, o desde lo universal hacia lo particular.
- Inducción. Recorre el camino inverso.
- Abducción o retroducción. Relacionado con la génesis de la hipótesis

<table>
<tr><td rowspan="3">Inferencia</td><td colspan="2">Deductiva o analítica</td></tr>
<tr><td rowspan="2">Sintética</td><td>Inducción</td></tr>
<tr><td>Hipótesis</td></tr>
</table>

Razonamiento

- **RAZONAMIENTO DEDUCTIVO**:

  De premisas generales a conclusiones particulares

  – Parte de categorías generales, para hacer afirmaciones sobre casos particulares.

  – La conclusión debe poder derivarse necesariamente de las premisas (las premisas implican lógicamente la conclusión), aplicando a éstas algunas de las reglas de inferencia.

- **RAZONAMIENTO ABDUCTIVO** :

  Dada una conclusión conocida, propone hipótesis que la expliquen.

  – En caso de conflicto entre hipótesis, se escoge la que mejor explique el suceso.

  – Consiste en derivar hipótesis novedosas sobre la causa

  Ejemplo: si sabemos que llover implica mojar algo, si se ve mojado se puede hipotetizar que llovió

Razonamiento

- **RAZONAMIENTO INDUCTIVO**:

  De premisas particulares a una conclusión general

  - consiste en obtener conclusiones generales a partir de premisas que contienen datos particulares. Por ejemplo, de la observación repetida de objetos o acontecimientos de la misma índole, se establece una conclusión para todos los objetos o eventos de dicha naturaleza.
    - *Premisas:*
      - He observado el cuervo número 1 y era de color negro.
      - El cuervo número 2 también era negro.
      - El cuervo número 3 también
    - *Conclusión:* todos los cuervos son negros.
  - En este razonamiento, se generaliza para todos los elementos de un conjunto la propiedad observada en un número finito de casos.
  - Ahora, la verdad de las premisas no convierten en verdadera la conclusión, ya que en cualquier momento podría aparecer una excepción.

  De ahí que la conclusión de un razonamiento inductivo sólo pueda considerarse probable, es siempre una información incierta y discutible.

Razonamiento

- **Deducción (Especifico)**:
  Regla: "Todas las bolas de la bolsa x son blancas".
  Caso:"Estas bolas provienen de la bolsa x".
  Deducción:"Estas bolas son blancas".

- **Inducción (Generalizo)**:
  Caso:"Estas bolas proceden de la bolsa x"
  Caso: "Estas bolas son blancas".
  Inducción:"En la bolsa x todas las bolas son blancas"

- **Abducción (Hipótesis)**:
  Regla:"Todos las bolas de la bolsa x son blancas".
  Caso:"Estas bolas son blancas"
  Abducción:"Estas bolas proceden de la bolsa x".

Razonamiento

1	Todos los hombres son mortales	∀x hombre(x) -> mortal (x)	Implicación (sentencia)
2	Juan es hombre	hombre(Juan)	hecho
3	Juan es mortal	mortal(Jean)	conclusión

- Si tenemos 1 y 2, entonces podemos concluir (deducción).
- Si tenemos 2 y 3, entonces necesitamos encontrar una regla (inducción, aprendida desde ejemplos)
- Si tenemos 1 y 3 y no tenemos hipotesis (2), entonces necesitamos encontrarla (abducción, donde encontramos las causas de las consecuencias usando reglas. Tipicamente es del dominio de diagnóstico)

Representación del Conocimiento

Expresar el conocimiento de forma que sea manejable por el computador, de modo que pueda ser utilizado por los agentes.

- Aproximación Simbólica vs. No simbólica

- Paradigmas de representación de conocimiento
  - Procedural
  - Relacional
  - Jerárquico
  - Lógico

Representación del Conocimiento

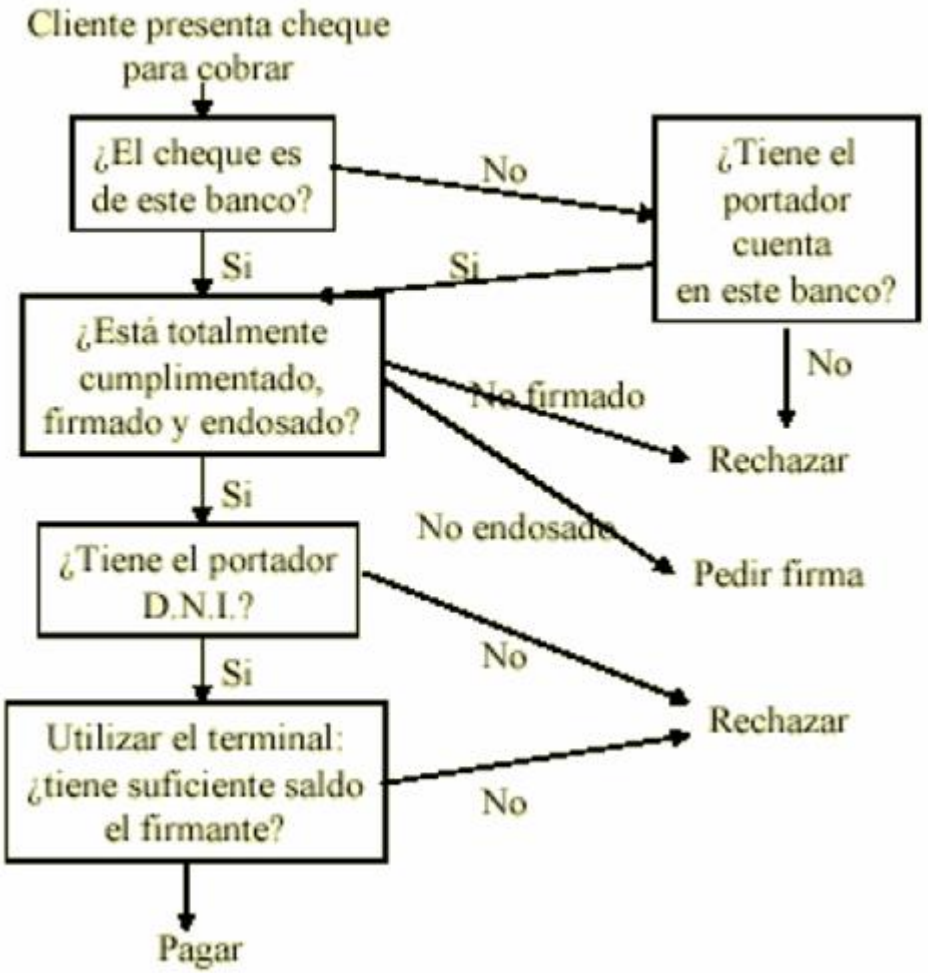
Un lenguaje consta de dos aspectos:

- **Sintaxis.**
  - Explica las posibles **configuraciones** mediante las cuales se forma las oraciones o sentencias (lenguaje).
  - Define las **sentencias** en el lenguaje

- **Semántica.**
  - Determina los **hechos** del mundo a los que se hace alusión en las oraciones o sentencias.
  - Define el "**significado**" de las sentencias

Representación del conocimiento

PROCEDIMENTAL



DECLARATIVO

(1) **Si** cheque completo y portador conocido y fondos suficientes **entonces** pagar

(2) **Si** fecha correcta y firmado y fondos suficientes y portador identificado y ... **entonces** cheque completo

(3) **Si** fecha cheque es hoy o fecha cheque entre 1 y 90 días antes de hoy **entonces** fecha correcta

Representación del Conocimiento

- **Lenguajes lógicos**: son lenguajes formales para representar información tales que conclusiones puedan ser alcanzadas

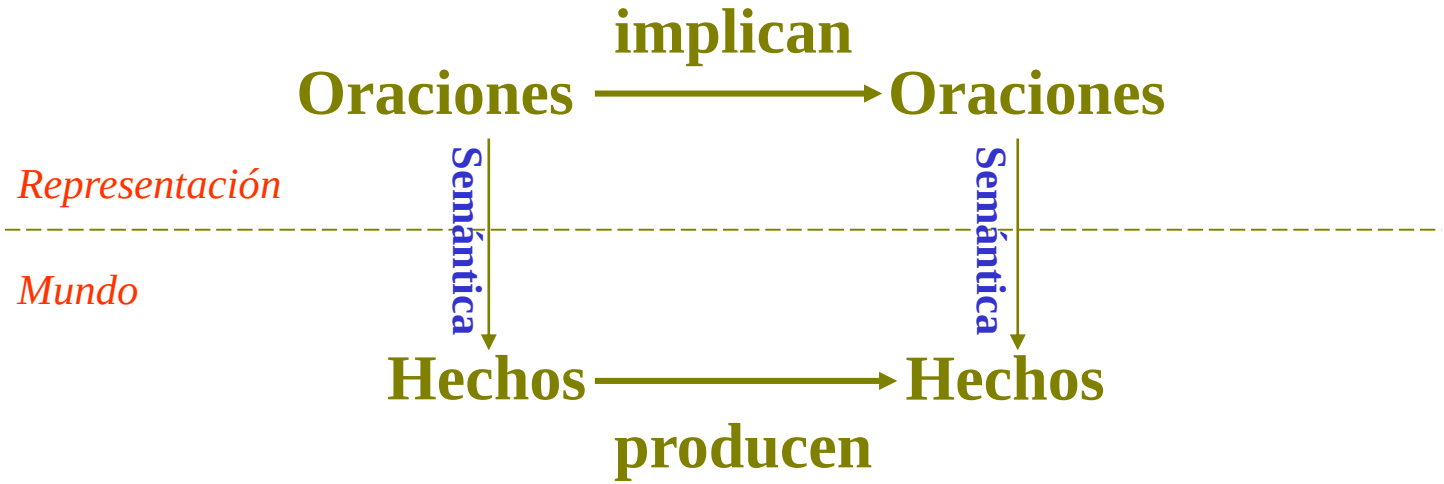
  =>inferencia

- **Ejemplos de lenguajes lógicos**
  - Calculo Proposicional (hechos)=>V/F/D
  - Calculo de Predicado de Primer Orden (Hechos, Objetos, Relaciones) =>V/F/D
  - Lógica Temporal(hechos, Objetos, Relaciones y tiempo) =>V/F/D
  - Teoría de probabilidad (hechos) =>Grado de Creer
  - Lógica Difusa (Grados de Verdad) =>Grado de Creer

Lenguajes lógicos

- Procedimiento de Inferencia
  - Determinar Activación de Sentencias dada una Sentencia

- La inferencia lógica genera nuevas oraciones que son consecuencia de oraciones ya existentes.



Inferencia

$BC \vdash_i \alpha$ = sentencia α puede ser derivada desde *BC por* procedimiento *i*

- Inferencia:
  - Es una **evaluación que realiza la mente** entre conceptos que, al interactuar, crean un punto axiomático o circunstancial, que permite **trazar una línea lógica de causa-efecto** entre los diferentes puntos inferidos en la resolución del problema.

- Inferencia lógica:
  - aplicación de una **regla de transformación** para transformar una fórmula o expresión bien formada (EBF) de un sistema formal en otra EBF.
  - La inferencia lógica es un proceso mediante el que se implanta la **relación de implicación** que existe entre dos oraciones.

Mecanismo de Inferencia

$\alpha \models \beta$, que significa que β se puede obtener desde α mediante inferencia.

- Realiza razonamiento
- Verifica la consistencia de una sentencia dada.
  - Es "completo" si puede encontrar una secuencia de pasos para cada sentencia que se puede producir .
  - Es "robusto" si los pasos que se siguen conducen solamente a sentencias que son consistentes con la base de conocimiento

Reglas de Inferencia

$$\Delta \mid =_{\rho} \omega$$

desde Δ se obtiene ω

ρ : reglas de inferencia

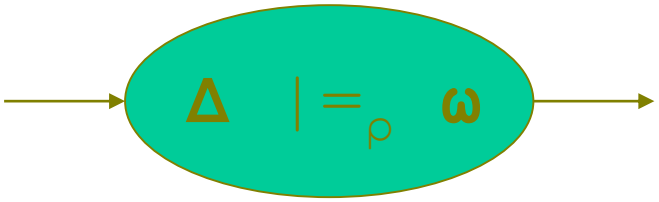
Δ : conjunto de fórmulas bien formada

ω : teoremas que se pueden deducir desde Δ

- Modus Ponens
- Y-Eliminación
- Y-Introducción.
- O-Introducción.
- Doble Negación Eliminación.
- Resolución Unitaria

INFERENCIA

Reglas + Observaciones (nuevos hechos)



Reglas de Inferencia

- **Modus ponendo ponens** (Latín: *modo que afirmando afirma*) de una implicación de la verificación de la premisa se verifica la conclusión:

  Si P entonces Q.

  Expresado en la notación de operadores lógicos:

  P->Q o P ├ Q

- **Modus tollendo ponens** (Latín: *modo que negando afirma*) de una implicación y de la no verificación del consecuente, se concluye la no verificación de la premisa :

  Es verdad P entonces ¬Q.

  Expresado en la notación de operadores lógicos:

  P -> ¬Q o P ├ ¬Q

Reglas de inferencia

Modus Ponens : $\dfrac{a \Rightarrow b,\ a}{b}$

Modus Tollens: $\dfrac{a \Rightarrow b,\ \neg b}{\neg a}$

Eliminación-y : $\dfrac{a1 \wedge a2 \wedge \ldots \wedge an}{ai}$

ai puede ser inferido de **a1 ∧ a2 …**

Introducción-y: $\dfrac{a1,\ a2,\ \ldots,an}{a1 \wedge a2 \wedge \ldots \wedge an}$

a1∧a2.. puede ser inferida de **a1, a2..**

Introducción-o: $\dfrac{ai}{a1 \vee a2 \vee \ldots \vee an}$

Eliminación-doble-negación: $\dfrac{\sim\sim a}{a}$

Resolución Unitaria: $\dfrac{a \vee b,\ \sim b}{a}$

Resolución: $\dfrac{a \vee b,\ \sim b \vee c}{a \vee c}$ $\quad$ $\dfrac{\sim a \Rightarrow b,\ b \Rightarrow c}{\sim a \Rightarrow c}$

Calculo Proposicional

- Gramática

Sentencia->SentenciaAtomica|SentenciaCompleja

SentenciaAtomica->Verdad|Falso|P|Q, ...

SentenciaCompleja->(Sentencia)|Sentencia conexión Sentencia| ¬ Sentencia

Conexión->∧ |∨ | => | ⇔

Calculo Proposicional

- Ejemplos:

  $\neg P \vee Q \wedge R \Rightarrow S$

  $((\neg P) \vee (Q \wedge R)) \Rightarrow S$

Semántica (Tabla de la Verdad)

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
F	*F*	*V*	*F*	*F*	*V*	*V*
F	*V*	*V*	*F*	*V*	*V*	*F*
V	*F*	*F*	*F*	*V*	*F*	*F*
V	*V*	*F*	*V*	*V*	*V*	*V*

- Proceso recursivo para evaluar una sentencia, por ejemplo:

$$\neg P_1 \wedge (P_2 \vee P_3) = true \wedge (true \vee false) = true \wedge true = true$$

Validez e inferencia

- Se puede obtener la validez de una oración compleja de la siguiente manera:

infiriendo →

P	H	$P \vee H$	$(P \vee H) \wedge \neg P$	$((P \vee H) \wedge \neg P) \Rightarrow P$
F	F	F	F	V
F	V	V	F	V
V	F	V	V	V
V	V	V	F	V

Calculo Proposicional

- **Sintaxis**
  - Si S es una sentencia, $\neg S$ es una sentencia (negación)
  - Si S_1 y S_2 son sentencias, $S_1 \wedge S_2$ es una sentencia (conjunción)
  - Si S_1 y S_2 son sentencias, $S_1 \vee S_2$ es una sentencia (disyunción)
  - Si S_1 y S_2 son sentencias, $S_1 \Rightarrow S_2$ es una sentencia (implicación)
  - Si S_1 y S_2 son sentencias, $S_1 \Leftrightarrow S_2$ es una sentencia (bicondicional)

- **Semántica**

$\neg S$ es verdad sii S es falso

$S_1 \wedge S_2$ es verdad sii S_1 es verdad Y S_2 es verdad

$S_1 \vee S_2$ es verdad sii S_1 es verdad O S_2 es verdad

$S_1 \Rightarrow S_2$ es verdad sii S_1 es verdad entonces S_2 es verdad

o, es verdad sii S_1 es falso entonces S_2 es falso

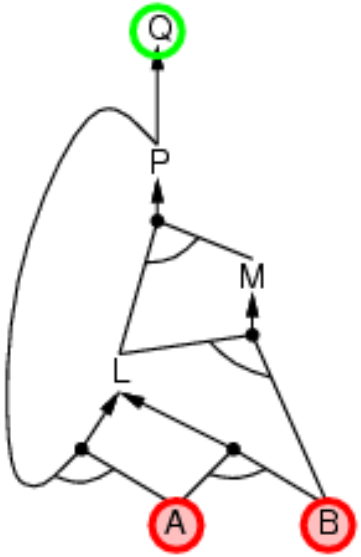
$S_1 \Leftrightarrow S_2$ es verdad sii $S_1 \Rightarrow S_2$ es verdad Y $S_2 \Rightarrow S_1$ es verdad

Calculo Proposicional

- Y-Eliminación $\alpha 1 \wedge \alpha 2 \wedge \alpha 3 \wedge .../ \alpha i$
- Y-Introducción $\alpha 1, \alpha 2, \alpha 3,... / \alpha 1 \wedge \alpha 2 \wedge \alpha 3 \wedge \ldots$
- O-Introducción $\alpha i / \alpha 1 \vee \alpha 2 \vee \alpha 3 \vee \ldots$
- Doble Negación Eliminación $\neg \neg \alpha / \alpha$
- Conmutación (para V y $\wedge$)
- Asociatividad (para V y $\wedge$)
- Distribución $(\alpha \text{ V } (\beta \wedge \gamma)) = ((\alpha \text{ V } \beta) \wedge (\alpha \text{ V } \gamma))$
- Morgan $\neg (\alpha \text{ V } \beta) = (\neg \alpha \text{ V } \neg \beta)$ (igual $\wedge$)

Hacia Adelante

- **Idea:** activar reglas cuyas premisas son satisfechas en la *BC*
  - Añadir sus conclusiones a la *BC*, hasta que consulta es conseguida
- **Es orientada a Datos.**
- Usada para **reconocimiento de objetos**



Hacia Atrás

- **Idea:** trabajar desde la respuesta *q*:
  - Chequear si *q* es conocida, y
  - probar en la BC todas las premisas de alguna regla que concluya *q*
- **Es orientada a Objetivos**,
- Apropiada para **resolver problemas**

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

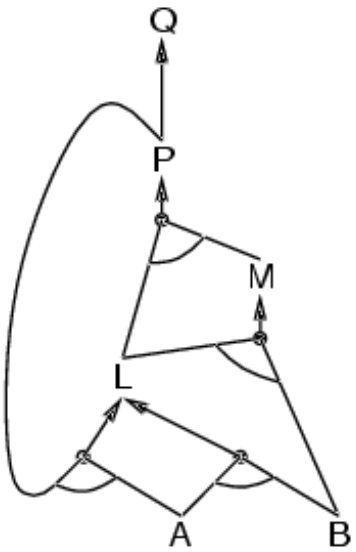
$B \wedge L \Rightarrow M$

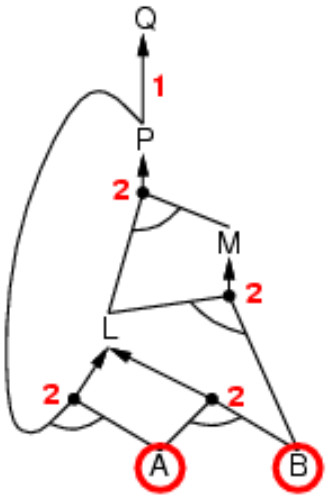
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B





Pro y contra de Calculo Proposicional

☺ Es declarativo

☺ Permite información parcial/negativa

☺ Es composicional:

$B \wedge P$ es derivado de los significados de B y P

☺ Significado es independiente-del-contexto

- lenguaje natural no lo es, donde significado depende del contexto

☹ Tiene limitado poder de expresión

- lenguaje natural es muy expresivo

Lógica de Predicados

- Lógica de primer orden.
- Es una lógica con suficiente expresividad para representar nuestro sentido común.
- La **lógica de predicados** tiene alcances ontológicos más amplios.
- Considera el mundo constituido por **objetos** y **propiedades** que los distinguen, a diferencia de la lógica proposicional que sólo permite representar **hechos**.

Lógica de Predicados

- Está basada en la idea de que las **sentencias** expresan **relaciones entre objetos**, así como también **cualidades** y **atributos** de tales objetos.
  - Los objetos pueden ser personas, objetos físicos, o conceptos.
  - Las cualidades, relaciones o atributos, se denominan **predicados**.
  - Los objetos se conocen como **argumentos** o **términos** del predicado.

- Al igual que las proposiciones, **los predicados tienen un valor de veracidad**, pero a diferencia de las preposiciones, su valor de veracidad, **depende de sus términos**.

  Un predicado puede ser verdadero para un conjunto de términos, pero falso para otro.

Proposiciones y Predicados

- Una proposición es una oración completa donde se afirma algo acerca de un sujeto identificado.

- Una sentencia en lógica de predicados es una oración completa, donde se afirma algo acerca de un sujeto. El sujeto puede ser una constante o una variable.

sentencia = oración = enunciado

Predicado

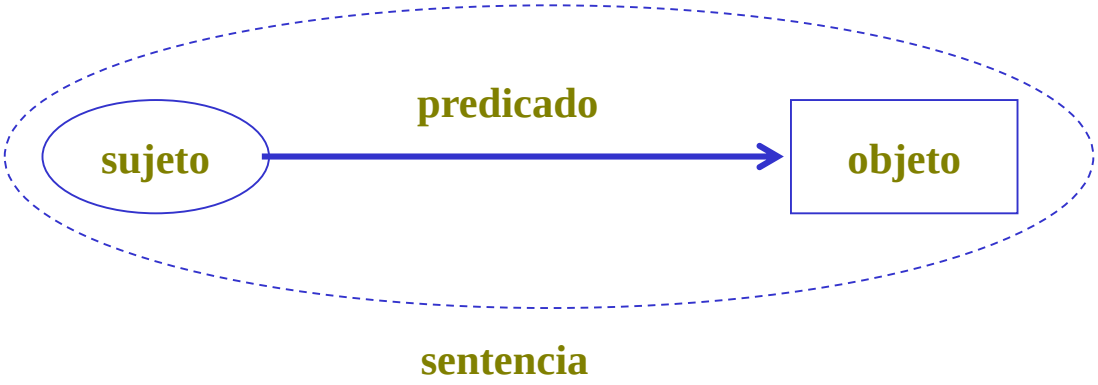
Un predicado es lo que se afirma del sujeto.

Predicado.

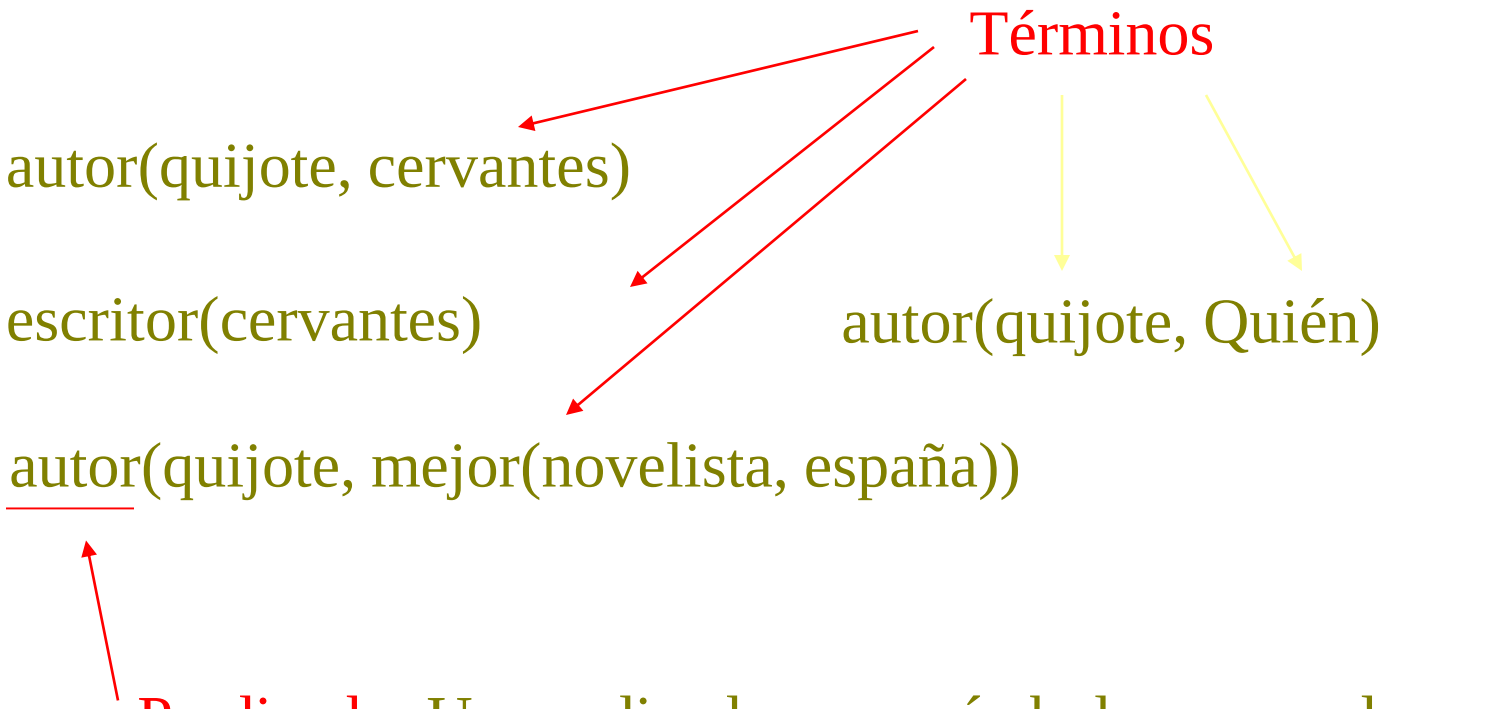
- Propiedades
- Cualidades
- Relaciones
- Atributos.
- Funciones

Sujeto.

- Términos, Cosas, Personas, Conceptos



Fórmulas atómicas



Predicado: Un predicado es un símbolo cuyo valor se encuentra en el dominio lógico (verdadero o falso) y representa alguna cualidad semántica, en un cierto contexto, acerca de las relaciones entre objetos o entidades

Lógica de orden cero:

Los predicados carecen de términos y el cálculo de predicados se reduce al cálculo de proposiciones o proposicional.

Lógica de orden uno:

Los predicados poseen términos que pueden ser constantes, variables o functores.

Lógica de orden superior:

Los predicados lógicos pueden ser en sí mismos variables.

Lógica de Predicado de Primer Orden

- Componentes
  - Objetos
  - Propiedades
  - Relaciones
  - Funciones
- Rector de la Universidad de los Andes es Joven
  - Objetos: Rector, ULA
  - Propiedad: Joven
- Juan es Rector de la Universidad de los Andes
  - Objetos: Juan, ULA
  - Relación: Rector

Lógica de Predicado 1er Orden

- Elementos del lenguaje:
  - Conectores: $\wedge$ (y), $\neg$ (no), $\vee$ (o), $\supset$ (implica) $\nRightarrow$(conllevar), $\Leftrightarrow$ (equivalente).
  - Predicados (relaciones/Función): Padrede(...) describe algo que es verdad.
  - Expresiones (hechos): Padrede(Vanessa, Jose)
  - Objetos: Vanessa
  - Sentencias:

[Mayorque(7,2) $\wedge$ Menorque(15,4)] $\vee$ Hermano(Jose, Luis)

Lógica de Predicado de Primer Orden

- Gramática

Sentencia->SentenciaAtomica|SentenciaCompleja|
Cuantificador Variable Sentencia

SentenciaAtomica->Predicado (termino, ...)

SentenciaCompleja->(Sentencia)|Sentencia conexión
Sentencia| ¬ Sentencia

Conexión->∧ |∨ | ⊃ | ⇔

Termino->Constante|Variable|función(termino)

Cuantificador->∃|∀

Sintaxis LPPO

El **alfabeto** está formado por:

- **Sentencia atómica:**

  `predicado (término, ....)`
  `termino = término`

- **Sentencias:**

  `¬ sentencia`
  `sentencias_atómicas.`
  `(sentencia conectiva sentencia)`

  `cuantificador variable, ....,`
  `  sentencia`

- **Término:**

  `función`
  `constante`
  `variable`

- **Símbolos de conectivas:**

  `( ∧, ∨, ⇔, ⇒, y ¬ )`

- **Cuantificador universal**:

  ∀ (para todos)

- **Cuantificador existencial**:

  ∃ (existe al menos uno)

Sintaxis LPPO

Oraciones atómicas

- Los términos y signos de predicado se combinan para formar oraciones atómicas, mediante las que se afirman hechos.

- Una oración atómica está formada por un signo de predicado y por una lista de términos entre paréntesis,

```
Hermano(Ricardo, Juan)
MadreDe(María, Juan)
```

- Una oración atómica es verdadera si la relación a la que alude el signo de predicado es válida para los objetos a los que aluden los argumentos.

Sintaxis LPPO

Oraciones Complejas

– Mediante los conectores lógicos se pueden construir oraciones más complicadas, ejemplo:

Hermano(Ricardo,Juan)∧Hermano(Juan, Jose)

Mayor(Juan, 30) ∨ Menor (Juan, 40)

Mayor(Juan, 30) ⇒ ¬Menor(Juan, 30)

¬Hermano(Robin, Juan)

Lógica de Predicado de Primer Orden

- **Sentencia Atómica**

  Hermano(Jose, Juan)

  Casados(Padrede(Jose), Madre de(Juan))

- **Sentencia Compleja**

  Hermano(Jose, Juan) Y Hermano(Juan, Jose)

  MasViejo(Juan, 30)=> ¬ MasJoven(Juan,30)

- **Cuantificadores:** permiten expresar propiedades de grupos completos de objetos en vez de enumerarlos por sus nombres

  ∃x Perro(x) => Amamanta(x) (existencial)

  ∀x Hermano(x, Juan) => Estudia (x) (universal)

  ∀x (Estudia(x) Y (∃x (Hermano(Jose ,x)))

  General: < *cuantificador*><*variable*> <*sentencia*>

Cuantificadores

Establecen el ámbito de existencia de las variables

Cuantificador Universal ($\forall$):

Cuando acompaña a una variable X establece que la fórmula es siempre válida para cada valor posible de la variable X. La variable X se dice entonces cuantificada universalmente.

"Todo cuerpo con masa cae"

$(\forall X)\ (\ [cuerpo(X) \wedge posee(masa, X)] \rightarrow cae(X)\ )$

Cuantificadores

Establecen el ámbito de existencia de las variables

Cuantificador Existencial (∃):

Establece que como mínimo existe un valor de la variable cuantificada que hace cierta la fórmula. La variable se dice cuantificada existencialmente.

"Alguién descubrió la penicilina"

(∃ X) (descubrió(X, penicilina)

Propiedades cuantificadores

- $\forall x\ \forall y$ es lo mismo que $\forall y\ \forall x$
- $\exists x\ \exists y$ es lo mismo que $\exists y\ \exists x$
- $\exists x\ \forall y$ no es lo mismo que $\forall y\ \exists x$

  $\exists x\ \forall y$ Quiere(x,y)

  "hay una persona quien ama a c/u en el mundo"

  $\forall y\ \exists x$ Quiere(x,y)

  "Cada quien en el mundo es amado por al menos una persona"

- Cuantificador de dualidad: uno puede ser expresado usando el otro

  $\forall x$ LeGusta(x, Helado) $\quad\quad \neg\exists x\ \neg$LeGusta (x, Helado)

  $\exists x$ LeGusta(x, Cebolla) $\quad\quad \neg\forall x\ \neg$LeGusta (x, Cebolla)

Lógica de Predicado de Primer Orden

- Alto Orden Lógico

  $\forall$ x,y (x=y) <=> $\forall$p p(x)=p(y)

- Operador λ

  λ x,y x-y

  λ25, 25 => 0 (hace la operación)

- Cuantificador $\exists$!

  $\exists$! x Rey(x)

- Axiomas: capturan los hechos básicos

- Dominio: coherente cuerpo de conocimiento sobre algo

Lógica de Predicado de Primer Orden

- Simple Agente Reflexivo

  $\forall$a Percibo(a, t) => Acción (recoger, a)

- Otros Operadores

  Ir(persona, sitio)

  En(posición, tiempo, Jose)

- Ejemplo: Todos los paquetes en hab. 27 son mas pequeños que los que están en la hab. 28

  $(\forall x,y)[\text{Paquete}(x) \wedge \text{Paquete}(y) \wedge \text{En}(x,27) \wedge \text{En}(y,28)] \supset \text{Maspequeno}(x,y)$

LPPO: Mas ejemplos

- Juan es Oso, Oso es animal, animal son cosas

  Oso(Juan)

  ∀b Oso(b)=>Animal(b)

  ∀a Animal(a)=>CosaFisica(a)

- Todos los animales tienen un cerebro, el cual es parte de todo animal

  ∀a Animal(a) <=>Cerebrode(a)

  ∀a Partede(a,Cerebrode(a))

- Carlos tiene cerebro pequeño

  Tamano(Cerebrode(carlos),cerebrode(tipicohumano))=pequeño

Ejercicio

Traduzcamos los siguientes ejemplos:

- "Todas las cosas no negras no son cuervos."

$$\forall x(\neg negro(x) \rightarrow \neg cuervo(x))$$
$$\neg \exists x(\neg negro(x) \wedge cuervo(x))$$
$$\forall x(negro(x) \vee \neg cuervo(x))$$

Ejercicio

Traduzcamos los siguientes ejemplos:

- "Ningún pájaro blanco llegara al Polo Sur".

$\neg \exists x\ (pájaro(x) \wedge blanco(x) \wedge llegara\ (x,PoloSur))$, o
$\forall x\ (pájaro(x) \wedge blanco(x) \rightarrow \neg llegara(x,PoloSur))$
$\forall x\ (\neg\ pájaro(x) \vee \neg\ blanco(x) \vee$
$\neg llegara(x,PoloSur))$

LPPO

- Razonamiento de un Agente (inferir)

  $En(x,y) \wedge Sucio(x) \supset absorber(x)$

  $En(0,0) \wedge Caminando(Sur) \wedge Sucio(0,1) \supset Hacer(rotar\_este)$

- Tipos de Razonamiento
  - Regla Causal (deduzco)

    $\forall$ l1,l2,s En(l1,s, jose) Y Camino(l1,l2)=> En(I2, s+t,jose)
  - Regla Diagnostico (Generalizo)

    $\forall$ x En(Florida, Agosto, x) Y LlueveFuerte(Florida)=> Tormenta(Florida)
  - Abducción

    $\forall$ x lluevesobre(x) => Mojado(x)

    Percibe(mojado(x))=> Hipótesis (ha llovido)

Lógica de Predicado de Primer Orden

- Preferencias entre acciones

BC

∀a,s Grande(a, s) Y ¬∃ b Grande (b,s) => acción (a)

∀a,s Mediano(a, s) Y ¬∃ b (Grande (b,s) O Mediano (b,s)) => acción (a)

- Agentes con Objetivos

∀ s Quiere(Oro, s) => ...

- Requiere:

Inferir

Buscar

Planificar

Uso Lógica de Predicado de Primer Orden

- BC:

  $\forall x$ Rey(x) $\wedge$ Codicioso(x) $\Rightarrow$ Malo(x)
  Rey(Juan)
  Codicioso(Juan)
  Hermano(Richard, Juan)

- Instanciando BC:

  Rey(Juan) $\wedge$ Codicioso(Juan) $\Rightarrow$ **Malo(Juan)**

BC

A. Marco era un hombre.
B. Marco era pompeyano (de Pompeya).
C. Todos los pompeyanos eran romanos.
D. César era un dirigente.
E. Todos los romanos o bien eran leales a César o bien le odiaban.
F. Todo el mundo es fiel a alguien.
G. La gente sólo trata de asesinar a aquellos dirigentes a los que no son leales.
H. Marco intentó asesinar a César.
I. Algún romano odia a César.

A. hombre(marco)

B. pompeyano(marco)

C. $(\forall X)(\text{pompeyano}(X) \rightarrow \text{romano}(X))$

D. dirigente(césar)

E. $(\forall X)(\text{romano}(X) \rightarrow \text{leal}(X,\text{césar}) \vee \text{odia}(X,\text{césar}))$

F. $(\forall X)(\text{hombre}(X) \rightarrow [(\exists Y)\ \text{leal}(X,Y)]$

G. $(\forall X)[(\forall Y)\ (\text{hombre}(X) \wedge \text{dirigente}(Y)$

$\wedge\ \text{intenta\_asesinar}(X,Y) \rightarrow \neg\ \text{leal}(X,Y))]$

H. intenta_asesinar(marco, césar)

I. $(\exists X)\ (\text{romano}(X) \wedge \text{odia}(X, \text{césar}))$

Lo que queremos hacer ...

Teorema + Axiomas (como fórmulas bien formadas, fbf)

+

Teorema + Axiomas (como cláusulas)

+

Método de resolución por refutación

Sustitución

Cláusula resolvente

Fórmulas bien formadas (fbf)

Una fórmula se dice bien formada si puede derivarse de alguna de las siguientes reglas de formación:

1. Cualquier fórmula atómica es una fbf.
2. Si p y q son fbf, entonces también serán fbf las siguientes:

   $\neg p,\ p \vee q,\ p \wedge q,\ p \rightarrow q$

3. Si X es una variable libre en la fbf p, entonces son fbf :

   $(\forall X)\ p(X),\ (\exists X)\ p(X)$

Cláusula

Es una fórmula que sólo contiene operadores disyuntivos y posiblemente negaciones sobre los átomos. Así la fórmula:

$$a_1 \vee a_2 \vee \ldots \vee a_n \vee\ b_1 \vee b_2 \vee \ldots \vee b_m$$

se transforma en :

$$\neg(\neg a_1 \wedge \neg a_2 \wedge \ldots \wedge \neg a_n) \vee\ b_1 \vee b_2 \vee \ldots \vee b_m$$

$$\neg a_1 \wedge \neg a_2 \wedge \ldots \wedge \neg a_n \rightarrow b_1 \vee b_2 \vee \ldots \vee b_m$$

Cláusulas de Horn.

Aquellas cuyo consecuente tiene un único predicado.

$$a_1 \wedge a_2 \wedge ... \wedge a_n \rightarrow b$$

Ventajas de una representación basada en cláusulas de Horn

- La demostración de teoremas o resolución de un conjunto de cláusulas de Horn es más sencilla en general.
- Las cláusulas de Horn admiten una interpretación directa en términos de un lenguaje de programación, lo que no es posible con cláusulas generales.

Proceso de paso a cláusulas

1. *Eliminar los símbolos de implicación*
2. *Mover las negaciones hasta las fórmulas atómicas*
3. *Renombrar variables*
4. *Eliminar los cuantificadores existenciales.*
5. *Desplazar los cuantificadores universales*
6. *Convertir los operadores AND en los más externos*
7. *Eliminar los cuantificadores universales.*
8. *Eliminar los conectores conjuntivos (AND).*
9. *Renombrar las variables.*

Equivalencias y leyes en LPO

Leyes de Morgan:

$\neg\exists x F \models\!\dashv \forall x \neg F$

$\neg\forall x F \models\!\dashv \exists x \neg F$

Si la variable x no figura en la fórmula G podemos decir:

$\forall x F(x) \wedge G \models\!\dashv \forall x(F(x) \wedge G)$

$\exists x F(x) \wedge G \models\!\dashv \exists x(F(x) \wedge G)$

$\forall x F(x) \vee G \models\!\dashv \forall x(F(x) \vee G)$

$\exists x F(x) \vee G \models\!\dashv \exists x(F(x) \vee G)$

$\forall x F(x) \wedge \forall x G(x) \models\!\dashv \forall x(F(x) \wedge G(x))$

$\exists x F(x) \vee \exists x G(x) \models\!\dashv \exists x(F(x) \vee G(x))$

$\forall x\, \forall y G(x,y) \models\!\dashv \forall y\, \forall x G(x,y)$

$\exists x\, \exists y G(x,y) \models\!\dashv \exists y\, \exists x G(x,y)$

Equivalencias y Leyes

Leyes de Morgan:

- "Todos los pájaros no llegan al horizonte"

$\neg\exists xF \models\!\!\models \forall x\neg F$

- "No es cierto que a todos les apetece la opera"

$\neg\forall xF \models\!\!\models \exists x\neg F$

Otros casis

- " A todos les gusta: ir al cine y la coca-cola"

$\forall xF(x) \wedge \forall xG(x) \models\!\!\models \forall x(F(x) \wedge G(x))$

- "Aqui hay gente con pelo rizado o hay gente con ojos azules."

$\exists xF(x) \vee \exists xG(x) \models\!\!\models \exists x(F(x) \vee G(x))$

- "A todas las personas todos los días les gusta ir al mar."

$\forall x\ \forall yG(x,y) \models\!\!\models \forall y\ \forall xG(x,y)$

- "Existen plantas y existen animales que comen plantas."
- "Es cierto que existen animales y plantas tales que los animales comen estas plantas?!"

$\exists x\ \exists yG(x,y) \models\!\!\models \exists y\ \exists xG(x,y)$

Lógica de Predicado de Primer Orden

- Resolución por Refutación

Misil(M1)
Dueno(Carlos,M1)
$\forall$x Misil(x) y Dueno(Carlos, x)=> Vender (Oeste,Carlos, x)

y se puede inferir

Vender(Oeste,Carlos,M1)

Lógica de Predicado de Primer Orden (Resolución por Refutación)

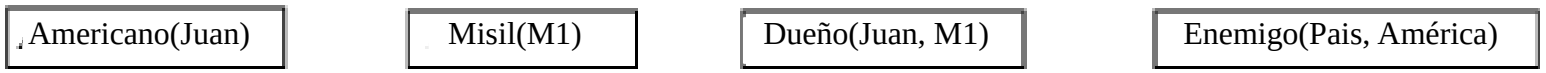
- **Cadena hacia adelante (Preguntar)**
  - Preguntar(BC, Abuelo (Juan, María))
  - Comenzar con sentencias y llegar a conclusiones (se determinan las consecuencias)
  - Hechos que llegan activan proceso de razonamiento

Americano(x) Y Arma(y) Y Hostil(z) Y Vende(z,y,x) =>Traidor(x)

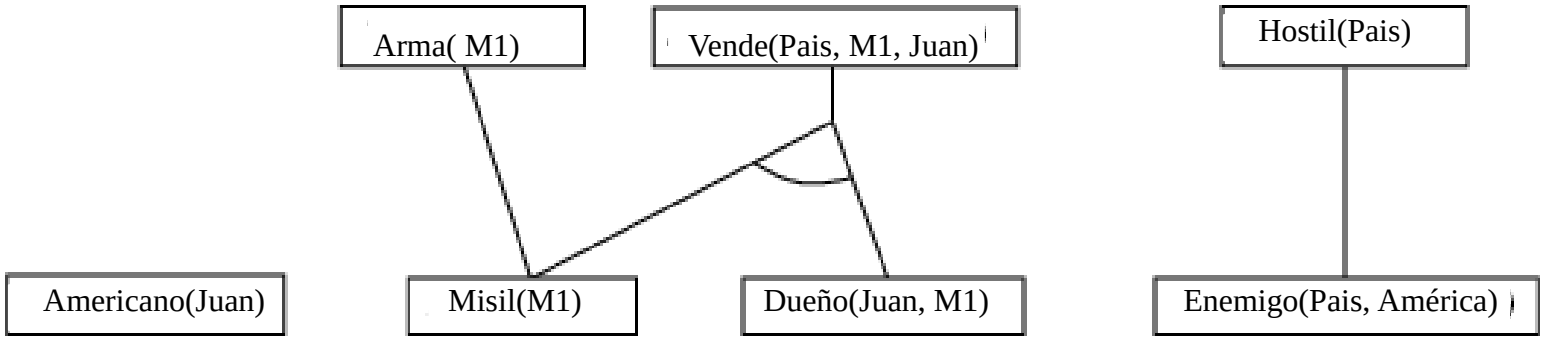
- **Cadena hacia atrás (Decir)**
  - Decir(BC, ∀ m,c Madre(c)=m <=> Hembra(m)Y Padres(m,c))
  - Se parte de algo probado a nivel de ocurrencias y se retrocede para ver las sentencias implicadas que permiten concluir eso

MODUS PONENS permite eso!!

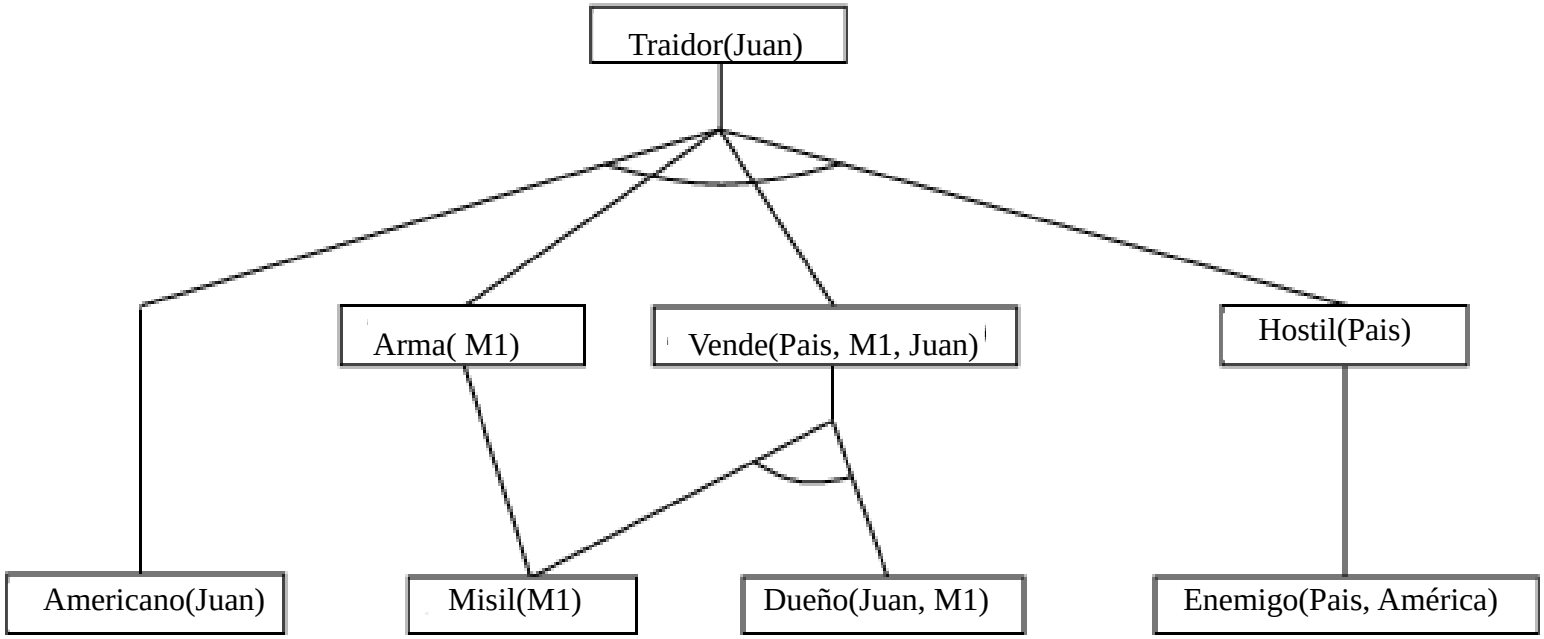
Hacia Adelante



Hacia Adelante



Hacia Adelante



Hacia Adelante

function FOL-FC-ASK(KB, α) **returns** a substitution or *false*

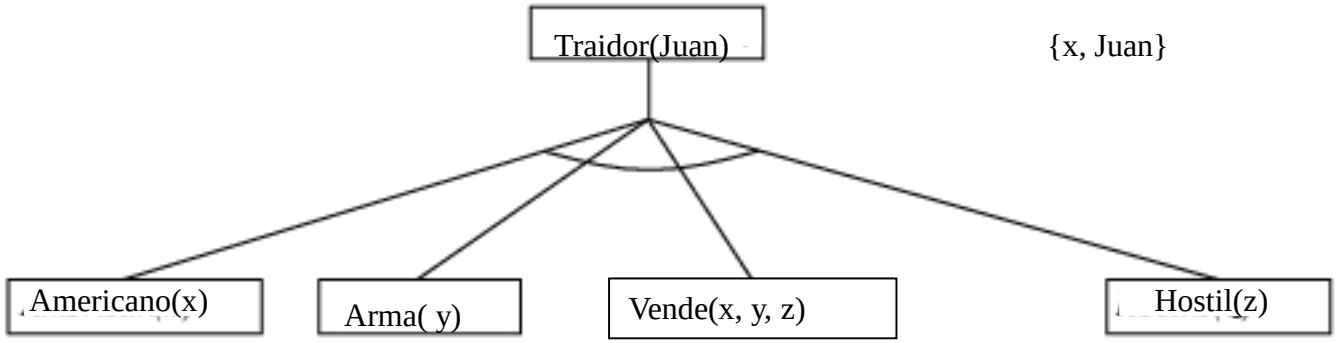
repeat until *new* is empty
- $new \leftarrow \{\}$
- **for each** sentence r **in** KB **do**
  - $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-APART(r)
  - **for each** θ such that $(p_1 \wedge \ldots \wedge p_n)\theta = (p'_1 \wedge \ldots \wedge p'_n)\theta$ for some $p'_1, \ldots, p'_n$ in KB
    - $q' \leftarrow$ SUBST(θ, q)
    - **if** q' is not a renaming of a sentence already in KB or new **then do**
      - add q' to new
      - $\phi \leftarrow$ UNIFY(q', α)
      - **if** ϕ is not *fail* **then return** ϕ
- add new to KB

return *false*

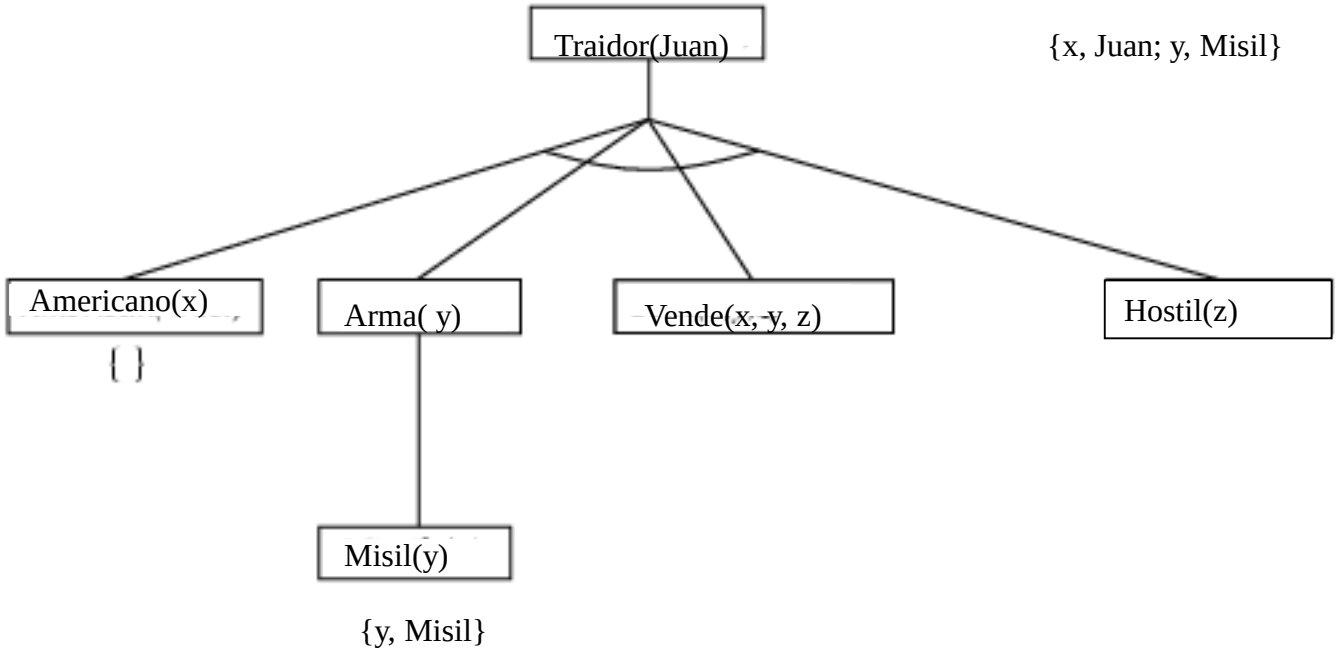
Hacia Atrás

- Traidor(Juan)

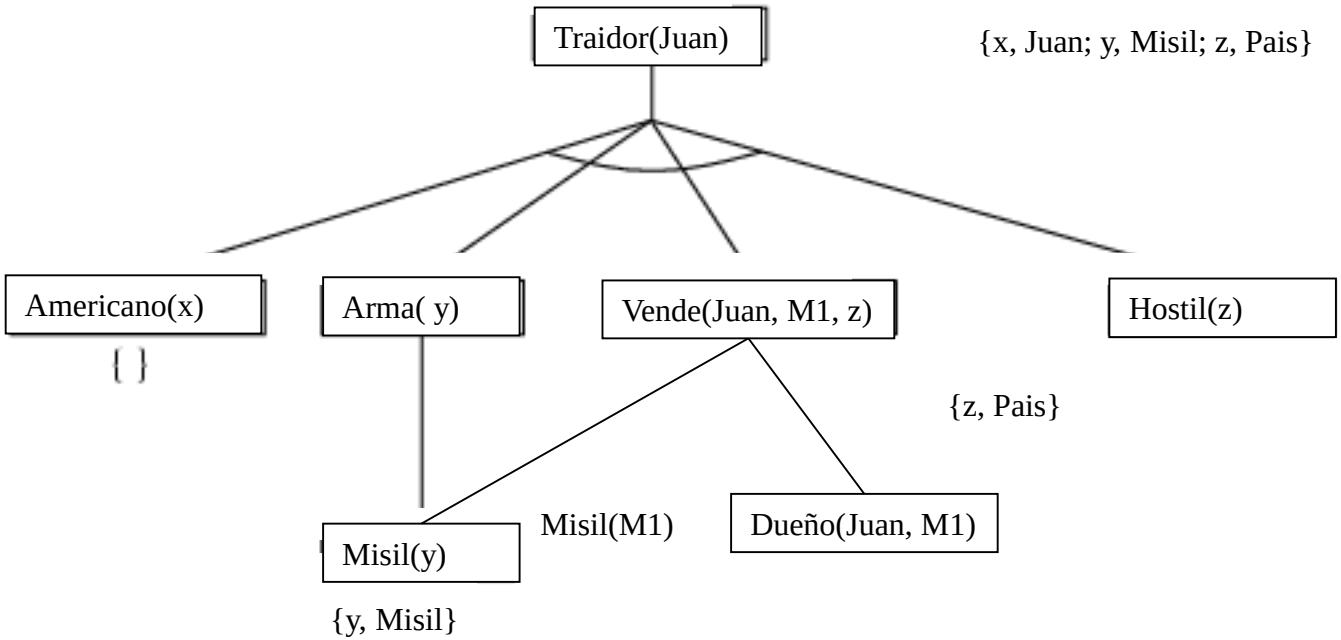
Hacia Atrás



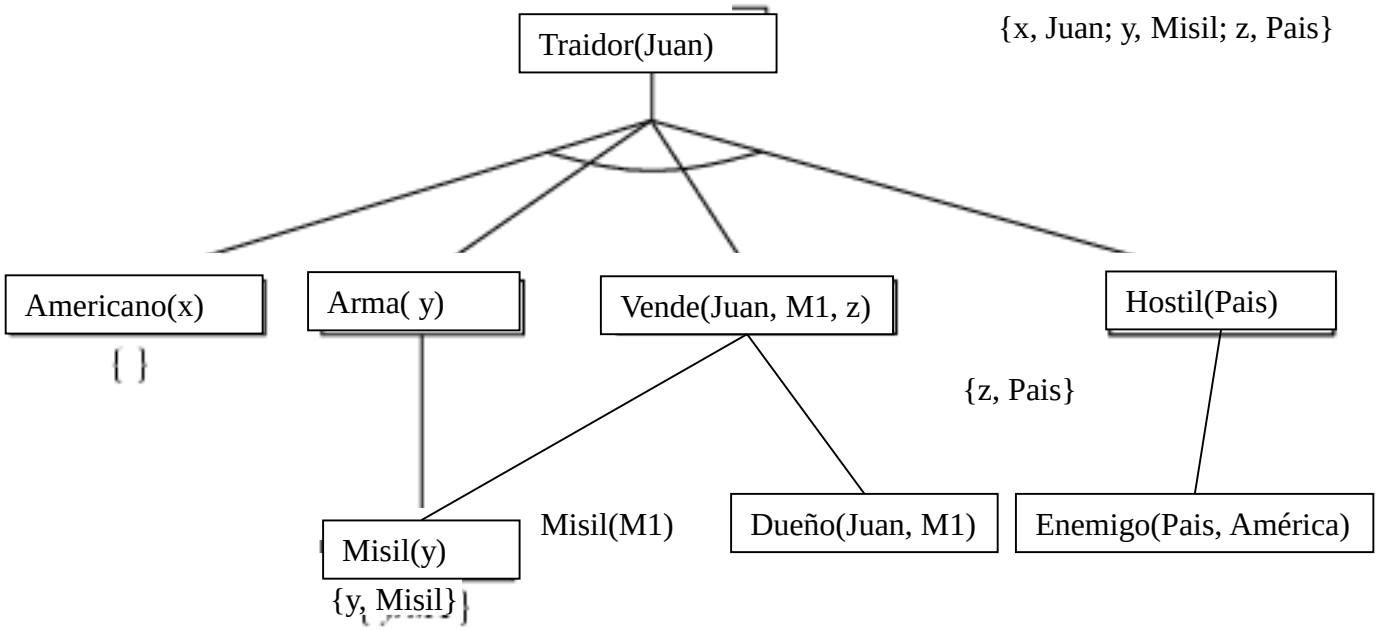
Hacia Atrás


Traidor(Juan)
{x, Juan; y, Misil}
Americano(x)
{ }
Arma(y)
Vende(x, y, z)
Hostil(z)
Misil(y)
{y, Misil}

Hacia Atrás


Traidor(Juan)
{x, Juan; y, Misil; z, Pais}
Americano(x)
{ }
Arma(y)
Vende(Juan, M1, z)
Hostil(z)
{z, Pais}
Misil(y)
{y, Misil}
Misil(M1)
Dueño(Juan, M1)

Hacia Atrás


Traidor(Juan)
{x, Juan; y, Misil; z, Pais}
Americano(x)
{ }
Arma(y)
Vende(Juan, M1, z)
Hostil(Pais)
{z, Pais}
Misil(y)
{y, Misil}
Misil(M1)
Dueño(Juan, M1)
Enemigo(Pais, América)

Hacia Atrás

function FOL-BC-ASK(KB, goals, θ) **returns** a set of substitutions

inputs: KB, a knowledge base

$goals$, a list of conjuncts forming a query

θ, the current substitution, initially the empty substitution $\{\}$

local variables: ans, a set of substitutions, initially empty

if $goals$ is empty **then return** $\{\theta\}$

$q' \leftarrow$ SUBST(θ, FIRST($goals$))

for each r **in** KB where STANDARDIZE-APART(r) $= (p_1 \wedge \ldots \wedge p_n \Rightarrow q)$

and $\theta' \leftarrow$ UNIFY(q, q') succeeds

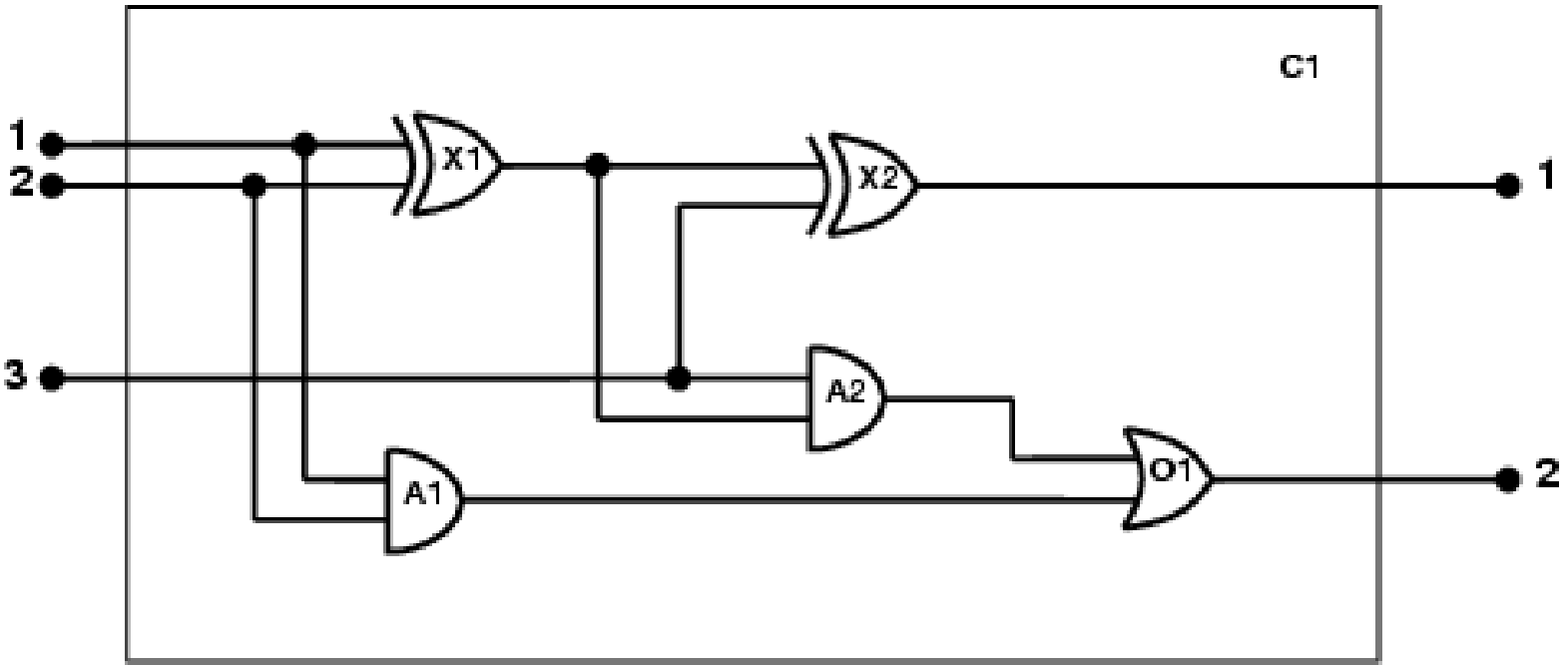
$ans \leftarrow$ FOL-BC-ASK(KB, $[p_1, \ldots, p_n | \text{REST}(goals)]$, COMPOSE($\theta, \theta'$)) $\cup$ ans

return ans

Ingeniería de Conocimiento en LPPO

1. Identificar la tarea
2. Identificar Conocimiento Relevante
3. Decidir vocabulario: conceptos, funciones, y constantes => **Ontología**
4. Codificar conocimiento => Axiomas de la ontologías (sentencias lógicas)
5. Codificar instancias del problema (hechos=>descripciones)
6. Hacer consultas al procedimiento de inferencia
7. Depurar la BC

Identificar la tarea: diseño de circuitos electrónicos



Diseño de circuitos electrónicos

1. Identificar la tarea
   - Diseño de circuitos
2. Identificar Conocimiento Relevante
   - Circuitos, señales, puertas, terminales
   - Tipos de puertas (AND, OR, XOR, NOT)
   - Irrelevante: tamaño, color, costo de las puertas
3. Decidir vocabulario: conceptos, funciones, y constantes, terminales
   - Tipos de puertas: XOR, …
   - Puertas: X_1, X_2, .. Terminales: Out(1, X_1)
   - Funciones:

     Type(X_1) = XOR o Type(X_1, XOR)

     Connected(Out(1, X_1), In(1, X_2)

Diseño de circuitos electrónicos

4. Codificar conocimiento

- $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Signal(t_1) = Signal(t_2)
- $\forall t$ Signal(t) = 1 $\vee$ Signal(t) = 0
- $1 \neq 0$
- $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Connected(t_2, t_1)
- $\forall g$ Type(g) = OR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 1
- $\forall g$ Type(g) = AND $\Rightarrow$ Signal(Out(1,g)) = 0 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 0
- $\forall g$ Type(g) = XOR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ Signal(In(1,g)) $\neq$ Signal(In(2,g))
- $\forall g$ Type(g) = NOT $\Rightarrow$ Signal(Out(1,g)) $\neq$ Signal(In(1,g))

Diseño de circuitos electrónicos

5. Codificar instancias del problema

$Type(X_1) = XOR$ $Type(X_2) = XOR$

$Type(A_1) = AND$ $Type(A_2) = AND$

$Type(O_1) = OR$

$Connected(Out(1,X_1),In(1,X_2))$ $Connected(In(1,C_1),In(1,X_1))$

$Connected(Out(1,X_1),In(2,A_2))$ $Connected(In(1,C_1),In(1,A_1))$

$Connected(Out(1,A_2),In(1,O_1))$ $Connected(In(2,C_1),In(2,X_1))$

$Connected(Out(1,A_1),In(2,O_1))$ $Connected(In(2,C_1),In(2,A_1))$

$Connected(Out(1,X_2),Out(1,C_1))$ $Connected(In(3,C_1),In(2,X_2))$

$Connected(Out(1,O_1),Out(2,C_1))$ $Connected(In(3,C_1),In(1,A_2))$

Diseño de circuitos electrónicos

6. Hacer consultas al procedimiento de inferencia

   ¿Cuales son los valores de los terminales?

   $\exists i_1,i_2,i_3,o_1,o_2$ Signal(In(1,C_1)) = $i_1 \wedge$ Signal(In(2,C_1)) = $i_2 \wedge$ Signal(In(3,C_1)) = $i_3 \wedge$ Signal(Out(1,C_1)) = $o_1 \wedge$ Signal(Out(2,C_1)) = o_2

7. Depurar la BC

   Eliminar sentencias, determinar sentencias redundantes, …

Tarea: realizar Ingeniería de Conocimiento en LPO para curso de IA

1. Identificar la tarea
2. Identificar Conocimiento Relevante
3. Decidir vocabulario
4. Codificar conocimiento
5. Codificar instancias del problema
6. Hacer consultas al procedimiento de inferencia
7. Depurar la BC

Reglas de Producción

- Forma más popular de representación del conocimiento
  - Pertenece al paradigma declarativo
  - Es de la forma si … entonces
  - Soporta los distintos tipos de razonamiento
  - Es fácil incorporar nueva información

Si antecedente entonces consecuente

- Una base de reglas puede tener varias reglas que se activan, la estrategia de control dirá cual se activa primero

Prolog: Programación Lógica

- Lenguaje de programación PROLOG (“PROgrammation en LOGique”) creado por Alain Colmerauer y sus colaboradores alrededor de 1970.

- Primer intento de diseñar un lenguaje de programación que posibilitara al programador especificar sus problemas en lógica.

- Juega un importante papel dentro de la Inteligencia Artificial, ya que se utiliza para los Sistemas de Procesamiento de Conocimiento

- PROLOG es un lenguaje de programación que se basa en la Lógica de Primer Orden

Prolog: Programación Lógica

- Algoritmo = Lógica + Control

- Base inferencia: hacia atrás

- Programa = conjunto de clausulas (clauses)

  Una clausula es head :- $literal_1$, … $literal_n$.

- Ejemplo:

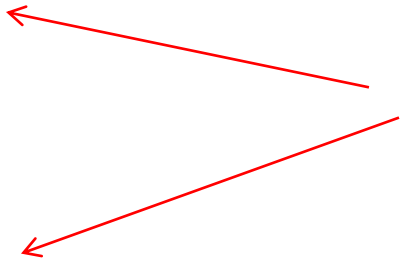
  traidor(X) :- americano(X), cañón(Y), vende (X,Y,Z), hostil(Z)

Prolog: Programación Lógica

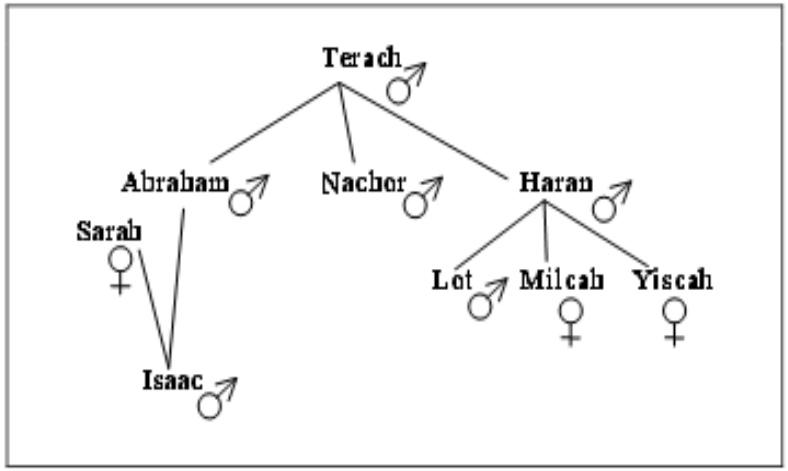
Ejemplos

```
precio( boli, 0.5 ).
precio( folios, 2.5 ).
precio( ordenador, 700 ).
precio( coche, 15000 ).
precio( tomates, 1 ).

%% ¿Qué puedo comprar con 100 euros?
?- precio( X,Y ), Y < 100.
X=boli, Y=0,5;
X=folios, Y=2,5;
X=tomates, Y=1;
No.
```

```
ganados( juan, 7 ).
ganados( susana, 6 ).
ganados( pedro, 2 ).
ganados( rosa, 5 ).
jugados( rosa, 10 ).
jugados( juan, 13 ).
jugados( pedro, 3 ).
jugados( susana, 7 ).
porcentaje( X,Y ):- ganados( X,Z ),jugados( X,Q ),Y is ( Z/Q )*100.

%% ¿Qué porcentaje de ganados tienen?
?- porcentaje( X,Y ),.
X=juan,Y=53.84;
X=susana, Y=85.71;
X=pedro, Y=66.66;
X=rosa,Y=50;
No.
```



Prolog: Definición de relaciones mediante hechos



Programa 1:

1. es_padre(terach, abraham).
2. es_padre(terach, nachor).
3. es_padre(terach, haran).
4. es_padre(abraham, isaac).
5. es_padre(haran, lot).
6. es_padre(haran, milcah).
7. es_padre(haran, yiscah).
8. es_madre(sarah, isaac).
9. es_hombre(terach).
10. es_hombre(abraham).
11. es_hombre(nachor).
12. es_hombre(haran).
13. es_hombre(isaac).
14. es_hombre(lot).
15. es_mujer(sarah).
16. es_mujer(milcah).
17. es_mujer(yiscah).

PREGUNTA	SIGNIFICADO	RESPUESTA
?es_padre(abraham, Isaac)	¿es padre Abraham de Isaac?	**Sí**, pues encuentra un hecho que lo afirma.
?es_padre(abraham, Loi)	¿es padre Abraham de Lot?	**No**, pues no hay ningún hecho que lo afirme.
?es_padre(abraham, X)	¿existe un X tal que es padre Abraham de X?	**Sí**. El programa contestará dando para X los valores que verifican dicha relación: X =Isaac
?es_padre(haran, X)	¿existe un X tal que es padre Haran de X?	**Sí**, dando todas las soluciones posibles: X = lot, X=milcah, X=yiscah

Prolog: Definición de relaciones mediante hechos

También se pueden hacer preguntas más complejas:

PREGUNTA	SIGNIFICADO	RESPUESTA
?es_padre(haran,lot), es_hombre(lot).	¿es padre Haran de Lot y es hombre Lot?	Sí, pues ambos fines son ciertos.
?es_padre(abraham,lot), es_hombre(lot).	¿es padre Abraham de Lot y es hombre Lot?	La respuesta es no pues algún fin no es cierto.
?es_padre(abraham,X), es_hombre(X).	¿existe un X tal que es padre Abraham de X y es hombre X?	La respuesta es sí pues ambos fines son ciertos para algún valor de X. El programa contestará X=isaac.
?es_padre(haran,X), es_mujer(X).	¿existe un X tal que es padre Haran de X y es mujer X?	El programa contestará que sí, dando todas las soluciones posibles: X=milcarh, X=yiscah

Prolog

El lenguaje Prolog contiene una serie de predicados de evaluación aritmética:

- Para realizar una operación se usa el predicado **is**. Ejemplo:

?- X is 1 + 2. X= 3

Donde ?- define una función

- El operador .**is**. también permite evaluar otras operaciones: resta (.-.), multiplicación(.*.), división(./.), potencia(.**.), módulo (.mod.).

Ejemplo:

?- X is 5/2, Y is 2 ** 3, Z is 5 mod 2.

X = 2.5 Y = 8 Z = 1

Prolog: Estructura de decisión

La estructura If-Then-Else en prolog es:

(a->b; c)

Lo cual significa si a entonces b de lo contrario c

Ejemplo:

Escribir el mayor entre 2 números.

```
imprime_mayor(X,Y) :- ( X > Y -> write(X), write(' es mayor');
write(Y), write(' es mayor') ).
```

Prolog: Reglas Recursivas

Las reglas son sentencias de la forma: A :- B1 , ... , Bn . donde A y cada Bi son átomos. A es la cabeza de la regla y los Bi's componen el cuerpo de la regla.

- Ejemplo: Una regla que define la relación “ser hijo” a partir de las relaciones dadas podría ser:

  es_hijo(X,Y) :- es_padre(Y,X), es_hombre(X).

  que se leería de la forma:

“para cualesquiera X e Y, X es hijo de Y si Y es padre de X y X es hombre"

- De igual forma se definirían otras relaciones mediante reglas:

es_hija(X,Y) :- es_padre(Y,X), es_mujer(X).

es_abuelo(X,Z) :- es_padre(X,U), es_padre(U,Z).

Prolog: Reglas Recursivas

Con estas tres nuevas relaciones entre objetos y los hechos de base anteriores podemos crear el siguiente Programa 2.

Programa 2=Programa 1+

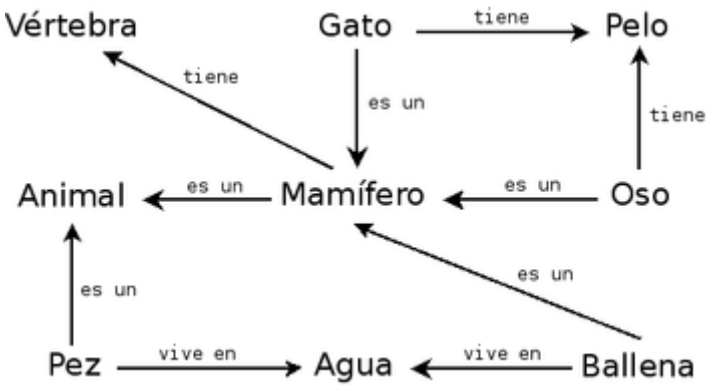
18. es_hijo(X,Y):- es_padre(Y,X), es_hombre(X).
19. es_hija(X,Y):- es_padre(Y,X), es_mujer(X).
20. es_abuelo(X,Z):- es_padre(X,Y), es_padre(Y,Z).

PREGUNTA	SIGNIFICADO	RESPUESTA
?es_hijo(lot, haran).	¿es hijo lot de Haran?	Si, pues este hecho puede deducirse, ya que según la regla 18 es equivalente a preguntar: ?es_padre(haran,lot), es_hombre(lot).
?es_hija(X,haran).	¿existe un X tal que es hija X de Haran?	Según la regla 19 es equivalente a preguntar ? es_padre(haran,X), es_mujer(X). por lo que el programa contestará que si, dando las soluciones: X=milcah ; X=yiscah.

PREGUNTA	RESPUESTA
?es_abuelo(terach,isaac).	si
?es_abuelo(terach,X9,es_mujer(X).	X=milcah ; X=yiscah
?es_abuelo(terach,X), es_mujer(X).	X=sarah Y=isaac
?es_abuelo(X,Y), es_hombre(Y).	X=terach Y=isaac ; X=terach Y=lot

Redes Semánticas

- Es una representación grafica del conocimiento
- Se usa para definir un concepto a partir de la definición de otro concepto
- Se usan grafos donde los conceptos son nodos y los arcos las relaciones entre ellos.
- Existen mecanismos para transformar esta representación en lógica de predicado de 1er orden



Base de las ontologías