

Curso de Inteligencia Artificial:

Representación del Conocimiento

Jose Aguilar

Cemisid, Facultad de Ingeniería

Universidad de los Andes

Mérida, Venezuela

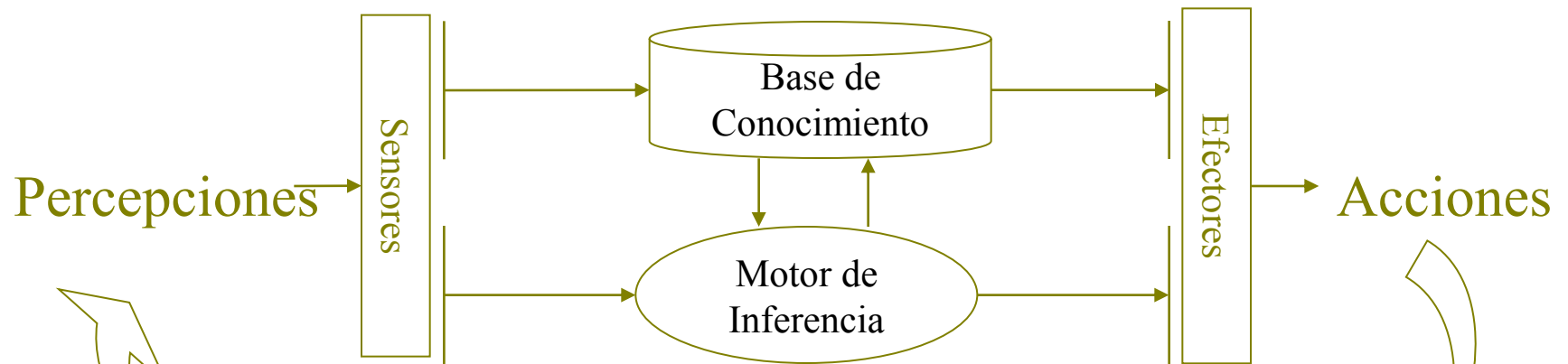
aguilar@ula.ve

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

Agente



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.**

diseñar

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

- **Nivel lógico.**

implementar

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

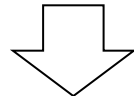
- **Nivel de implementación.**

usar

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

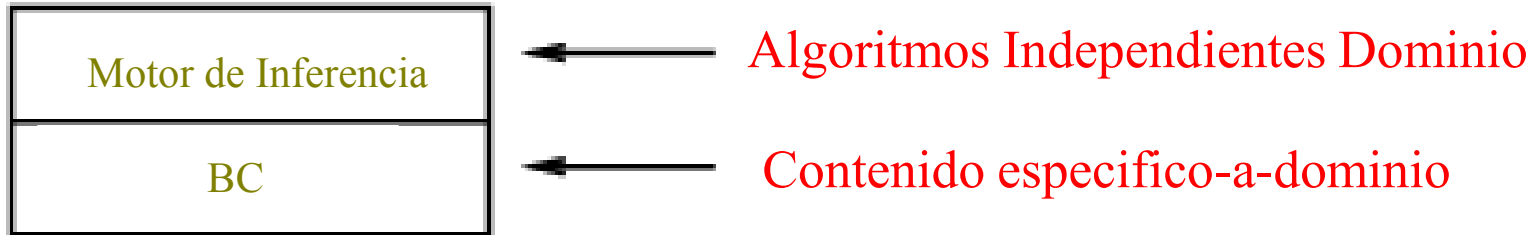
Base de Conocimiento (KB)

- **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



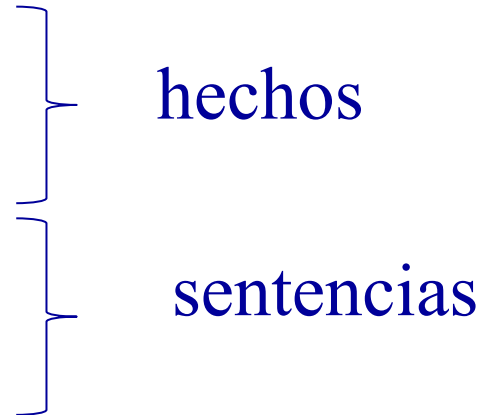
- **Ingeniería de Conocimiento**
 - Diseño y desarrollo de BC
 - Su objetivo es extraer, articular e informatizar el conocimiento de un experto.
 - Escritura de Sentencias que representan el conocimiento adquirido
- **Ontología**

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



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) \Rightarrow 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 intencionalidad.

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, intenciones 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 crítica
 - ⇒ **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

- Un Agente simple que razona acerca de las acciones implica:

- 1) Una *Medida de Rendimiento*
- 2) Una *secuencia percibida*
- 3) Lo que *conoce* del ambiente
- 4) Las *acciones* que puede ejecutar

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.



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.

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

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

Inferencia	Deductiva o analítica	
	Sintética	Inducción
		Hipótesis

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 27 probable, es siempre una información incierta y discutible.

Razonamiento

- **Deducción (Específico):**

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	$\forall x \text{ hombre}(x) \rightarrow \text{mortal}(x)$	Implicación (sentencia)
2	Juan es hombre	$\text{hombre}(\text{Juan})$	hecho
3	Juan es mortal	$\text{mortal}(\text{Jean})$	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 hipótesis (2), entonces necesitamos encontrarla (**abducción**, donde encontramos las causas de las consecuencias usando reglas. Típicamente 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

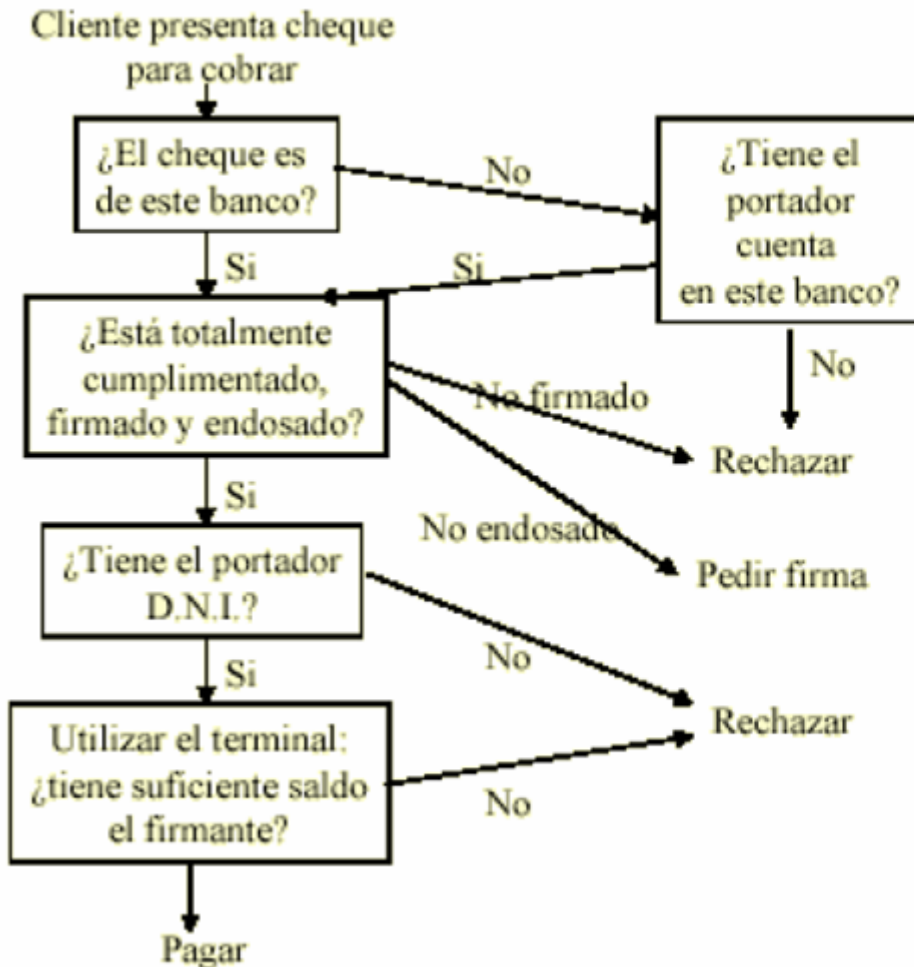
Representación del Conocimiento

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**

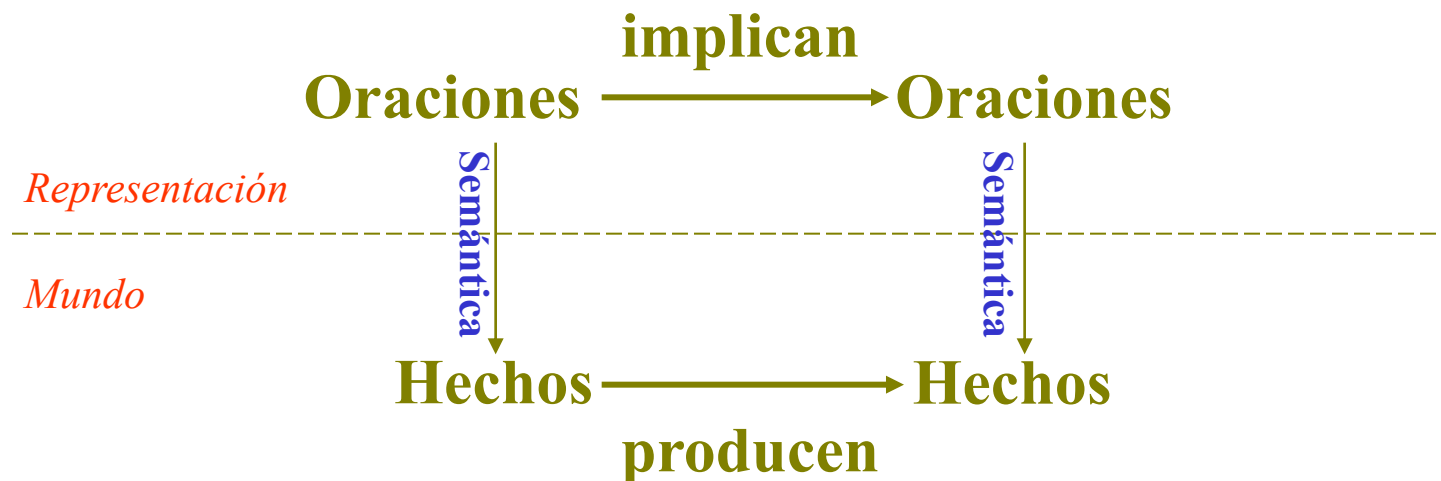
- Calculo Proposicional (hechos) => V/F/D
- Calculo de Predicado de Primer Orden (Hechos, Objetos, Relaciones) => V/F/D
- Lógica Temporal (hechos, Objetos, Relaciones y tiempo) => V/F/D
- Teoría de probabilidad (hechos) => Grado de Creer
- Lógica Difusa (Grados de Verdad) => Grado de Creer

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

$$\Delta \quad | \quad =_{\rho} \quad \omega$$

desde Δ se obtiene ω

ρ : reglas de inferencia

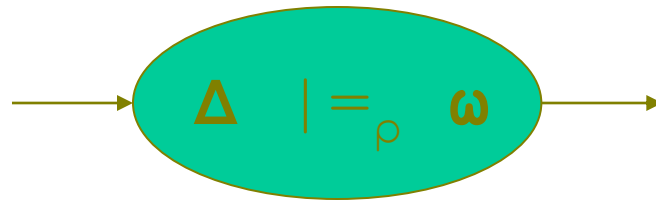
Δ : conjunto de fórmulas bien formada

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)



Modus Ponens

- **Modus ponendo ponens** (Latín: *modo que afirmando afirma*) es una regla de inferencia simple:

Si P entonces Q.

Expresado en la notación de operadores lógicos:

$$P \rightarrow Q \text{ o } P \vdash Q$$

- **Modus tollendo ponens** (Latín: *modo que negando afirma*) es una regla de inferencia simple:

Es verdad P entonces $\neg Q$.

Expresado en la notación de operadores lógicos:

$$P \rightarrow \neg Q \text{ o } P \vdash \neg Q$$

Reglas de inferencia

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

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

Eliminación-y : $\frac{a_1 \wedge a_2 \wedge \dots \wedge a_n}{a_i}$

Introducción-y: $\frac{a_1, a_2, \dots, a_n}{a_1 \wedge a_2 \wedge \dots \wedge a_n}$

Introducción-o: $\frac{a_i}{a_1 \vee a_2 \vee \dots \vee a_n}$

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

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

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

Calculo Proposicional

- Gramática

Sentencia \rightarrow Sentencia Atomica | Sentencia Compleja

Sentencia Atomica \rightarrow Verdad | Falso | P | Q, ...

Sentencia Compleja \rightarrow (Sentencia) | Sentencia conexión

Sentencia | \neg Sentencia

Conexión \rightarrow \wedge | \vee | \Rightarrow | \Leftrightarrow

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:

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

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**

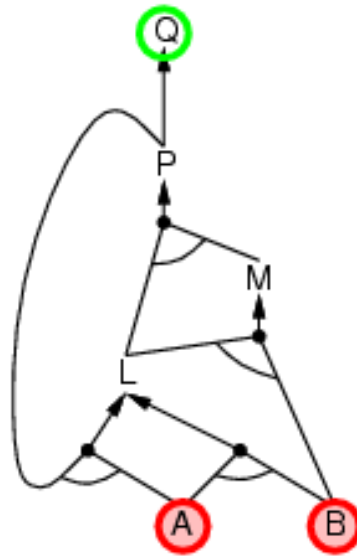
- $\neg S$ es verdad sii S es falso
- $S_1 \wedge S_2$ es verdad sii S_1 es verdad Y S_2 es verdad
- $S_1 \vee S_2$ es verdad sii S_1 es verdad O S_2 es verdad
- $S_1 \Rightarrow S_2$ es verdad sii S_1 es verdad entonces S_2 es verdad
o, es verdad sii S_1 es falso entonces S_2 es falso
- $S_1 \Leftrightarrow S_2$ es verdad sii $S_1 \Rightarrow S_2$ es verdad Y $S_2 \Rightarrow S_1$ es verdad

Calculo Proposicional

- Y-Eliminación $\alpha_1 \wedge \alpha_2 \wedge \alpha_3 \wedge \dots / \alpha_i$
- Y-Introducción $\alpha_1, \alpha_2, \alpha_3, \dots / \alpha_1 \wedge \alpha_2 \wedge \alpha_3 \wedge \dots$
- O-Introducción $\alpha_i / \alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \dots$
- Doble Negación Eliminación $\neg \neg \alpha / \alpha$
- Conmutación (para \vee y \wedge)
- Asociatividad (para \vee y \wedge)
- Distribución $(\alpha \vee (\beta \wedge \gamma)) = ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$
- Morgan $\neg (\alpha \vee \beta) = (\neg \alpha \wedge \neg \beta)$ (igual \wedge)

Hacia Adelante

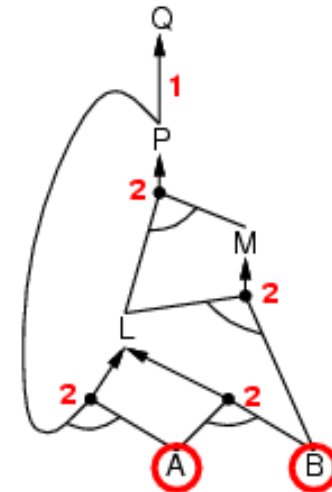
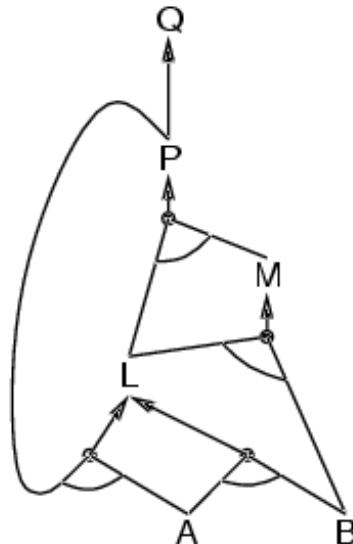
- **Idea:** activar reglas cuyas premisas son satisfechas en la *BC*
 - Añadir sus conclusiones a la *BC*, hasta que consulta es conseguida
- **Es orientada a Datos.**
- Usada para **reconocimiento de objetos**



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,**
- Apropiaada 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 proposiciones, 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

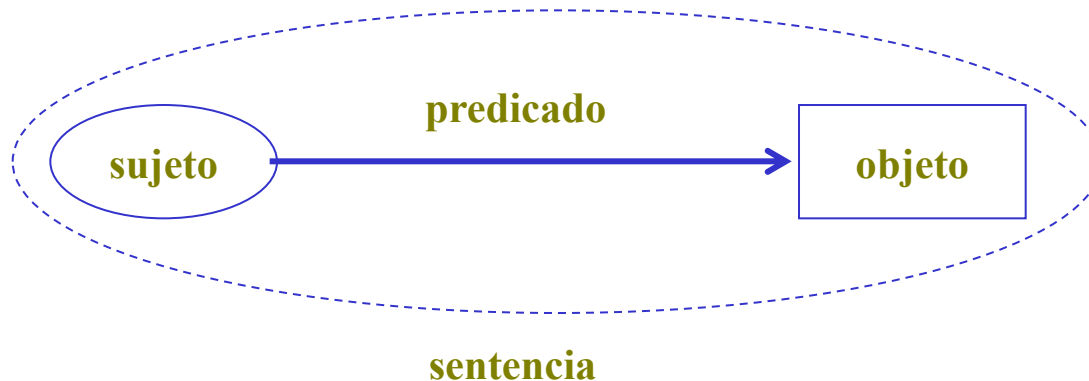
Un predicado es lo que se afirma del sujeto.

Predicado.

- Propiedades
- Cualidades
- Relaciones
- Atributos.
- Funciones

Sujeto.

- Términos, Cosas, Personas, Conceptos



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)
 \Rightarrow (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 \rightarrow SentenciaAtomica | SentenciaCompleja |

Cuantificador Variable Sentencia

SentenciaAtomica \rightarrow Predicado (termino, ...)

SentenciaCompleja \rightarrow (Sentencia) | Sentencia conexión

Sentencia | \neg Sentencia

Conexión $\rightarrow \wedge$ | \vee | \supset | \Leftrightarrow

Termino \rightarrow Constante | Variable | función(termino)

Cuantificador $\rightarrow \exists$ | \forall

Sintaxis LPPO

El **alfabeto** está formado por:

- **Sentencia atómica:**

predicado (término,)
termino = término

- **Sentencias:**

\neg sentencia
sentencias_atómicas.
(sentencia conectiva sentencia)

cuantificador variable,,
sentencia

- **Término:**

función
constante
variable

- **Símbolos de conectivas:**

(\wedge , \vee , \Leftrightarrow , \Rightarrow , \Uparrow , \neg)

- **Cuantificador universal:**

\forall (para todos)

- **Cuantificador existencial:**

\exists (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) \wedge Hermano (Juan, Jose)

Mayor (Juan, 30) \vee Menor (Juan, 40)

Mayor (Juan, 30) $\Rightarrow \neg$ Menor (Juan, 30)

\neg 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) \Rightarrow \neg MasJoven(Juan,30)

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

$\exists x$ Perro(x) \Rightarrow Amamanta(x) (existencial)

$\forall x$ Hermano(x, Juan) \Rightarrow Estudia (x) (universal)

$\forall x$ (Estudia(x) Y ($\exists x$ (Hermano(Jose ,x))))

General: \langle cuantificador \rangle \langle variable \rangle \langle sentencia \rangle

Cuantificación universal (\forall)

- Facilita la expresión de **reglas generales**, ejemplo: en vez de decir “Mancha es un gato” y “Mancha es un mamífero” se usa:
 - $\forall x \text{ Gato}(x) \Rightarrow \text{Mamífero}(x)$
- Lo cual equivale a
 - $\text{Gato}(\text{Mancha}) \Rightarrow \text{Mamífero}(\text{Mancha}) \wedge \text{Gato}(\text{Rebeca}) \Rightarrow \text{Mamífero}(\text{Rebeca}) \wedge \text{Gato}(\text{Félix}) \Rightarrow \text{Mamífero}(\text{Félix}) \wedge \text{Gato}(\text{Juan}) \Rightarrow \text{Mamífero}(\text{Juan}) \wedge \dots$
- Por lo tanto, la primera expresión será válida si y sólo si todas estas últimas son también verdaderas, es decir, si P es verdadera para todos los objetos x del universo.
 - a \forall se le conoce como **cuantificador universal**.

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 $\forall y \exists x$
 - $\exists x \forall y \text{ Quiere}(x,y)$
“hay una persona quien ama a c/u en el mundo”
 - $\forall y \exists x \text{ Quiere}(x,y)$
“Cada quien en el mundo es amado por al menos una persona”
- **Cuantificador de dualidad:** uno puede ser expresado usando el otro
 - $\forall x \text{ LeGusta}(x, \text{Helado})$ $\neg \exists x \neg \text{LeGusta}(x, \text{Helado})$
 - $\exists x \text{ LeGusta}(x, \text{Cebolla})$ $\neg \forall x \neg \text{LeGusta}(x, \text{Cebolla})$

Lógica de Predicado de Primer Orden

- Alto Orden Lógico

$$\forall x,y (x=y) \Leftrightarrow \forall p p(x)=p(y)$$

- Operador λ

$$\lambda x,y x-y$$

$$\lambda 25, 25 \Rightarrow 0 \text{ (hace la operación)}$$

- Cuantificador $\exists!$

$$\exists! x \text{ Rey}(x)$$

- Axiomas: capturan los hechos básicos
- Dominio: coherente cuerpo de conocimiento sobre algo

Lógica de Predicado de Primer Orden

- Simple Agente Reflexivo

$\forall a \text{ Percibo}(a, t) \Rightarrow \text{Acción}(\text{recoger}, a)$

- Otros Operadores

$\text{Ir}(\text{persona}, \text{sitio})$

$\text{En}(\text{posición}, \text{tiempo}, \text{Jose})$

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

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

LPPPO: Mas ejemplos

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

Oso(Juan)

$\forall b \text{ Oso}(b) \Rightarrow \text{Animal}(b)$

$\forall a \text{ Animal}(a) \Rightarrow \text{CosaFisica}(a)$

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

$\forall a \text{ Animal}(a) \Leftrightarrow \text{Cerebrode}(a)$

$\forall a \text{ Partede}(a, \text{Cerebrode}(a))$

- Carlos tiene cerebro pequeño

$\text{Tamano}(\text{Cerebrode}(\text{carlos}), \text{cerebrode}(\text{tipicohumano})) = \text{pequeño}$

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**)

$\forall I1,I2,s \quad En(I1,s, jose) \wedge Camino(I1,I2) \Rightarrow En(I2, s+t, jose)$

- Regla Diagnostico (**Generalizo**)

$\forall x \quad En(Florida, Agosto, x) \wedge LlueveFuerte(Florida) \Rightarrow$
 $Tormenta(Florida)$

- **Abducción**

$\forall x \quad lluevesobre(x) \Rightarrow Mojado(x)$
 $Percibe(mojado(x)) \Rightarrow Hipótesis (ha llovido)$

Lógica de Predicado de Primer Orden

- Preferencias entre acciones

BC { $\forall a,s \text{ Grande}(a, s) \text{ Y } \neg \exists b \text{ Grande}(b,s) \Rightarrow \text{acción}(a)$
 $\forall a,s \text{ Mediano}(a, s) \text{ Y } \neg \exists b (\text{Grande}(b,s) \text{ O } \text{Mediano}(b,s))$
 $\Rightarrow \text{acción}(a)$

- Agentes con Objetivos

$$\forall s \text{ Quiere}(\text{Oro}, s) \Rightarrow \dots$$

- Requiere:

Inferir

Buscar

Planificar

Uso Lógica de Predicado de Primer Orden

- **BC:**

$\forall x \text{ Rey}(x) \wedge \text{Codicioso}(x) \Rightarrow \text{Malo}(x)$

$\text{Rey}(\text{Juan})$

$\text{Codicioso}(\text{Juan})$

$\text{Hermano}(\text{Richard}, \text{Juan})$

- **Instanciando BC:**

$\text{Rey}(\text{Juan}) \wedge \text{Codicioso}(\text{Juan}) \Rightarrow \text{Malo}(\text{Juan})$

Lógica de Predicado de Primer Orden

- Otro Ejemplo

Misil(M1)

Dueno(Carlos,M1)

$\forall x \text{ Misil}(x) \text{ y Dueno}(\text{Carlos}, x) \Rightarrow \text{Vender}(\text{Oeste}, \text{Carlos}, x)$

y se puede inferir

Vender(Oeste,Carlos,M1)

Lógica de Predicado de Primer Orden

- **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, $\forall m,c$ Madre(c)=m \Leftrightarrow 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

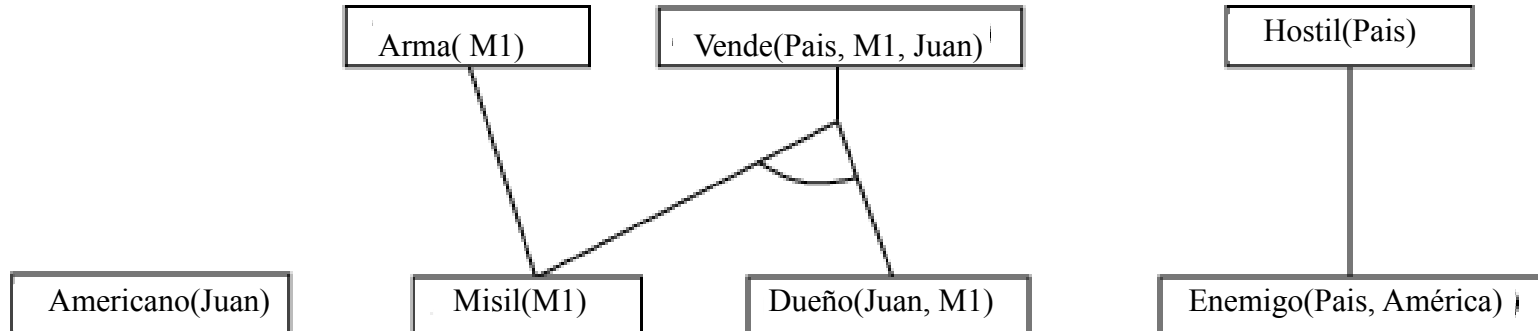
Americano(Juan)

Misil(M1)

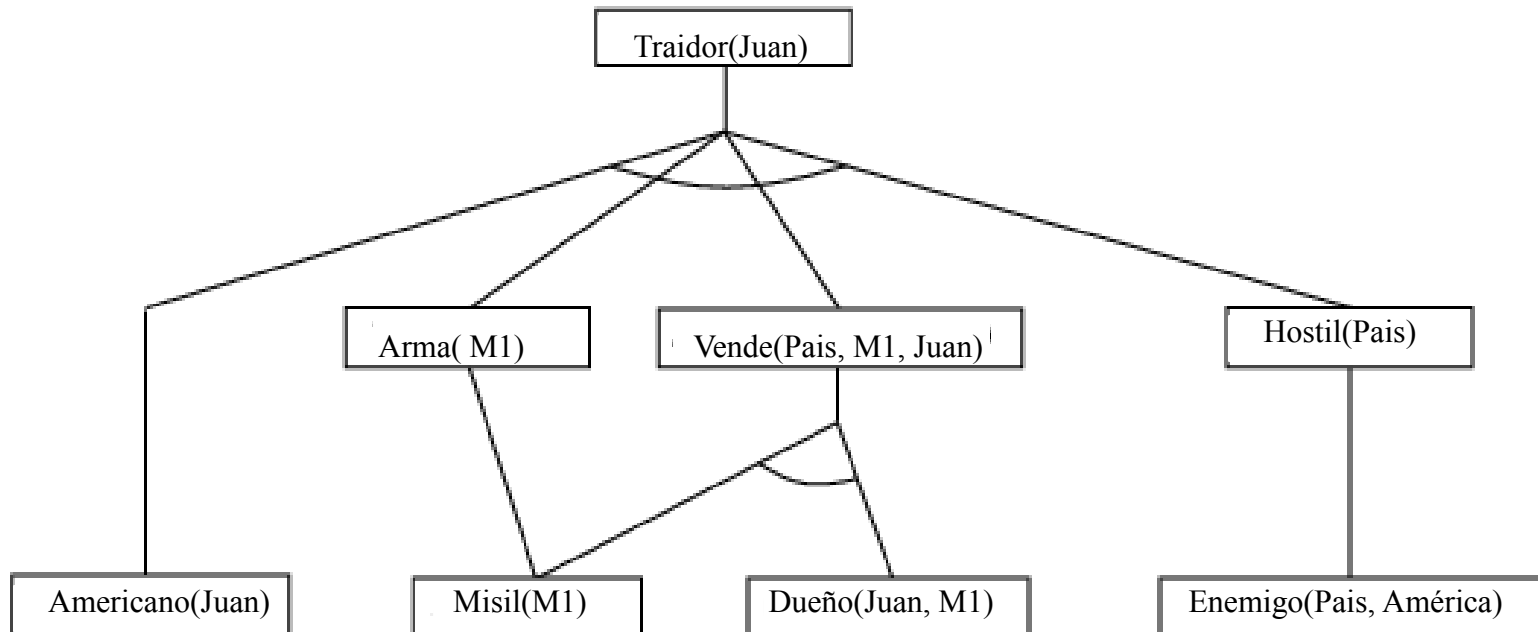
Dueño(Juan, M1)

Enemigo(Pais, América)

Hacia Adelante



Hacia Adelante



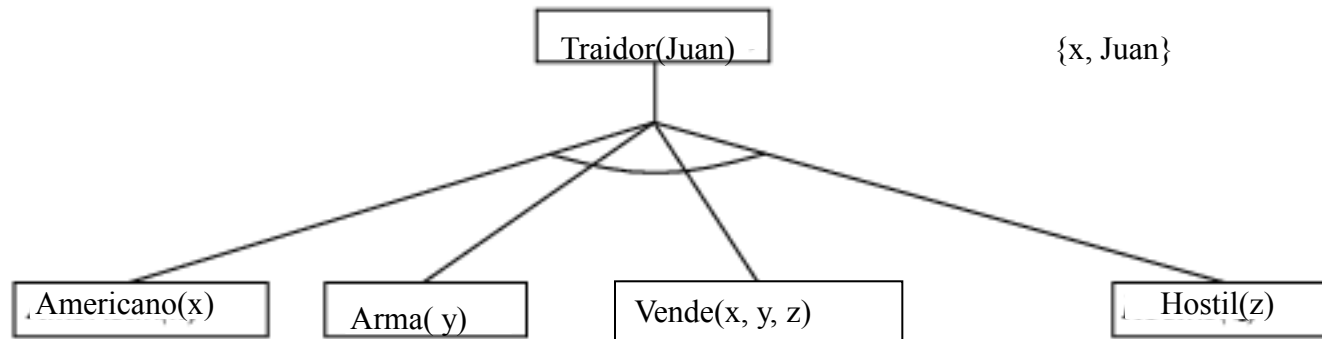
Hacia Adelante

```
function FOL-FC-ASK( $KB, \alpha$ ) returns a substitution or false
  repeat until new is empty
     $new \leftarrow \{ \}$ 
    for each sentence  $r$  in  $KB$  do
       $(p_1 \wedge \dots \wedge p_n \Rightarrow q) \leftarrow \text{STANDARDIZE-APART}(r)$ 
      for each  $\theta$  such that  $(p_1 \wedge \dots \wedge p_n)\theta = (p'_1 \wedge \dots \wedge p'_n)\theta$ 
        for some  $p'_1, \dots, p'_n$  in  $KB$ 
           $q' \leftarrow \text{SUBST}(\theta, q)$ 
          if  $q'$  is not a renaming of a sentence already in  $KB$  or new then do
            add  $q'$  to new
             $\phi \leftarrow \text{UNIFY}(q', \alpha)$ 
            if  $\phi$  is not fail then return  $\phi$ 
    add new to  $KB$ 
  return false
```

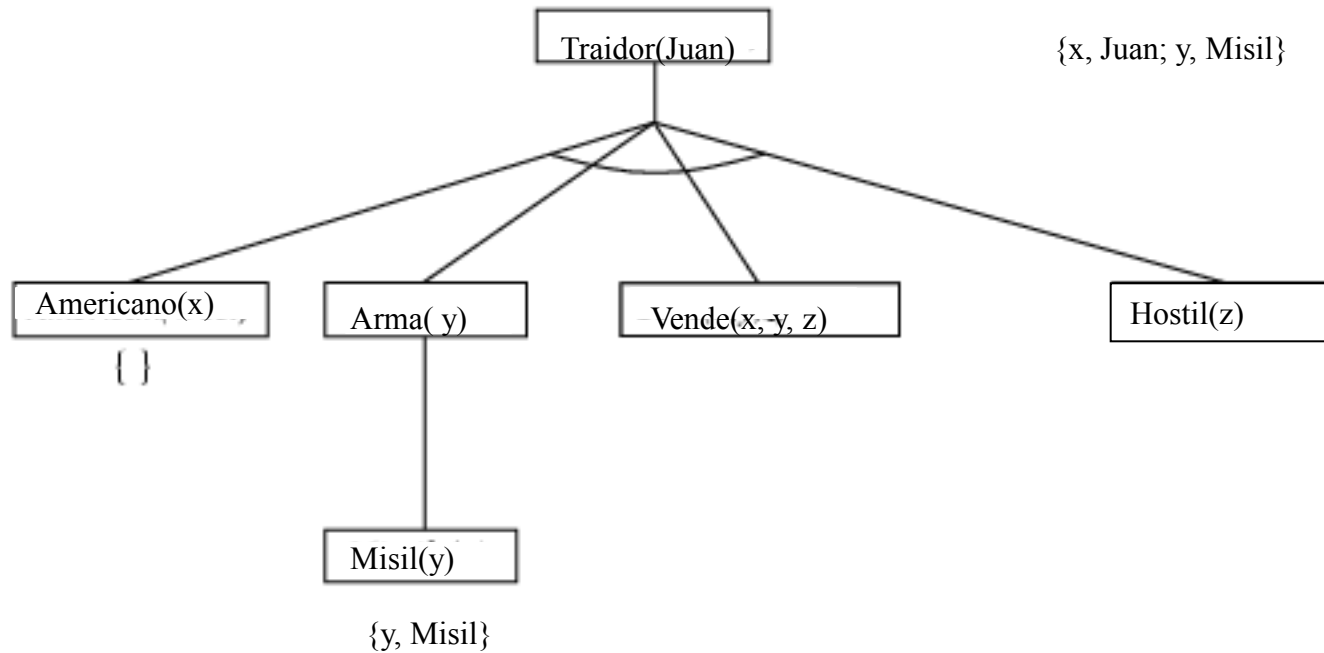
Hacia Atrás

-Traidor(Juan) *

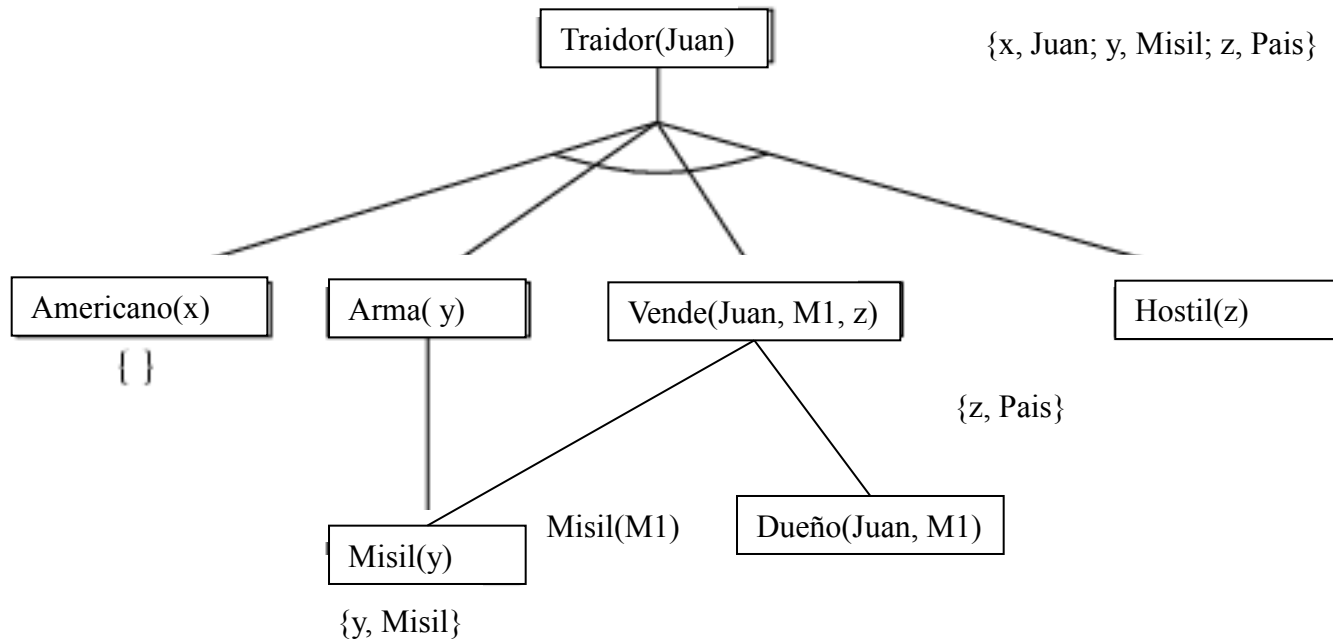
Hacia Atrás



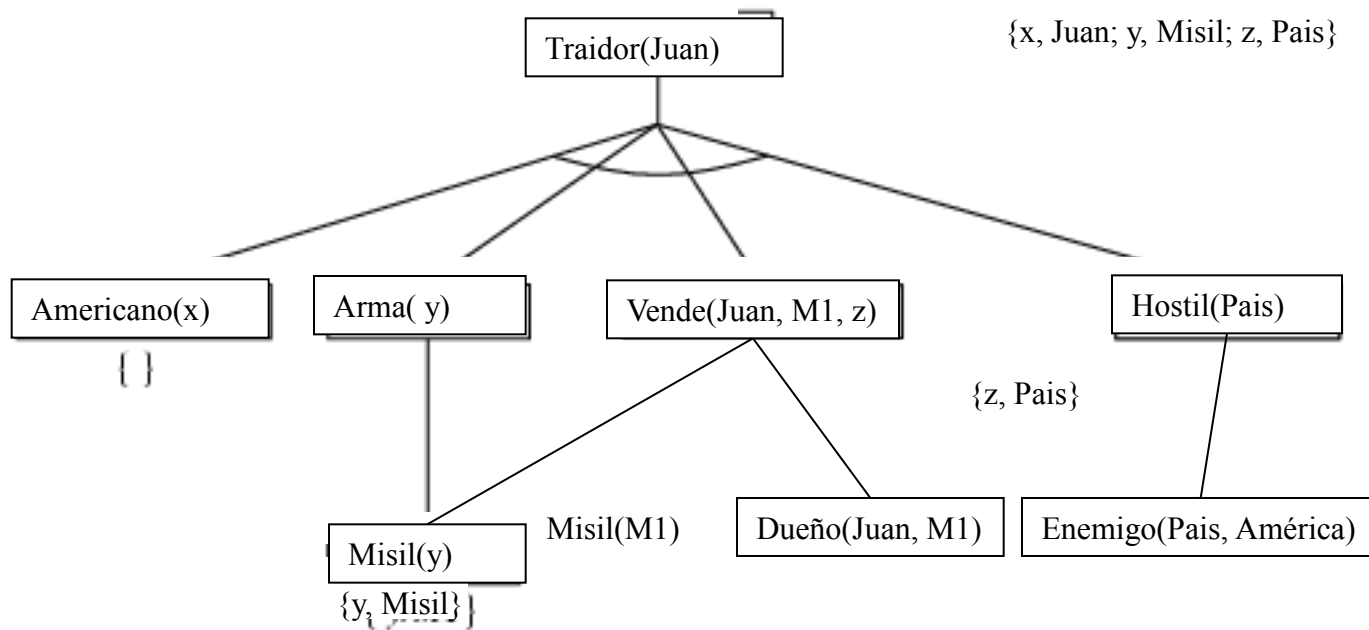
Hacia Atrás



Hacia Atrás



Hacia Atrás



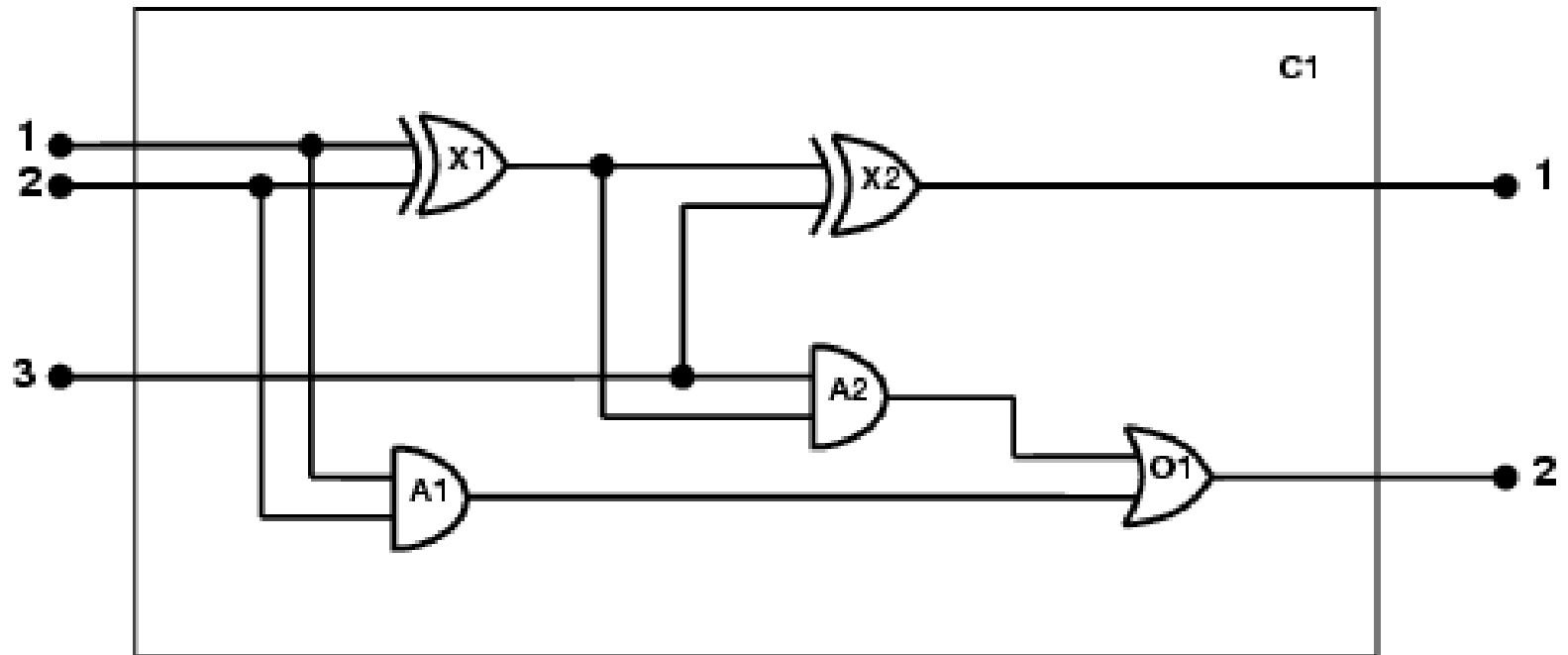
Hacia Atrás

```
function FOL-BC-ASK(KB, goals,  $\theta$ ) returns a set of substitutions
  inputs: KB, a knowledge base
            goals, a list of conjuncts forming a query
             $\theta$ , 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( $\theta$ , FIRST(goals))
  for each r in KB where STANDARDIZE-APART(r) = ( $p_1 \wedge \dots \wedge p_n \Rightarrow q$ )
    and  $\theta' \leftarrow$  UNIFY(q,  $q'$ ) succeeds
     $ans \leftarrow$  FOL-BC-ASK(KB, [ $p_1, \dots, p_n$  | 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: $\text{Out}(1, X_1)$

- Funciones:

$\text{Type}(X_1) = \text{XOR}$ o $\text{Type}(X_1, \text{XOR})$

$\text{Connected}(\text{Out}(1, X_1), \text{In}(1, X_2))$

Diseño de circuitos electrónicos

4. Codificar conocimiento

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

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?

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

7. Depurar la BC

Eliminar sentencias, determinar sentencias redundantes, ... J Aguilar

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 $\text{head} \text{ :- literal}_1, \dots \text{literal}_n.$

- Ejemplo:

$\text{traidor}(X) \text{ :- americano}(X), \text{cañón}(Y), \text{vende}(X, Y, Z), \text{hostil}(Z)$

Prolog: Programación Lógica

Ejemplos

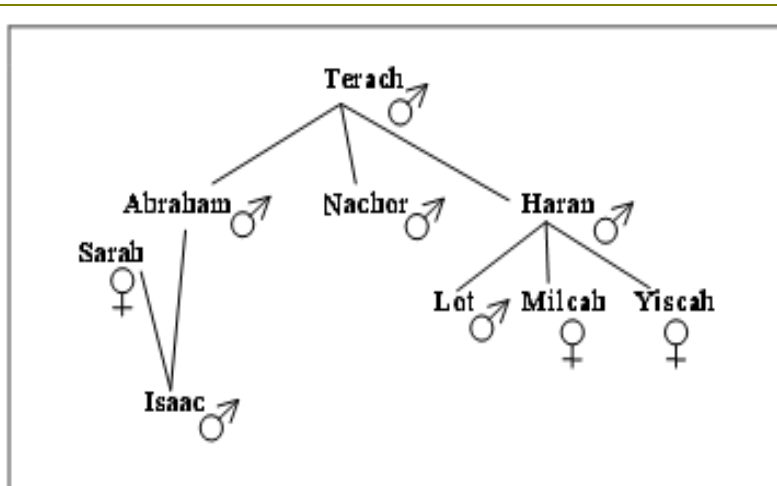
```
precio( boli, 0.5 ).  
precio( folios, 2.5 ).  
precio( ordenador, 700 ).  
precio( coche, 15000 ).  
precio( tomates, 1 ).  
  
%% ¿Qué puedo comprar con 100 euros?  
?- precio( X,Y ), Y < 100.  
X=boli, Y=0,5;  
X=folios, Y=2,5;  
X=tomates, Y=1;  
No.
```

```
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.
```

hechos

sentencias

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, Lot)

¿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 si 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 si , 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: Reglas Recursivas

Las reglas son sentencias de la forma: $A :- B_1 , \dots , B_n$. donde A y cada B_i son átomos. A es la cabeza de la regla y los B_i '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).

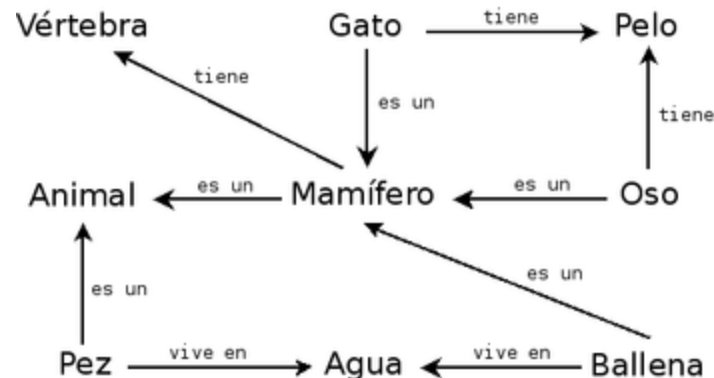
20. es_abuelo(X,Z):- es_padre(X,Y), es_padre(Y,Z).

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=milcab ; X=yiscab.

PREGUNTA	RESPUESTA
?es_abuelo(terach,isaac).	si
?es_abuelo(terach,X),es_mujer(X).	X=milcab ; X=yiscab
?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 gráfica 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

Lógica 1er Orden y Redes Semánticas

- A subconjunto B $A \subset B$
gato \subset mamíferos
- A miembro B $A \in B$
Kitty \in gato
- A Relacionado B $R(A,B)$
Kitty edad 12
- A Relacionado B $\forall x x \in A \Rightarrow R(x,B)$
Kitty tiene 4 patas
- A Relacionado B $\forall x \exists y x \in A \Rightarrow y \in B \text{ y } R(x,y)$
Kitty padre Katty