Tema 5. Comportamiento inteligente: Representación del Conocimiento e inferencia basados en lógica

- Representación del conocimiento en IA.
- El cálculo proposicional.
- Resolución en el cálculo proposicional.
- Cálculo de predicados.
- Resolución en el cálculo de predicados.
- Introducción a los Sistemas Basados en el Conocimiento.
- Sistemas Basados en el Conocimiento.

Objetivos

- Adquirir las habilidades básicas para construir sistemas capaces de resolver problemas mediante técnicas de IA.
- Entender que la resolución de problemas en IA implica definir una representación del problema y un proceso de búsqueda de la solución.
- Comprender la necesidad de representar el conocimiento y realizar inferencia para que un sistema pueda exhibir comportamiento inteligente.
- Conocer los fundamentos de la representación del conocimiento en lógica proposicional y de predicados y sus mecanismos de inferencia asociados.
- Aplicar los aspectos de representación basada en la lógica y mecanismos de inferencia, mediante técnicas y herramientas de programación lógica.

Estudia este tema en …

- Nils J. Nilsson, "*Inteligencia Artificial: Una nueva síntesis*", Ed. Mc Graw Hill, 2000. pp. 215-284

Representación del conocimiento en IA

- Hemos estudiado varias formas de modelar el mundo de un agente, entre ellas:
  - **Representaciones icónicas:** Simulaciones del mundo que el agente podía percibir.
  - **Representaciones descriptivas:** Valores binarios que describían aspectos ciertos o falsos sobre el mundo.
- Las representaciones descriptivas tienen ciertas ventajas sobre las icónicas:
  - Son más sencillas.
  - Son fáciles de comunicar a otros agentes.
  - Se pueden descomponer en piezas más simples.

Representación del conocimiento en IA

- Además, hay información del entorno del agente que no se puede representar mediante modelos icónicos, tales como:
  - **Leyes generales**. “Todas las cajas azules pueden ser cogidas”.
  - **Información negativa**. “El bloque A no está en el suelo”, sin decir dónde está el bloque A.
  - **Información incierta.** “O bien el bloque A está sobre el bloque C, o bien el bloque A está sobre el bloque B”.
- Sin embargo, este tipo de información es fácil de formular como conjunto de restricciones sobre los valores de las características binarias del agente.
- Estas restricciones representan **conocimiento sobre el mundo**.

Representación del conocimiento en IA

- A menudo, este **conocimiento sobre el mundo** puede utilizarse para razonar sobre él y hallar nuevas características del mismo.
  Ejemplo:
  - El conocimiento que se tiene es “Los hombres aman a las mujeres”; y “Juan es un hombre”.
  - Se puede ***razonar***, por tanto, que “Juan ama a las mujeres”.
- **Otro Ejemplo:** Un robot sólo puede levantar un bloque si tiene suficiente batería y el bloque es elevable. Entonces, el conocimiento sobre el mundo es: “Si el bloque es elevable y hay suficiente batería, entonces es posible levantar el bloque”.
- El robot “sabrá” si es capaz de levantar el bloque a partir de este **conocimiento** sobre su entorno.

Representación del conocimiento en IA

- Estudiaremos 2 tipos básicos para representar el conocimiento y razonar sobre él:
  - Cálculo proposicional.
  - Cálculo de predicados.

Cálculo Proposicional

- Distinguiremos entre los siguientes elementos del lenguaje del cálculo proposicional:
  - **Átomos:** **V** (verdadero) y **F** (falso), las proposiciones (en mayúscula) **P, Q, P1, P2, SOBRE_A_B**, etc.
  - **Conectivas:** “y”, “o”, “implica”, “no”, representadas como:

$$\wedge\textbf{(y)}, \vee\textbf{(o)}, \supset\textbf{(implica)}, \neg\textbf{(no)}$$

  - **Sentencias (Fórmulas Bien Formadas, FBF)**:
    - Cualquier átomo es una FBF.
    - Si A y B sobre FBF, entonces también lo son:

$$\mathbf{A \wedge B, A \vee B, A \supset B, \neg A}$$

Cálculo Proposicional

- En las implicaciones, el primer término se denomina **antecedente** y el segundo **consecuente**. Ejemplos de FBFs:

$$(P \wedge Q) \supset \neg P$$

$$P \supset \neg P$$

$$P \vee P \supset P$$

$$(P \supset Q) \supset (\neg Q \supset \neg P)$$

Reglas de inferencia

- Las **reglas de inferencia** nos permiten producir nuevas FBFs a partir de las que ya existen, Algunas de las más comunes son:
  - **Q** puede inferirse a partir de **P** y $\mathbf{P} \supset \mathbf{Q}$ (*modus ponens*)
  - $\mathbf{P} \wedge \mathbf{Q}$ se puede inferir a través de la conjunción de **P** y **Q**
  - $\mathbf{Q} \wedge \mathbf{P}$ se puede inferir desde $\mathbf{P} \wedge \mathbf{Q}$ (*conmutatividad*)
  - **P** (también **Q**) se puede inferir desde $\mathbf{Q} \wedge \mathbf{P}$
  - $\mathbf{P} \vee \mathbf{Q}$ se puede inferir bien desde **P**, bien desde **Q**
  - **P** se puede inferir desde $\neg(\neg\mathbf{P})$

Demostración

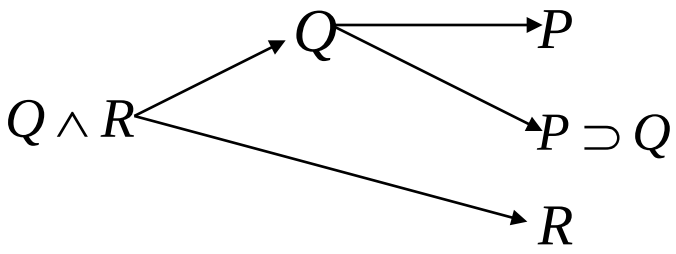
- Supongamos Δ un conjunto de FBFs, y una secuencia de **n** FBFs **$\{w_1, w_2, w_3, \ldots, w_n\}$**.
- Esta secuencia de FBFs se llama **demostración** o **deducción** de $\mathbf{w_n}$ a partir de Δ si, y sólo si, cada $\mathbf{w_i}$ de la secuencia pertenece a Δ o puede inferirse a partir de FBFs en Δ.
- Si existe tal demostración, entonces decimos que $\mathbf{w_n}$ es un **teorema de** Δ, y decimos que $\mathbf{w_n}$ puede demostrarse desde Δ con la siguiente notación: $\Delta \vdash \mathbf{w_n}$,
- o como $\Delta \vdash_{\mathbf{R}} \mathbf{w_n}$ para indicar que $\mathbf{w_n}$ se demuestra desde Δ mediante las reglas de inferencia **R**.

Demostración

- **Ejemplo:**
  - Sea el conjunto de FBFs Δ, $\Delta = \{P, R, P \supset Q\}$
  - Entonces, la siguiente secuencia es una demostración de la Fórmula Bien Formada $R \wedge Q$:

$$\{P, P \supset Q, Q, R, Q \wedge R\}$$

  - La demostración se puede llevar a cabo fácilmente a través del siguiente **árbol de demostración**, utilizando Δ y las reglas de inferencia:



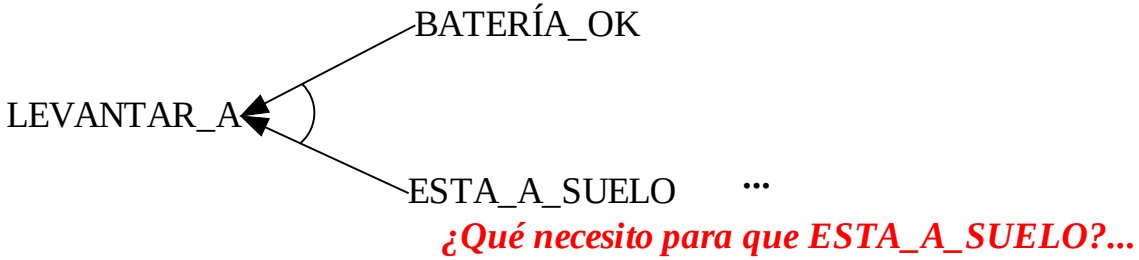
Interpretación

- A la hora de resolver problemas con IA, el papel de la semántica es esencial: Hay que hacer una correcta **interpretación** del sistema lógico subyacente.
- Conlleva asociar conceptos del lenguaje lógico con su significado (semántica) en el mundo real o en el mundo del entorno del agente.
- **Ejemplo:** Se desea implantar el conocimiento "Si la batería funciona y el bloque A está en el suelo, entonces se puede levantar" dentro de un agente.
  - Definimos los átomos ***BATERIA_OK***, ***ESTA_A_SUELO***, ***LEVANTAR_A***.
  - Definimos la FBF:

$$BATERIA\_OK \wedge ESTA\_A\_SUELO \supset LEVANTAR\_A$$

Interpretación

- En un agente orientado a metas, cuya meta sea "*levantar el bloque A*", con este conocimiento puede especificar las acciones que debe llevar a cabo para realizar su acción.
- Esta planificación se puede hacer mediante árboles de demostración.
- La representación de **grafos Y/O** es muy útil en este tipo de problemas.
- **Ejemplo:** "*Debo levantar el bloque A, ¿qué necesito para poder levantarlo?*"



Tablas de la verdad

- Las **tablas de verdad** establecen la semántica de las conectivas proposicionales.
- Para una representación interna de un agente con **n** características, el número de combinaciones (formas de ver el mundo) es $\mathbf{2^n}$.
- Para dos características **A** y **B**:

A	B	A y B	A o B	No A	A implica B
V	V	V	V	F	V
V	F	F	V	F	F
F	V	F	V	V	V
F	F	F	F	V	V

Satisfacibilidad y Modelos

- Una **interpretación satisface una FBF** cuando a la FBF se le asocia el valor **V** bajo esa interpretación.
- A la interpretación que satisface una FBF se le denomina **modelo**.
- Bajo una interpretación una FBF debe indicar una restricción que nos informe sobre algún aspecto del mundo. **Ejemplo:** $A \wedge B \supset C$
  - Para la interpretación A=BATERIA_OK, B=ESTA_A_SUELO, C=LEVANTAR_A, la semántica se corresponde con el entorno y el mundo a modelar.
  - Para la interpretación A=TOCA_LOTERIA, B=TENGO_SALUD, C=TIRAR_POR_VENTANA, la semántica es inconsistente con lo que se modela. **Esta interpretación no es válida porque no satisface la FBF.**

Validez y equivalencia

- Se dice que una FBF es **válida** si se cumple independientemente de la interpretación que se le asocie. Ejemplo: $P \supset P, \neg(P \wedge \neg P)$
- Las interpretaciones válidas no modelan aspectos del mundo y deben ser evitadas en el diseño del comportamiento del agente.

- Dos FBFs son **equivalentes** si sus tablas de verdad son idénticas. Ejemplo:$\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
- En el diseño de agentes, debemos evitar FBFs con interpretaciones equivalentes para hacer más eficiente el proceso de razonamiento.

Resolución en el cálculo proposicional

- **Claúsulas como FBFs:**
  - **Un Literal** es un átomo o su negación: Ejemplo: **P**, **¬P**
  - **Una cláusula Σ** es un conjunto de literales. Ejemplo: **Σ={P,Q,¬R}**
  - Una cláusula es equivalente a la disyunción de sus elementos. Por tanto, la cláusula **Σ** anterior equivale a: $\mathbf{P \vee Q \vee \neg R}$
- **Idea general de la resolución:**
  - Siendo **λ** un literal y $\mathbf{\Sigma_1, \Sigma_2}$ dos cláusulas, a partir de $\mathbf{\{\lambda\} \cup \Sigma_1}$ y $\mathbf{\{\neg\lambda\} \cup \Sigma_2}$ se puede inferir $\mathbf{\Sigma_1 \cup \Sigma_2}$.
- Ejemplo: Resolviendo $R \vee P, \neg P \vee Q$ se obtiene $R \vee Q$
- Denominaremos **Nil** a la cláusula vacía **Σ={}**

Resolución en el cálculo proposicional

- **Cualquier FBFs en el cálculo proposicional puede escribirse como conjunción de cláusulas.**
- Si una FBF está escrita como conjunción de cláusulas, se dice que está en **FNC** (Forma Normal Conjuntiva).
- Si una FBF está escrita como disyunción de cláusulas, se dice que está en FND (Forma Normal Disyuntiva).
- Para transformar una FBF a FNC se siguen 3 pasos:
  - 1. Eliminar implicaciones mediante su equivalente $A \supset B \equiv \neg A \vee B$
  - 2. Reducir el ámbito de aplicación de la negación ¬ mediante las leyes de DeMorgan $\neg(A \vee B) \equiv \neg A \wedge \neg B; \neg(A \wedge B) \equiv \neg A \vee \neg B$
  - 3. Usar las leyes asociativas y distributivas para pasar a FNC.

Resolución en el cálculo proposicional

- **Ejemplo:** $\neg(P \supset Q) \vee (R \supset P)$
  - 1. Eliminar implicaciones.

    $$\neg(\neg P \vee Q) \vee (\neg R \vee P)$$

  - 2. Reducir el ámbito de aplicación de la negación ¬.

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

  - 3. Usar las leyes asociativas y distributivas para pasar a FNC.
    - Distributiva: $(P \vee \neg R \vee P) \wedge (\neg Q \vee \neg R \vee P)$
    - Simplificamos: $(P \vee \neg R) \wedge (\neg Q \vee \neg R \vee P)$
  - **Solución:** El conjunto de cláusulas $\{(P \vee \neg R), (\neg Q \vee \neg R \vee P)\}$

Resolución en el cálculo proposicional

- **La refutación** es útil para demostrar que la negación de una cláusula es inconsistente en el sistema, quedando así demostrada, por tanto, la veracidad de dicha cláusula.
- La idea es que, si "algo" no se puede demostrar, entonces se demuestra que lo contrario de ese "algo" no puede ser cierto.
- **Se siguen 4 pasos p**ara refutar una FBF **w:**
  - 1. Convertir las FBFs de Δ como conjunciones de cláusulas
  - 2. Convertir $\neg$**w** como conjunción de cláusulas
  - 3. Unir el resultado de los pasos 1 y 2 en un único conjunto $\mathbf{\Gamma}$
  - 4. Aplicar la resolución a las cláusulas de $\mathbf{\Gamma}$ de forma iterativa, hasta que no haya nada más que resolver o se llegue a **Nil**

Resolución en el cálculo proposicional

- **Se podrán producir dos resultados:**
  - **Se llega a generar la cláusula vacía Nil**. El proceso de refutación termina con éxito y queda demostrada **w**.
  - **El proceso termina sin generar la cláusula vacía**. Entonces no se puede demostrar **w**.
- **Ejemplo:** Supongamos que en el mundo de los bloques tenemos que **BATERIA_OK** y que **¬ROBOT_SE_MUEVE.**
  - El conocimiento que se tiene es:

    $$BATERIA\_OK \wedge OBJETO\_ELEVABLE \supset ROBOT\_SE\_MUEVE$$

  - Queremos demostrar **w**= ¬OBJETO_ELEVABLE

Resolución en el cálculo proposicional

- **Ejemplo:**
  - 1. Convertir las FBFs de Δ como conjunciones de cláusulas

$a) \; BATERIA\_OK$

$b) \; \neg ROBOT\_SE\_MUEVE$

$c) \; \neg BATERIA\_OK \vee \neg OBJETO\_ELEVABLE \vee ROBOT\_SE\_MUEVE$

  - 2. Convertir **¬w** como conjunción de cláusulas

$$OBJETO\_ELEVABLE$$

Resolución en el cálculo proposicional

- **Ejemplo:**
  - 3. Unir el resultado de los pasos 1 y 2 en un único conjunto **Γ**

$$\Gamma = \{$$
$$a) \; BATERIA\_OK$$
$$b) \; \neg ROBOT\_SE\_MUEVE$$
$$c) \; \neg BATERIA\_OK \vee \neg OBJETO\_ELEVABLE \vee ROBOT\_SE\_MUEVE$$
$$d) \; OBJETO\_ELEVABLE$$
$$\}$$

Resolución en el cálculo proposicional

- **Ejemplo:**
  - 4. Aplicar la resolución a las cláusulas de Γ de forma iterativa, hasta que no haya nada más que resolver o se llegue a **Nil**
    - Resolviendo c) y d), tenemos que

$$e) \neg BATERIA\_OK \vee ROBOT\_SE\_MUEVE$$

    - Resolviendo e) y b), tenemos:

$$f) \neg BATERIA\_OK$$

    - Resolviendo f) y a), tenemos **Nil**.
    - Queda demostrado que ¬OBJETO_ELEVABLE

Resolución en el cálculo proposicional

- Como hemos visto, el procedimiento de refutación mediante resolución consiste en "*aplicar resoluciones hasta que se genere la cláusula vacía o no se puedan hacer más resoluciones*".
- La selección de cláusulas para su resolución, de forma manual, es sencilla.
- Sin embargo, si queremos elaborar un procedimiento automático que implemente este método, ¿cómo debemos seleccionar las cláusulas a resolver?

Resolución en el cálculo proposicional

- ¿En qué orden se deben realizar las resoluciones para obtener **Nil** con la mayor brevedad?
  - Esta cuestión es análoga a la de qué nodo/estado hay que expandir en la búsqueda en espacios de estados (temas 2 y 3):
    - Estrategias de primero en anchura.
    - Estrategias de primero en profundidad.etc.
- Llamamos **resolvente de nivel 0** a las cláusulas iniciales, incluída la negación de la cláusula a demostrar.
- Una **resolvente de nivel i+1** es aquella que se obtiene tras resolver una resolvente de nivel **i** con una resolvente de nivel **j**, con **j<=i**.

Resolución en el cálculo proposicional

- **Estrategias de primero en profundidad.etc.**
  - En primer lugar se generaría una resolvente de nivel 1.
  - Luego se usaría la resolvente del nivel 1 con alguna del nivel 0 para generar una nueva resolvente de nivel 2.
  - Seguidamente, se usaría la resolvente de nivel 2 y alguna entre las resolventes de los niveles 1, 0 para generar la resolvente del nivel 3
  - Etc.
- Si fijamos un nivel de profundidad máximo, se puede aplicar una técnica de vuelta atrás para hacer la refutación.

Resolución en el cálculo proposicional

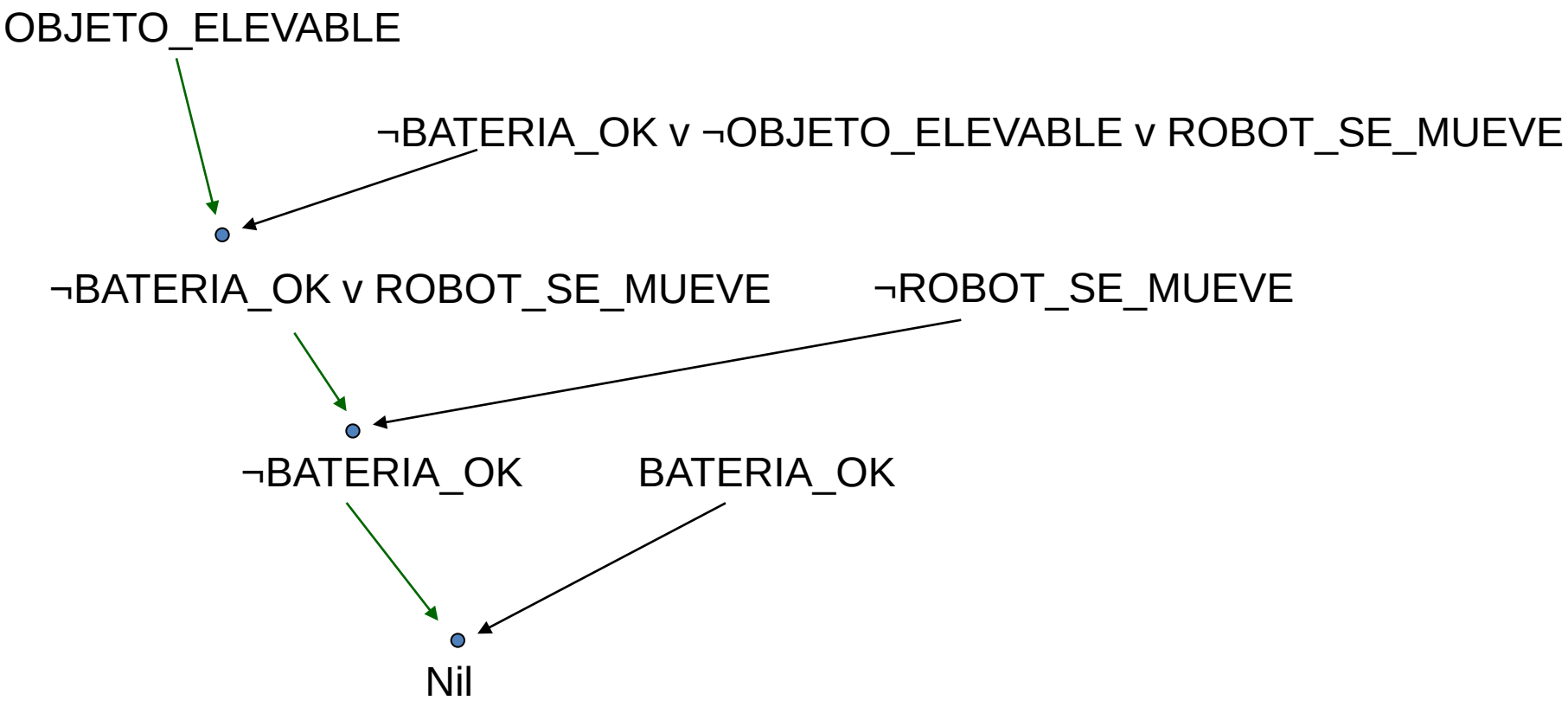
- **Estrategia de preferencia unitaria:**
  - Se da preferencia a resolver las cláusulas en las que, al menos una de las cláusulas a resolver, tienen un único literal.
  - Esta estrategia se puede generalizar en “resolver primero las más simples”, intentando que al menos una de las dos cláusulas a resolver sea unitaria. En caso de que no pueda ser, se escogerá que al menos una de las dos cláusulas tenga sólo dos literales, etc.

Resolución en el cálculo proposicional

- **Estrategia de refinamiento de conjunto soporte:**
  - Se dice que una cláusula C_2 es descendiente de otra cláusula C_1 si:
    - a) $\mathbf{C_2}$ es resultado de resolver $\mathbf{C_1}$ y alguna otra cláusula.
    - b) $\mathbf{C_2}$ es resultado de resolver alguna cláusula sucesora de $\mathbf{C_1}$ y alguna otra cláusula.
- **Conjunto soporte**: Conjunto de cláusulas que provienen de la negación de la cláusula a demostrar, o bien descendientes de estas cláusulas.
- **La estrategia del conjunto soporte** sólo permite realizar resoluciones en las que una de las cláusulas que se resuelven pertenecen al conjunto soporte.

Resolución en el cálculo proposicional

- **Ejemplo de estrategia del conjunto soporte:** El proceso de refutación del ejemplo anterior sigue esta estrategia:



El Cálculo de Predicados

- **El cálculo proposicional es limitado.** Supongamos nuestro mundo de bloques. Para decir que el bloque **A** está sobre el bloque **B**, deberíamos establecer una interpretación **SOBRE_A_B**.
- Para representar esta situación con todos los bloques usando cálculo proposicional, necesitaríamos tantos literales como posibilidades.
- Además, supongamos dos literales **P** y **Q**, con la semántica asociada **P≡SOBRE_A_B**, **Q≡SOBRE_B_C.**
- En lenguaje natural y mediante el conocimiento que tenemos del problema, nosotros (diseñadores, personas, etc.) ***sabemos*** que **A** está sobre **B**, y que **B** está sobre **C**. Por tanto, **C** está por debajo de **A**. Sin embargo, necesitaríamos más proposiciones y más complejas para implementar este conocimiento utilizando únicamente cálculo proposicional.

El Cálculo de Predicados

- Sería de gran utilidad un lenguaje que permitiese definir objetos y relaciones entre ellos.
- El **cálculo de predicados** nos permite esta opción y, además solventa los problemas planteados en la diapositiva anterior.
- Ejemplo: Para decir que $SOBRE\_B\_C \supset \neg LIBRE\_C$, para cualquier bloque, el cálculo de predicados nos evita tener que reescribir todas las proposiciones del cálculo proposicional de las situaciones que pueden darse. Podemos abstraer los objetos a variables y escribir:

$$SOBRE(x, y) \supset \neg LIBRE(y)$$

- El significado sería "*cuando un objeto x esté sobre otro y, entonces y no estará libre*".

El Cálculo de Predicados

- **Componentes del lenguajes del cálculo de predicados:**
  - **Constantes de nombres de objetos:** TorreEiffel, Paco, Marta, 127, 4B
  - **Constantes de nombres de funciones:** monumento, padreDe, distanciaEntre, codigoProducto
  - **Constantes de relaciones:** B17, Padre, Largo, Libre
  - **Conectivas proposicionales:** $\wedge, \vee, \neg, \supset$
  - **Delimitadores y separadores:** "(", ")", "[", "]", ","
- Llamaremos **aridad** de una función al número de parámetros de entrada que tiene una llamada correcta a dicha función.

El Cálculo de Predicados

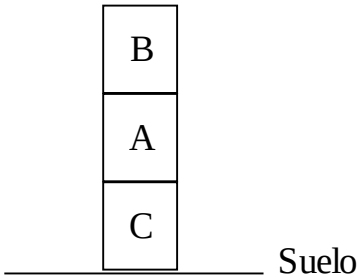
- **Términos.** Se definen como términos:
  - Las constantes de objetos: Juan, Pepe, A, PuertaEntrada
  - Las funciones de aridad **n**, seguidas de **n** términos separados por comas: monumento(TorreEiffel), padreDe(Marta, Juan), distanciaEntre(París, Ceuta, 2800)
- **FBFs.** Son FBFs:
  - Los atomos: Constantes de relaciones de aridad **n** seguidas de **n términos** separados por comas. Ejemplo: MayorQue(7, 3), esPadreDe(Juan, Pepe).
  - FBFs de predicados: MayorQue(7,3)^MenorQue(2,3) v ¬Hermano(Juan, Pepe)

Modelos e interpretaciones

- Con el cálculo de predicados disponemos de un lenguaje para referirnos a los objetos del mundo, y funciones y relaciones sobre ellos.
- En cálculo de predicados, una **interpretación** es una asignación entre los objetos del mundo y las constantes de objetos, funciones n-arias a las constantes de funciones n-arias, y relaciones n-arias a las constantes de relaciones n-arias.
- Dada una interpretación, un **átomo tendrá valor V** sólo si es sostenible (que sea correcto en el mundo). Si no es sostenible, tendrá el **valor F**.
- Las veracidad o falsedad del resto de FBFs se determinan mediante las tablas de verdad.

Modelos e interpretaciones

- **Ejemplo:** En el mundo de los bloques, definimos **A, B, C, Suelo**, y les asignamos su interpretación con los bloques reales y el suelo. Definimos también las relaciones **Sobre(x,y)** y **Libre(x)**, indicando la primera que **x** está sobre **y**, y la segunda que **x** está libre (no tiene nada encima).
- Supongamos que la relación **Sobre** se da para: **Sobre(B, A), Sobre(A, C), Sobre(C, Suelo)**, y que **Libre** se da para **Libre(B)**. Estas relaciones definen el estado del mundo actualmente representada como sigue:



Modelos e interpretaciones

- **Ejemplo:** Esta asignación se representa en la siguiente tabla:

Cálculo de Predicados	Mundo
A	A
B	B
C	C
Suelo	Suelo
Sobre	<B,A>, <A,C>, <C,Suelo>
Libre	<B>

Modelos e interpretaciones

- **Ejemplo:** La información de la tabla anterior se puede utilizar para conocer la veracidad de otras FBFs del cálculo de predicados. Ejemplo:
  - **Sobre(A,B)** es falsa porque **<A,B>** no está en la lista de relaciones **Sobre**.
  - **Libre(B)** es verdadera porque **<B>** está en la relación **Libre**
  - **Sobre(C, Suelo)^¬Sobre(A,B)** es verdadera porque ambas relaciones son verdaderas.

Modelos e interpretaciones

- Una interpretación **satisface** una FBF, si la FBF es verdadera bajo esa interpretación.
- Una interpretación que satisface una FBF es un **modelo** de esta.
- Las FBF con valor **V** en cualquier interpretación son **FBF válidas**.
- Las FBF que no tiene ningún modelo se llaman **inconsistentes** o **insatisfacibles**.
- Si una FBF **w** es **V** para todas las interpretaciones posibles en las que el conjunto Δ es verdadero, entonces **w** se demuestra desde Δ.
- Dos FBF son **equivalentes** si sus valores **V** son idénticos en cualquier interpretación.

Modelos e interpretaciones

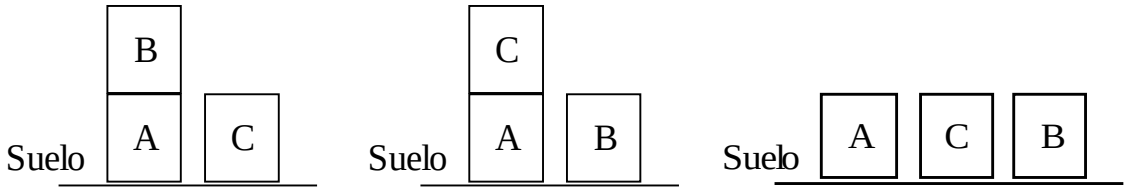
- Las fórmulas del cálculo de predicados se pueden utilizar para representar el conocimiento que tiene un agente sobre el mundo.
- Al conjunto Δ formado por este tipo de fórmulas se le llama **Base de Conocimiento.**
- Al construir una Base de Conocimiento, debemos pensar en generar el suficiente número de fórmulas que permitan identificar el modelo del mundo que se representa.
- Pero también, a su vez, debemos evitar ambigüedades e intentar evitar excluir otros modelos posibles adicionales.

Modelos e interpretaciones

- **Ejemplo:** Supongamos la siguiente Base de Conocimiento (B.C.):

$$1.\ Sobre(A, Suelo) \supset Libre(B)$$
$$2.\ Libre(B) \wedge Libre(C) \supset Sobre(A, Suelo)$$
$$3.\ Libre(B) \vee Libre(A)$$
$$4.\ Libre(B)$$
$$5.\ Libre(C)$$

- Entonces, cualquiera de los siguientes modelos podría ser válido:



- Por ejemplo, añadiendo **Libre(A)** a la B.C., se elimina toda ambigüedad, definiendo claramente el modelo del centro.

Cuantificación

- El **Cálculo de predicados** tiene otras componentes más aparte de las indicadas anteriormente:
  - **Símbolos de variables:** x, y, z, etc.
  - **Cuantificadores (universal, existencial):** $\forall$ **(para todo),** $\exists$ **(existe)**
- **FBFs:**
  - Si **w** es una FBF y **x** es una variable entonces son FBFs:

$$(\forall x)w$$

$$(\exists x)w$$

  - Ejemplos: $(\forall x)[P(x) \supset R(x)]$

$$(\exists x)[P(x) \supset (\exists y)[R(x,y) \supset S(f(x))]]$$

Cuantificación

- **¡Cuidado con la cuantificación!:** $(\forall x)(\forall y) \equiv (\forall y)(\forall x)$

  $(\exists x)(\exists y) \equiv (\exists y)(\exists x)$

  $(\forall x)(\exists y) \neq (\exists y)(\forall x)$

- La primera parte de la última sentencia quiere decir que "Para cualquier objeto **x**, existe al menos un objeto **y** tal que...".La segunda parte de la última sentencia quiere decir que "Existe al menos un objeto **y** tal que todos los objetos **x**...".

- **Ejemplo:** a) Parte 1. "**Para cualquier modelo de pasarela existe al menos un hombre fan**". b) Parte 2. "**Existe al menos un hombre que es fan de todas las modelos de pasarela**".

$$(\forall x)(\exists y) Modelo(x) \wedge Hombre(y) \wedge Fan(x, y)$$

$$(\exists y)(\forall x) Modelo(x) \wedge Hombre(y) \wedge Fan(x, y)$$

Cuantificación

- Equivalencias útiles entre cuantificadores:

$$\neg(\forall x)P(x) \equiv (\exists x)\neg P(x)$$
$$\neg(\exists x)P(x) \equiv (\forall x)\neg P(x)$$

- Semántica de cuantificadores:
  - $(\forall x)P(x)$ es cierta si, para cualquier valor que pueda tomar **x**, entonces **P(x)** es cierta.
  - $(\exists x)P(x)$ es cierta si, como mínimo para un valor que pueda tomar **x, P(x)** es cierta.
  - Ejemplos: $(\forall x)[Sobre(x, C) \supset \neg Libre(C)]$

    $(\exists x)[Sobre(x, Suelo)]$

Cuantificación

- **Se utilizan las reglas de inferencia del cálculo proposicional**.
- Además, para tratar con los cuantificadores, se introducen dos nuevas reglas:
  - **Eliminación del cuantificador universal**. Este se elimina instanciando la/s variable/s cuantificadas en los objetos que hacen que la FBF sea cierta, dentro de la interpretación actual.

    $$(\forall x)P(x,f(x),B), \text{instanciando } x \text{ en } A : P(A,f(A),B)$$

  - **Introducción del cuantificador existencial.** Se introduce abstrayendo los objetos de una FBF.

    $$(\forall x)Q(A,f(A),x), \text{abstrayendo } A : (\exists y)(\forall x)Q(y,f(y),x)$$

Conceptualización

- El gran problema en IA no es cómo representar, sino **qué** representar. Requiere de una gran habilidad del diseñador. Ejemplos:
  - *"Todos los paquetes que están en la habitación 27 son más pequeños que los de la 28"*

    $$(\forall x, y)[[\mathbf{Paquete}(x) \wedge \mathbf{Paquete}(y) \wedge \mathbf{EnHabitacion}(x, \mathbf{H27}) \wedge \mathbf{EnHabitacion}(y, \mathbf{H28})] \supset \mathbf{MasPequeño}(x, y)]$$

  - *"Cada paquete de la habitación 27 es más pequeño que uno de los paquetes de la habitación 29"*
    - Esta frase es ambigua (cosa frecuente en el lenguaje natural), ya que puede representar dos situaciones.

Conceptualización

- El gran problema en IA no es cómo representar, sino **qué** representar. Requiere de una gran habilidad del diseñador. Ejemplos:
  - *"Cada paquete de la habitación 27 es más pequeño que uno de los paquetes de la habitación 29"*
    - Situación 1:

$$(\exists y)(\forall x)[\mathbf{Paquete}(x) \wedge \mathbf{Paquete}(y) \wedge \mathbf{EnHabitacion}(x, \mathbf{H27}) \wedge \mathbf{EnHabitacion}(y, \mathbf{H29})] \supset \mathbf{MasPequeño}(x, y)]$$

    - Situación 2:

$$(\forall x)(\exists y)[\mathbf{Paquete}(x) \wedge \mathbf{Paquete}(y) \wedge \mathbf{EnHabitacion}(x, \mathbf{H27}) \wedge \mathbf{EnHabitacion}(y, \mathbf{H29})] \supset \mathbf{MasPequeño}(x, y)]$$

Conceptualización

- El cálculo de predicados nos permite también conceptualizar la noción del tiempo. Por ejemplo, si en una empresa de mensajería se desea decir que "*El paquete A ha llegado antes que el paquete B*", con cálculo de predicados lo haríamos de la siguiente forma, introduciendo una nueva variable que modele el tiempo:

$$(\exists z1, z2)\mathbf{[HaLlegado(A, z1)} \wedge \mathbf{H}aLlegado\mathbf{(B, z2)} \wedge \mathbf{A}ntes\mathbf{(z1, z2)]}$$

Resolución en el cálculo de predicados

- Antes de empezar…
  - … Por simplificar vamos a escribir todas las disyunciones de literales **(una cláusula)** sin el cuantificador universal, suponiendo siempre que ese cuantificador está presente en la expresión (sólo por comodidad al leer las diapositivas).

$$(\forall x, y, z, ...)[w_1 \vee w_2 \vee w_3 \vee ... \vee w_n] \Rightarrow w_1 \vee w_2 \vee w_3 \vee ... \vee w_n$$

- Si dos cláusulas tienen literales similares pero complementarios, se pueden resolver de igual forma que en el cálculo proposicional.
- **Unificación:** Proceso de sustitución de variables que permite unificar dos cláusulas de modo que al menos un término y su negación sean resolubles.

Resolución en el cálculo de predicados

- **Sustitución de variables:** Supongamos una expresión de ejemplo **P(x,f(y),B)**. Ejemplos de sustituciones serían:
  - S1={z/x, w/y} → P(z, f(w), B)
  - S2={A/y} → P(x, f(A), B)
  - S3={g(z)/x, A/y} → P(g(z), f(A), B)
  - S4={C/x, A/y} → P(C, f(A), B)
- La sustitución **S1** es una **variante alfabética**, dado que todas las variables se han sustituido por otras.
- La sustitución **S4** es una **instancia base**, dado que no aparece ninguna variable.

Resolución en el cálculo de predicados

- **Composición de sustituciones de variables:** Es la aplicación concatenada de dos sustituciones **S1** y **S2**, y se escribe como **S1S2**.
- **Ejemplo:** S1={g(x,y)/z} y S2={A/x, B/y, C/w}

- **Para resolver una composición de dos sustituciones S1 y S2, primero se aplica S2 sobre S1.** La sustitución resultante es lo que quede de esta sustitución de S2 sobre S2:
- **Ejemplo:** S3= S1S2= {g(A,B)/z, A/x, B/y, C/w}
- Diremos que un conjunto de expresiones **$\Gamma=\{w_i\}$** son unificables si existe una sustitución **S** que las simplifique de modo que **$w_1S=w_2S=w_3S=\ldots$**. Ejemplo: **S={A/x, B/y}** unifica **Γ={P(x,f(y),B), P(x, f(B), B}** dando como resultado **P(A,f(B), B)**.

Resolución en el cálculo de predicados

- **Algoritmo de unificación del conjunto de expresiones Γ:**

UNIFICA(Γ)

1. **K**←0; Γ_k← Γ; S_k← {}
2. Si $\boldsymbol{\Gamma_k}$ es unitario, salir y devolver $\boldsymbol{\Gamma_k}$, $\boldsymbol{S_k}$
3. $\boldsymbol{D_k}$← Conjunto de discrepancias de $\boldsymbol{\Gamma_k}$
4. **Si** no existen variable $\boldsymbol{v_k}$ y término $\boldsymbol{t_k}$, tal que $\boldsymbol{v_k}$ no aparece en $\boldsymbol{t_k}$, **hacer:** Salir con fallo: No unificable.
5. En otro caso,
6. Hacer $\boldsymbol{S_{k+1}}$←$\boldsymbol{S_k\{t_k/v_k\}}$; $\boldsymbol{\Gamma_{k+1}}$← $\boldsymbol{\Gamma_k\{t_k/v_k\}}$
7. Hacer **k**← **k+1** y volver al paso 2.

- La idea clave es el conjunto de discrepancias, que se obtiene localizando el primer símbolo que no es exactamente el mismo en las expresiones de Γ. Por ejemplo, para **P(x,f(A),y)** y **P(x,f(x),B)** es el conjunto **{A/z}**.

Resolución en el cálculo de predicados

- Supongamos que tenemos dos cláusulas $\mathbf{C_1}$ y $\mathbf{C_2}$. Si hay un átomo $\mathbf{\Phi}$ en $\mathbf{C_1}$ y un literal $\mathbf{\neg\Psi}$ en $\mathbf{C_2}$, tales que existe una unificación mediante **S** para $\mathbf{\Phi}$ y $\mathbf{\Psi}$, entonces $\mathbf{C_1}$ y $\mathbf{C_2}$ tienen un resolvente aplicando **S** sobre las mismas.
- **Ejemplo:**
  - Sean las cláusulas **C1={P(x), Q(x,y)}, C2={¬P(A), R(B,z)}** Mediante **S={A/x}** se unifican **P(x)** y **P(A)**. Por tanto, **C1** y **C2** se resuelven y generan **{Q(A,y), R(B,z)}**.
  - Sean las cláusulas **C1={P(x,x), Q(x), R(x)}, C2={¬P(A,z), Q(B)}**. Hay 2 formas de resolverlo, generando:
    - a) **{Q(A), R(A), ¬Q(B)}**, mediante **S={A/x, A/z}**
    - b) **{P(B,B), R(B), ¬P(A,z)}**, mediante **S={B/x}**

Resolución en el cálculo de predicados

- Se siguen 9 pasos:
  - **1. Eliminar implicaciones** (como en cálculo proposicional).
  - **2. Reducir ámbito de negaciones** (como en c. proposicional).
  - **3. Estandarizar variables** para que cada cuantificador tenga su símbolo de variable. Ejemplo:

    $$(\forall x)[\neg P(x) \vee (\exists x)Q(x)] \text{ queda como } (\forall x)[\neg P(x) \vee (\exists y)Q(y)]$$

  - **4. Eliminar cuantificadores existenciales**, reemplazando la ocurrencia de la variable existencial por una función (*función de Skolem*) dependiente de la/s variable/s cuantificada/s universalmente en su ámbito (cuidado: ¡Que no se repitan funciones en ámbitos diferentes!). Ejemplo:

    $$(\forall x)[(\exists y)\text{Altura}(x,y)] \text{ queda como } (\forall x)[\text{Altura}(x,h(x))]$$

Resolución en el cálculo de predicados

- Se siguen 9 pasos (continuación):
  - Otro ejemplo (eliminación de cuantificador existencial):

$$[(\forall w)Q(w)] \supset (\forall x)\{(\forall y)\{(\exists z)[P(x,y,z) \supset (\forall u)R(x,y,u,z)]\}\ \textbf{queda como}$$

$$[(\forall w)Q(w)] \supset (\forall x)\{(\forall y)[P(x,y,g(x,y)) \supset (\forall u)R(x,y,u,g(x,y))]\}$$

  - **5. Pasar todos los cuantificadores universales al principio de la cadena** *(forma prenex).* Ejemplo:

$$(\forall w,x,y,u)[Q(w) \supset [P(x,y,g(x,y)) \supset R(x,y,u,g(x,y))]]$$

  - **6. Pasar a FNC.** Ejemplo:

$$(\forall x,y)[\neg P(x) \vee \{[\neg P(y) \vee P(f(x,y))] \wedge [Q(x,h(x)) \wedge \neg P(h(x))]\}] \Rightarrow$$

$$\Rightarrow (\forall x,y)[[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge [\neg P(x) \vee Q(x,h(x))] \wedge$$

$$\wedge [\neg P(x) \vee \neg P(h(x))]]$$

Resolución en el cálculo de predicados

- Se siguen 9 pasos (continuación):
  - **7. Eliminar cuantificadores universales.**
  - **8. Eliminar los símbolos** ^. Las expresiones del tipo $\mathbf{w_1 \wedge w_2}$ se pueden reemplazar por conjuntos de FBFs $\mathbf{\{w_1, w_2\}}$. Ejemplo para el resultado obtenido en el paso 6:

$$\mathbf{\neg P(x) \vee \neg P(y) \vee P(f(x,y))}$$

$$\mathbf{\neg P(x) \vee Q(x,h(x))}$$

$$\mathbf{\neg P(x) \vee \neg P(h(x))}$$

  - **9. Renombrar variables**, para que el mismo símbolo no aparezca en las diferentes cláusulas

$$\mathbf{\neg P(x1) \vee \neg P(y) \vee P(f(x1,y))}$$

$$\mathbf{\neg P(x2) \vee Q(x2,h(x2))}$$

$$\mathbf{\neg P(x3) \vee \neg P(h(x2))}$$

Nota para el paso 9: tener en cuenta que

$$\forall x[P(x) \wedge Q(x)] \equiv (\forall x)P(x) \wedge (\forall y)Q(y)$$

Resolución para demostrar teoremas

- **Se hace de forma análoga como lo hacíamos en el cálculo proposicional.**
- **Ejemplo:** La Base de Conocimiento de un agente contiene los siguientes elementos:

$$(\forall x, y)[\{Paquete(x) \wedge Paquete(y) \wedge$$
$$\wedge EnHabitacion(x, H27) \wedge EnHabitacion(y, H28)\} \supset MasPequeño(x, y)]$$
$$Paquete(A)$$
$$Paquete(B)$$
$$EnHabitacion(B, H27)$$
$$MasPequeño(B, A)$$

- **Pregunta: ¿Dónde está el paquete A?**

Resolución para demostrar teoremas

- **Ejemplo (continuación):** El robot puede inferir fácilmente que EnHabitacion(A,H27) v EnHabitacion(A,H28).
- Mediante refutación, se comienza a probar si EnHabitacion(A,H27), introduciendo su negación ¬EnHabitacion(A, H27) en la Base de Conocimiento y comprobando si se llega a **Nil**.

- **Ejercicio:** Elaborar el árbol de refutación mediante resolución, partiendo de EnHabitacion(A,H27) v EnHabitacion(A,H28) y de ¬EnHabitacion(A, H27).

Sistemas Basados en el Conocimiento

- Una gran cantidad de aplicaciones reales de la IA se basan en la existencia de una gran masa de conocimiento:
  - Diagnóstico médico.
  - Diseño de equipos.
  - Sistemas de Recomendación.
  - Etc.
- Este tipo de sistemas se denominan **Sistemas Basados en el Conocimiento**, ya que este ocupa la parte central de la solución al problema a resolver.

Sistemas Basados en el Conocimiento

- Un **Sistema Basado en el Conocimiento (SBC)** necesita 3 componentes básicas:
  - Una **Base de Conocimiento (BC)**, que contenga el conocimiento experto necesario sobre el problema a resolver. Puede ser:
    - **Estática**, si la BC no varía a lo largo del tiempo.
    - **Dinámica,** cuando se añaden nuevos hechos o reglas, o se modifican las existentes a lo largo del tiempo.
  - Un **Motor de Inferencia**, que permite razonar sobre el conocimiento de la BC y los datos proporcionados por un usuario.
  - Una **interfaz de usuario** para entrada/salida de datos.

Sistemas Basados en el Conocimiento

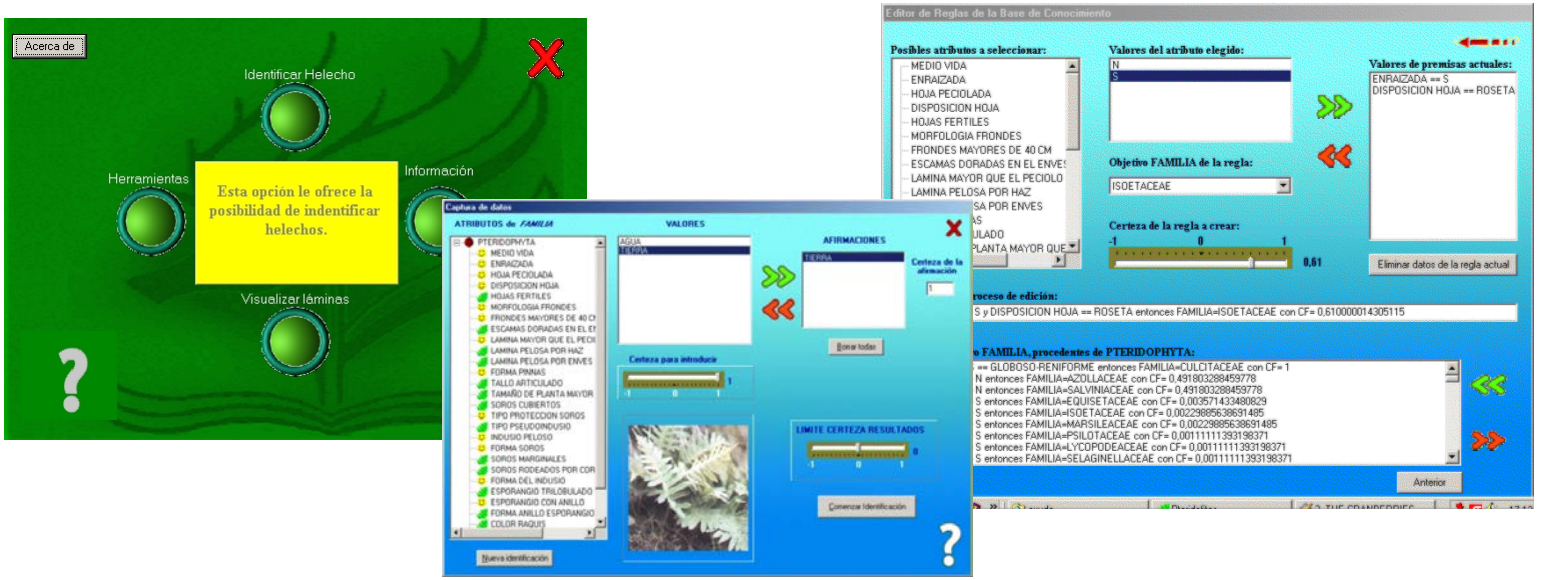
- Ejemplos de **SBC** que incorporan conocimiento específico sobre algún tema:
  - **Anna** (Agente conversacional de IKEA). Su Base de Conocimiento está descrita en lenguaje AIML (*Artificial Intelligence Markup Language*).

Sistemas Basados en el Conocimiento

- Ejemplos de **SBC** que incorporan conocimiento específico sobre algún tema:
  - El **Paciente Simulado Virtual (PSV)** es un agente que simula enfermedades. Se utiliza para la formación de especialistas en medicina: El médico pregunta y/o visualiza síntomas que el paciente presenta y debe decidir un diagnóstico válido.

Sistemas Basados en el Conocimiento

- Ejemplos de **SBC** que incorporan conocimiento específico sobre algún tema:
  - **Pteridophita** es un experto en helechos que, de forma interactiva con un usuario, permite identificar tipos de helechos y proporcionar información sobre los mismos.



Sistemas Expertos basados en Reglas (SEBR)

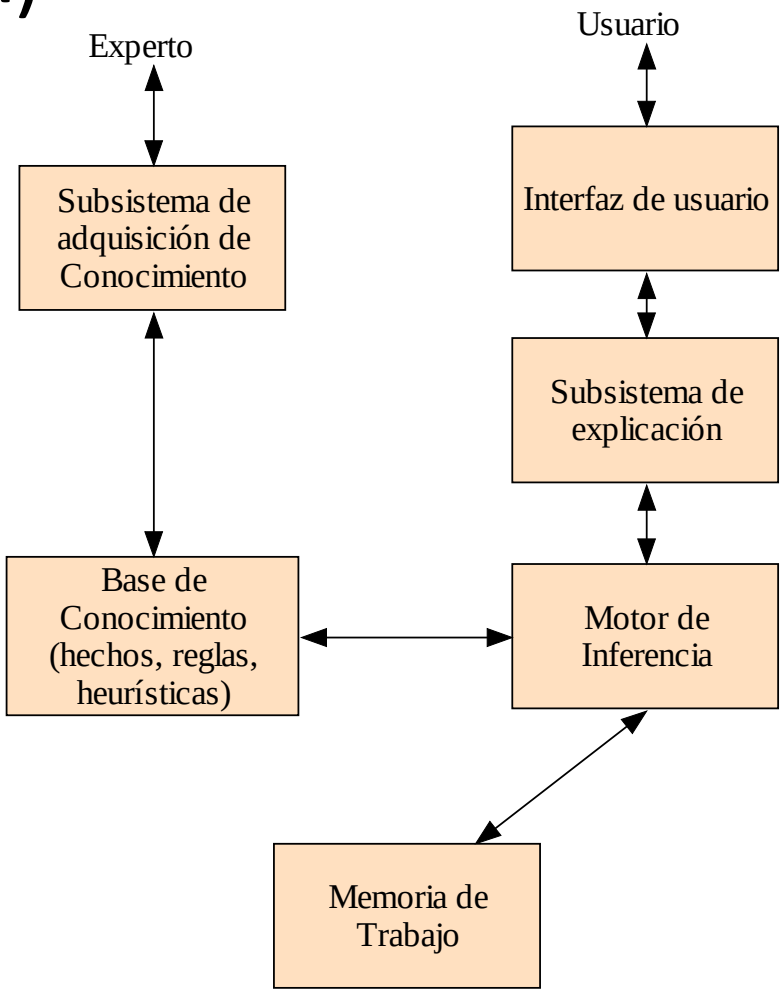
- Un **SEBR** es un **SBC** donde el conocimiento se incluye en forma de reglas y hechos.
- Estas reglas y hechos pueden implementarse, por ejemplo, mediante el **cálculo de predicados**.
- El proceso de construcción de un SEBR es el siguiente:
  - Se **extrae el conocimiento** experto (bibliografía, entrevistas a expertos reales, etc.).
  - Se **modela y se adquiere el conocimiento**, utilizando un lenguaje adecuado (cálculo de predicados, otras lógicas más avanzadas, etc.)
  - Se **crea la Base de Conocimiento** con el conocimiento adquirido.

Sistemas Expertos basados en Reglas (SEBR)

- Por otra parte, también se necesita:
  - Una **interfaz de usuario**, para poder utilizar el sistema y adquirir/enviar datos.
  - Un **subsistema de explicación**, para los casos en los que sea necesario indicar al usuario porqué se llega a las conclusiones que se llegan.
  - Un **Motor de Inferencia**, para razonar sobre la Base de Conocimiento y los datos proporcionados por el usuario.

Sistemas Expertos basados en Reglas (SEBR)

- El esquema general de diseño de un SEBR es el siguiente:
  - La **memoria de trabajo** contiene la información relevante que el Motor de Inferencia está usando para razonar las respuestas para el usuario.



Extracción del conocimiento en SEBR

- La forma más sencilla de representar los datos de un experto para la posterior extracción de conocimiento es mediante Bases de Datos relacionales y tablas relacionales. Ejemplo (Sistema Pteridophita). El objetivo es conocer la familia del helecho que se tiene

FAMILIA	MEDIO VIDA	DISPOSICIÓN HOJA	FRONDES > 40 CM	LAMINA > PECIOLO
ADIANTACEAE	TIERRA	INDIVIDUAL	S	S
PTERIDACEAE	TIERRA	INDIVIDUAL	S	S
ISOETACEAE	TIERRA	ROSETA	N	S
ADIANTACEAE	TIERRA	INDIVIDUAL	N	S/N
ADIANTACEAE	TIERRA	INDIVIDUAL	S	N
PTERIDACEAE	TIERRA	INDIVIDUAL	N	S

- Claramente, de la tabla se puede inferir visualmente que se extraerá la regla $DisposicionHoja(ROSETA) \supset Familia(ISOETACEAE)$, entre otras.

Extracción del conocimiento en SEBR

- Los métodos para extraer el conocimiento se estudian en el área de **aprendizaje automático** dentro de la IA. En SEBR, la extracción del conocimiento también se conoce como **extracción de reglas** o **aprendizaje de reglas.**
- Además, normalmente, la percepción del mundo por parte de un agente no es perfecta.
- Del mismo modo, es posible que una regla no sea aplicable siempre (aunque sí en un gran número de casos). Este hecho no permite que la regla sea admitida en un sistema de cálculo proposicional o de predicados, dado que daría lugar a sistemas inconsistentes.
- Se hace necesario establecer mecanismos para **tratar con incertidumbre**.

Otros modelos/problemas de representación del conocimiento

- Representación del conocimiento de sentido común
- Organización jerárquica del conocimiento
- Razonamiento temporal
- Lógica difusa
- …