

Tema 2: Lógica computacional para IA: Lógica proposicional

Joaquín Borrego Díaz

Dpto. de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE SEVILLA

Contenido

- Sintaxis
- Semántica
- Deducción (I): Tableros
- Formas clausales
- Deducción (II): Resolución
- Adecuación y completitud de los métodos deductivos
- Teoría de la demostración
- Algunas aplicaciones
- Limitaciones de la lógica proposicional

Lógica proposicional: Sintaxis (I)

- Una lógica de *oraciones* y/o propiedades
- No distingue entre elementos no explícitos en el lenguaje
- Semántica: Verdadero/falso
- El *lenguaje proposicional* consta de:
 1. Un conjunto numerable, SP , de **símbolos proposicionales**: p_0, p_1, \dots
 2. Las **conectivas lógicas**: \neg (negación) y \vee (disyunción).
 3. **Símbolos auxiliares**: “(” y “)”.
- Los símbolos p, q, r, \dots representarán símbolos proposicionales, y los símbolos F, G, H, \dots representarán fórmulas.

Lógica proposicional: Sintaxis (II)

- El conjunto $PROP$ de las *fórmulas proposicionales* es el conjunto de expresiones \mathcal{C} que contiene a SP y verifica que
 - $F \in \mathcal{C} \implies \neg F \in \mathcal{C}$
 - $F, G \in \mathcal{C} \implies \vee FG \in \mathcal{C}$
- Para facilitar la lectura y el manejo de las fórmulas, se usarán paréntesis y la siguiente notación:
 1. $F \vee G$ por $\vee FG$, y $F \wedge G$ en lugar de $\neg(\neg F \vee \neg G)$.
 2. $F \rightarrow G$ en lugar de $\neg F \vee G$, y $F \leftrightarrow G$ en lugar de $(F \rightarrow G) \wedge (G \rightarrow F)$.
 3. Las conectivas tienen una precedencia de asociación. De mayor a menor, están ordenadas por:
 $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.
 4. Cuando una conectiva se usa repetidamente, se asocia por la derecha:
 $F \wedge G \rightarrow \neg F \vee G$ es $((F \wedge G) \rightarrow (\neg F \vee G))$, y $F \vee G \vee H$ es $(F \vee (G \vee H))$.

Semántica (I)

- Los elementos del conjunto $\{0, 1\}$ se llaman *valores de verdad*. Se dice que 0 es el valor *falso* y el 1 es el valor *verdadero*.

- La *función de verdad* de la negación es la función $H_{\neg}(i) = \begin{cases} 1, & \text{si } i = 0; \\ 0, & \text{si } i = 1. \end{cases}$

- La *función de verdad* de la disyunción es la función $H_{\vee}(i, j) = \begin{cases} 0, & \text{si } i = j = 0; \\ 1, & \text{en otro caso.} \end{cases}$

- Interpretación de los símbolos proposicionales: Una aplicación

$$v : SP \rightarrow \{0, 1\}$$

- Es decir, asigna un valor de verdad a cada símbolo proposicional

Semántica (II)

- Cada v se extiende sobre las fórmulas como:

- $v(\neg F) = H_{\neg}(v(F))$

- $v((F \vee G)) = H_{\vee}(v(F), v(G))$.

- Se dice que $v(F)$ es el **valor de verdad** de F respecto de v .

- $H_{\wedge}(i, j) = \begin{cases} 1, & \text{si } i = j = 1; \\ 0, & \text{en otro caso.} \end{cases}$

- $H_{\rightarrow}(i, j) = \begin{cases} 0, & \text{si } i = 1, j = 0; \\ 1, & \text{en otro caso.} \end{cases}$

- $H_{\leftrightarrow}(i, j) = \begin{cases} 1, & \text{si } i = j; \\ 0, & \text{en otro caso.} \end{cases}$

- Ejemplo: si $v(p) = v(q) = 0$ y $v(r) = 1$, entonces

$$\begin{aligned} v((p \rightarrow q) \vee r) &= H_{\vee}(v(p \rightarrow q), v(r)) = \\ &= H_{\vee}(H_{\rightarrow}(p, q), 1) = 1 \end{aligned}$$

Validez y consistencia

- F válida para v (o v modelo de F , $v \models F$) si $v(F) = 1$.
 - v es modelo de un conjunto Γ , $v \models \Gamma$, si v es modelo de toda fórmula de Γ .
- F es *lógicamente válida (tautología)* si es válida para toda interpretación (notación $\models F$).
- Un conjunto de fórmulas Γ es consistente (satisfactible) si existe una interpretación que es modelo de Γ . En caso contrario diremos que es inconsistente.
- Ejemplo: Tomemos el lenguaje tal que $SP = \{\text{Llueve}, \text{Mojado}\}$
 - Sea la fórmula: $\text{Llueve} \rightarrow \text{Mojado}$
 - Es consistente: v tal que $v(\text{Llueve}) = 0$.
 - No es tautología: v tal que $v(\text{Llueve}) = 1$ y $v(\text{Mojado}) = 0$.
 - Es consistente, pero aún siendo válida en el mundo real no es lógicamente válida.

Problemas

- Problema(s): Decidir mecánicamente la consistencia y/o validez
 - Resoluble algorítmicamente pero no de manera eficiente
 - La consistencia es un problema NP-completo
- Relación entre ellos:
 - F válida $\implies \neg F$ insatisfactible
 - F válida $\implies F$ satisfactible
- Acercamiento directo: Las tablas de verdad
- Acercamiento indirecto: Cálculos lógicos
- No existen algoritmos incrementales:
 - Un pequeño cambio en la base de conocimiento implica comenzar el análisis

Consecuencia Lógica

- Una fórmula F es consecuencia lógica de un conjunto Γ , $\Gamma \models F$, si todo modelo de Γ es modelo de F . Es decir, para toda v interpretación,

$$v \models \Gamma \implies v \models F$$

- La consecuencia lógica no coincide con la consecuencia real: Supongamos que $\Gamma = \{\text{Llueve}\}$. Entonces $\{\text{Llueve}\} \not\models \text{Mojado}$
- Relación entre consecuencia lógica, consistencia y validez: Si $\Gamma = \{F_1, \dots, F_n\}$, entonces son equivalentes:
 - $\{F_1, \dots, F_n\} \models F$
 - $\models F_1 \wedge \dots \wedge F_n \rightarrow F$
 - $\{F_1, \dots, F_n, \neg F\}$ es inconsistente.
- Monotonía de la consecuencia lógica: Si $\Gamma_1 \models F$, entonces $\Gamma_1 \cup \Gamma_2 \models F$

Equivalencias

- Dos fórmulas F_1, F_2 se dicen equivalentes ($F_1 \equiv F_2$) si tienen los mismos modelos
 - $F_1 \equiv F_2$ si y sólo si $F_1 \models F_2$ y $F_2 \models F_1$
 - Ejemplos:
 - Dos fórmulas inconsistentes son equivalentes
 - Dos tautologías son equivalentes
 - $\neg\neg A \equiv A$
 - $\neg(A \vee B) \equiv \neg A \wedge \neg B$, y $\neg(A \wedge B) \equiv \neg A \vee \neg B$
 - $(A \vee B) \vee C \equiv A \vee (B \vee C)$, y $(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$
 - $(A \vee B) \wedge C \equiv (A \wedge C) \vee (B \wedge C)$
 - $(A \wedge B) \vee C \equiv (A \vee C) \wedge (B \vee C)$

Deducción (II): Tableros

- Sea F una fórmula. Diremos que F es:
 1. Una α -fórmula si existen α_1 y α_2 tales que $F \equiv \alpha_1 \wedge \alpha_2$
 2. Una β -fórmula si existen β_1 y β_2 tales que $F \equiv \beta_1 \vee \beta_2$
- La negación de una α -fórmula es una β -fórmula, y recíprocamente.
- Las siguientes fórmulas son α -fórmulas y β -fórmulas:

α	α_1	α_2
$\neg\neg F$	F	
$F_1 \wedge F_2$	F_1	F_2
$\neg(F_1 \vee F_2)$	$\neg F_1$	$\neg F_2$
$\neg(F_1 \rightarrow F_2)$	F_1	$\neg F_2$
$F_1 \leftrightarrow F_2$	$F_1 \rightarrow F_2$	$F_2 \rightarrow F_1$

β	β_1	β_2
$F_1 \vee F_2$	F_1	F_2
$\neg(F_1 \wedge F_2)$	$\neg F_1$	$\neg F_2$
$(F_1 \rightarrow F_2)$	$\neg F_1$	F_2
$\neg(F_1 \leftrightarrow F_2)$	$\neg(F_1 \rightarrow F_2)$	$\neg(F_2 \rightarrow F_1)$

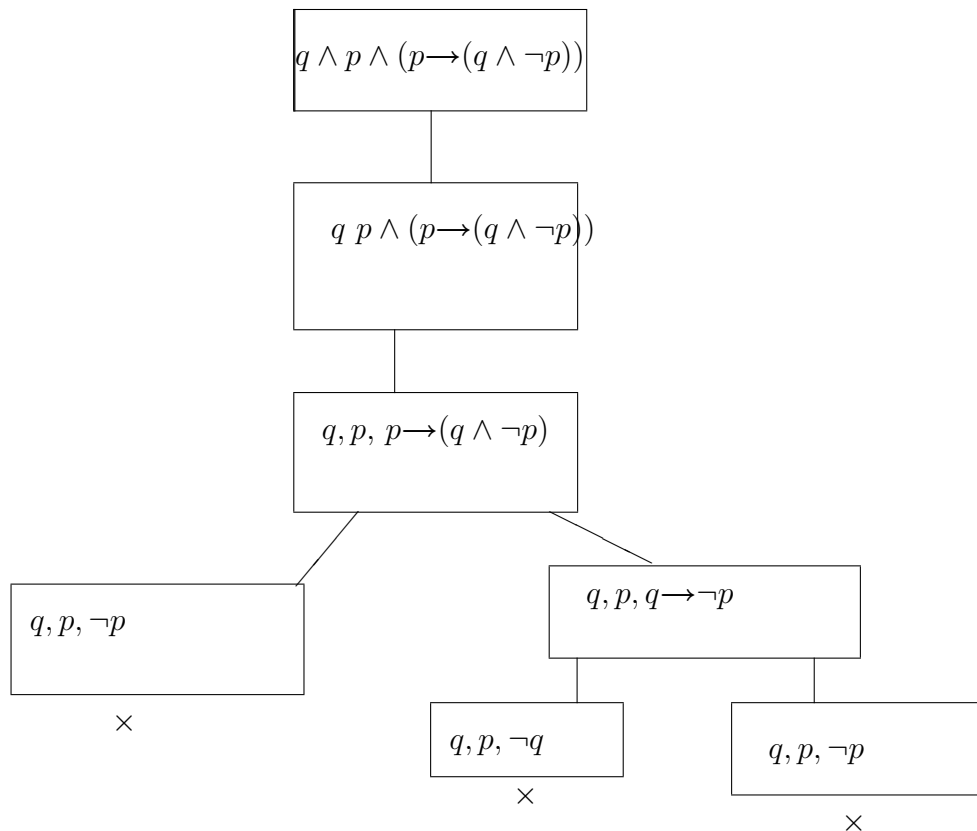
Definición de Tablero

- Un tablero para $\{A_1, \dots, A_n\}$ es un árbol T , con nodos etiquetados por conjuntos de fórmulas, tal que:
 - La raíz r de T está etiquetado por $U(r) = \{A_1, \dots, A_n\}$.
 - Para cada nodo l de T , con etiqueta $U(l)$, no marcado, hacer:
 1. Si $U(l)$ es un conjunto de literales, entonces:
 - (a) Si existe un par de literales complementarios en $U(l)$, marcar con \times (y se denomina **hoja cerrada**).
 - (b) Si no existe tal par, marcar con \bigcirc (**hoja abierta**).
 2. Si $U(l)$ no es un conjunto de literales, elegir A de $U(l)$ no literal.
 - (a) Si A es una α -fórmula, entonces añadir un hijo l' de l con $U(l') = (U(l) \setminus \{A\}) \cup \{\alpha_1, \alpha_2\}$ (α_2 puede no existir).
 - (b) Si A es una β -fórmula, entonces añadir dos hijos l', l'' con etiquetas

$$U(l') = (U(l) \setminus \{A\}) \cup \{\beta_1\} \quad \text{y} \quad U(l'') = (U(l) \setminus \{A\}) \cup \{\beta_2\}$$

Ejemplo

La fórmula $q \wedge p \wedge (p \rightarrow (q \wedge \neg p))$ es inconsistente:



Propiedades de los tableros

- La construcción siempre termina. El tablero final se denomina **tablero completo**.
- Un tablero T es **cerrado** si todas sus hojas son cerradas. En otro caso es **abierto**.

Teorema de corrección:

- *Sea S un conjunto de fórmulas, y T un tablero semántico para S . Si T es cerrado, entonces S es inconsistente.*

Teorema de completitud:

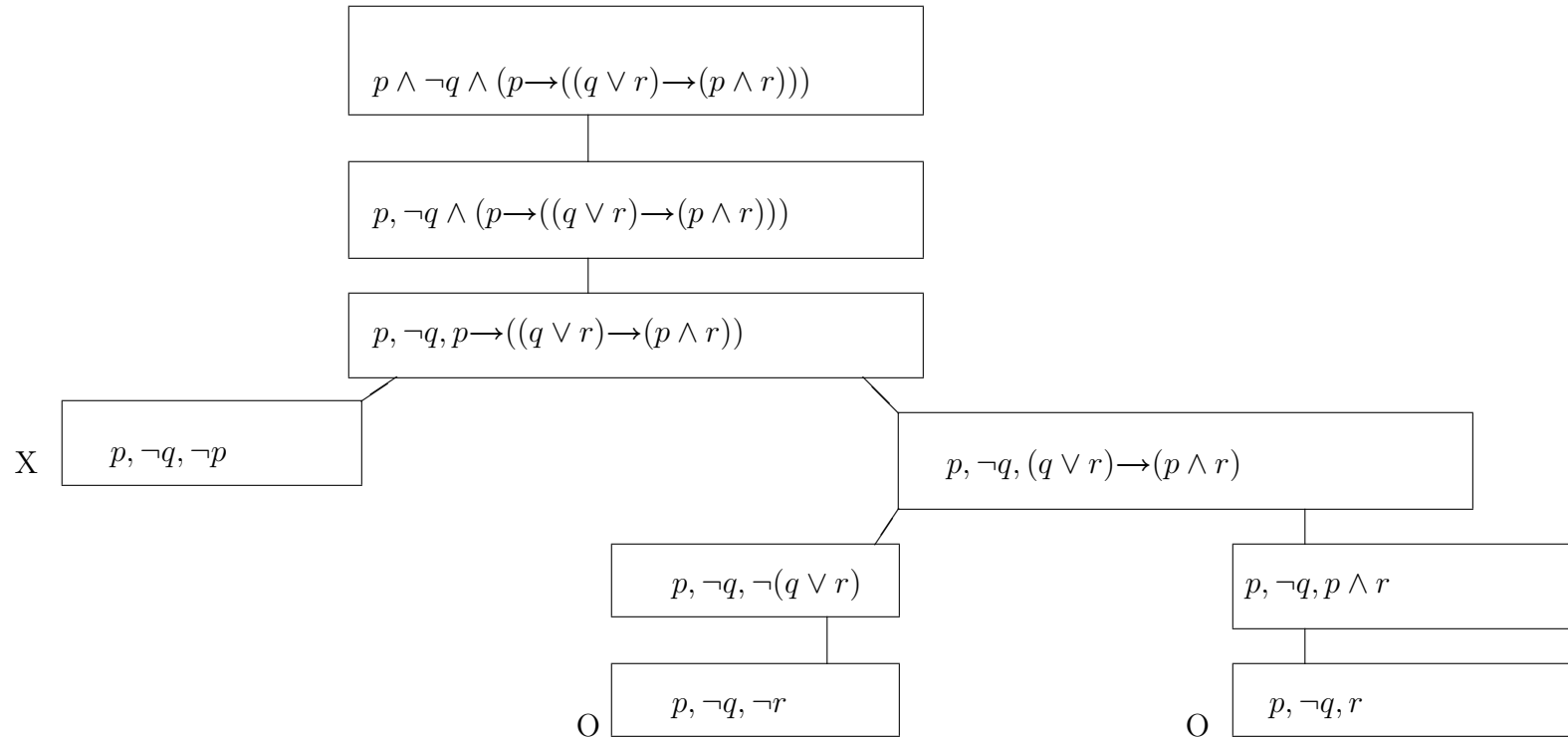
- *Sea S un conjunto de fórmulas y T un tablero para S . Si S es inconsistente entonces T es cerrado.*

- $\{A_1, \dots, A_n\}$ admite un tablero abierto si y sólo si es un conjunto consistente. Además la rama abierta define un modelo: Si U es la etiqueta de la hoja abierta,

$$v(p) = 1 \text{ si } p \in U \text{ ó } \neg p \notin U, \quad \text{y} \quad v(p) = 0 \text{ si } \neg p \in U.$$

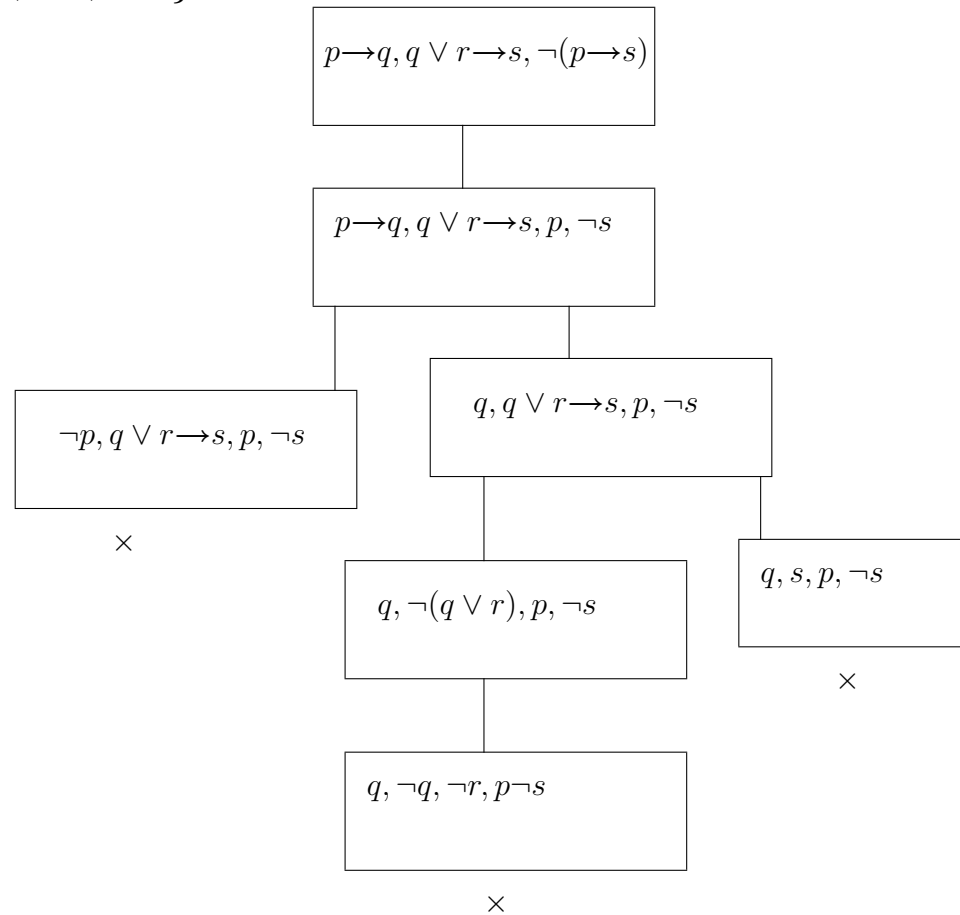
Ejemplo de consistencia:

- Un nodo abierto induce $v(p) = v(r) = 1, v(q) = 0$:



Consecuencia lógica

$\{A_1, \dots, A_n\} \models A \iff \{A_1, \dots, A_n, \neg A\}$ admite un tablero cerrado.



Por ejemplo, $\{p \rightarrow q, q \vee r \rightarrow s\} \models p \rightarrow s$

Formas clausales

- Objetivo: Restringir la sintaxis para obtener eficiencia. Opciones:
 - Fórmulas C , conjunción de un conjunto finito de literales; $C = L_1 \wedge \cdots \wedge L_n$
 - Cláusulas: disyunción de un conjunto finito de literales; $D = L_1 \vee \cdots \vee L_n$
 - Estos dos tipos no son expresivos
 - **Forma normal conjuntiva (FNC)** si es una conjunción de un conjunto de disyunciones;

$$F = \bigwedge_{i=1}^n \left(\bigvee_{j=1}^{m_i} L_{i,j} \right)$$

- **Forma normal disjuntiva (FND)** si es una disyunción de un conjunto de conjunciones;

$$F = \bigvee_{i=1}^n \left(\bigwedge_{j=1}^{m_i} L_{i,j} \right)$$

Normalización

- Para toda G existe F en forma normal conjuntiva tal que $F \equiv G$.
- Para toda G existe F en forma normal disyuntiva tal que $F \equiv G$.
- Procedimiento para transformar G en F.N.C.:

- Eliminar todas las implicaciones mediante la equivalencia: $A \rightarrow B \equiv \neg A \vee B$

- Reducir el ámbito de \neg mediante las leyes de Morgan:

$$\neg(A \wedge B) \equiv \neg A \vee \neg B$$

$$\neg(A \vee B) \equiv \neg A \wedge \neg B$$

- Convertir a f.n.c. utilizando la ley asociativa: $A \vee (B_1 \wedge B_2) \equiv (A \vee B_1) \wedge (A \vee B_2)$

- Ejemplo: $(p \wedge q) \rightarrow (q \wedge r) \vee p \implies \neg(p \wedge q) \vee (q \wedge r) \vee p \implies \neg p \vee \neg q \vee (q \wedge r) \vee p \implies (\neg p \vee \neg q \vee q \vee p) \wedge (\neg p \vee \neg q \vee r \vee p)$

Propiedades semánticas

- Sean L_1, \dots, L_n literales. Son equivalentes:

1. $\bigvee_{i=1}^n L_i$ es una tautología.

2. $\bigwedge_{i=1}^n L_i$ es inconsistente.

3. $\{L_1, \dots, L_n\}$ contiene un par de literales complementarios.

- Una fórmula en forma normal conjuntiva es una tautología syss cada una de sus disyunciones es una tautología.
- Una fórmula en forma normal disjuntiva es inconsistente syss cada una de sus conjunciones es inconsistente.

Algoritmos

- Algoritmo de inconsistencia mediante FND: *Entrada*: Una fórmula F .
Salida: Consistente, si F es consistente; Inconsistente, en caso contrario.
Procedimiento $INC_{FND}(F)$
 $G = G_1, \vee \dots \vee G_n \leftarrow FND(F)$
para $i = 1$ **hasta** n
 si en G_i no ocurren literales complementarios,
 entonces devolver consistente, **para**
 devolver inconsistente
- Algoritmo de validez mediante FNC:
Entrada: Una fórmula F
Salida: Tautología, si F es una tautología; No-tautología, en caso contrario.
Procedimiento $VAL_{FNC}(F)$
 $G = G_1 \wedge \dots \wedge G_n \leftarrow FNC(F)$
para $i = 1$ **hasta** n
 si en G_i no ocurren literales complementarios,
 entonces devolver No-tautología; **para**
 devolver Tautología

Ejemplos

- $F_1 = (p \wedge q) \rightarrow (q \wedge r) \vee p$. Su forma normal conjuntiva es

$$(\neg p \vee \neg q \vee q \vee p) \wedge (\neg p \vee \neg q \vee r \vee p)$$

- Es tautología (y por tanto consistente)

- $F_2 = \neg(p \vee q) \vee (p \rightarrow q)$. Su forma normal disyuntiva es:

$$(\neg p \wedge \neg q) \vee \neg p \vee q$$

- Es consistente

- Su forma normal disyuntiva es

$$(\neg p \vee q) \wedge (\neg q \vee \neg p \vee q)$$

- No es tautología

Deducción (II): Resolución

- Objetivo: Restringir la sintaxis del lenguaje para obtener eficiencia, trabajando con cláusulas y conjuntos de cláusulas (en notación conjuntista):

- La cláusula $L_1 \vee \dots \vee L_n$ (los L_i literales) se escribe como: $\{L_1, \dots, L_n\}$
- La fórmula en forma normal conjuntiva

$$(L_{1,1} \vee \dots \vee L_{1,n_1}) \wedge \dots \wedge (L_{m,1} \vee \dots \vee L_{m,n_m})$$

se escribe (*forma clausal*):

$$\{\{L_{1,1}, \dots, L_{1,n_1}\} \dots \{L_{m,1} \vee \dots \vee L_{m,n_m}\}\}$$

- La cláusula vacía (inconsistente) se denota por \square .
- Toda fórmula posee una forma clausal equivalente.
- Si dos fórmulas tienen una misma forma clausal, entonces son equivalentes.

Resolución entre cláusulas

- Si $L \in C_1$ y $L' \in C_2$ son literales complementarios, entonces la resolvente de C_1 y C_2 respecto a L es $(C_1 \setminus \{L\}) \cup (C_2 \setminus \{L'\})$
(es una regla de *inferencia*)

- Ejemplos
$$\frac{\{p, q, \neg r\}, \{\neg p, r, s\}}{\{q, \neg r, r, s\}} \quad \frac{\{p, q, \neg r\}, \{\neg p, r, s\}}{\{p, \neg p, q, , s\}}$$

- Definiciones de prueba y de teorema (por resolución) a partir de un conjunto de cláusulas
Notación: $\Gamma \vdash_r C$

- Teorema de adecuación:

- Si C es una resolvente de C_1 y C_2 , entonces $\{C_1, C_2\} \models C$
- Si C es un teorema (por resolución) a partir de Γ , entonces $\Gamma \models C$

Completitud de la refutación

- Si Γ es finito, el conjunto de teoremas por resolución a partir de Γ es finito y se puede obtener algorítmicamente: saturación de Γ por resolución

- Incompletitud de la resolución:

$$\{\{q\}\} \models \{q, r\} \quad \text{pero} \quad \{\{q\}\} \not\vdash_r \{q, r\}$$

- Refutación: Prueba por resolución de la cláusula vacía

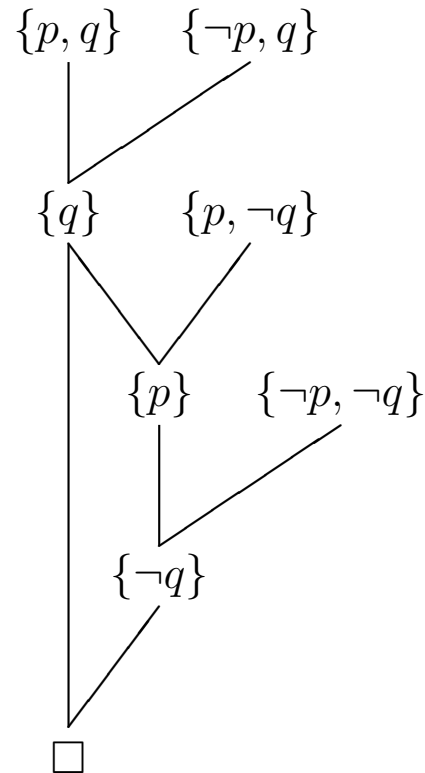
- Teorema de completitud de la refutación:

Si Γ es inconsistente, entonces $\Gamma \vdash_r \square$

- El problema de la consecuencia lógica: Para decidir si $\Gamma \models C$, se obtiene una forma clausal D de la negación de C y se intenta obtener una refutación a partir de $\Gamma \cup \{D\}$

Ejemplos

- $S = \{\{p, q\}, \{p, \neg q\}, \{\neg p, q\}, \{\neg p, \neg q\}\}$ es inconsistente;



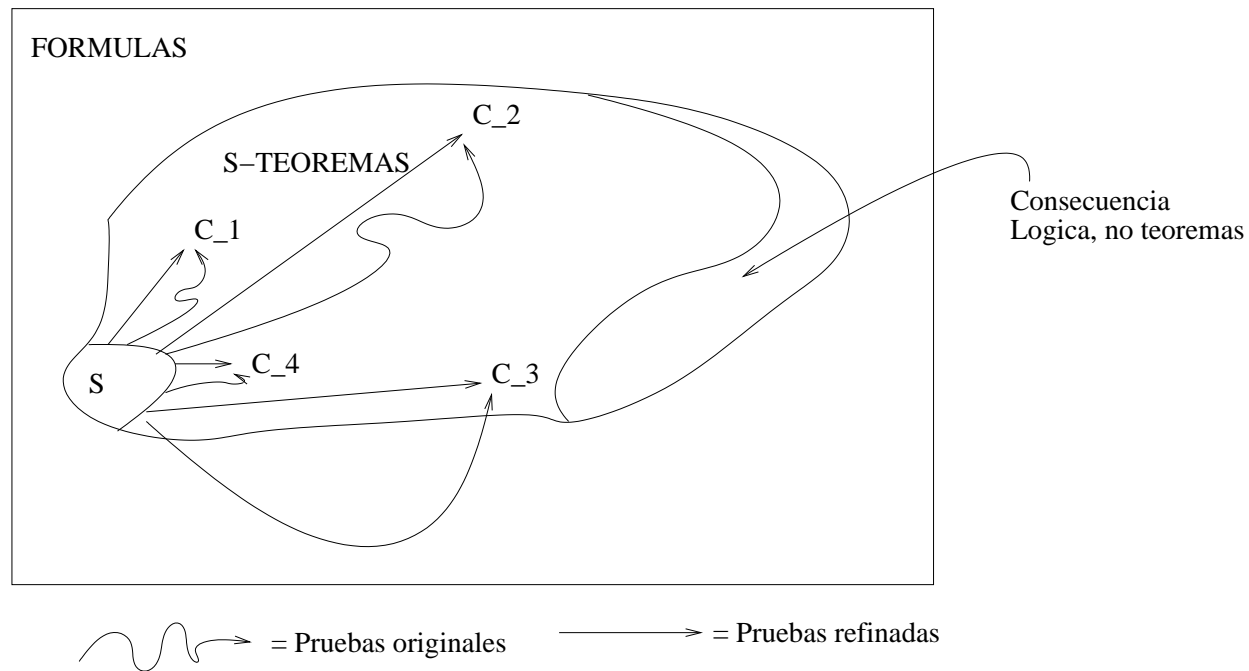
- Sea $S = \{\{p, \neg q\}, \{\neg p, q\}, \{\neg p, \neg q\}\}$. Se verifica que $S \models \{\neg q, r\}$, pues $S \cup \{\{q\}, \{\neg r\}\} \vdash_r \square$

Completitud y eficiencia

- Deducción: método mecánico de decidir la validez, consistencia, consecuencia lógica, etc.
- Completitud de la deducción (tablero, resolución, etc.):
 - Existe una prueba con ciertas propiedades (*hacia adelante*).
 - En el caso de la lógica proposicional, la existencia de la prueba es decidible, y el conjunto de pruebas es finito (si la base de conocimiento es finita): Saturación por resolución.
 - Eficiencia de la deducción: No es eficiente, en general
- Otra restricción al lenguaje: Cláusulas de Horn
 - Cláusulas con a lo sumo un literal positivo
 - El problema de decidir si un conjunto de cláusulas de Horn (proposicionales) es consistente es decidible de manera eficiente

Teoría de la demostración

- Existen diferentes estrategias para reducir el número o la complejidad de las pruebas, sin perder la adecuación y manteniendo la completitud



Opciones:

- (1) Aumentar el número de reglas de inferencia que usa el demostrador
 - Acorta la longitud de las pruebas
 - Justificar su adecuación
- (2) Reducir el ámbito de aplicación de las reglas
 - Acorta el tiempo de comprobación de la aplicabilidad
- (3) Heurísticas
 - Basadas en comprobaciones empíricas

Estrategias y refinamientos

- Opción (1): Aumentar el número de reglas, por ejemplo:

- Modus Ponens:
$$\frac{A \rightarrow B, A}{B}$$

- Eliminación de \wedge :
$$\frac{A_1 \wedge A_2}{A_1}$$

- Introducción del \wedge, \vee :
$$\frac{A_1, A_2}{A_1 \wedge A_2}, \quad \frac{A_1}{A_1 \vee A_2}$$

- Eliminación de la doble negación:
$$\frac{\neg\neg A}{A}$$

- Resolución unidad:
$$\frac{A \vee B, \neg B}{A}$$
 donde B es un literal.

- Nueva regla para la construcción del tablero: Si un nodo (no necesariamente terminal) posee un par complementario de fórmulas, entonces marcarlo como hoja cerrada

- Hiperresolución:
$$\frac{\{\neg A_1, \dots, \neg A_n, B_1, \dots, B_m\}, \{A_1\}, \dots, \{A_n\}}{\{B_1, \dots, B_m\}}$$

Estrategias y refinamientos (II)

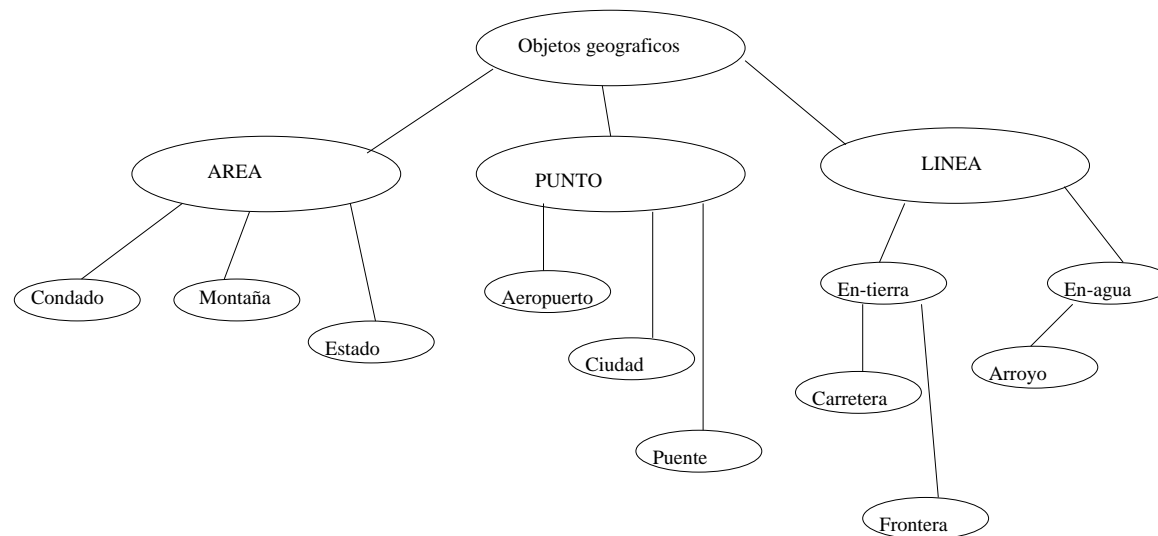
- Opción (2): Reducir el ámbito de aplicación de las reglas
 - Refinamientos para la resolución (a partir de S conjunto de cláusulas):
 - Resolución semántica: Fijada una interpretación v , sólo se permiten resolventes entre cláusulas con distinto valor de verdad con respecto a v . Es refutacionalmente completa.
 - Resolución positiva (resp. negativa): Una de las dos cláusulas a resolver tiene todos sus literales positivos (resp. negativos). Es refutacionalmente completa.
 - Resolución lineal: Las resolvente se hace entre la cláusula inmediatamente anterior en la prueba y cualquier otra de S o anterior en la prueba. Es refutacionalmente completo.
 - Resolución con soporte: Dado $T \subset S$ tal que $S \setminus T$ es consistente, sólo se permiten resolventes si una de las cláusulas pertenece a T . Refutacionalmente completa.
 - Resolución unidad: Una de las cláusulas es unitaria. No es refutacionalmente completa, pero sí lo es si nos restringimos a cláusulas de Horn.

Estrategias y refinamientos (III)

- Opción (3): Heurísticas
 - Tableros: Si es posible elegir, expandir antes con las α -reglas que con las β -reglas.
 - Si se *satura* el conjunto, no repetir resolventes
 - Estrategias generales:
 - Encadenamiento hacia adelante: Cuando se añade nuevos hechos, se aplica todas las posibles las reglas que posea el sistema.
 - Encadenamiento hacia atrás: El objetivo se descompone en sub-objetivos teniendo en cuenta las reglas y axiomas del sistema.

Representación del conocimiento (I)

- Representado relaciones entre conceptos:
 - Llueve → Suelo – mojado
- Representado una jeraquía, por ejemplo, una ontología geográfica:



Representación del conocimiento (II)

- Subsunción de objetos entre categorías

Puente \rightarrow Punto

Frontera \rightarrow En - tierra \wedge Línea

(si el objeto es frontera, entonces es una línea y está en tierra)

- Categorías disjuntas

Punto $\leftrightarrow \neg$ Area

- Deficiencia: No es posible expresar la propiedad: *Existe un objeto que es punto y no es línea* sin explicitar tal objeto

Observaciones

- La implicación no representa *pertenecer a*:

Dumbo es un elefante. Elefante es una especie animal. Entonces Dumbo es una especie

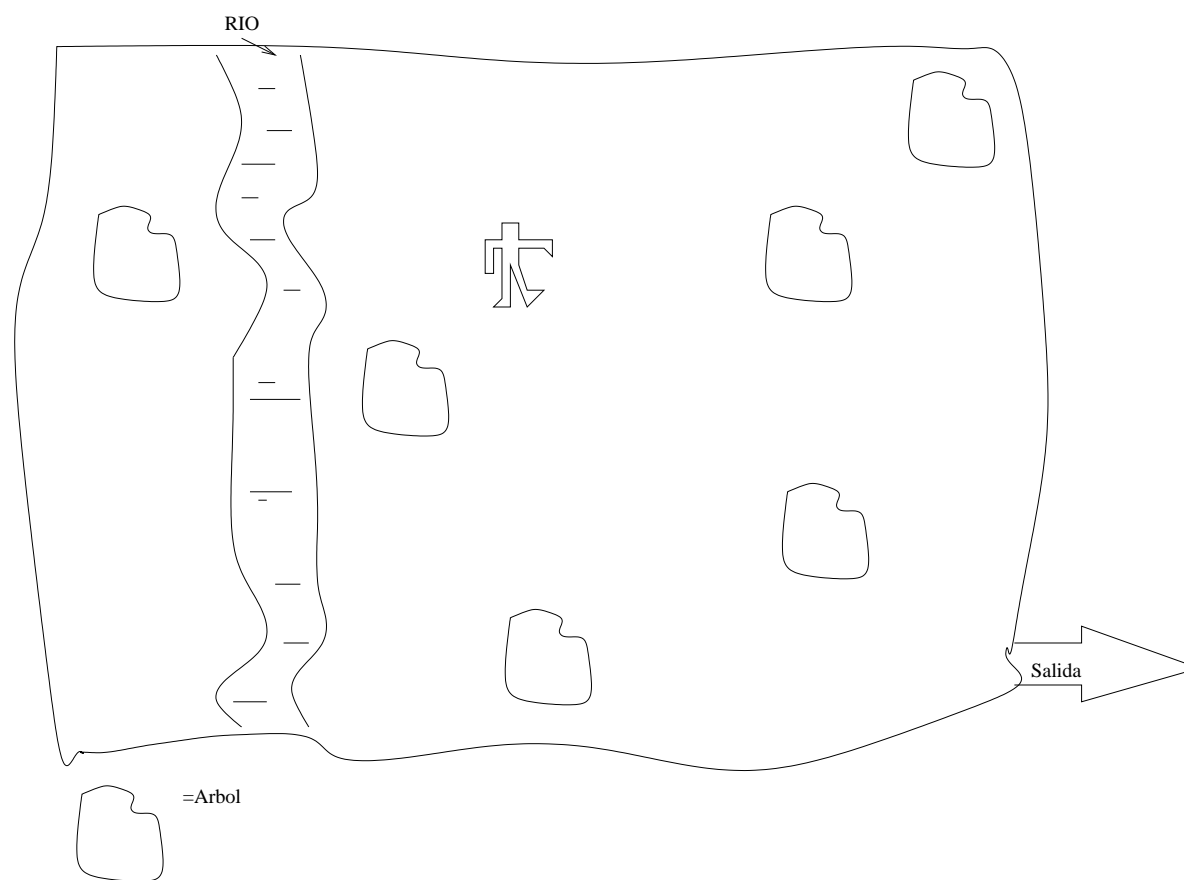
$$\{\text{Dumbo} \rightarrow \text{Elefante}, \text{Elefante} \rightarrow \text{Especie}\} \models \text{Dumbo} \rightarrow \text{Especie}$$

- Mal representado
- Buena opción: $\text{Dumbo} \in \text{Elefante}$
 - No es de primer orden
- Reglas: **Si Condición entonces Acción**
 - Hacer *predicativa* la acción
 - Hacer referencia al efecto: $\text{Acción} \rightarrow \text{efecto}$

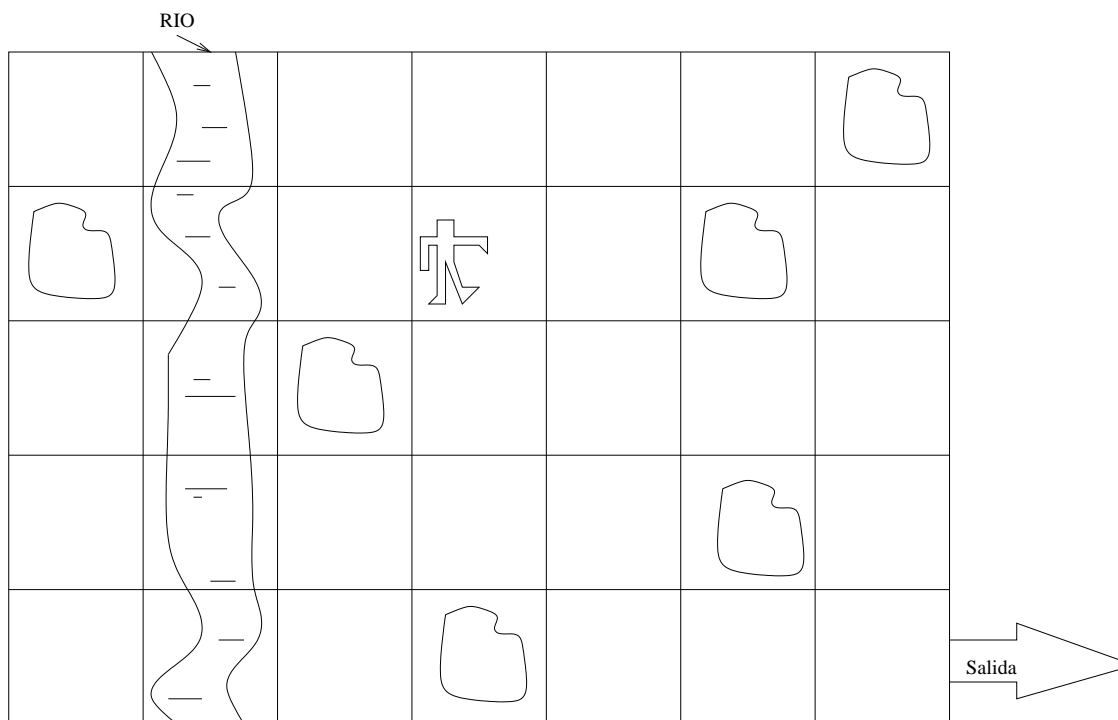
Ejemplo

- Ejemplo: Supongamos que un robot está navegando por un parque, y debe llegar a la salida de éste
 - Percepciones: Su entorno
 - Acciones: Deplazarse (paso a paso)
 - Objetivo: Alcanzar la salida
 - Entorno: Estático, discreto, determinista, episódico
 - Entorno absoluto: Guarda en memoria el mapa completo del lugar (accesible)
 - Entorno local: Guarda la información sólo de su entorno físico más próximo (efectivamente accesible) (veremos un ejemplo más adelante)
 - Obstáculos: Río, árboles

Gráfico



Representación del ejemplo



Representando el ejemplo (I)

- Variables:
 - A_{ij} = En la coordenada (i,j) hay un árbol
 - R_{ij} = Por la coordenada (i,j) pasa el río
 - $D_{i,j}$ = Acción de desplazarse i casillas horizontales, j verticales ($i, j \in \{-1, 0, 1\}$)
 - E_{ij} = El robot puede alcanzar (i,j)
- Base de conocimiento:
 - Hechos: $\{E_{24}, \neg A_{12}, \dots, A_{21}, \dots, R_{12}, \dots\}$
 - Condiciones: $\neg(E_{11} \wedge E_{12})$

Representando el ejemplo (II)

- Componente pensante:
 - Acciones (reglas E-C-A): $E_{34} \wedge \neg(R_{35} \vee A_{35}) \rightarrow D_{01}$
 - Restricciones: $\neg(A_{11} \wedge E_{11})$
 - Prohibiciones: $E_{11} \rightarrow \neg D_{-1,-1}$
 - Definiciones: La casilla (i,j) está libre (L_{ij})
 - $L_{ij} \leftrightarrow \neg(R_{ij} \vee E_{ij} \vee A_{ij})$
(j No es proposicional!)
 - Creencia: por ejemplo, $E_{11} \rightarrow L_{12}$ (¿falsa?)

Representando el ejemplo (III)

- Resultado de la acción (mandato a los sensores; movimiento):
 - $D_{10} \wedge E_{22} \rightarrow E_{32}$ (¡No es disparable!)
 - $E_{11} \wedge A_{01} \rightarrow E_{12}$
- Estado: $\{E_{ij}\}$ (entorno representado localmente)
- Estado: $\{E_{ij}, A_{21}, \dots\}$ (entorno global)
- La componente reactiva está formada por reglas E-C-A

Representando el ejemplo (IV)

- Componente racional:
 - Objetivo: E_{57}
 - Planificación: Obtener un conjunto adecuado de reglas del tipo:

$$E_{67} \wedge D_{10} \rightarrow E_{57}$$

para ejecutarlas ordenadamente

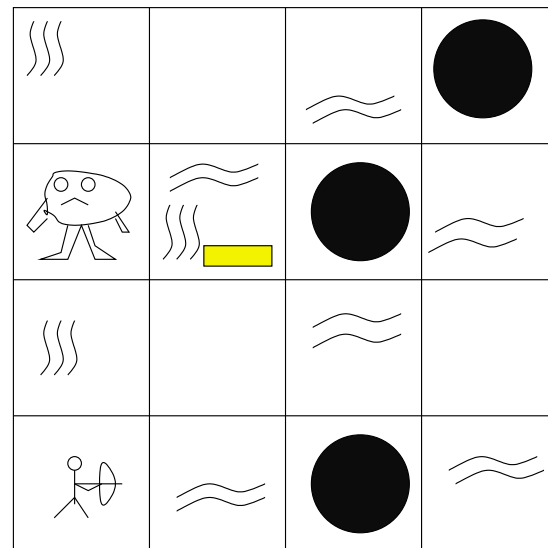
- Componente pro-activa: Si E_{34} entonces Objetivo : E_{44}
- Objetivo: E_{71}
- Ejercicio: Elegir una base de conocimiento K relativamente pequeña y *natural* de tal modo que $K \models E_{71}$
 - Se puede extraer de la prueba por tablero el camino que debe seguir el robot?

Limitaciones de la lógica proposicional para el problema

- Grado de autonomía:
 - Se puede diseñar un conjunto de reglas para alcanzar la salida, si se puede, para ese parque, independientemente del sitio donde se encuentra el robot
 - Si se cambia a otro parque, ya no sirve el sistema
- Expresividad deficiente: No es posible expresar: *Existen* (i, j) y (j', j') tales que $E_{ij} \wedge \neg E_{i'/j'}$
- En el ejemplo anterior, el tamaño de la base de conocimiento es $O(n)$ (siendo n el número de casillas), pues el entorno es accesible en su totalidad (con respecto al conocimiento del agente)

El mundo de Wumpus:

- El mundo de Wumpus:



↑
Salida



Descripción PAGE del agente cazador

- Percepciones (restringidas a la casilla que ocupa):
 - El agente percibe si en su casilla se encuentra el wumpus
 - En los cuadros adyacentes al wumpus, percibe su hedor
 - En los cuadros adyacentes a un pozo, percibe la brisa
 - Donde está el oro, percibe su brillo
 - Si avanza hasta un muro, percibe el choque
 - Cuando mata al wumpus, percibe un grito

La percepción es representable por cinco símbolos (hedor,brisa,brillo,choque,grito)

- Acciones: Avanzar, girar 90° grados a izq. o der., lanzar flecha (la flecha llega hasta el wumpus o a la pared), y *salir* (si se encuentra en la casilla de salida)
- El agente muere si entra en una cueva, u ocupa una casilla en la que está un wumpus vivo
- Objetivo: encontrar el oro y volver a la salida lo más rápidamente posible (vivo, claro)

Ejecución

- Supongamos que el agente ha visitado ya las casillas (1,1), (1,2) y (2,1), encontrándose en esta última:

<div style="border: 1px solid black; display: inline-block; padding: 2px;">A</div> visitada Hedor			
visitada	visitada Brisa	POZO	

↑
Salida

Representación

- La sucesión de percepciones ha sido (con respecto a (hedor,brisa,brillo,choque,grito)):
 $(0, 0, 0, 0, 0), (0, 1, 0, 0, 0), (1, 0, 0, 0, 0)$

- En la base de conocimiento tiene:

$$\left\{ \begin{array}{ll} \neg H_{11} & \neg B_{11} \\ \neg H_{21} & B_{21} \\ \neg H_{12} & \neg B_{12} \end{array} \right. + \left\{ \begin{array}{l} \neg H_{11} \rightarrow \neg W_{11} \wedge \neg W_{12} \wedge \neg W_{21} \\ \neg H_{21} \rightarrow \neg W_{11} \wedge \neg W_{21} \wedge \neg W_{22} \wedge \neg W_{31} \\ \neg H_{12} \rightarrow \neg W_{11} \wedge \neg W_{12} \wedge \neg W_{22} \wedge \neg W_{13} \\ H_{12} \rightarrow W_{13} \vee W_{12} \vee W_{22} \vee W_{11} \end{array} \right.$$

- Cuestión: ¿El agente puede deducir que Wumpus está en W_{13} ?
 - Ejercicio: Deducirlo por resolución

Ejemplo: Lenguaje natural

- Validar el razonamiento en lenguaje natural
 - Si Juan es comunista, entonces es ateo. Juan es ateo. Por tanto, Juan es comunista
 - $\{p \rightarrow q, q\} \not\models p$
 - Si la temperatura y la presión atmosférica permanecen constantes, entonces no llueve. La temperatura permanece constante. Por tanto, si llueve entonces la presión atmosférica no permanece constante.
 - $\{(t \wedge p) \rightarrow \neg l, t\} \models l \rightarrow \neg p$
 - Si un número x es divisible por 5, entonces acaba en 0. El número x no acaba en 0. Por tanto, x no es divisible por 5.
 - Si x es positivo, entonces y es negativo. Si z es negativo, entonces y es negativo. Por tanto, si x es positivo ó z es negativo, entonces y es negativo.

Ventajas y Limitaciones (I)

- La lógica proposicional es muy útil si no es importante el contenido de cada proposición, sólo es importante la estructura de la información
 - En estos casos, muy manejable
- Los problemas fundamentales son decidibles
- Es una lógica Bivalente, pero se puede alterar la semántica para obtener lógicas multi-valoradas
- En algunos casos no es importante que sean deducibles todas las tautologías

Ventajas y limitaciones (II)

- Los procedimientos basados en *demostración/teorema* no son directamente mecanizables
 - Razonamiento hacia delante
 - Es difícil encontrar una heurística
- La lógica proposicional NO se corresponde con la lógica de la demostrabilidad
 - $F \vee \neg F$ es una tautología, por tanto deducible por cualquier sistema completo. Sin embargo, no es posible decidir si F ó $\neg F$ son teoremas (en este caso, ninguno de las dos). Sin embargo, en un entorno sólo es cierta una de las dos
 - La lógica de la demostrabilidad (lógica intuicionista) es más adecuada para analizar el comportamiento de ciertos sistemas inteligentes
- Imposibilidad de distinguir entre objetos con respecto a una propiedad:
 - Existe x que verifica la propiedad p y existe y que no la verifica

Bibliografía

- M. Ben-Ari: Mathematical Logic for Computer Science, 2 ed. (2001)
- Russell & Norvig cap. 6
- Alonso Jiménez, Borrego Díaz: Deducción automática (Vol. 1: Construcción lógica de sistemas lógicos). Ed. Kronos 2002
 - Dispongo de una copia en el despacho