Representación del Conocimiento (1)

Carlos Hurtado L.

Depto de Ciencias de la Computación,
Universidad de Chile

Representación del Conocimiento

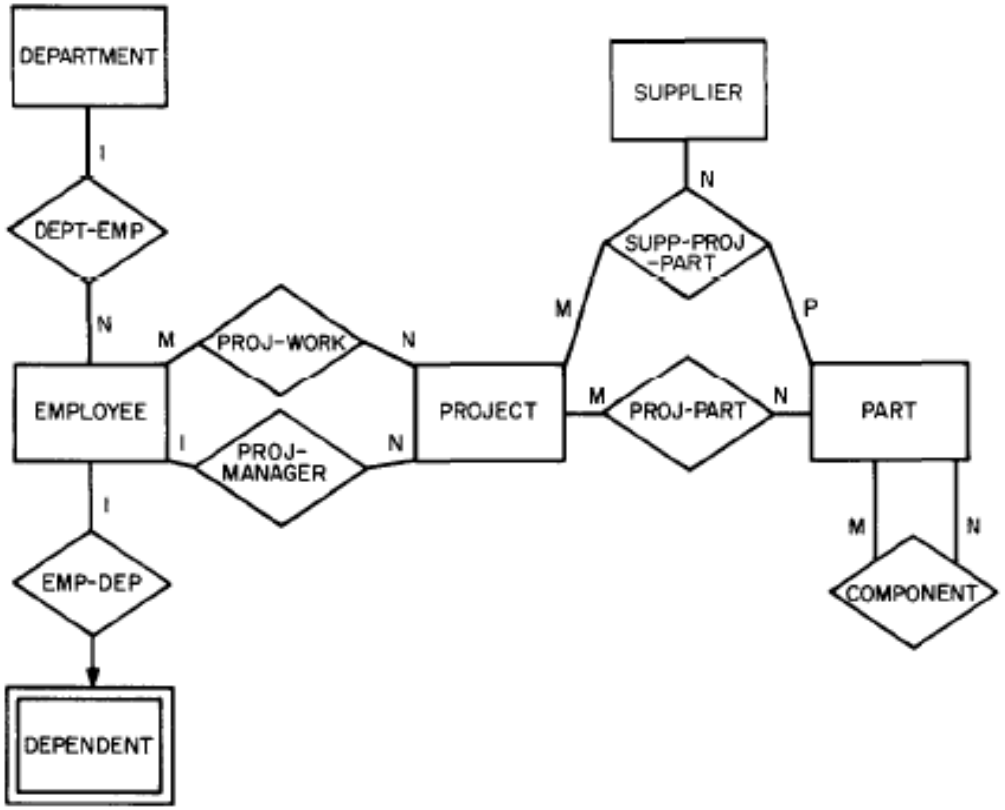
- Disciplina que estudia el problema representar conocimiento en sistemas computacionales
- Se centra en dos ideas fundamentales:
  - Representaciones que consisten en objetos explícitos (individuos, clases) de algún dominio y aserciones sobre ellos (relaciones, hechos).
  - Inferencia Lógica: uso de estas representaciones para deducir nuevo conocimiento.

Conocimiento

- Conocimiento:
  - Modelo de un dominio: Los objetos, conceptos y relaciones que se asumen existen en algún dominio de interés.
  - Lo que es verdadero en un dominio: "Knowledge consists in the perception of the truth of affirmative or negative propositions. --Locke.
- Propiedad fundamental del conocimiento:
  - nuevo conocimiento puede ser obtenido de conocimiento previo usando inferencia.
- Ejemplo: programa en Prolog con una colección de hechos y reglas sobre un dominio.

Ejemplo: Lenguaje para Representación de Concimiento

- Peter Chen. "The Entity-Relationship Model - Toward a Unified View of Data".

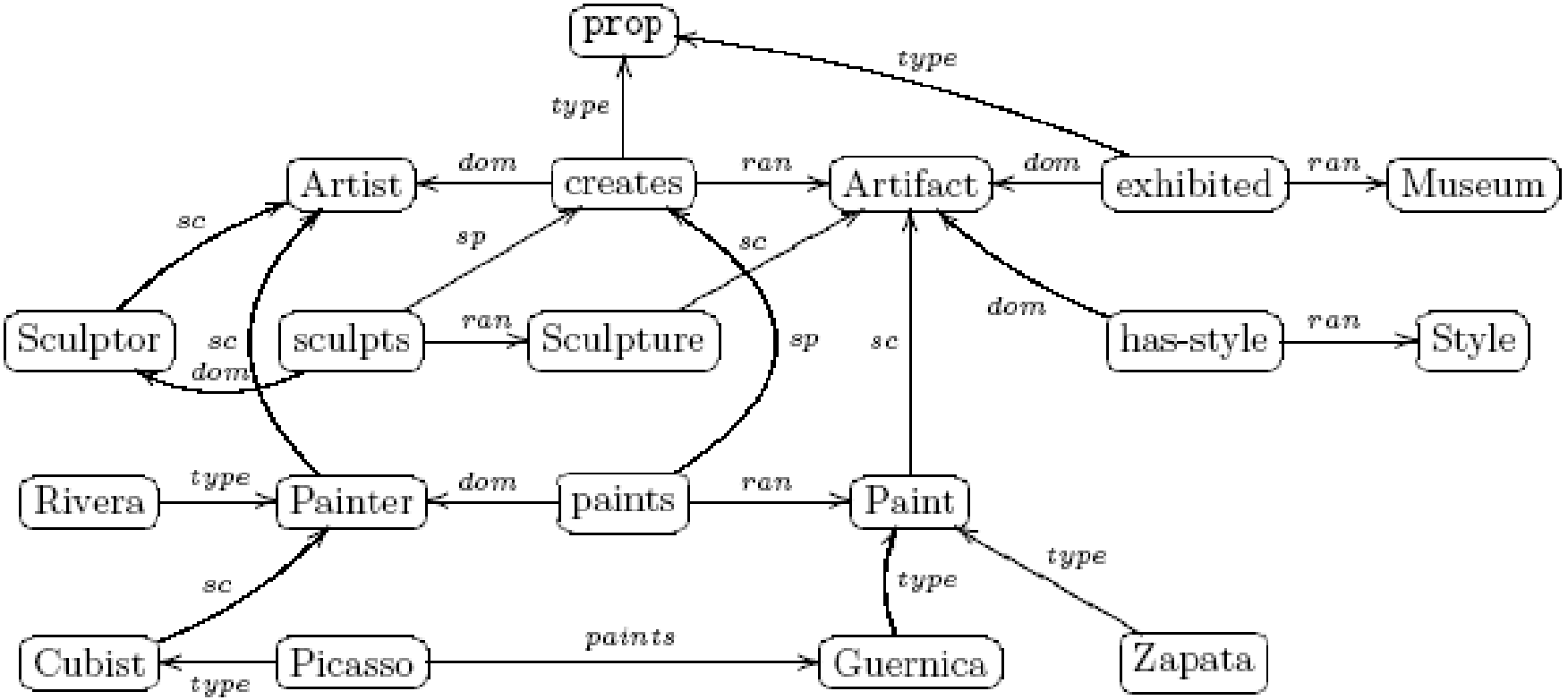


Lenguajes para Representación de Conocimiento

Knowledge representation formalisms and methods is the name of section I.2.4 of the ACM Computing Classification System.

- frames and scripts
- Modal logic
- Predicate logic
- Relation systems
- Representation languages
- Representations (procedural and rule-based)
- Semantic networks
- Temporal logic

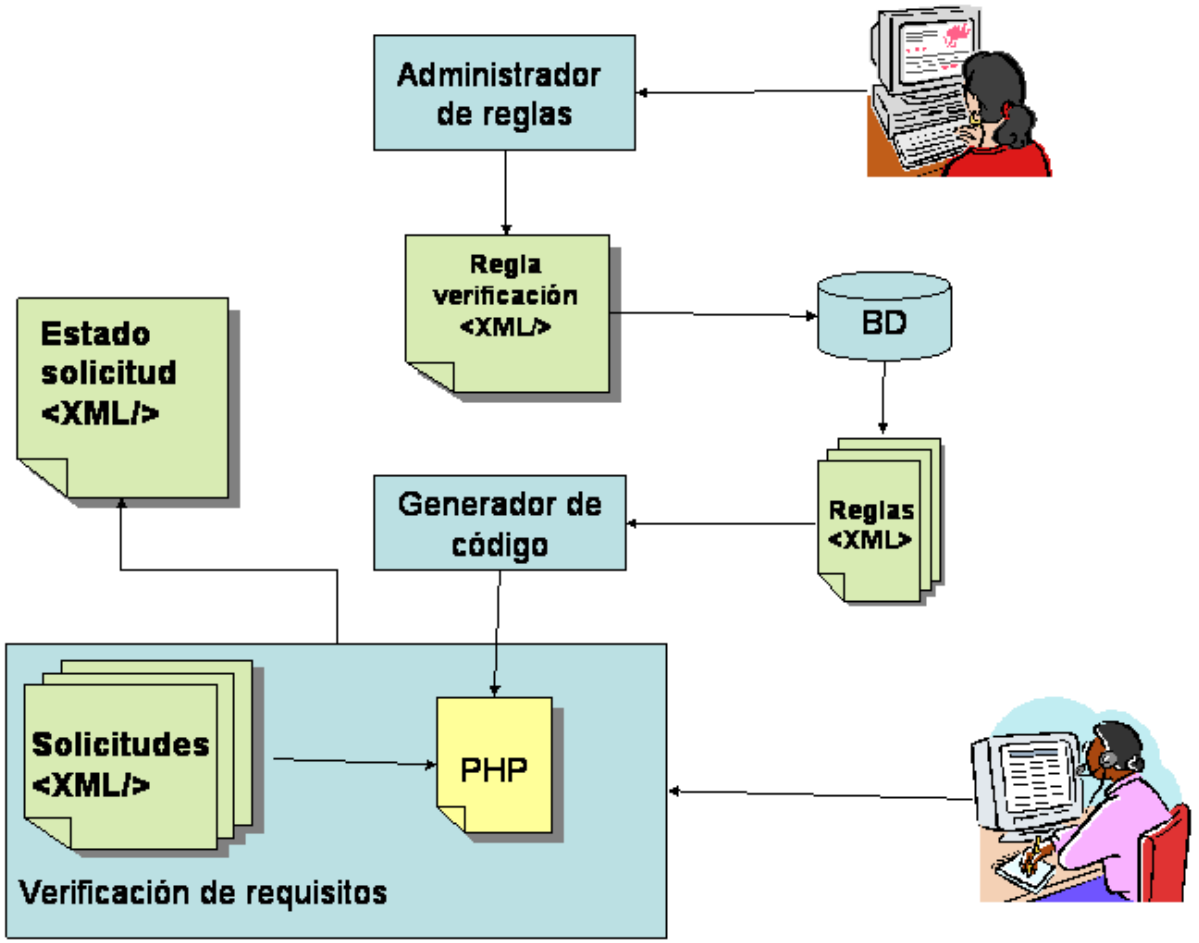
Ejemplo: RDF


prop
type
type
Artist
dom
creates
ran
Artifact
dom
exhibited
ran
Museum
sc
sp
sc
Sculptor
sc
dom
sculpts
ran
Sculpture
sp
sc
dom
has-style
ran
Style
Rivera
type
Painter
dom
paints
ran
Paint
sc
type
type
Cubist
type
Picasso
paints
Guernica
Zapata

Base de Conocimiento

- Colección de conocimiento, representado usando algún lenguaje formal
- Tiene asociado un programa para realizar inferencia y consultas a la base de conocimiento.
- Como ejemplo, una base de conocimiento sobre una familia debería contener hechos como:
  - Juan es hijo de Diego
  - Diego es hijo de Pedro
  - Si X hijo de Y e Y hijo de Z entonces X nieto de Z
  - Si X es hijo de Y e Z es hijo de Y entonces X hermano de Z

Ejemplo: Asignación de Garantías Estatales



Lenguajes Lógicos

- Un lenguaje lógico involucra:
  - Sintaxis: especifique cuáles son las sentencias correctas del lenguaje
  - Semántica: define el significado de una sentencia al asociar los elementos del lenguaje con los elementos de un dominio o mundo.
  - Inferencia: Mecanismo para deducir nuevas sentencias a partir de una BC
- Base de conocimiento (BC): conjunto de sentencias sobre un mundo

Ejemplo: Sintaxis del Lenguaje de la Aritmética

- Define *expresiones aritméticas* con *variables*, *valores*, *funciones* y *operadores*. Ejemplo:

$$x + y = 4$$

- Esta expresión aritmética pertenece al lenguaje de la aritmética, es decir, es una palabra aceptada por su gramática
- Sin embargo, la expresión $x2y = +$ no pertenece al lenguaje de la aritmética.

Ejemplo: Semántica del Lenguaje de la Aritmética

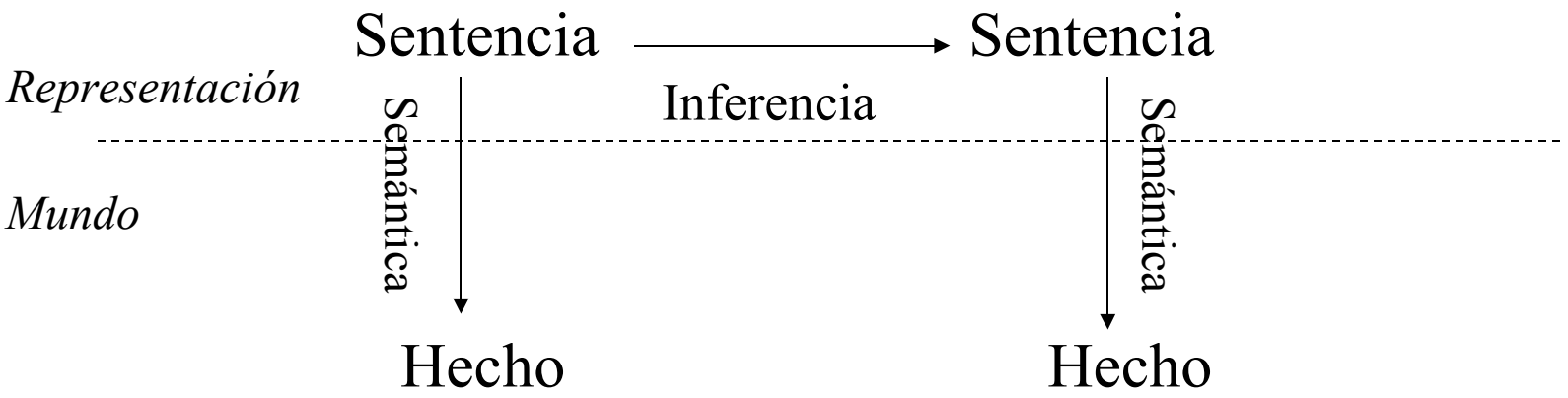
- La semántica usual de este lenguaje nos dice que

$$x + y = 4$$

es *verdadera* en un "mundo" donde $x = 2$ e $y = 2$.

- Sin embargo, es $x + y = 4$ es *falso* en un mundo donde $x = 1$ e $y = 1$.

Inferencia



$KB \models \alpha$

α es Consecuencia Lógica de KB

$KB \vdash \alpha$

α se deriva de KB usando reglas

Lenguajes Lógicos

Language	Ontological commitment (what exists in the world)
Propositional logic	Facts
First-order logic	Facts, objects, relations
Temporal logic	Facts, objects, relations, times
Probability theory	Facts
Fuzzy logic	Degree of truth

Cálculo proposicional

- Lenguaje lógico para describir y razonar sobre proposiciones.
- Proposiciones pueden representar "hechos" de un dominio
  - Ejemplos: el_gato_duerme, el_gato_come, el_raton_está_escondido, etc.

Sintaxis del Cálculo Proposicional

- Las sentencias se construyen en base a los siguientes conjuntos de símbolos
  - Atomos o variables proposicionales: P,Q,R,S,…
  - Constantes: *Verdadero, Falso*
  - Conectivos: $\neg$, $\wedge$,$\vee$, $\Leftrightarrow$, $\Rightarrow$
  - Paréntesis: ()
- Definición de una sentencia (fórmula bien formada):
  - Todo átomo o constante es una sentencia
  - Si A y B son sentencias, las siguientes tb. lo son:

  $\neg A$, $(A \wedge B)$, $(A \vee B)$.
- Abreviaciones:

  $(A \Rightarrow B)$ es una abreviación de $(\neg A \vee B)$

  $(A \Leftrightarrow B)$ es una abreviación de $((A \Rightarrow B) \wedge (B \Rightarrow A))$
- Precedencia para eliminar paréntesis: $\neg$, $\wedge$,$\vee$, $\Leftrightarrow$, $\Rightarrow$
  - Ejemplo: $\neg P \vee Q \wedge R$ es equivalente a $((\neg P) \vee (Q \wedge R))$

Semántica del Cálculo Proposicional

- El mundo al que se refiere una sentencia se representa como una función I: Atomos → {V,F}
- Una interpretación se puede extender para asignar V o F a sentencias como sigue:
  - I(Verdadero)=V
  - I(Falso)=F
  - I(¬A)=V sii I(A)=F
  - I(A $\wedge$ B)=V sii I(A)=V y I(B)=V
  - I(A $\vee$ B)=V sii I(A)=V o I(B)=V (se cumple al menos una de las dos)
- Es decir, dada una sentencia S y una interpretación I, denotamos I(S) al valor de verdad de S.

Ejemplo

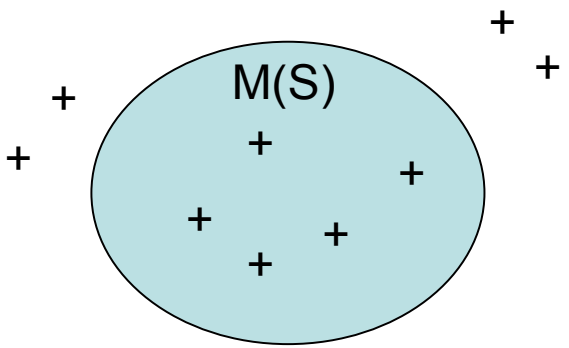
$$S = (A \wedge B) \vee (\neg A \wedge B)$$

I(A)	I(B)	I(S)
F	F	F
V	F	F
F	V	V
V	V	V

Modelos de S

Modelos

- Una interpretación I satisface una sentencia S ssi I(S)=V.
  - En este caso decimos que I es un modelo de S.
  - Denotamos como M(S) al conjunto de modelos de S.



Satisfabilidad y Validez

- S es satisfacible ssi existe una interpretación I tal que I(S)=V
  - S tiene al menos un modelo
- S es válida (o es tautología) ssi para toda interpretación I, I(S) =V.
  - En este cas, todas las interpretaciones son modelos de S

Consecuencia Lógica

- Una sentencia S es *consecuencia lógica* de una sentencia P, denotado P $\models$ S ssi

$$M(P) \subseteq M(S)$$

- Ejemplos:

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

$$Verdadero \not\models (A \wedge B) \vee (\neg A \wedge B)$$

- S es *equivalente* a P ssi P $\models$ S y S $\models$ P

Equivalencia Lógica

$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge$$
$$(\alpha \vee \beta) \equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee$$
$$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge$$
$$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee$$
$$\neg(\neg\alpha) \equiv \alpha \quad \text{double-negation elimination}$$
$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition}$$
$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination}$$
$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}$$
$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad \text{de Morgan}$$
$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad \text{de Morgan}$$
$$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee$$
$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge$$

Problema de Inferencia

- El problema fundamental en toda lógica es verificar si $P \models S$
- Teorema de Deducción

  $P \models S$ ssi $(P \Rightarrow S)$ es válida
- A partir de esto tenemos

  $P \models S$ ssi $(P \wedge \neg S)$ es insatisfacible

Métodos para Deducción

- Verificación de Modelos:
  - **Fuerza Bruta**: enumerar todas las interpretaciones para determinar validez de $(P \Rightarrow S)$
  - **Búsqueda de Modelos**: buscar un modelo para $(P \wedge \neg S)$

Ejemplo: Fuerza Bruta

$$KB = (A \vee C) \vee (B \vee \neg C) \quad \alpha = (A \vee B)$$

¿ $KB \models \alpha$?

A	B	C	$A \vee C$	$B \vee \neg C$	KB	α
$False$	$False$	$False$	$False$	$True$	$False$	$False$
$False$	$False$	$True$	$True$	$False$	$False$	$False$
$False$	$True$	$False$	$False$	$True$	$False$	$True$
$False$	$True$	$True$	$True$	$True$	$True$	$True$
$True$	$False$	$False$	$True$	$True$	$True$	$True$
$True$	$False$	$True$	$True$	$False$	$False$	$True$
$True$	$True$	$False$	$True$	$True$	$True$	$True$
$True$	$True$	$True$	$True$	$True$	$True$	$True$

Métodos para Deducción (cont.)

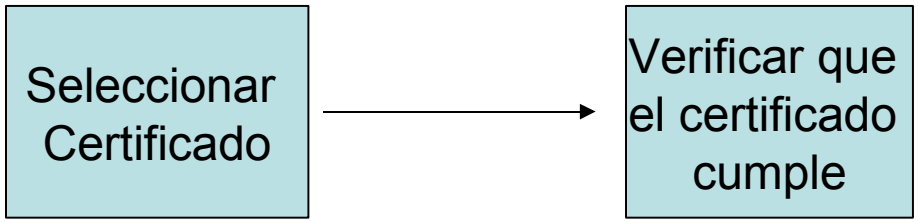
- Procedimientos de prueba:
  - Tenemos un conjunto de reglas de deducción
  - Buscar una demostración desde P a S.
  - Procedimiento de prueba: algoritmo que genera demostraciones.
- Prueba por refutación: buscar una demostración de $(P \wedge \neg S)$ a Falso.

Complejidad de Inferencia en Cálculo Proposicional

- Método de fuerza bruta requiere verificar 2n interpretaciones para inferencia que involucre n variables proposicionales.
- ¿Se puede hacer más eficiente?
- Cook, 1971, demuestra que 3SAT es NP-completo.
- 3SAT: es satisfacible $(x1 \vee x5 \vee x6) \wedge (x2 \vee \neg x5 \vee x6)$ …?

Problemas NP

¿ Es $(x1 \vee x5 \vee x6) \wedge (x2 \vee \neg x5 \vee x6)$ satisfacible ?



Interpretación

Se hace en forma eficiente.

Problema fundamental no resuelto: ¿P = NP?

Complejidad de Inferencia en Cálculo Proposicional

- Bajo el supuesto usual de que P ≠ NP, satisfacibilidad no se puede resolver "eficientemente" (en tiempo polinomial).
- Si encontramos una forma de resolverlo en forma eficiente, entonces resolveriamos en forma eficiente todos los problemas NP-completos.
  - Vendedor Viajero, Problema de la Mochila, Programación entera, diversos problemas de grafos, criptografía, etc.

Procedimiento de Prueba: Regla de Inferencia

Ejemplos:

Modus Ponens: $\dfrac{\alpha \Rightarrow \beta, \alpha}{\beta}$

Eliminación de $\wedge$: $\dfrac{\alpha \wedge \beta}{\alpha}$

Introducción de $\wedge$: $\dfrac{\alpha, \beta}{\alpha \wedge \beta}$

Demostración

- Una demostración desde KB a α es usando un conjunto de reglas R es una secuencia de sentencias w1, w2, …, wn donde
  - Cada w_i pertenece a KB o puede ser inferida por una regla de R aplicada a w_k,w_j tales que j,k<i.
  - Tenemos $w_n = \alpha$
- Decimos que $KB \vdash_R \alpha$ ssi existe una demo desde KB a α usando R

Ejemplo Demostración

- KB={ P, R, $P \Rightarrow Q$ }, $\alpha = Q \wedge R$. Una demo de KB a α usando las reglas anteriores sería:

$$P, P \Rightarrow Q, Q, R, Q \wedge R$$

- Esta demo se puede ver como un árbol.

Propiedades de Reglas de Inferencia

Un conjunto de reglas de inferencia R es:

- Correcto ssi $KB \vdash_R \alpha$ implica $KB \models \alpha$
- Completo ssi $KB \models \alpha$ implica $KB \vdash_R \alpha$
- Ejemplo:
  - ¿qué regla es trivialmente completa para cálculo proposicional?
  - ¿qué regla es trivialmente correcta para CP?

Ejemplo: Modus Ponens (MP)

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

- ¿Es MP correcto?
  - Sí. Ejercicio: demostrarlo
- ¿Es MP completo?

Ejemplo: Modus Ponens (MP)

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

- ¿Es MP correcto?
  - Sí. Ejercicio: demostrarlo
- ¿Es MP completo?
  - No. Contraejemplo:
  - A partir de $\alpha \Rightarrow \beta, \beta \Rightarrow \delta$ no podemos demostrar $\alpha \Rightarrow \delta$ usando MP