Métodos de Inteligencia Artificial

L. Enrique Sucar (INAOE)

esucar@inaoep.mx

ccc.inaoep.mx/esucar

Tecnologías de Información

UPAEP

Contenido

- Lógica proposicional
- Lógica de predicados
- Inferencia en lógica
- Representación de conocimiento en lógica

Lógica:

Lenguaje que permite expresar conocimiento y razonar a partir de ciertas expresiones para deducir otras (deducción)

Características:

- sintaxis y semántica bien definidas
- reglas de inferencia

Lógica Proposicional

Permite expresar y razonar con declaraciones que son o verdaderas o falsas

Ejemplos:

- la clase de IA es lo mejor que me ha pasado en mi vida
- lógica es fácil

Lógica Proposicional

Este tipo de declaraciones se llaman *proposiciones* y se denotan en lógica proposicional con letras mayúsculas

(v.g., *P*,*Q*,…)

P's y *Q*'s también se llaman *proposiciones atómicas* o *átomos*

Los átomos se pueden combinar con *conectores lógicos* (dando proposiciones compuestas)

negación: $\sim$, $\neg$

conjunción: &, $\wedge$

disjunción: $\vee$

implicación: $\supset$, $\rightarrow$

doble implicación: $\leftrightarrow$

Por ejemplo,
G = “esto ya lo ví”
D = “me estoy aburriendo”

$G \wedge D$ = “esto ya lo ví” Y “me estoy aburriendo”

Sólo algunas combinaciones de átomos y conectores son permitidas: *fórmulas bien formadas (wff)*

Una *wff* en lógica proposicional es una expresión que puede ser de la siguiente forma:

1. Un átomo es un *wff*
2. Si *F* es *wff* entonces $\neg F$ también lo es
3. Si *F* y *G* son *wff* entonces:

$F \wedge G, F \vee G, F \rightarrow G$ y $F \leftrightarrow G$ son *wff*

4. Ninguna otra fórmula es *wff*

Por ejemplo: $F \vee (G \rightarrow H)$ y $F \wedge \neg G$ son *wff*, mientras que: $\rightarrow H$ y $\wedge G$ no lo son

wff es sólo sintáxis, no dice si la fórmula es verdadera o falsa (i.e., no dice nada de su semántica)

El significado de una fórmula proposicional se puede expresar por medio de una función:

$$w{:}prop \rightarrow \{\ verdadero\ (true), falso\ (false)\ \}$$

La función *w* es una *función de interpretación* que satisface:

F	G	$\neg F$	$F \wedge G$	$F \vee G$	$F \rightarrow G$	$F \leftrightarrow G$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

$$w(\neg F) = true \text{ si } w(F) = false$$

$$w(\neg F) = false \text{ si } w(F) = true$$

...

Si *w* es una interpretación que asigna a una fórmula dada el valor de verdad (*true)*, entonces *w* se dice ser un *modelo* de *F*

Una fórmula se dice *válida* si es verdadera bajo cualquier interpretación (tautología)

Por ejemplo:

P V¬ P $\qquad$ $((P \rightarrow Q) \wedge P) \rightarrow Q$

Una fórmula es *inválida* si no es *válida*

Una fórmula es i*nsatisfascible* o *inconsistente* si es falsa bajo cualquier interpretación (contradicción) sino, es *satisfascible* o *consistente* , e.g.:

Por ejemplo: $P \wedge \neg P$ y $(P \rightarrow Q) \wedge (P \wedge \neg Q)$ son insatisfascibles.

Una fórmula es válida cuando su negación es insatisfascible y viceversa

<table>
<tr><td>válido</td><td colspan="2">inválido</td></tr>
<tr><td>siempre cierto</td><td>a veces T o F</td><td>siempre falso</td></tr>
<tr><td colspan="2">satisfacible</td><td>insatisfascible</td></tr>
</table>

Dos fórmulas F y G son equivalentes ($F \equiv G$) si los valores de verdad de F y G son iguales bajo cualquier interpretación

Existen muchas leyes de equivalencias, por ejemplo: $F \rightarrow G \equiv \neg F \vee G$

Una fórmula *G* se dice que es una *consecuencia lógica* de un conjunto de fórmulas : $F = \{F_1 \ldots, F_n\}$, denotado por $F \models G$

si para cada interpretación *w* para la cual

$$w(F_1 \wedge F_2 \wedge \ldots F_n) = true$$

entonces $w(G) = true$

Satisfacibilidad, validez, equivalencia y consecuencia lógica son nociones semánticas (generalmente establecidas por medio de tablas de verdad)

Para derivar consecuencias lógicas también se pueden hacer por medio de operaciones exclusivamente sintácticas (*reglas de derivación*).

Reglas de Derivación

- Modus ponens

$$A, A \rightarrow B \models B$$

- Modus tollens

$$\neg B, A \rightarrow B \models \neg A$$

Lógica de predicados de primer orden

En lógica proposicional los átomos son los constituyentes de las fórmulas y son: *verdaderos* o *falsos*

Limitación: no puede expresar propiedades generales de casos similares.

Por ejemplo, "todos los alumnos de Met. de I.A. se están durmiendo"

Símbolos:

- Símbolos de predicados (mayúsculas) asociados con su aridad (N) o número de argumentos (Si aridad = 0 $\Longrightarrow$ proposiciones (átomos))

- Variables: minúsculas (x,y,z)

- Símbolos funcionales: minúsculas asociados con su número de argumentos (funciones con aridad = 0 $\Longrightarrow$ constantes)

- Conectores lógicos
- Cuantificadores: universal (para toda x) $\forall x$ y existencial (existe una x) $\exists x$
- Símbolos auxiliares '(', ')', ','.

Un *término* es: una constante, variable o una función de términos

Una *fórmula atómica* o *átomo* es un predicado de N términos

Una fórmula bien formada (*wff*) en lógica de predicados es:

- un átomo
- si *F* es *wff* entonces $\neg F$ también lo es
- Si *F* y *G* son *wff*, $F \wedge G$, $F \vee G$, $F \rightarrow G$, $F \leftrightarrow G$

son *wff*

- Si *F* es *wff* y x es una variable libre en *F*, entonces $\forall x$ *F* y $\exists x$ *F* son *wff* (la variable x se dice *acotada* o "*bounded*")

- ninguna otra fórmula es *wff*

Ejemplo:
$\forall x$ *F(x) --> G(x)*

Semántica

En lógica de primer orden se asocia una estructura representando la “realidad” (básicamente el dominio)

La *estructura* *S* tiene:

- un conjunto no vacío de elementos *D*, llamados el dominio de *S*

- un conjunto de funciones de aridad n definidas en $D^n, \{f_i^n : D^n \rightarrow D\}$
- un conjunto no vacío de mapeos, predicados, de D^m a $\{true, false\}$

No se puede saber el valor de verdad de una fórmula hasta que no se especifique con qué elementos de la estructura se deben de asociar los elementos de la fórmula

Una *asignación* ν al conjunto de fórmulas *F* dada una estructura *S* con dominio *D* es un mapeo del conjunto de variables en *F* a *D*

$\exists xF$ es *true* si existe una asignación para la cual *F* sea verdadera

$\forall xF$ es *true* si para toda asignación *F* es verdadera

Una fórmula cerrada con un modelo se dice *satisfascible*

Ejemplo:

$P = C(x) \longrightarrow A(x)$

$D = \{$tubería, caldera, pipa, …$\}$

C = componente hidráulico

A = transporta agua

C (tubería) = T, C(caldera) = T, C (pipa) = F

A (tubería) = T, A(caldera) = F, A(pipa) = T

Para las asignaciones de x =tubería y pipa, P = T, para x =caldera, P = F

e.g., (2)

Cláusulas

Forma utilizada en prueba de teoremas y programación lógica

Una literal: un átomo o su negación
Una cláusula: es una fórmula cerrada de la forma:

$$\forall x_1 \ldots \forall x_s (L_1 \vee \ldots \vee L_m)$$

Equivalencias:

$$\forall x_1 \ldots \forall x_s (A_1 \vee \ldots A_n \vee \neg B_1 \ \ldots \ \vee \neg B_m) \equiv$$

$$\forall x_1 \ldots \forall x_s (B_1 \wedge \ldots \wedge B_m \rightarrow A_1 \vee \ldots A_n)$$

Se escribe normalmente como:

$$A_1, \ldots, A_n \leftarrow B_1, \ldots, B_m$$

Interpretación procedural: las *A*'s son las conclusiones y las *B* 's las condiciones

Razonamiento en lógica: reglas de inferencia

Existen varias reglas de inferencia, por ejemplo, *Modus Ponens*.

Estas reglas sólo hacen manipulación sintáctica (son formas procedurales)

Lo interesante es ver como las formas procedurales sintácticas están relacionadas con las semánticas

Una fórmula es *robusta / válida* (*sound*) si
S |- *F* entonces S ⊨ *F*

Una colección de reglas de inferencia es válida si preserva la noción de verdad bajo las operaciones de derivación

Una fórmula es *completa (complete)* si *S* ⊨ *F* entonces *S* |- *F*

Modus Ponens es *sound:*

$$\{P \rightarrow Q, P\} \vdash Q$$

ya que bajo cualquier interpretación:

$$\{P \rightarrow Q, P\} \models Q$$

pero no es *complete*:

$P \vee Q, \neg Q \models P$, pero no $\vdash P$ usando *modus ponens.*

Lo importante es: ¿existe un procedimiento de prueba mecánica, usando una colección de reglas de inferencia que son válidas y completas, que sea capaz de determinar si una fórmula F puede o no derivarse de un conjunto de fórmulas S?

En 1936, Church y Turing mostraron independientemente que ese procedimiento no existe para lógica de primer orden: *indecibilidad*

Sólo se puede mostrar si se sabe que *F* es consecuencia lógica de *S* (semi-decidible).

Lógica proposicional si es decidible.

Resolución

Es *sound* y *refutation complete*

Resolución sólo sirve para fórmulas en forma de cláusulas

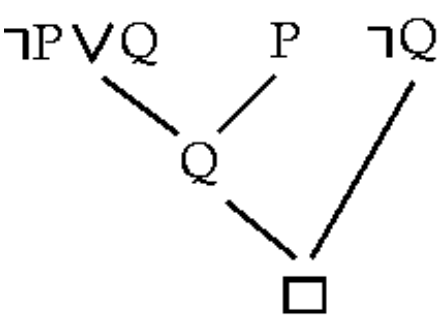
Idea: prueba por refutación

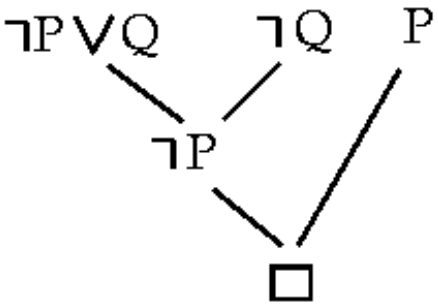
Para probar: P |- Q , hacer $W = P \cup \{\neg Q\}$
y probar que *W* es insatisfascible

EJEMPLOS DE DERIVACIÓN

Modus Ponens:

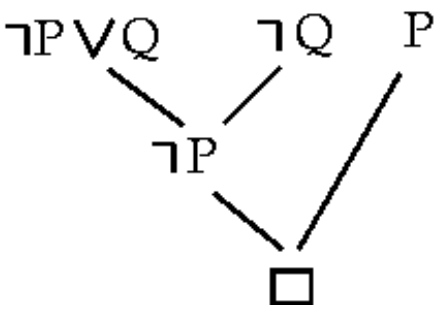
$$\frac{P \rightarrow Q \qquad P}{Q}$$

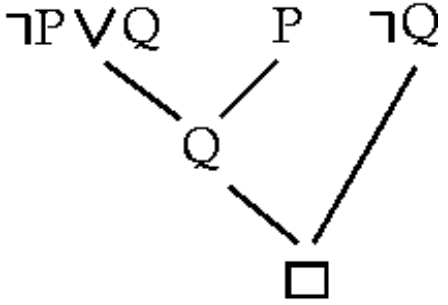




Modus Tollens:

$$\frac{P \rightarrow Q \qquad \neg Q}{\neg P}$$





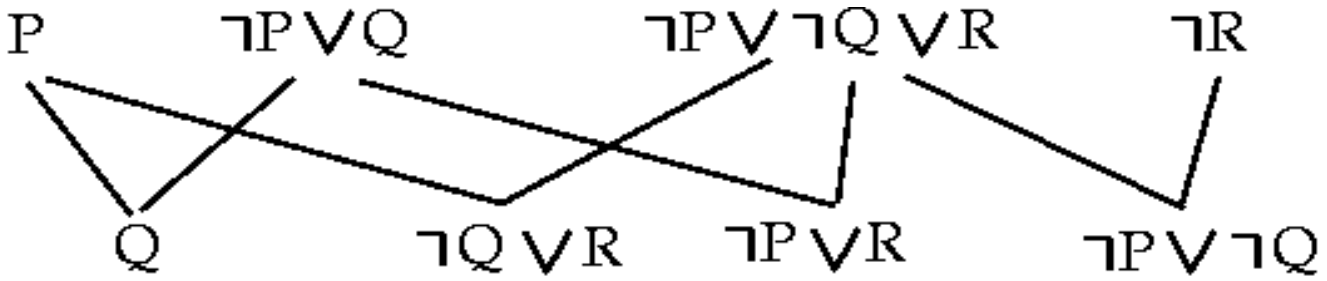
Para lógica de primer orden: *substitución* y *unificación*

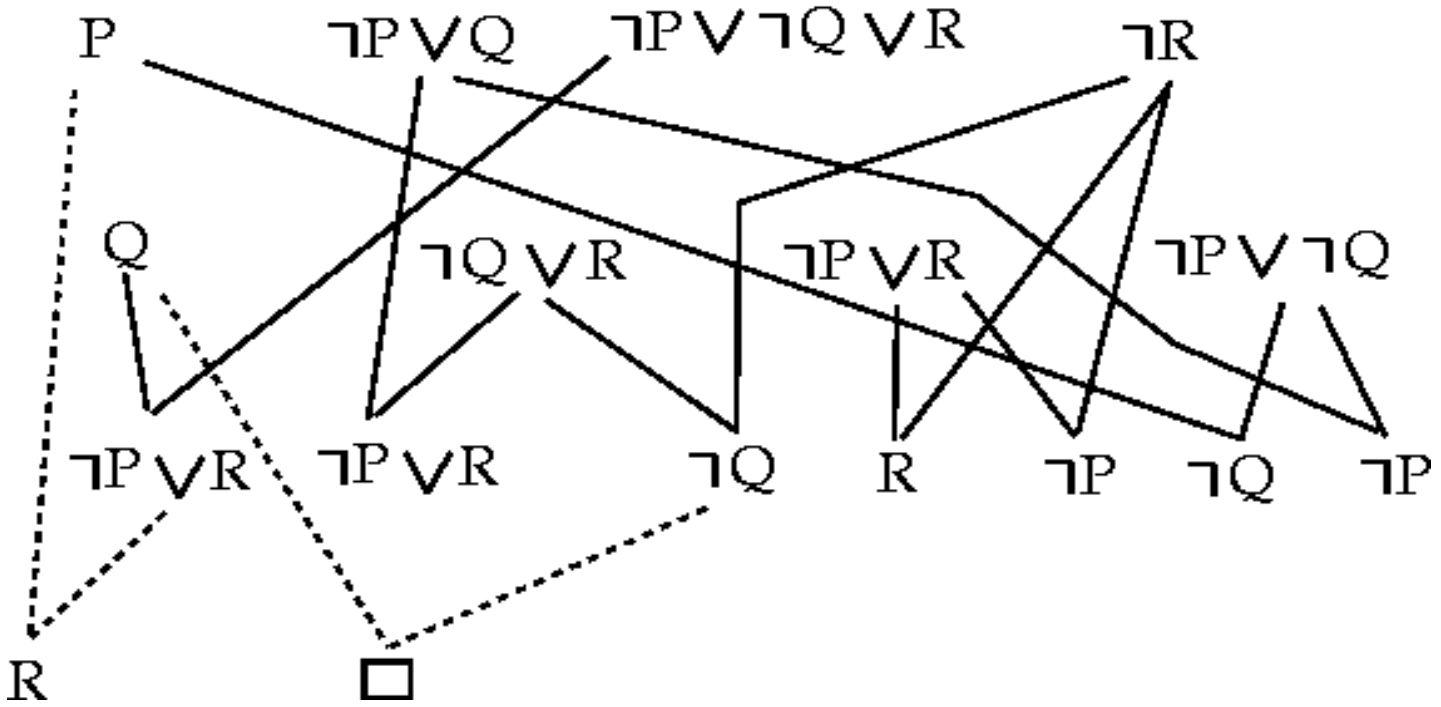
Una *substitución* σ es un conjunto finito de la forma: $\{t_1/x_1,\ldots,t_n/x_n\}$, donde las x_i son variables diferentes y las t_i son términos diferentes a las x_i

Para hacer resolución en lógica de primer orden tenemos que comparar si dos literales complementarias *unifican*.

El algoritmo de *unificación* construye el unificador más general (*mgu)* de un conjunto de expresiones, e.g. (2),

RESOLUCION: TODOS VS. TODOS





Una de las estrategias de resolución más utilizadas en programación lógica es la que utiliza Prolog.

La idea es tomar la primera meta, seleccionar la primera cláusula con quien se pueda unificar, y añadir el cuerpo de esa cláusula *al frente* de la lista de metas (variante de estrategia SLD).

En escencia está haciendo una búsqueda en profundidad con *backtracking* (ahorro de memoria).

Aunque resolución SLD es *sound* y (*refutation*) *complete* para cláusulas de Horn, en la práctica (por razones de eficiencia) se hacen variantes.

Esto es lo que usa básicamente PROLOG

EJEMPLO DE REPRESENTACION EN LÓGICA:

Plantas Eléctricas

Lógica como representación de conocimiento

Si se quiere realmente representar conocimiento, i.e., correspondencia entre las expresiones y el mundo real, cualquier formalismo debe de tener una semántica bien definida.

En este sentido lógica es la técnica de representación de conocimiento en donde más se ha trabajado al respecto.

Se requiere definir qué atributos lógicos se requieren para una representación de propósito general.

Un atributo básico de lógica sería: representar al mundo en términos de objetos, sus propiedades y relaciones (donde un objeto puede ser casi cualquier cosa).

- La cuantificación existencial permite decir que algo tiene una propiedad sin especificar cual
- La cuantificación universal permite decir que todos tienen una propiedad sin tener que enumerarlos
- La disjunción nos permite decir que al menos una de dos (o más) expresiones es verdadera sin tener que especificar cual

- La negación nos permite distinguir entre saber que algo es falso o no saber si es verdadero

- Podemos tener diferentes expresiones sin saber que se refieren al mismo objeto a menos que lo digamos por medio de igualdad

Algunos de los atributos son generales y deben de estar en cualquier representación de cualquier dominio.

El problema no está en la lógica o en la deducción, pero en saber qué inferencias hacer (el espacio de búsqueda crece exponencialmente con el número de fórmulas).

Otro punto importante es que muchas veces la eficiencia depende de cómo formalizar las cosas y el tipo de razonamiento que se utiliza

Resumiendo:

En general lógica es adecuada, lo que se requiere son mejores procesos deductivos y/o extensiones a la lógica más que pensar en desecharla.

Lógica proposicional es en general poco expresiva. Sin embargo, existe una gran cantidad de sistemas bajo esta representación.

Por ejemplo, árboles de falla, árboles de decisión, muchos de los sistemas expertos que se usan en la actualidad, aplicaciones en circuitos lógicos, etc.

Lógica de primer orden es, en general, suficientemente expresiva, pero el método de razonamiento es NP-completo y la lógica es indecidible

Cláusulas de Horn, aunque menos expresivas, son generalmente adecuadas.

Problemas de lógica de primer para representar conocimiento

- difícil expresar todo en fórmulas lógicas
- razonar con tiempo, meta-inferencia
- información incompleta o imprecisa
- excepciones

Tarea

- Leer Capítulos 7 (hasta 7.5) y 8 de Russell
- Representar en lógica de predicados el conocimiento sobre un tema (cada quien lo selecciona, no repetir!), incluyendo al menos 5:
  - Predicados
  - Funciones
  - “Reglas”
  - Constantes