Sintaxis y Semántica de la Lógica Proposicional

Dpto. Ciencias de la Computación e Inteligencia Artificial
UNIVERSIDAD DE SEVILLA

Contenido

Introducción
Lógica Proposicional
Sintaxis
Fórmulas
Inducción sobre fórmulas
Semántica
Funciones de verdad
Valoraciones
Consecuencia lógica y satisfactibilidad
Problemas de decisión
Sistemas deductivos
Reglas y Hechos
Encadenamieto Adelante
Encadenamieto Atrás
Limitaciones

Problema básico

Dados un conjunto de **afirmaciones** (hechos, hipótesis,...), $\mathcal{BC}$, y una afirmación, A, decidir si A ha de ser necesariamente cierta, suponiendo que todos las afirmaciones de $\mathcal{BC}$ lo son.

La **Lógica** proporciona formulaciones precisas de este problema y diferentes soluciones.
Para abordar este problema formalmente hemos de:

1. Dar un lenguaje para expresar las afirmaciones (*representación*).
2. Concretar qué entendemos por *afirmación cierta*.
3. Proporcionar mecanismos efectivos para garantizar la *corrección de las deducciones*.

A lo largo de este curso estudiaremos estas cuestiones en los dos casos más comunes: la **Lógica Proposicional**, y la **Lógica de Primer Orden**.

Lógica Proposicional

Características generales de la Lógica Proposicional:

- Sus expresiones (fórmulas) representan *afirmaciones* que pueden considerarse *verdaderas* o *falsas*.
- Se construyen a partir de expresiones básicas usando operadores (*conectivas*).
- Las conectivas se corresponden con formas sencillas de construir afirmaciones complejas en el lenguaje natural partiendo de otras más sencillas:
  - Conjunción: "...tal ... **y**... cual..."
  - Disyunción: "...tal ... **o**... cual ..."
  - Implicación "**Si** ...tal ... **entonces**... cual ..."
  - Negación: "**No** es cierto que tal ..."
- Sólo permite analizar las formas de razonamiento ligadas a este tipo de construcciones.

El Lenguaje de la Lógica Proposicional

Formalmente, el *lenguaje de la Lógica Proposicional* consta de:

1. Un conjunto numerable de **variables proposicionales**: $VP = \{p, p_0, p_1, \ldots, q, q_0, q_1, \ldots, r, r_0, \ldots\}$
2. Operadores básicos de construcción, **Conectivas lógicas**:
   - De aridad 1: $\neg$ (negación).
   - De aridad 2: $\vee$ (disyunción), $\wedge$ (conjunción), $\rightarrow$ (condicional) y $\leftrightarrow$ (bicondicional).
3. **Símbolos auxiliares**: "(" y ")".

Fórmulas

El conjunto de las *fórmulas proposicionales*, **PROP**, es el menor conjunto de expresiones que verifica:

- $VP \subseteq \mathbf{PROP}$,
- Es cerrado bajo las conectivas, es decir:
  - Si $F \in \mathbf{PROP}$, entonces $\neg F \in \mathbf{PROP}$.
  - Si $F, G \in \mathbf{PROP}$, entonces $(F \vee G)$, $(F \wedge G)$, $(F \to G)$, $(F \leftrightarrow G) \in \mathbf{PROP}$.

La sintaxis del lenguaje pretende evitar la ambigüedad en la interpretación de las fórmulas. Esa es la función de los símbolos auxiliares.

Árboles de formación

- Asociamos a cada fórmula un *árbol de formación* (esencialmente único) que describe el modo en que se construye la fórmula a partir de otras más sencillas.
- **Ejemplo**:

$$\neg(\neg(p \lor q) \to (\neg r \land s))$$

$$(\neg(p \lor q) \to (\neg r \land s))$$

$$\neg(p \lor q) \qquad (\neg r \land s)$$

$$(p \lor q) \qquad \neg r \qquad s$$

$$p \qquad q \qquad r$$

- Las fórmulas que aparecen en el árbol de formación de una fórmula F se denominan **subfórmulas** de F.

Reducción de paréntesis

Para facilitar la lectura de las fórmulas adoptaremos los siguientes convenios de notación:

1. Omitiremos los paréntesis externos.
2. Daremos a las conectivas una precedencia de asociación. De mayor a menor, están ordenadas por: $\neg, \wedge, \vee, \rightarrow$.
   - **Ejemplo**: $F \wedge G \rightarrow \neg F \vee G$ es $((F \wedge G) \rightarrow (\neg F \vee G))$.
3. Siempre se dejarán los paréntesis para la conectiva $\leftrightarrow$.
4. Cuando una conectiva se usa repetidamente, se asocia por la derecha: $F \vee G \vee H$ es $(F \vee (G \vee H))$.

Principio de Inducción sobre fórmulas

Gracias a la definición de **PROP** si deseamos probar que toda fórmula proposicional satisface cierta propiedad Ψ, podemos probarlo por **inducción sobre fórmulas**.

Para ello probamos:

1. **Caso base:** Todos los elementos de VP tienen la propiedad Ψ.
2. **Paso de inducción:**
   2.1 Si $F \in$ **PROP** tiene la propiedad Ψ, entonces $\neg F$ tiene la propiedad Ψ.
   2.2 Si $F, G \in$ **PROP** tienen la propiedad Ψ, entonces las fórmulas $(F \vee G)$, $(F \wedge G)$, $(F \to G)$ y $(F \leftrightarrow G)$ también tienen la propiedad Ψ.

Funciones de verdad

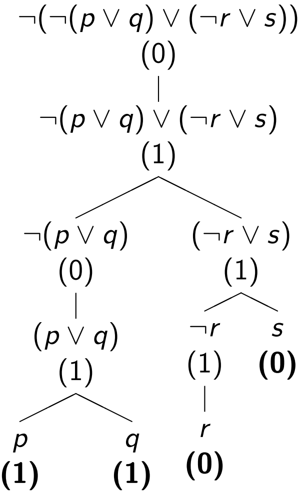
- Los elementos del conjunto $\{0,1\}$ se llaman **valores de verdad**. Se dice que 0 es el valor **falso** y el 1 es el valor **verdadero**.
- El significado de una conectiva se determina mediante su **función de verdad**:
  - $H_{\neg}(i) = \begin{cases} 1, & \text{si } i = 0; \\ 0, & \text{si } i = 1. \end{cases}$
  - $H_{\vee}(i,j) = \begin{cases} 0, & \text{si } i = j = 0; \\ 1, & \text{en otro caso.} \end{cases}$
  - $H_{\wedge}(i,j) = \begin{cases} 1, & \text{si } i = j = 1; \\ 0, & \text{en otro caso.} \end{cases}$
  - $H_{\rightarrow}(i,j) = \begin{cases} 0, & \text{si } i = 1, j = 0; \\ 1, & \text{en otro caso.} \end{cases}$
  - $H_{\leftrightarrow}(i,j) = \begin{cases} 1, & \text{si } i = j; \\ 0, & \text{en otro caso.} \end{cases}$

Valoraciones

- Las variables proposicionales se interpretan mediante una **valoración de verdad** (o interpretación), es decir, una aplicación
$$v : VP \to \{0, 1\}$$
- Podemos extender cada valoración, v, **de forma única**, al conjunto de todas las fórmulas de manera que para cada fórmula F se verifique:
  - $v(\neg F) = H_\neg(v(F))$.
  - $v((F \vee G)) = H_\vee(v(F), v(G))$.
  - $v((F \wedge G)) = H_\wedge(v(F), v(G))$.
  - $v((F \to G)) = H_\to(v(F), v(G))$.
  - $v((F \leftrightarrow G)) = H_\leftrightarrow(v(F), v(G))$.
- Se dice que $v(F)$ es el **valor de verdad** de F respecto de v.

Valor de verdad

Veamos el cálculo de $v(\neg(\neg(p \vee q) \vee (\neg r \vee s))$ en el árbol de formación (de abajo a arriba):



Tablas de verdad

Dada una valoración, v, el valor de verdad de una fórmula F respecto de v está determinado por los valores de verdad de las subfórmulas de F.

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

$$\begin{aligned} v(\neg((p \to q) \vee r)) &= H_{\neg}(H_{\vee}(v(p \to q), v(r))) = \\ &= H_{\neg}(H_{\vee}(H_{\to}(v(p), v(q)), 1)) = 0 \end{aligned}$$

Fijada v podemos presentar el cálculo de F mediante una tabla que recorre los valores de sus subfórmulas:

p	q	r	$p \to q$	$(p \to q) \vee r$	$\neg((p \to q) \vee r)$
0	0	1	1	1	0

Una **tabla de verdad** para F es una tabla similar que contiene una fila por cada posible valoración que asigne valores a las variables proposicionales que aparecen en F.

Validez y satisfactibilidad (I)

- Decimos que una fórmula F es **válida en** v, o que v es un **modelo** de F, si $v(F) = 1$.
  - Notación: $v \models F$.
  - Una valoración v es *modelo* de un conjunto de fórmulas U, $v \models U$, si v es modelo de todas las fórmulas de U.
- Una fórmula F es una **tautología** (o **válida**) si es válida para toda valoración (notación $\models F$).
- Una fórmula F es **satisfactible** (o consistente) si existe una valoración que es modelo de F. En caso contrario diremos que es **insatisfactible** (o inconsistente).
  - Análogamente, un conjunto de fórmulas U es satisfactible (o consistente) si existe una valoración que es modelo de U. En caso contrario diremos que es insatisfactible (o inconsistente).

Validez y satisfactibilidad (II)

Relación entre ambos conceptos:

Lema. Para cada $F \in \mathbf{PROP}$ se verifica:

- Si F es un tautología entonces F es satisfactible.
- F es una tautología si y sólo si $\neg F$ insatisfactible.

Ejemplos:

- Son tautologías: $(p \vee \neg p)$ y $((p \rightarrow q) \rightarrow p) \rightarrow p$.
- $p \wedge \neg p$ es insatisfactible y, por tanto, $\neg(p \wedge \neg p)$ es una tautología.
- $(p \rightarrow q) \rightarrow p$ es satisfactible pero no es una tautología.

Consecuencia Lógica

- Una fórmula F es **consecuencia lógica** de un conjunto de fórmulas U, si todo modelo de U es modelo de F. Es decir, para toda valoración, v,

$$v \models U \quad \Longrightarrow \quad v \models F$$

  - Notación: $U \models F$.
  - La relación de consecuencia lógica permite formular el problema básico en el marco de la lógica proposicional.

Relación entre consecuencia lógica, consistencia y validez:

Proposición. Sea $\{F_1, \dots F_n\} \subseteq \mathbf{PROP}$. Son equivalentes:

- $\{F_1, \dots, F_n\} \models F$
- $F_1 \wedge \cdots \wedge F_n \rightarrow F$ es un tautología.
- $\{F_1, \dots, F_n, \neg F\}$ es insatisfactible.

Algoritmos de decisión (I)

Dado un conjunto de fórmulas proposicionales, U, un **algoritmo de decisión** para U es un algoritmo que dada $A \in PROP$, devuelve `SI` cuando $A \in U$, y `NO` si $A \notin U$.

Casos especialmente interesantes:

- **SAT** $= \{A \in PROP : \; A$ es satisfactible$\}$
- **TAUT** $= \{A \in PROP : \; A$ es una tautología$\}$
- Fijado $U \subseteq PROP$, la **Teoría de** U es

$$\mathcal{T}(U) = \{A \in PROP : \; U \models A\}$$

Un algoritmo de decisión para $\mathcal{T}(U)$ propociona una respuesta al Problema Básico expuesto al principio del tema.

Algoritmos de decisión (II)

Problema Básico:
Obtener un algoritmo que, dado un conjunto finito de fórmulas proposicionales, U, y una fórmula F, decida si $U \models F$.

El problema anterior se reduce a decidir la **satisfactibilidad** de una cierta fórmula (o si se prefiere, la **validez** de otra). Por tanto,

- La construcción de tablas de verdad proporciona un algoritmo (ineficiente) para decidir la consecuencia lógica.
- El Problema Básico es resoluble algorítmicamente, aunque no se conoce ninguna solución eficiente y se duda de la existencia de algoritmos de decisión eficientes para este problema, ya que ...
- ... determinar la satisfactibilidad de una fórmula proposicional es un problema **NP-completo**.

Algoritmos de decisión (III)

Problema Básico (bis):
Obtener un algoritmo <u>eficiente</u> que, dado un conjunto finito de fórmulas proposicionales, U, y una fórmula F, decida si $U \models F$.

Observaciones:

- Este problema es equivalente al de obtener un algoritmo eficiente para determinar la satisfactibilidad de una fórmula proposicional.
- Se trata de un **problema abierto**, que posiblemente tendrá una respuesta negativa (se cree que no existen algoritmos eficientes para resolver SAT).
- Para propósitos prácticos puede bastar con algoritmos eficientes para alguna clase especial de fórmulas.

Sistemas deductivos

Un **sistema deductivo**, **T**, (o teoría proposicional) consta de:

- Un conjunto, $Ax(\mathbf{T})$, de fórmulas proposicionales que llamamos <u>los axiomas de **T**</u>.
- <u>Reglas de inferencia</u> de la forma:

$$\frac{A_1, \ldots, A_n}{A}$$

siendo $A_1, \ldots, A_n, A$ fórmulas proposicionales. Las fórmulas $A_1, \ldots, A_n$ se denominan *premisas* y la fórmula A *conclusión*.

Demostraciones y Teoremas

Definición.
Una **demostración** en $\mathbf{T}$ es una sucesión de fórmulas proposicionales $A_1, \ldots, A_k$ cada una de las cuales es un axioma de $\mathbf{T}$, o bien, se obtiene a partir de fórmulas anteriores de la sucesión mediante la aplicación de una regla de inferencia.

Definición.
Una fórmula A es un **teorema** de $\mathbf{T}$, $\vdash_{\mathbf{T}} A$, si existe una demostración en $\mathbf{T}$, $A_1, \ldots, A_k$ tal que $A = A_k$. La sucesión $A_1, \ldots, A_k$ se denomina una **demostración de** A en $\mathbf{T}$.

Ejemplo: Sistema deductivo

- $Ax(\mathbf{T}) = \{p,\ q,\ p \wedge q \to (\neg s \vee p \to r)\}$
- Reglas de inferencia: (A y B fórmulas cualesquiera)

$$I_\wedge : \quad \frac{A,\ B}{A \wedge B} \qquad I_\vee : \quad \frac{A}{A \vee B}$$

$$C_\vee : \quad \frac{A \vee B}{B \vee A} \qquad MP : \quad \frac{A,\ A \to B}{B}$$

La siguiente sucesión es una demostración de r en $\mathbf{T}$ ($\vdash_{\mathbf{T}} r$):

1.	p	$[\![$Hip.$]\!]$
2.	q	$[\![$Hip.$]\!]$
3.	$p \wedge q$	$[\![I_\wedge$ aplicada a 1. y 2.$]\!]$
4.	$p \wedge q \to (\neg s \vee p \to r)$	$[\![$Hip.$]\!]$
5.	$\neg s \vee p \to r$	$[\![MP$ aplicada a 3. y 4.$]\!]$
6.	$p \vee \neg s$	$[\![I_\vee$ aplicada a 1.$]\!]$
7.	$\neg s \vee p$	$[\![C_\vee$ aplicada a 6.$]\!]$
8.	r	$[\![MP$ aplicada a 7. y 5.$]\!]$

Procedimientos de demostración

- Un sistema deductivo introduce una noción de prueba formal, pero no proporciona ningún procedimiento efectivo para generar demostraciones de las fórmulas que deseamos probar.
- Un **sistema de razonamiento** proporciona, además de un sistema deductivo, un **procedimiento de demostración**, que implementa una estrategia de búsqueda de demostraciones.
- En general, no podemos encontrar estrategias que sean efectivas para todos los sistemas deductivos, y habrá que buscar estrategias que funcionen solo para casos particulares.
- Ilustraremos un ejemplo de estrategia eficiente en el caso del encadenamiento con cláusulas de Horn.

Sistemas de Reglas de Disparo y Hechos

- Un **hecho** es una fórmula formada por una sola variable proposicional.
- Una **regla de disparo** es una fórmula de la forma:

$$R: \; V_1 \wedge \cdots \wedge V_n \to V$$

  donde $V_1, \ldots, V_n, V$ son variables proposicionales.
  - Al conjunto $\{V_1, \ldots, V_n\}$, se le llama **cuerpo** de la regla, R_b.
  - A V, se le llama **cabeza** de la regla, R_h.
  - Obsérvese que los hechos se pueden ver como casos particulares de reglas de disparo donde $R_b = \emptyset$.
- Si tenemos un conjunto de hechos, **H**, una regla es **disparable** en **H** si $R_b \subseteq \mathbf{H}$.
  - Por ejemplo, $P \wedge Q \to S$ es disparable en $\{A, B, P, Q\}$.

Encadenamiento hacia adelante

Consideraremos el siguiente sistema deductivo, $EA(\Sigma)$, para trabajar con reglas y hechos.

- **Axiomas**: Un conjunto finito, Σ, de reglas y hechos.
- **Reglas de inferencia**: Si A y B son conjunciones de literales, entonces:

$$I_\wedge : \quad \frac{A,\ B}{A \wedge B} \qquad MP : \quad \frac{A,\ A \to B}{B}$$

Teorema. Se tiene que, para todo **hecho**, H

$$\Sigma \models H \quad \Longleftrightarrow \quad \vdash_{EA(\Sigma)} H$$

Ejemplo: deducción en $EA(\Sigma)$

$$\Sigma = \{P \to Q,\ L \wedge M \to P,\ B \wedge L \to M,\ A \wedge P \to L,\ A \wedge B \to L,\ A,\ B\}$$

Veamos que $\vdash_{EA(\Sigma)} Q$,

1.	A	[[Hip.]]
2.	B	[[Hip.]]
3.	$A \wedge B$	[[$I_\wedge$ aplicada a 1. y 2..]]
4.	$A \wedge B \to L$	[[Hip.]]
5.	L	[[MP aplicada a 3. y 4.]]
6.	$B \wedge L$	[[$I_\wedge$: 2. y 5.]]
7.	$B \wedge L \to M$	[[Hip.]]
8.	M	[[MP: 6. y 7.]]
9.	$L \wedge M$	[[$I_\wedge$: 5. y 8.]]
10.	$L \wedge M \to P$	[[Hip.]]
11.	P	[[MP: 9. y 10.]]
12.	$P \to Q$	[[Hip.]]
13.	Q	[[MP: 11. y 12.]]

Algoritmo para EA

Podemos diseñar un algoritmo de decisión eficiente para demostrar hechos en $EA(\Sigma)$, que se denomina **encadenamiento hacia adelante**, y que consiste en ir *disparando* las reglas disparables y comprobar en cada disparo si obtenemos el hecho buscado.

- Entrada:
  - Un hecho, G.
  - Un conjunto Σ formado por reglas, $\mathbf{R}$, y hechos, $\mathbf{H}$.
- Salida:
  - SI, si $\vdash_{EA(\Sigma)} G$.
  - NO, en caso contrario.
- Algoritmo:
  1. Si $G \in \mathbf{H}$, parar y devolver SI.
  2. Sea $\mathbf{D}$ las reglas de $\mathbf{R}$ disparables en $\mathbf{H}$:
     2.1 Si $\mathbf{D} = \emptyset$, parar y devolver NO.
     2.2 Si no, tomar $\mathbf{R} = \mathbf{R} - \mathbf{D}$ y $\mathbf{H} = \mathbf{H} \cup \{R_h : \ R \in \mathbf{R}\}$, y volver a 1.

Ejemplo: encadenamiento hacia adelante

Veamos que $\vdash_{EA(\Sigma)} Q$,

H	**D**
A, B	$A \wedge B \rightarrow L$
A, B, L	$B \wedge L \rightarrow M$
A, B, L, M	$L \wedge M \rightarrow P$
A, B, L, M, P	$P \rightarrow Q, A \wedge P \rightarrow L$
A, B, L, M, P, Q	$\emptyset$

$$\Sigma = \{ \quad P \rightarrow Q, L \wedge M \rightarrow P, B \wedge L \rightarrow M, A \wedge P \rightarrow L, A \wedge B \rightarrow L, A, B\}$$

Encadenamiento hacia atrás

Consideramos ahora el sistema deductivo alternativo, $EB(\Sigma)$, para trabajar también con reglas de disparo y hechos.

- **Axiomas**: Un conjunto finito, Σ de reglas y hechos.
- **Reglas de inferencia**: (A y C pueden ser conjunciones vacías)

$$SR: \quad \frac{A \wedge L \wedge L \wedge C \to D}{A \wedge L \wedge C \to D} \qquad CR: \quad \frac{A \wedge L_1 \wedge L_2 \wedge C \to D}{A \wedge L_2 \wedge L_1 \wedge C \to D}$$

$$BC: \quad \frac{A \to B,\ B \wedge C \to D}{A \wedge C \to D}$$

Deducción en $EB(\Sigma)$

Veamos que $\vdash_{EB(\Sigma)} Q$,

1.	$P \to Q$	[[Hip.]]
2.	$L \wedge M \to P$	[[Hip.]]
3.	$L \wedge M \to Q$	[[BC aplicada a 1. y 2.]]
4.	$M \wedge L \to Q$	[[CR aplicada a 3.]]
5.	$B \wedge L \to M$	[[Hip.]]
6.	$B \wedge L \wedge L \to Q$	[[BC aplicada a 4. y 5.]]
7.	$B \wedge L \to Q$	[[SR aplicada a 6.]]
8.	B	[[Hip.]]
9.	$L \to Q$	[[BC: 7. y 8.]]
10.	$A \wedge B \to L$	[[Hip.]]
11.	$A \wedge B \to Q$	[[BC aplicada a 9. y 10.]]
12.	A	[[Hip.]]
13.	$B \to Q$	[[BC aplicada a 11. y 12.]]
14.	Q	[[BC aplicada a 13. y 8.]]

$$\Sigma = \{P \to Q,\ L \wedge M \to P,\ B \wedge L \to M,\ A \wedge P \to L,\ A \wedge B \to L,\ A,\ B\}$$

Algoritmo para EB

También podemos diseñar un algoritmo de decisión para demostrar hechos en $EB(\Sigma)$, que se denomina **encadenamiento hacia atrás**.

- Entrada:
  - Un hecho, G.
  - Un conjunto Σ formado por reglas, $\mathbf{R}$, y hechos, $\mathbf{H}$.
- Salida:
  - SI, si $\vdash_{EB(\Sigma)} G$.
  - NO, en caso contrario.
- $EB(G) = Aux(\mathbf{L_G})$, donde inicialmente $\mathbf{L_G} = (G)$ (lista de objetivos pendientes). Se consideran los hechos como reglas (de cuerpo vacío) y el procedimiento auxiliar $Aux(\mathbf{L})$ es:
  1. Si $\mathbf{L} = ()$, parar y devolver SI.
  2. Si no, $E = head(\mathbf{L})$, $\mathbf{R}_E = \{R \in \mathbf{R}\colon R_h = E\}$.
  3. Si para algún $R \in \mathbf{R}_E$, $Aux(R_b \cdots tail(\mathbf{L}))$ da respuesta positiva, parar y devolver SI.
  4. Si no, parar y devolver NO.

Ejemplo: encadenamiento hacia atrás

Veamos que $\vdash_{EB(\Sigma)} Q$,

L	**E**	$\mathbf{R_E}$
(Q)	Q	$P \rightarrow Q$
(P)	P	$L \wedge M \rightarrow P$
(L, M)	L	$A \wedge B \rightarrow L$
(A, B, M)	A	A
(B, M)	B	B
(M)	M	$B \wedge L \rightarrow M$
(B, L)	B	B
(L)	L	$A \wedge B \rightarrow L$
(A, B)	A	A
(B)	B	B
$()$		

$$\begin{aligned}\Sigma = \{ \quad & P \rightarrow Q, L \wedge M \rightarrow P, B \wedge L \rightarrow M, \\ & A \wedge P \rightarrow L, A \wedge B \rightarrow L, \rightarrow A, \rightarrow B\}\end{aligned}$$

Limitaciones de la lógica proposicional

- La lógica proposicional posee un semántica sencilla y existen algoritmos de decisión para sus problemas básicos, como SAT o la consecuencia lógica.
- Sin embargo, la expresividad de la lógica proposicional es bastante limitada.
- Existen problemas cuya descripción mediante lógica proposicional es complicada, ya que requieren un gran número de fórmulas o bien fórmulas de gran tamaño.
- Más aún, existen formas de razonamiento válido que no pueden ser expresadas mediante la lógica proposicional, por ejemplo:
  - Todos los hombres son mortales
  - Sócrates es un hombre.
  - Por tanto, Sócrates es mortal.
- La Lógica de Primer Orden extiende a la Lógica Proposicional proporcionando mayor expresividad.