Parte II: Lógica Proposicional

9 de septiembre de 2015

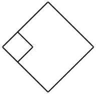
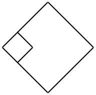
¿Qué es lógica (proposicional)?

- La lógica se entiende, en término generales, como el *análisis de los razonamientos correctos*.
- A fines del siglo XIX y principios del XX, se despierta un interés por dar bases sólidas a la matemática.
- Para ello se introducen sistemas lógicos en los que las fórmulas y los razonamiento válidos están establecidos sin ninguna referencia a lenguajes naturales.
- De esta manera se puede comprobar la validez de razonamientos por medios puramente sintácticos.

Ejemplo de razonamiento correcto

- Premisa 1: Si L es un álgebra de Boole, entonces L es complementado.
- Premisa 2: D_{30} es un álgebra de Boole.
- Conclusión: D_{30} es complementado.

Ejemplo de razonamiento correcto

- Premisa 1: Si un reticulado tiene un sub-reticulado isomorfo a N_5, entonces no es distributivo.
- Premisa 2:  tiene un reticulado isomorfo a N_5.
- Conclusión:  no es distributivo.

El patrón de esos ejemplos

- Ambos razonamientos tienen el siguiente esquema:
- Premisa 1: Si P, entonces Q.
- Premisa 2: P
- Conclusión: Q

Sintaxis y Semántica

- Para estudiar matemáticamente los razonamientos válidos debemos saber representar los mismos de manera matemática. Esto es, definir un conjunto de proposiciones, la *sintaxis*.
- Cada proposición la podremos interpretar como cierta o falsa; a esta interpretación la llamamos la *semántica*.
- Lo primero que descubrimos es que los razonamientos demuestran la *validez* de una *afirmación* a partir de la validez de otras siguiendo ciertas *reglas*. Esta noción se formaliza como *pruebas*.

El alfabeto

- Asumimos un conjunto numerable $\mathcal{V}$ de variables proposicionales que representan las afirmaciones más básicas.
- A los elementos de $\mathcal{V}$ los escribiremos simplemente como p_0, p_1, $\ldots$.
- Definimos $At = \mathcal{V} \cup \{\perp\}$, el símbolo $\perp$ representa la afirmación "es falso". A este conjunto At lo llamamos el conjunto de *átomos*.
- Las proposiciones serán ciertas palabras construidas sobre el alfabeto: $\Sigma = At \cup \{\neg, \vee, \wedge\} \cup \{(,)\}$.

Palabras

- Al conjunto de palabras que se construyen con el alfabeto Σ lo denotamos con Σ^*.
- Ejemplos de palabras sobre Σ son:

$$p_0 \qquad \wedge\, p_0(p_1 \qquad (p_{23} \wedge p_9) \qquad (\neg\bot)$$

- Ejemplos de cadenas que *NO* son palabras sobre Σ:

$$4+0 \qquad x \leqslant y \wedge z \qquad A \vee B$$

- Pero no todas las palabras serán proposiciones.

Proposiciones

- Algunos elementos de Σ^* representan proposiciones:

$$p_2 \qquad p_{35} \qquad (\neg p_{35}) \qquad ((\neg p_{35}) \wedge p_2)$$

- ..., pero otras no:

$$\wedge p_0(p_1 \qquad p_{23} \wedge \neg p_9 \vee p_2$$

- En la última podemos descubrir la necesidad de utilizar paréntesis.
- Pero, cómo definir un sub-conjunto $Prop \subseteq \Sigma^*$ que sólo contenga proposiciones?

Proposiciones

- Ahora daremos una definición *inductiva* del conjunto $Prop$.

$$\begin{array}{rl} A \in At & \text{Si } A \in At \text{, entonces } A \in Prop; \\ (\neg P) & \text{Si } P \in Prop \text{, entonces } (\neg P) \in Prop; \\ (P \vee Q) & \text{Si } P \in Prop \text{ y } Q \in Prop \text{, entonces } (P \vee Q) \in Prop. \\ (P \wedge Q) & \text{Si } P \in Prop \text{ y } Q \in Prop \text{, entonces } (P \wedge Q) \in Prop. \\ (P \to Q) & \text{Si } P \in Prop \text{ y } Q \in Prop \text{, entonces } (P \to Q) \in Prop. \end{array}$$

- Observación: podemos unificar las tres últimas cláusulas usando una meta-variable $\square$ que puede ser $\vee, \wedge,$ ó $\to$:

$$\begin{array}{rl} (P \,\square\, Q) & \text{Si } P \in Prop \text{ y } Q \in Prop \text{, entonces } (P \,\square\, Q) \in Prop. \end{array}$$

Proposiciones

- ¿Cómo sabemos que esas cláusulas definen un conjunto?
- Podemos definir conjuntos cada vez más grandes:

$$
\begin{aligned}
Prop_0 =& At \\
Prop_1 =& Prop_0 \cup \{(\neg P) \mid P \in Prop_0\} \\
& \cup \{(P \,\square\, Q) \mid P, Q \in Prop_0\} \\
\ldots & \\
Prop_{k+1} =& Prop_k \cup \{(\neg P) \mid P \in Prop_k\} \\
& \cup \{(P \,\square\, Q) \mid P, Q \in Prop_k\}
\end{aligned}
$$

- El conjunto de proposiciones es la unión de todos esos conjuntos:

$$Prop = \bigcup_{n \in \mathbb{N}} Prop_n$$

Proposiciones: parseo

- ¿Cómo sabemos si una palabra en Σ^* está en $Prop$?
  1. Si tiene un único símbolo, ese símbolo debe estar en At
  2. Si tiene más de un símbolo:
     - el primer símbolo debe ser $($
     - el último símbolo debe ser un $)$
     - si el segundo símbolo es $\neg$, entonces lo que nos queda debería ser una $Prop$ (es decir repetimos el proceso);
     - si el segundo símbolo no es $\neg$, entonces lo que queda debe ser de la forma $P_1 \square P_2$ con P_1 y P_2 proposiciones.

Proposiciones como TAD

- El TAD apropiado para las proposiciones está parametrizado por $\mathcal{V}$:
- Constructores:

$$\begin{aligned} \bot &: \mathit{Prop} \\ \text{Symbol} &: \mathcal{V} \to \mathit{Prop} \\ \neg &: \mathit{Prop} \to \mathit{Prop} \\ \vee &: \mathit{Prop} \times \mathit{Prop} \to \mathit{Prop} \\ \wedge &: \mathit{Prop} \times \mathit{Prop} \to \mathit{Prop} \\ \to &: \mathit{Prop} \times \mathit{Prop} \to \mathit{Prop} \end{aligned}$$

- Ecuaciones:

Recursión en los naturales

- Si queremos definir una función $f\colon \mathbb{N} \to X$ de los naturales en algún conjunto X alcanza con:

$$\begin{array}{rl} n = 0 & \text{elegir } a \in X \text{ y definir } f(0) = a; \\ n = k+1 & \text{definir } f(n) \text{ en términos de } f(k): \\ & f(k+1) = \ldots f(k) \ldots \end{array}$$

- Es fácil ver que de esa manera tenemos bien definida la función f.
- Puesto que $Prop$ también es un conjunto definido inductivamente, podemos utilizar un esquema semejante.

Recursión en $Prop$

- Supongamos ahora que queremos definir una función $f\colon Prop \to X$.
- Ahora tenemos muchos casos base: $\bot$, p_0, p_1, ..., p_{2356},... por lo tanto debemos definir:

$$f(P) = \begin{cases} x_i & \text{si } P = p_i \\ y & \text{si } P = \bot \end{cases}$$

- Y tenemos varios casos recursivos, uno para cada conectivo; en el caso de la negación:

$$f((\neg P)) = \ldots f(P) \ldots$$

- Para las fórmulas de la forma $(P_1 \square P_2)$, podemos utilizar la llamada recursiva tanto en P_1 como en P_2:

$$f((P_1 \square P_2)) = \ldots f(P_1) \ldots f(P_2) \ldots$$

Recursión, ejemplos

- Cantidad de conectivos en una proposición:

$$\begin{aligned} con(-)\colon Prop &\to \mathbb{N} \\ con(P) &= 0 \qquad \text{si } P \in At \\ con((\neg P)) &= con(P) + 1 \\ con((P_1 \square P_2)) &= con(P_1) + con(P_2) + 1 \end{aligned}$$

- Cantidad de símbolos "(" y ")" en una proposición:

$$\begin{aligned} paren(-)\colon Prop &\to \mathbb{N} \\ paren(P) &= 0 \qquad \text{si } P \in At \\ paren((\neg P)) &= paren(P) + 2 \\ paren((P_1 \square P_2)) &= paren(P_1) + paren(P_2) + 2 \end{aligned}$$

Otros ejemplos de recursión

- Sub-fórmulas:

$$
\begin{aligned}
sub(-) &\colon Prop \to \mathcal{P}(Prop) \\
sub(P) &= \{P\} \qquad \text{si } P \in At \\
sub((\neg P)) &= sub(P) \cup \{(\neg P)\} \\
sub((P_1 \square P_2)) &= sub(P_1) \cup sub(P_2) \cup \{(P_1 \square P_2)\}
\end{aligned}
$$

- Sustitución del símbolo p_i por la proposición Q:

$$
\begin{aligned}
- \, [Q/p_i] &\colon Prop \to Prop \\
p_j \, [Q/p_i] &= \begin{cases} Q & \text{si } i = j \\ p_j & \text{si } i \neq j \end{cases} \\
\bot \, [Q/p_i] &= \bot \\
(\neg P) \, [Q/p_i] &= (\neg \, P \, [Q/p_i]) \\
(P_1 \square P_2) \, [Q/p_i] &= (P_1 \, [Q/p_i] \, \square \, P_2 \, [Q/p_i])
\end{aligned}
$$

Inducción en sub-fórmulas

- Así como podemos definir funciones recursivamente, también podemos probar propiedades sobre $Prop$ utilizando *inducción*.
- Para probar que todo $P \in Prop$ satisface un predicado A, entonces alcanza con probar:

$P \in At$	$A(P)$, para todo $P \in At$;
$(\neg P)$	Si $A(P)$, entonces $A((\neg P))$;
$(P \square Q)$	Si $A(P)$ y $A(Q)$, entonces $A((P \square Q))$.

- Notemos que ese principio de inducción es análogo al principio de inducción para los naturales.

Inducción, ejemplo

Teorema

Para toda $P \in Prop$, $paren(P) = 2 * con(P)$.

- Antes de intenar probarlo, enunciemos las hipótesis inductivas que tendremos a nuestra disposición:
- Si $P = (\neg P')$, entonces la hipótesis inductiva vale para P', es decir:

$$paren(P') = 2 * con(P')$$

- Si $P = (P_1 \square P_2)$, ahora disponemos de la hipótesis inductiva tanto sobre P_1 como sobre P_2:

$$paren(P_1) = 2 * con(P_1) \qquad paren(P_2) = 2 * con(P_2)$$

Inducción, ejemplo

- Si $P \in At$, es fácil

$$paren(P) = 0 = 2 * 0 = 2 * con(P)$$

- Si $P = (\neg\, P')$, utilizando la hipótesis inductiva para P' calculamos:

$$\begin{aligned}
& paren((\neg P')) \\
=&\{ \text{ definición } \} \\
& paren(P') + 2 \\
=&\{ \text{ por h.i. en } P' \} \\
& (2 * con(P')) + 2 \\
=&\{\text{distributividad}\} \\
& 2 * (con(P') + 1) \\
=&\{ \text{ definición } \} \\
& 2 * con((\neg P'))
\end{aligned}$$

Inducción, ejemplo

- Si $P = (P_1 \square P_2)$, ahora disponemos de la hip. inductiva sobre P_1 y sobre P_2.

$$
\begin{aligned}
& paren((P_1 \square P_2)) \\
=& \{ \text{ definición } \} \\
& paren(P_1) + paren(P_2) + 2 \\
=& \{ \text{ por h.i. en ambas sub-fórmulas } \} \\
& 2 * con(P_1) + 2 * con(P_2) + 2 \\
=& \{\text{distributividad}\} \\
& 2 * (con(P_1) + con(P_2) + 1) \\
=& \{ \text{ definición } \} \\
& 2 * con((P_1 \square P_2))
\end{aligned}
$$