Lógica de enunciados (proposicional)

Lógica computacional

13 de marzo de 2024

1. Elementos

La intención en este primer apartado es ver como se construye un lenguaje capaz de expresar ideas, pero estrictamente formal. Que a partir de una cantidad fija de reglas podamos construir enunciados. Si recuerdan sus clases de español en la primaria seguro recordarán que así se inicio su estudio de la lengua, pero al igual rememorarán que siempre salían las excepciones a la regla, los verbos con conjugaciones distintas, y aún más, que hay demasiadas formas de decir una misma cosa[1]. Ahora queremos reducirlo al mínimo, lo que apenas nos alcance para dar sentido pero no llegando a que sea un lenguaje capaz de parecerse a cualquier lenguaje humano. Un dato chusco es que existe el lenguaje de programación *Shakespeare*, (SPL por sus siglas en inglés) donde el programa asemeja al guión de una obra de teatro de Shakespeare, los personajes son pilas que guardarán datos, los diálogos es como interactúan con esos datos y las preguntas que se hacen son enunciados condicionales, pueden ver un poco más de este lenguaje en: `https://en.wikipedia.org/wiki/Shakespeare_Programming_Language`.

Ya que queremos que las expresiones creadas en este lenguaje se les pueda aplicar una función, por ejemplo la concatenación, la negación o alguna transformación, y el resultado siga teniendo sentido, siga siendo una expresión del lenguaje, definimos de acuerdo a lo visto en el capítulo anterior, como un conjunto recursivo. Para ello necesitamos:

- Átomos, en este caso las proposiciones $A_1, A_2, ...$
- Los sómbolos: $\neg$, $\vee$, $\wedge$, $\rightarrow$ y $\iff$. Los conectivos de enunciados.
- Los paréntesis "(" y ")".

Todos estos elementos forman un conjunto S, pero el que nos interesa es un $\bar{S} \subset S$, el subconjutno de las combinaciones de estos elementos que tienen sentido como expresión lógica.

Para poder definirlo inductivamente es necesario asegurar la cerradura. Por ello toda proposición en este lenguaje debe cumplir:

- Todo símbolo A_i está en el lenguaje ($\bar{S}$) así como los símbolos $\{verdadero, falso\}$ también está en $\bar{S}$.
- Si A está en $\bar{S}$, $\neg A$ también está en $\bar{S}$.

[1] Afortunadamente, eso hace a la lengua, sea el español, el inglés, el nahuatl, sumamente rica e interesante, gracias a ello es que hay literatura.

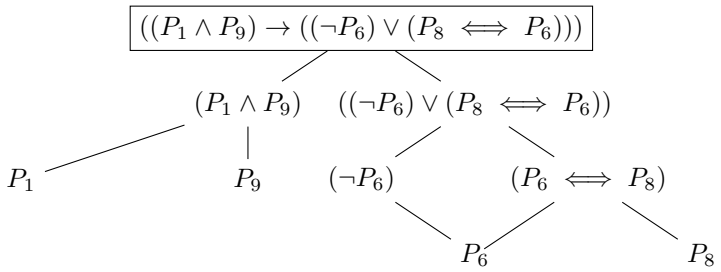


Figura 1: Construcción de una expresión lógica.

- Si A y B están en $\bar{S}$, $(A \vee B)$, $(A \wedge B)$, $(A \rightarrow B)$ y $(A \iff B)$ también están en $\bar{S}$.
- Una cadena está en $\bar{S}$ si y sólo si es formada a través de los pasos anteriores.

Los paréntesis aseguran que $\bar{S}$ está libremente generado. Ejemplos de proposiciones bien armadas: A_1, $\neg A_2$, $(A_1 \vee A_2) \rightarrow (\neg A_1 \wedge A_2)$. Proposiciones sin sentido: $(())$, $(A_1 \wedge A_2)\neg$.

Otra manera de ver estas construcciones es a partir de su árbol de generación, por ejemplo para la expresión $((P_1 \wedge P_9) \rightarrow ((\neg P_6) \vee (P_8 \iff P_6)))$, puede generarse por el árbol que se muestra en la imagen **??**.

El siguiente paso es evaluar cada una de las proposiciones P_i a alguno de los valores de verdad[2]. A partir del árbol podemos saber si la expresión tiene un valor de verdad de *falso* o *verdadero*.

$$\bar{v}(A_1) = \{verdadero, falso\}$$

La evaluación es una función del tipo:

$$\bar{v} : \bar{S} \rightarrow \{verdadero, falso\}.$$

Pero hacer este procedimiento por medio de un árbol para expresiones con demasiadas proposiciones se vuelve una tarea complicada, quizá convenga algún otro método, las tablas de verdad.

Para las operaciones que ya definimos antes, sus tablas de verdad serían:

A_1	$\neg A_1$
F	V
V	F

A_1	A_2	$A_1 \vee A_2$
F	F	F
V	F	V
F	V	V
V	V	V

A_1	A_2	$A_1 \wedge A_2$
F	F	F
V	F	F
F	V	F
V	V	V

A_1	A_2	$A_1 \rightarrow A_2$
F	F	V
V	F	F
F	V	V
V	V	V

A_1	A_2	$A_1 \iff A_2$
F	F	V
V	F	F
F	V	F
V	V	V

[2]Se dice que se le da un valor de verdad, aunque esa es la acción, los valores en realidad pueden ser *verdadero* o *falso*.

Hay más funciones booleanas que pueden definirse, sin argumentos como V^0 y F^0 que son las que regresan los valores $verdadero$ y $falso$ respectivamente sin necesidad de darles argumento alguno.

2. Tautologías, contradicciones y contingencias

Con estos elementos podemos construir distintas expresiones. En general para saber que valor de verdad tendrán al final deberemos evaluar, pero haciendo una división de nuestras posibilidades podemos tener algunos casos bastante particulares, algunos en los que sin hacer la evaluación podremos saber si es verdadero o falso, casos en los que podrá reducirse la expresión, y muchos otros en los que sin evaluare no podremos decir más.

3. Formas distintas de decir lo mismo

Un primer ejemplo que ya vimos son las leyes de DeMorgan:

$$\neg(A_1 \wedge A_2) \iff (\neg A_1) \vee (\neg A_2)$$
$$\neg(A_1 \vee A_2) \iff (\neg A_1) \wedge (\neg A_2)$$

Pero podemos sacar más jugo de esto en otros casos
Forma normal conjuntiva: La conjunción de una disyunción de fórmulas.

$$\wedge_{i=1}^{n}(\vee_{j=1}^{m}\alpha_{i,j})$$

Forma normal disyuntiva: La disyunción de una conjunción de fórmulas.

$$\vee_{i=1}^{n}(\wedge_{j=1}^{m}\alpha_{i,j})$$

p	q	$p \to q$	$\neg p \vee q$
F	F	V	V
V	F	F	F
F	V	V	V
V	V	V	V

Expresar $A \iff B$ en forma normal conjuntiva.

A	B	$A \iff B$	$(\neg A \vee B) \wedge (A \vee \neg B)$
F	F	V	V
V	F	F	F
F	V	F	F
V	V	V	V

Pasos para convertir a forma normal conjuntiva:

1. Cambiar $\to$ y $\iff$ por su correspondiente.
2. Usar las leyes de DeMorgan para introducir $\neg$ a los paréntesis

$$(p \wedge q) \to (t \wedge s)$$

Expresar $A \iff B$ en forma normal disyuntiva.
Ejercicio: Convierte a forma normal conjuntiva:

$$A \iff (\neg B \wedge \neg C)$$
$$\neg(A \wedge B \wedge \neg C)$$

4. Sistemas de demostraciones

Para esta sección nos centraremos en el sistema de Łukasiewicz, para ello partamos de las siguientes tautologías:

1. $A_1 : p \to (q \to p)$
2. $A_2 : (p \to (q \to r)) \to ((p \to q) \to (p \to r))$
3. $A_3 : (\neg p \to \neg q) \to (q \to p)$

Son tautologías, pueden comprobarlo con su tabla de verdad. La idea es partir de estos valores como axiomas, traducir toda expresión posible a estos tres axiomas y así, de manera directa resolver el sistema, sin necesidad de engorrosas tablas de verdad.
Lo único que hace falta es la regla de derivación, llamada *modus ponens*[3]:

$$\frac{p \to q,\ p}{q}$$

Ejercicio: Prueba que $A \to A$ es una consecuencia lógica.

$$\begin{aligned}
&(A \to (B \to A)) \to A && (A_1)\\
&A \to ((B \to A) \to A) && (r.p.)\\
(A \to ((B \to A) \to A)) \to &((A \to (B \to A)) \to (A \to A)) && (A_2)\\
&((A \to (B \to A)) \to (A \to A)) && (M.P.)\\
&(A \to A) && (A_1, M.P.)\square.
\end{aligned}$$

Ejercicio: Asumiendo que $(A \wedge B) \iff \neg(A \to \neg B)$ y $\neg(A \vee B) \iff (\neg A \to B)$ son consecuencias lógicas, muestra que:

- $(\neg B \to \neg A) \iff (A \to B)$
- $(A \wedge B) \to A$

Ejercicio: Muestra que las siguientes son expresiones lógicas válidas o no:

$$\{B \to A, C \to C, D \to B, B \wedge c \wedge D\} \models A \wedge B$$
$$\{A \wedge B \to C, A \to B\} \models A \to C$$

[3]El nombre se deriva de los conceptos de Diogenes Laeritus, quizá uno de los primeros historiadores de la filosofía, biógrafo de los filósofos griegos.

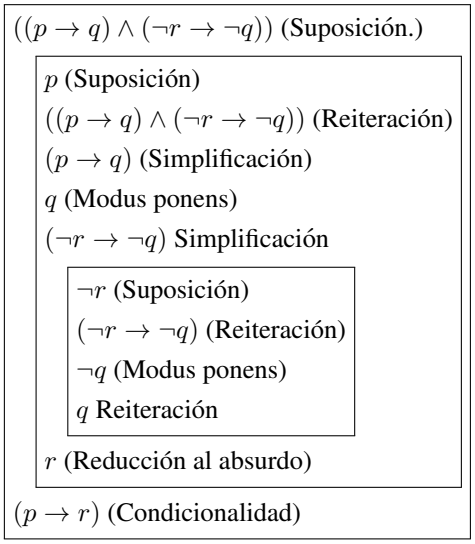
5. Deducción natural

La *deducción natural* es un tipo de sistema lógico, un sistema para poder obtener demostraciones siguiendo algunos pasos establecidos. Su principal característica es tener *subpruebas*, partes de la prueba general que depende de premisas temporales.

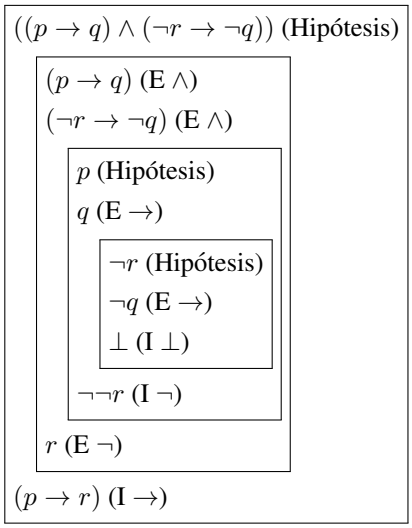
Al mencionar que un enunciado puede asignársele un valor de verdad ($verdadero$ o $falso$) secciones atrás, se mencionó que era parecido a hacer un árbol con cada átomo como hoja y lo que los une es la operación lógica, binaria o unitaria. Eso ya es un sistema de deducción natural, muy parecido a lo que propuso Gentzen en su cálculo-$\mathcal{N}$.

Jaśkowski en 1934, en forma independiente propone un método pictórico de representar una demostración a partir de cajas que encierran porciones de la prueba.

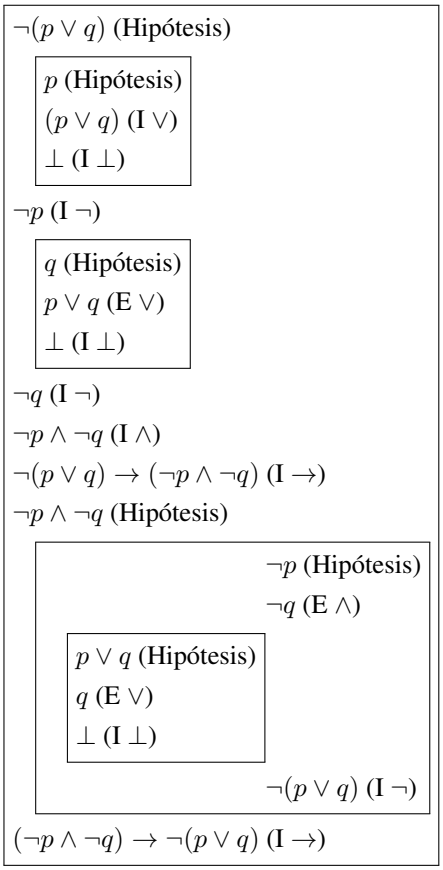
Ejemplo: Probar que $((p \to q) \land (\neg r \to \neg q)) \to (p \to r)$ es una consecuencia lógica:



Pero viéndolo como está en clase:



Ahora un ejemplo conocido. **Ejemplo**: Mostrar que $\neg(p \vee q) \iff (\neg p \wedge \neg q)$



Ejercicio: Prueba las siguientes proposiciones por deducción natural:

$$(A \wedge (A \to B)) \to B$$
$$((A \to B) \wedge (A \to C)) \to (A \to (B \wedge C))$$

Referencias

[1] Enderton, Herbert B. "Una Introducción matemática a la lógica" Universidad Nacional Autónoma de México, Instituto de Investigaciones Filosóficas, 2a edición (2004), reimpresión (2021).

[2] Gallier, Jean. "Logic for computer science. Foundations of automatic theorem proving" University of Pensylvania (2003) `https://www.researchgate.net/publication/31634432_Logic_for_computer_science_foundations_of_automatic_theorem_proving_JH_Gallier`