Notas de Clase para IL

2. Definición de la Lógica Proposicional

Rafael Farré, Robert Nieuwenhuis,
Pilar Nivela, Albert Oliveras, Enric Rodríguez

3 de septiembre de 2009

1. ¿Qué es una Lógica?

Consideraremos en esta asignatura que *una lógica* es la unión de:

1. **Sintaxis:** ¿De qué símbolos dispone el lenguaje y de qué maneras los puedo combinar para obtener una fórmula F?
2. **Semántica:**
   - Una definición de *interpretación* I:
     ¿Qué posibles significados existen para los símbolos?
   - Una definición del concepto de *satisfacción*:
     ¿Cuándo una interpretación I *satisface* una fórmula F?

En una lógica también suele haber (pero no necesariamente) métodos de *deducción*, que describen cómo a partir de unas fórmulas dadas se pueden inferir otras nuevas, dando lugar a gran diversidad de aplicaciones prácticas de las lógicas. Aquí, para comenzar, veremos un ejemplo de lógica muy simple, la *lógica proposicional*, que, a pesar de su sencillez, permite expresar conceptos muy importantes en el mundo de la informática y tiene muchas aplicaciones.

2. Definición de la Lógica Proposicional

Sintaxis: En esta lógica, el *vocabulario* es un conjunto de *símbolos proposicionales* o *de predicado* $\mathcal{P}$ (cuyos elementos escribiremos normalmente como $p, q, r, \ldots$). Una *fórmula* de lógica proposicional sobre $\mathcal{P}$ se define como:

- Todo símbolo de predicado de $\mathcal{P}$ es una fórmula.
- Si F y G son fórmulas, entonces $(F \wedge G)$ y $(F \vee G)$ son fórmulas.
- Si F es una fórmula, entonces $\neg F$ es una fórmula.
- Nada más es una fórmula.

Interpretación: Una *interpretación* I sobre el vocabulario $\mathcal{P}$ es una función $I : \mathcal{P} \to \{0, 1\}$, es decir, I es una función que, para cada símbolo de predicado, nos dice si es 1 (cierto, true) o 0 (falso, false).

Satisfacción: Sean I una interpretación y F una fórmula, ambas sobre el vocabulario $\mathcal{P}$. La *evaluación* en I de F, denotada $eval_I(F)$, es una función que por cada fórmula da un valor de $\{0, 1\}$. Definimos $eval_I(F)$ para todos los casos posibles de F, usando *min*, *max* y $-$ que denotan el mínimo, el máximo, y la resta (sobre números del conjunto $\{0, 1\}$), donde, como sabemos:

$$\begin{array}{llllllll} min(0,0) & = & min(0,1) & = & min(1,0) & = & 0 \quad \text{y} & min(1,1) = 1 \\ max(1,1) & = & max(0,1) & = & max(1,0) & = & 1 \quad \text{y} & max(0,0) = 0: \end{array}$$

- si F es un símbolo p de $\mathcal{P}$ entonces $eval_I(F) = I(p)$
- $$\begin{array}{lll} eval_I(\quad (F \wedge G) \quad) & = & min(\ eval_I(F),\ eval_I(G)\ ) \\ eval_I(\quad (F \vee G) \quad) & = & max(\ eval_I(F),\ eval_I(G)\ ) \\ eval_I(\quad \neg F \quad) & = & 1 - eval_I(F) \end{array}$$

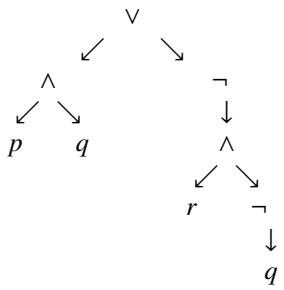
Definimos: si $eval_I(F) = 1$ entonces I *satisface* F, denotado $I \models F$. En este caso también se dice que F *es cierto en* I, o que I *es un modelo de* F.

3. Explicaciones sobre la definición de la lógica proposicional

Otras conectivas: En lógica proposicional, a $\wedge$, $\vee$ y $\neg$ se les llama *conectivas lógicas*. A menudo también se consideran otras conectivas, como $\rightarrow$ o $\leftrightarrow$. Aquí no las hemos introducido porque son expresables mediante las tres conectivas dadas. Por ejemplo, las fórmulas $F \rightarrow G$ y $F \leftrightarrow G$ pueden verse como abreviaturas de $(\neg F \vee G)$ y de $((\neg F \vee G) \wedge (\neg G \vee F))$ respectivamente.

No ambigüedad, representación en árbol: Las fórmulas, vistas como una cadena de símbolos (conectivas, símbolos de predicado y paréntesis), no son *ambiguas*: existe una única forma de entenderlas. Formalmente: para cada fórmula F existe un único árbol que representa la construcción de F según la definición de la sintaxis de la lógica proposicional.

Por ejemplo, la fórmula $((p \wedge q) \vee \neg(r \wedge \neg q))$ sólo puede leerse como un $\vee$ de la fórmula $(p \wedge q)$ y de la fórmula $\neg(r \wedge \neg q)$. Éstas a su vez sólo son legibles de una única manera, y así sucesivamente. El árbol correspondiente es:



Prioridades de conectivas y paréntesis: A veces podemos omitir paréntesis innecesarios: quitarlos no introduce ambigüedad. Por ejemplo, podemos escribir $p \wedge q$ en vez de $(p \wedge q)$. Además se asumen las siguientes *prioridades* entre las conectivas; de más prioritario a menos, tenemos: $\neg \ \wedge \ \vee \ \rightarrow \ \leftrightarrow$. Así, podemos escribir $p \wedge q \vee r$ para referirnos a $(p \wedge q) \vee r$, pero no podemos omitir paréntesis en $\neg(p \wedge (q \vee r))$. Nótese la similitud con la suma, el producto, y el "$-$" unario en aritmética: con $-x * y + z$ nos referimos a $((-x) * y) + z$ y no a, por ejemplo, $-(x * (y + z))$, ni a $(-x) * (y + z)$.

¿Qué cosas se modelan con esta lógica?: La idea intuitiva de esta lógica es que sirve para modelar (y razonar formalmente sobre) situaciones de la vida real en las que tratemos con enunciados o *proposiciones* que sólo pueden ser o bien ciertas o bien falsas.

Por ejemplo, si $\mathcal{P}$ es { *llueve*, *hace_sol*, *esta_nublado* }, cada interpretación I para esta $\mathcal{P}$ modela una situación real del tiempo del estilo de "sí llueve, no hace sol y no está nublado", es decir, $I(llueve) = 1$, $I(hace\_sol) = 0$, $I(esta\_nublado) = 0$. Con esta interpretación I, si F es la fórmula

$$llueve \wedge \neg(hace\_sol \vee esta\_nublado)$$

entonces, por la definición de $eval_I$, tenemos $eval_I(F) =$

$$\begin{array}{lllllll} min(& I(llueve), & 1 - & max(& I(hace\_sol), & I(esta\_nublado) &)) = \\ min(& 1, & 1 - & max(& 0, & 0 &)) = 1, \end{array}$$

luego F es cierta en I, es decir, tenemos $I \models F$. Similarmente, si G es la fórmula $\neg llueve \vee (hace\_sol \vee esta\_nublado)$, entonces I no satisface G, es decir, $I \not\models G$.

Podemos listar todas las posibles interpretaciones y la evaluación en cada una de ellas de la fórmula $llueve \wedge \neg(hace\_sol \vee esta\_nublado)$ mediante una *tabla de verdad* (escribiendo p, q, r en vez de *llueve*, *hace_sol*, y *esta_nublado*):

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

y vemos que en este ejemplo sólo hay una interpretación que satisface la fórmula, que es cuando llueve, no hace sol y no está nublado.

4. Satisfactibilidad, tautología, consecuencia, y equivalencia

Ahora podemos introducir la siguiente nomenclatura básica. Estas nociones no dependen de la lógica concreta. Se definen de manera idéntica en otras lógicas distintas a la lógica proposicional, como la lógica de primer orden. Por eso no se repetirán cuando tratemos a la lógica de primer orden.

1. Una fórmula F es *satisfactible* si tiene algún modelo, es decir, si existe alguna interpretación I tal que $I \models F$. Un fórmula F es *insatisfactible* (o es una *contradicción*) si no es satisfactible.

2. Una fórmula F es una *tautología* (o es *válida*), si *toda* interpretación es modelo de F, es decir, si para toda interpretación I tenemos $I \models F$.

3. Sean F y G fórmulas construidas sobre un vocabulario $\mathcal{P}$.
   Definimos: *F es consecuencia lógica de G* si todo modelo de G también lo es de F, es decir, si para toda interpretación I sobre $\mathcal{P}$ tal que $I \models G$ tenemos $I \models F$.
   Sobrecargando el operador "$\models$", esto se denotará por $G \models F$.

4. Sean F y G fórmulas construidas sobre un vocabulario $\mathcal{P}$.
   Definimos: *F y G son lógicamente equivalentes* si tienen los mismos modelos, es decir, si para toda interpretación I sobre $\mathcal{P}$ tenemos $I \models G$ si y sólo si $I \models F$.
   Esto se denotará por $G \equiv F$.

Nótese que el símbolo "$\models$" denota satisfacción cuando a su izquierda hay una interpretación I; en ese caso, $I \models F$ denota que I satisface F. En cambio, cuando a la izquierda hay una fórmula G, el símbolo "$\models$" denota consecuencia lógica; así, $G \models F$ denota que *todo modelo de G* satisface F.

Este tipo de propiedades (satisfactibilidad o tautología de una fórmula, consecuencia lógica o equivalencia entre fórmulas, etc.) se plantean en diversas aplicaciones prácticas de una lógica. Por ejemplo, en Intel pueden estar interesados en saber si dos circuitos (dos fórmulas) son equivalentes, o saber si un circuito F cumple cierta propiedad G (es decir, si $F \models G$).

La lógica proposicional es especialmente interesante porque todas estas propiedades son *decidibles*: para cada una de ellas hay algún programa de ordenador que siempre termina y nos da una respuesta correcta sí/no. Por ejemplo, para saber si una fórmula F dada es satisfactible, podemos construir su tabla de verdad con la lista de todas las posibles interpretaciones I para el vocabulario $\mathcal{P}$ sobre el que F está construido, y averiguar si hay algún 1 en la columna de F, esto es, si $eval_I(F) = 1$ para alguna I.

Similarmente, tenemos $F \models G$ para dos fórmulas F y G dadas si y sólo si en la tabla de verdad (para el vocabulario $\mathcal{P}$ de F y G), por cada fila con un 1 en la columna de F también hay un 1 en la columna de G.

Sin embargo, estos métodos basados en la tabla de verdad no son los más eficientes. En los ejercicios veremos que cualquiera de estas propiedades se puede expresar en términos del problema de la satisfactibilidad (abreviado, *SAT*), esto es, el de determinar si una fórmula proposicional dada es satisfactible o no. Por ejemplo, para dos fórmulas F y G tenemos $F \models G$ si, y sólo si, $F \wedge \neg G$ es insatisfactible. Por eso, para las aplicaciones prácticas se usa un *SAT solver*, esto es, un programa de ordenador que, dada una fórmula F, decide si F es satisfactible o no.

5. Ejercicios

1. (dificultad 1) ¿Cuántas interpretaciones posibles hay en función de $|\mathcal{P}|$?
   (nota: si S es un conjunto, $|S|$ denota su *cardinalidad*, es decir, el número de elementos de S).

2. (dificultad 1) Demuestra que $p \wedge \neg p$ es insatisfactible.

3. (dificultad 1) Demuestra que $p \vee \neg p$ es una tautología.

4. (dificultad 1) Escribe la tabla de verdad para las fórmulas $(p \wedge \neg(q \vee r))$ y $(\neg p \vee (q \vee r))$. Di, por cada una de las fórmulas si es tautología o insatisfactible y averigua las posibles relaciones de consecuencia o equivalencia lógica entre ellas.

5. (dificultad 1) Sean F y G dos fórmulas. ¿Es cierto que $F \vee G$ es tautología si y sólo si alguna de las dos fórmulas F o G lo es? Demuéstralo.

6. (dificultad 2) Sea F una fórmula. Demuestra que F es tautología si y sólo si $\neg F$ es insatisfactible.

7. (dificultad 2) Sean F y G dos fórmulas. Demuestra que F es consecuencia lógica de G si y sólo si $G \wedge \neg F$ es insatisfactible.

8. (dificultad 2) Sean F y G dos fórmulas. Demuestra que F es lógicamente equivalente a G si y sólo si $G \leftrightarrow F$ es tautología si y sólo si $(G \wedge \neg F) \vee (F \wedge \neg G)$ es insatisfactible.

9. (dificultad 2) Da un ejemplo de tres fórmulas F_1, F_2, y F_3 tales que $F_1 \wedge F_2 \wedge F_3$ sea insatisfactible y donde cualquier conjunción de todas ellas menos una sea satisfactible. Generalízalo a n fórmulas.

10. (dificultad 1) Demuestra que son tautologías:

    a) $(p \wedge (p \to q)) \to q$

    b) $((p \to q) \wedge \neg q) \to \neg q$

11. (dificultad 2) Dada una fórmula F, si $c(F)$ denota el número de sus conectivas binarias (contando repeticiones) y $p(F)$ el número de sus símbolos de predicado (contando repeticiones), demuestra que $p(F) = c(F) + 1$.

12. (dificultad 1) Demuestra de la forma más sencilla que encuentres:
    $(\neg q \vee \neg r \vee p) \wedge (p \vee q) \wedge (r \vee p) \quad \not\equiv \quad p \vee q$

13. (dificultad 1) Demuestra de la forma más sencilla que encuentres:
    $(p \vee q) \wedge (r \vee p) \wedge (\neg q \vee \neg r \vee p) \quad \equiv \quad p$

14. (dificultad 3) Sea F una fórmula que no contiene el símbolo $\neg$. Demuestra que si el número de símbolos de predicado es $k + 1$ entonces la longitud de la fórmula es $4k + 1$.

15. (dificultad 2) Si $F \to G$ es tautología y F es tautología, entonces ¿G es tautología? Demuéstralo.

16. (dificultad 2) Si $F \to G$ es satisfactible y F es satisfactible, entonces ¿G es satisfactible? Demuéstralo.

17. (dificultad 2) Si $F \to G$ es tautología y F satisfactible, entonces ¿G es satisfactible? Demuéstralo.

18. (dificultad 2) Demuestra las siguientes equivalencias entre fórmulas:

$F \wedge F$	$\equiv F$	idempotencia de $\wedge$
$F \vee F$	$\equiv F$	idempotencia de $\vee$
$F \wedge G$	$\equiv G \wedge F$	conmutatividad de $\wedge$
$F \vee G$	$\equiv G \vee F$	conmutatividad de $\vee$
$F \wedge (F \vee G)$	$\equiv F$	absorción 1
$F \vee (F \wedge G)$	$\equiv F$	absorción 2
$(F \wedge G) \wedge H$	$\equiv F \wedge (G \wedge H)$	asociatividad de $\wedge$
$(F \vee G) \vee H$	$\equiv F \vee (G \vee H)$	asociatividad de $\vee$
$(F \wedge G) \vee H$	$\equiv (F \vee H) \wedge (G \vee H)$	distributividad 1
$(F \vee G) \wedge H$	$\equiv (F \wedge H) \vee (G \wedge H)$	distributividad 2
$\neg\neg F$	$\equiv F$	doble negación
$\neg(F \wedge G)$	$\equiv \neg F \vee \neg G$	ley de De Morgan 1
$\neg(F \vee G)$	$\equiv \neg F \wedge \neg G$	ley de De Morgan 2
Si F es tautología entonces:		
$F \wedge G$	$\equiv G$	ley de tautología 1
$F \vee G$	$\equiv F$	ley de tautología 2
Si F es insatisfactible entonces:		
$F \wedge G$	$\equiv F$	ley de insatisfactible 1
$F \vee G$	$\equiv G$	ley de insatisfactible 2

19. (dificultad 2) Dadas fórmulas F, A, B, ¿es cierto que $F \wedge (A \vee B)$ es satisfactible si y sólo si al menos una de las fórmulas $F \wedge A$ y $F \wedge B$ lo es? Demuéstralo.

20. (dificultad 2) Demuestra que si $A \wedge B \models D$ y $A \wedge C \models D$ entonces $A \wedge (B \vee C) \models D$. ¿Es cierto el recíproco?

21. (dificultad 3) Una *relación binaria* sobre un conjunto S es un subconjunto del producto cartesiano $S \times S$, es decir, un conjunto de pares de elementos de S. Sea $\approx$ una relación binaria sobre S que escribiremos con notación infija, es decir, si para dos elementos a y b de S tenemos $(a, b) \in \approx$, escribiremos $a \approx b$. Se dice que $\approx$ es una *relación de equivalencia* en S si se cumplen las siguientes condiciones:

    a) Para cada $x \in S$ tenemos $x \approx x$ (*reflexividad*).

    b) Para cada $x, y \in S$, si $x \approx y$ entonces $y \approx x$ (*simetría*).

    c) Para cada $x, y, z \in S$, si $x \approx y$ y $y \approx z$ entonces $x \approx z$ (*transitividad*).

Demuestra que la equivalencia lógica $\equiv$ es una relación de equivalencia en el conjunto de las fórmulas proposicionales definidas sobre un conjunto $\mathcal{P}$ de símbolos. Demuestra también que es una relación compatible con las operaciones lógicas $\neg$, $\wedge$ y $\vee$; es decir:

a) Si $F \equiv G$ entonces $\neg F \equiv \neg G$.

b) Si $F_1 \equiv G_1$ y $F_2 \equiv G_2$ entonces $F_1 \wedge F_2 \equiv G_1 \wedge G_2$.

c) Si $F_1 \equiv G_1$ y $F_2 \equiv G_2$ entonces $F_1 \vee F_2 \equiv G_1 \vee G_2$.

22. (dificultad 2) Definimos recursivamente las subfórmulas de una fórmula F de la forma siguiente:

    - Si F es un símbolo de predicado p, entonces p es su única subfórmula.
    - Si F es $\neg G$, entonces las subfórmulas de F son F y las subfórmulas de G.
    - Si F es de la forma $(G \wedge H)$ o de la forma $(G \vee H)$, entonces las subfórmulas de F son F y las subfórmulas de G y las de H (contando con repetición, es decir, todas las ocurrencias).

    Demuestra que para toda fórmula F, el número de subfórmulas de F, denotado $nsf(F)$, cumple que $nsf(F) \leq |F|$, donde $|F|$ es el número de símbolos (conectivas, paréntesis y símbolos de predicado) de F.

23. (dificultad 3) Demuestra que si en una fórmula F sustituimos una ocurrencia de una subfórmula G por otra G' lógicamente equivalente a G, obtenemos una nueva fórmula F' que es lógicamente equivalente a F. Este resultado tiene un interés obvio para poder manipular fórmulas, por lo que tiene nombre: es el *Lema de Sustitución*.

24. (dificultad 2) Demuestra utilizando los resultados de los dos ejercicios anteriores que $q \vee (p \wedge (\neg p \vee r))$ es lógicamente equivalente a $(q \vee p) \wedge ((q \vee r) \vee (p \wedge \neg p))$.

25. (dificultad 2) Demuestra que para todas las fórmulas A, B, C se cumple que $A \wedge B \models C$ si y sólo si $A \models (B \rightarrow C)$.

26. (dificultad 2) Supongamos que $|\mathcal{P}| = 100$ y que nos interesa determinar si una fórmula F construida sobre $\mathcal{P}$ es satisfactible o no. Si el algoritmo está basado en un análisis de la tabla de verdad, y evaluar F en una interpretación I dada cuesta un microsegundo (10^{-6} segundos), ¿cuántos años tardará? Más adelante veremos técnicas que muchas veces funcionan mejor.

27. (dificultad 2) Una *función booleana* de n entradas es una función $f\colon \{0,1\}^n \rightarrow \{0,1\}$, es decir, una función que toma como entrada una cadena de n bits y devuelve un bit. ¿Cuántas funciones booleanas de n entradas hay?

28. (dificultad 2) Cada fórmula F representa una única función booleana: la que devuelve 1 exactamente para aquellas cadenas de bits I tales que $eval_I(F) = 1$. Por eso, dos fórmulas son lógicamente equivalentes si y sólo si representan la misma función booleana. ¿Cuántas funciones booleanas (o cuántas fórmulas lógicamente no-equivalentes) hay en función de $|\mathcal{P}|$?

29. (dificultad 2) Consideremos los siguientes símbolos de predicado:

    p es *María abusa de la comida basura*

    q es *Tomás abusa de la comida basura*

    r es *María está enferma*

    s es *Tomás está enfermo*

    t es *María fuma*

    u es *Tomás fuma*

    a) Traduce las siguientes fórmulas a tu idioma:

    1) $p \wedge \neg q$
    2) $(\neg p \wedge \neg t) \rightarrow \neg r$
    3) $(t \vee u) \rightarrow (t \wedge u)$
    4) $s \leftrightarrow (q \wedge u)$

    b) Traduce las siguientes frases a fórmulas de la lógica proposicional. Si en algún caso hay varias fórmulas posibles, indica si son lógicamente equivalentes o no. Nótese que, si no lo son, estamos ante un caso de ambigüedad del lenguaje natural.

    1) Tomás no fuma salvo que María lo haga.
    2) Ni Tomás ni María abusan de la comida basura.
    3) María esta sana si y sólo si no fuma y no abusa de la comida basura
    4) María no fuma si Tomás no lo hace
    5) Si María no fuma entonces está sana si no abusa de la comida basura

30. (dificultad 3) Demuestra para $n \geq 1$ las dos siguientes equivalencias lógicas:

    $(F_1 \wedge \ldots \wedge F_n) \vee G \equiv (F_1 \vee G) \wedge \ldots \wedge (F_n \vee G)$

    $(F_1 \vee \ldots \vee F_n) \wedge G \equiv (F_1 \wedge G) \vee \ldots \vee (F_n \wedge G)$

31. (dificultad 3) Escribe en una tabla de verdad las 16 funciones booleanas de 2 entradas. ¿Cuántas de ellas sólo dependen de una de las dos entradas? ¿Cuántas dependen de cero entradas? ¿Las otras, vistas como conectivas lógicas, tienen algún nombre? Ya sabemos que podemos expresar cualquier función booleana con el conjunto de tres conectivas $\{\wedge, \vee, \neg\}$, es decir, cualquier función booleana es equivalente a una fórmula construida sobre estas tres conectivas. ¿Es cierto esto también para algún conjunto de sólo dos de las 16 funciones? (Hay varias maneras, pero basta con dar una sola.)

32. (dificultad 3) Demuestra que cualquier función booleana de dos entradas se puede expresar con sólo *nor* o bien con sólo *nand*, donde $nor(F, G)$ es $\neg(F \vee G)$, y $nand(F, G)$ es $\neg(F \wedge G)$.

33. (dificultad 3) Tres estudiantes A, B y C son acusados de introducir un virus en la salas de ordenadores de la FIB. Durante el interrogatorio, las declaraciones son las siguientes:

- *A* dice: "*B* lo hizo y *C* es inocente"
- *B* dice: "Si *A* es culpable entonces *C* también lo es"
- *C* dice: "Yo no lo hice, lo hizo al menos uno de los otros"

a) ¿Son las tres declaraciones contradictorias?

b) Asumiendo que todos son inocentes, ¿quién o quiénes mintieron en la declaración?

c) Asumiendo que nadie mintió, ¿quién es inocente y quién es culpable?

34. (dificultad 2) Inventa y define formalmente alguna otra lógica distinta a la lógica proposicional. Por ejemplo, si las interpretaciones son funciones $I: \mathcal{P} \to \{0, 1, \bot\}$ que también pueden dar "indefinido" $\bot$, se puede adaptar la noción de satisfacción de manera razonable, aunque la respuesta ya no será binaria: la "evaluación" de una fórmula F en una interpretación I puede dar 1 (I satisface F) o 0 (I no satisface F) o $\bot$ (indefinido).

35. (dificultad 2) Como el ejercicio anterior, pero considerando $I: \mathcal{P} \to [0 \ldots 1]$, es decir, la interpretación de un símbolo p es una probabilidad (un número real entre 0 y 1). En este caso, la evaluación de una fórmula F en una interpretación I puede dar algo (remotamente) parecido a la probabilidad de satisfacción de F en I. En la lógica que has definido, ¿la evaluación de F en una I determinada, y la de $F \wedge F$ en esa misma I dan el mismo resultado?

36. (dificultad 2) Tienes delante de tí tres cajas cerradas, dos de ellas vacías y una llena de oro. La tapa de cada caja contiene una afirmación respecto de su contenido. La caja *A* dice "El oro no está aquí", la caja *B* dice "El oro no está aquí" y la caja *C* dice "El oro está en la caja *B*". Si sabemos que una y sólo una las afirmaciones es cierta, qué caja contiene el oro?

37. (dificultad 3) Considera el siguiente fragmento de código, que devuelve un booleano:

```
int i;
bool a, b;
...
if (a and i>0)
  return b;
else if (a and i<=0)
  return false;
else if (a or b)
  return a;
else
  return (i>0);
```

Simplifícalo sustituyendo los valores de retorno por un solo valor de retorno que sea una expresión booleana en $i > 0$, a y b:

```
int i;
bool a, b;
return ...;
```

38. (dificultad 2) Demuestra que I satisface la fórmula $((\cdots(p_1 \leftrightarrow p_2) \leftrightarrow \cdots) \leftrightarrow p_n)$ (donde $n \geq 1$) si y sólo si asigna el valor 0 a un número par de símbolos de predicado de la fórmula.

39. (dificultad 3)

    a) La fórmula $(((p \to q) \to p) \to p)$ es una tautología?

    b) Si definimos recursivamente A_0 como $(p \to q)$ y A_{n+1} como $(A_n \to p)$, ¿para qué valores de n es A_n una tautología?

40. (dificultad 3) Sea A^* la fórmula resultante de intercambiar en A los símbolos $\wedge, \vee$ y reemplazar cada símbolo de predicado por su negación. Demuestra que A^* es lógicamente equivalente a $\neg A$.

41. (dificultad 2) Dada una cadena de símbolos w, escribimos $pa(w)$ para denotar el número de paréntesis de abrir que hay en w. Similarmente, escribimos $pc(w)$ para los de cerrar. Demuestra que para toda fórmula F tenemos $pa(F) = pc(F)$.