Examen Final IL. 1a parte: Lógica Proposicional. Junio 2007. Tiempo: 70 min.

Publicación notas: 29 jun. Revisión: 2 jul. 10h, con solicitud previa via mail razonado el 29 y 30 jun.

---

1. (6 puntos) Demuestra en todo detalle si son ciertas o falsas las siguientes afirmaciones para fórmulas proposicionales cualesquiera F y G. Utiliza sólo las definiciones de la lógica proposicional, de satisfactibilidad, de tautología y de equivalencia y consecuencia lógica. Si das algún contraejemplo, demuestra en detalle que realmente es un contraejemplo.

   a) Si $F \rightarrow G$ es una tautología entonces $\neg F \rightarrow \neg G$ también lo es.
   Solución: Falso. Contraejemplo: F es $p \wedge \neg p$ y G es q.
   Esta F es insatisfactible: para toda I, tenemos $eval_I(F) = eval_I(p \wedge \neg p) = min(eval_I(p), 1 - eval_I(p)) = min(I(p), 1 - I(p)) = 0$.
   Y $\neg F$ es una tautología: para toda I, tenemos $eval_I(\neg F) = 1 - eval_I(F) = 1$.
   Ahora, por un lado, $F \rightarrow G$ es tautología (toda interpretación I es modelo de $F \rightarrow G$), ya que $eval_I(F \rightarrow G) = eval_I(\neg F \vee G) = max(\ eval_I(\neg F), eval_I(G)\ ) = 1$.
   Por otro lado, $\neg F \rightarrow \neg G$ no es tautología: para la interpretación I con $I(q) = I(p) = 1$, tenemos $eval_I(\neg F \rightarrow \neg G) = eval_I(\neg\neg F \vee \neg G) = max(\ 1-(1-eval_I(F)),\ 1-eval_I(G)\ ) = max(\ eval_I(F),\ 1 - eval_I(G)\ ) = max(0, 1 - I(q)) = max(0, 0) = 0$.

   b) Si $F \models G$ y $F \models \neg G$, entonces F es insatisfactible.
   Solución: Cierto.
   Por reducción al absurdo. Si existiera un modelo I de F, entonces:
   -puesto que $F \models G$ tendríamos $I \models G$, es decir, $eval_I(G) = 1$.
   -puesto que $F \models \neg G$ tendríamos $I \models \neg G$, es decir, $eval_I(\neg G) = 1$, lo cual implica $1 - eval_I(G) = 1$, es decir, $eval_I(G) = 0$.
   Como $eval_I(G)$ no puede dar tanto 1 como 0, hemos llegado a una contradicción: el supuesto modelo I de F no puede existir, es decir, F es insatisfactible.

   c) Si en F no aparece la conectiva $\neg$ (es decir, si $\wedge$ y $\vee$ son las únicas conectivas), entonces F es satisfactible.
   Solución: Cierto. Sea I la interpretación donde $I(p) = 1$ para todo p en $\mathcal{P}$. Demostramos $I \models F$ por inducción sobre n, la longitud (número total de símbolos) de F.
   Si $n = 1$, F es un símbolo de predicado p y $I \models p$.
   Si $n > 1$, F es de la forma $(F_1 \wedge F_2)$ o de la forma $(F_1 \vee F_2)$, donde, por H.I., tenemos $I \models F_1$ y $I \models F_2$, es decir, $eval_I(F_1) = eval_I(F_2) = 1$. Pero entonces $max(\ eval_I(F_1),\ eval_I(F_2)\ ) = min(\ eval_I(F_1),\ eval_I(F_2)\ ) = 1$, por lo que en ambos casos $I \models F$.

2. (4 puntos) Los n trabajadores de una empresa quieren elegir entre ellos una comisión de K representantes sindicales, con $n > K$. Para ello se hace una votación en la que cada trabajador entrega una papeleta con K nombres.

   Se desea averiguar, usando SAT, si, dadas las n papeletas, es posible (y cómo) formar la comisión de manera que cada trabajador esté representado (es decir, que al menos una de las K personas a las que ha votado esté en la comisión).

   Para ello, expresa este problema como un conjunto de cláusulas con $K \cdot n$ símbolos p_{ij} que signifiquen: "el i-ésimo miembro de la comisión es el trabajador j". Notación: supón que cada papeleta es un subconjunto $\{j_1, \ldots, j_K\}$ de $\{1, \ldots, n\}$.

   Solución:

- Cada miembro i de la comisión es al menos uno de los n trabajadores:
  para toda i con $1 \leq i \leq K$, una cláusula $p_{i,1} \vee \ldots \vee p_{i,n}$
- Cada miembro i de la comisión es como máximo uno de los n trabajadores:
  para toda i con $1 \leq i \leq K$, y para todos los j, j' con $1 \leq j < j' \leq n$, una cláusula $\neg p_{ij} \vee \neg p_{ij'}$.
- Ningún trabajador j puede ser a la vez dos miembros distintos de la comisión:
  para toda j con $1 \leq j \leq n$, y para todos los i, i' con $1 \leq i < i' \leq K$, una cláusula $\neg p_{ij} \vee \neg p_{i'j}$.
- Al menos uno de los miembros de cada papeleta $\{j_1, \ldots j_K\}$ estará en la comisión:
  $p_{1,j_1} \vee \ldots \vee p_{1,j_K} \ \vee \ldots \vee \ p_{K,j_1} \vee \ldots \vee p_{K,j_K}$ (una cláusula de K^2 literales por papeleta)

Examen Final IL. 2a parte: LPO y Prog. Lógica. Junio 2007. Tiempo: 90 min.

Publicación notas: 29 jun. Revisión: 2 jul. 10h, con solicitud previa via mail razonado el 29 y 30 jun.

1. (3 puntos) Demuestra la verdad o falsedad de las siguientes afirmaciones. Está permitido usar reglas deductivas o contraejemplos, sin necesidad de usar el $eval_I$.

   A: $\forall x\ \exists y\ q(f(x), y) \models \exists y\ \forall x\ q(f(x), y)$
   Solución: No es cierta. Considera la interpretación I con $D_I = \{a, b\}$, $q_I(a, a) = q_I(b, b) = 1$, $q_I(a, b) = q_I(b, a) = 0$, $f_I(a) = a$, y $f_I(b) = b$. Entonces I es modelo de $\forall x\ \exists y\ q(f(x), y)$ pero no de $\exists y\ \forall x\ q(f(x), y)$. [véase el problema 18 de la hoja 4].

   B: $\forall x\ \forall y\ (p(x, y) \vee p(y, x)) \models \forall x\ p(x, x)$.
   Solución: Es cierta. Demostramos que $\forall x\ \forall y\ (p(x, y) \vee p(y, x)) \wedge \neg\forall x\ p(x, x)$ es insatisfactible. Por resolución a partir de las cláusulas $p(x, y) \vee p(y, x)$ y $\neg p(a, a)$ (que viene de la conclusión negada $\exists x \neg p(x, x)$), obtenemos $\Box$ en dos pasos.

2. (3.5 puntos) Consideremos un conjunto de personas.

   A: En este conjunto, la relación binaria "ser-amigo-de" es simétrica (si x es amigo de y entonces y es amigo de x).
   B: También es transitiva (si x es amigo de y, e y es amigo de z, entonces x es amigo de z).
   C: Si x es amigo de y, entonces, si una persona del conjunto es amigo de y también es amigo de x.

   Es cierto que C es consecuencia lógica de $A \wedge B$? Formalízalo y demuestra tu respuesta.

   Solución: Utilizamos un predicado binario p para esta relación:

   A es: $\forall x \forall y (p(x, y) \rightarrow p(y, x))$, que en forma clausal queda:

   1. $\neg p(x, y) \vee p(y, x)$

   B es: $\forall x \forall y \forall z (p(x, y) \wedge p(y, z) \rightarrow p(x, z))$, que en forma clausal queda:

   2. $\neg p(x, y) \vee \neg p(y, z) \vee p(x, z)$

   C es: $\forall x \forall y (\ p(x, y) \rightarrow \forall z (\ p(z, y) \rightarrow p(z, x)\ )\ )$

   Luego $\neg C$ (moviendo el $\neg$ hacia dentro) es: $\exists x \exists y\ (\ \ p(x, y) \wedge \exists z (\ p(z, y) \wedge \neg p(z, x)\ )\ \ )$, que en forma clausal queda:

   3. $p(a, b)$
   4. $p(c, b)$
   5. $\neg p(c, a)$

   donde a, b, c son las constantes de Skolem para las variables existencialmente cuantificadas x, y, z.

   Aplicando resolución podemos ver que $A \wedge B \wedge \neg C$ es insatisfactible:

Resolvente	Cláusulas utilizadas	Unificador
6) $p(b, a)$	$1 + 3$	$\{x = a, y = b\}$
7) $\neg p(b, z) \vee p(c, z)$	$2 + 4$	$\{x = c, y = b\}$
8) $p(c, a)$	$7 + 6$	$\{z = a\}$
9) $\Box$	$8 + 5$	$\{\}$

   Por lo tanto, C es consecuencia lógica de $A \wedge B$.

3. (3.5 puntos) Resuelve los cuatro apartados siguientes sobre programas en Prolog. Hay que hacer todos los predicados, incluso si son conocidos de clase o de los apuntes. Está permitido utilizar resultados de algún apartado previo de este ejercicio, incluso si ese apartado no te ha salido.

   A: Escribe en Prolog el predicado `subset(L,S)` que significa "S es un subconjunto de L".

   B: Conviértelo en el predicado `subset2(L,S,R)` que significa "S es un subconjunto de L y R es el resto de L". Por ejemplo:

```
?- subset2( [a,b,c], S, R ), write(S), write('   '), write(R), nl, fail.
[]   [a,b,c]
[c]   [a,b]
[b]   [a,c]
[b,c]   [a]
[a]   [b,c]
[a,c]   [b]
[a,b]   [c]
[a,b,c]   []
no
```

   C: Haz un predicado `partición(L)`, que dado un conjunto de naturales (una lista `L`), escribe todas las maneras de partirlo en dos subconjuntos `L1` y `L2` tales que la suma de los elementos de `L1` sea igual a la suma de los elementos de `L2`. Por ejemplo:

```
?- partición([2,3,4,5,6,8]).
[6,8]  [2,3,4,5]
[3,5,6]  [2,4,8]
[2,4,8]  [3,5,6]
[2,3,4,5]  [6,8]
yes
```

   D: Modifica tu programa del apartado C para que sólo escriba los dos subconjuntos si además tienen el mismo número de elementos. Por ejemplo:

```
?- partición([2,3,4,5,6,8]).
[3,5,6]  [2,4,8]
[2,4,8]  [3,5,6]
yes
```

Solución:

```
subset([],[]).
subset([_|C],S):-subset(C,S).
subset([X|C],[X|S]):-subset(C,S).

subset2([],[],[]).
subset2([X|C],S,[X|R]):-subset2(C,S,R).
subset2([X|C],[X|S],R):-subset2(C,S,R).

partición(L):-  subset2(L,L1,L2), suma(L1,N), suma(L2,N),
                write(L1), write('  '), write(L2), nl, fail.
partición(_).

partición2(L):- subset2(L,L1,L2), suma(L1,N), suma(L2,N),
```

```
                long(L1,NL), long(L2,NL),
                write(L1), write('  '), write(L2), nl, fail.
partición2(_).

long([],0).
long([_|L],M):- long(L,N),M is N+1.

suma([],0).
suma([X|L],P):- suma(L,P1), P is P1+X.
```