Ronda de rescate

Diego Méndez Medina

Me fue bien en los examenes, pero al final del semestre me dio covid y no pude hacer el ultimo semanal. Hago estas cinco preguntas esperando suplan la del semanal.

1. Dada una fŕrmula de la lógica proposicional φ, se define su fórmula complementaria, denotadacomo φ^c, como el resultado de sustituir en φ cada presencia de las variables proposicionales por su negación. Por ejemplo para la fórmula $p \vee q$ su complementaria es $\neg p \vee \neg q$.

   a) Da la definición recursiva de una función `fcomp` que recibe una fórmula de la lógica proposicional y regresa su complementaria.

$$
\begin{aligned}
\texttt{fcomp} &: LPROP \to LPROP \\
\texttt{fcomp}(\bot) &= \bot && \texttt{$\bot$ es una constante, no es una variable} \\
\texttt{fcomp}(\top) &= \top && \texttt{$\top$ es una constante, no es una variable} \\
\texttt{fcomp}(p) &= \neg p && \texttt{Si p es variable prop} \\
\texttt{fcomp}(\neg\varphi) &= \neg\texttt{fcomp}(\varphi) \\
\texttt{fcomp}(\varphi \vee \psi) &= \texttt{fcomp}(\varphi) \vee \texttt{fcomp}(\psi) \\
\texttt{fcomp}(\varphi \wedge \psi) &= \texttt{fcomp}(\varphi) \wedge \texttt{fcomp}(\psi) \\
\texttt{fcomp}(\varphi \to \psi) &= \texttt{fcomp}(\varphi) \to \texttt{fcomp}(\psi) \\
\texttt{fcomp}(\varphi \leftrightarrow \psi) &= \texttt{fcomp}(\varphi) \leftrightarrow \texttt{fcomp}(\psi)
\end{aligned}
$$

   b) Demuestra que si $\models \varphi$ entonces $\models \varphi^c$.

2. Considera el siguiente argumento lógico:

   Si mi cliente es culpable, entonces el cuchillo estaba en el cajón. El cuchillo no estaba en el cajón o Juan Pablo escondió el cuchillo. No es cierto que, si encontraron el cuchillo el 10 *de Septiembre entonces Juan Pablo escondió el cuchillo. Además, si no encontraron el cuchillo el* 10 *de Septiembre, entonces el cuchillo estaba en el cajón y el martillo estaba en el establo. Pero sabemos que el martillo no estaba en el establo. Por lo tanto mi cliente es inocente.*

   a) Traduce el argumento anterior a lógica preposicional, indicando el glosario empleado.
   Consideramos el siguiente glosario:

$$
\begin{aligned}
p &: \text{Mi cliente es culpable} \\
q &: \text{El cuchillo estaba en el cajón} \\
r &: \text{Juan Pablo escondió el cuchillo} \\
s &: \text{Encontraron el cuchillo el 10 de Septiembre.} \\
t &: \text{El martillo estaba en el establo.}
\end{aligned}
$$

Tenemos el siguiente argumento:

$p \to q$
$\neg q \vee r$
$\neg(s \to r)$
$\neg s \to (q \wedge t)$
$\neg t$

$\therefore \neg p$

b) Decide si el argumento es correcto o no. Indica qué método vas a utilizar para decidirlo.

Vamos a resolver con resolución binaria. Pasando el argumento a forma normal conjuntiva:

$fnc(p \to q) = \neg p \vee q$
$fnc(\neg q \vee r) = \neg q \vee r$
$fnc(\neg(s \to r)) = \neg(\neg s \vee r) = s \wedge \neg r$
$fnc(\neg s \to (q \wedge t)) \equiv s \vee (q \wedge t) = (s \wedge q) \vee (s \wedge t)$
$fnc(\neg t) = \neg t$

$\therefore \neg p$

Sea $\sigma = \{\neg p \vee q, \neg q \vee r, s \wedge \neg r, (s \wedge q) \vee (s \wedge t), \neg t\}$, las premisas del argumento dado, queremos ver si:

$$\sigma \models \neg p$$

Basta demostrar que $\sigma \cup \{p\}$ es insatisfasible. Procedemos a hacerlo:

1. $\neg p \vee q$	Hip.
2. $\neg q \vee r$	Hip.
3. $s \wedge \neg r$	Hip.
4. $(s \wedge q) \vee (s \wedge t)$	Hip.
5. $\neg t$	Hip.
6. p	Hip.
7. $s \wedge q$	Res$(4, 5)$
8. $\neg r$	Por 3.
9. q	Por 7.
10. r	Res$(2, 9)$
11. $\square$	Res$(8, 10)$

4. Considera la siguiente fórmula A:

$$A =_{def} (\forall x \exists z(\neg P(x) \to T(y, z))[y := a][z := x] \wedge \forall v(\exists x R(v, x, y) \wedge P(x)))[x, y := f(w), g(v)]$$

a) Determina el alcance de cada uno de los cuantificadores de A antes de realizar las sustituciones.

Los cuantificadores $\forall x \exists z$ tienen como alcance $\neg P(x) \to T(y, z)$.

El cuantificador $\forall v$ tiene como alcance $\exists x R(v, x, y) \wedge P(x)$.

Por último el cuantificador $\exists x$ tiene como alcance $R(v, x, y)$.

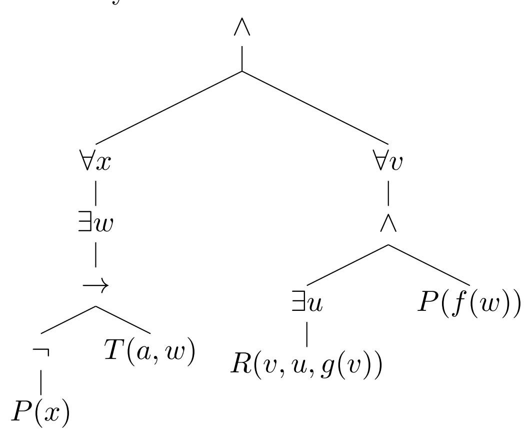
b) Obtén el conjunto de variables libres de A antes de realizar las sustituciones.
y figura libre en $\neg P(x) \to T(y, z)$.
y figura libre tambíen en $R(v, x, y)$.
x figura libre en $P(x)$.

$$FV(A) = \{y, x\}$$

c) Realiza las sustituciones indicadas para obtener una fórmula B, indicando explícitamente los pasos realizados de acuerdo al algoritmo de sustitución visto en clase.

$$\begin{aligned}
\forall x \exists z(\neg P(x) \to T(y, z))[y := a][z := x] \land \forall v(\exists x R(v, x, y) \land P(x))[x, y := f(w), g(v)] &\equiv_\alpha \\
\forall x \exists w(\neg P(x) \to T(y, w))[y := a][z := x] \land \forall v(\exists u R(v, u, y) \land P(x))[x, y := f(w), g(v)] &\equiv \\
\forall x \exists w(\neg P(x)[y := a][z := x] \to T(y, w)[y := a][z := x]) \land \forall v(\exists u R(v, u, y) \land P(x))[x, y := f(w), g(v)] &\equiv \\
\forall x \exists w(\neg P(x) \to T(a, w)) \land \forall v(\exists u R(v, u, y)[x, y := f(w), g(v)] \land P(x)[x, y := f(w), g(v)]) &\equiv \\
\forall x \exists w(\neg P(x) \to T(a, w)) \land \forall v(\exists u R(v, u, g(v)) \land P(f(w))
\end{aligned}$$

d) Construye el árbol de sintaxis abstracta de la expresión B resultante del inciso anterior.



5. Sea $\mathcal{I} = \{P^(3), Q^(2), f^(1)\}$ y $\mathcal{M} =< M, \mathcal{I} >$ donde $M = \{a, b, c\}$ y

$$\begin{aligned}
P^{\mathcal{I}} &= \{(a, a, a), (b, b, b), (b, a, b)\} \\
Q^{\mathcal{I}} &= \{(a, a), (b, c), (c, a)\} \\
f^{\mathcal{I}}(a) &= a, f^{\mathcal{I}}(b) = c, f^{\mathcal{I}}(c) = a
\end{aligned}$$

a) Decide si $\mathcal{M} \models_\sigma \forall x(Qxfx \land \exists x(Payfx \lor Pbyfx))$ donde $\sigma(y) = a$
Lo que em realidad queremos ver es si:

$$\begin{aligned}
\mathcal{M} &\models \forall x(Qxfx \land \exists x(Paafx \lor Pbafx)) \\
\mathcal{M} &\models \forall x(Qxfx \land \exists y(Paafy \lor Pbafy)) \\
\mathcal{M} &\models \forall x Qxfx \land \exists y(Paafy \lor Pbafy)
\end{aligned}$$

Primero veremos que

$$\mathcal{M} \models \forall x Qxfx$$

Checamos los elementos del universo:

- Para $x = a$

$$Qafa =_{\mathcal{I}} Qaa \in Q^{\mathcal{I}}$$

- Para $x = b$

$$Qbfb =_{\mathcal{I}} Qbc \in Q^{\mathcal{I}}$$

- Para $x = c$

$$Qcfc =_{\mathcal{I}} Qca \in Q^{\mathcal{I}}$$

Entonces en efecto

$$\mathcal{M} \models \forall x Qxfx$$

Ahora falta ver que:

$$\mathcal{M} \models \exists y(Paafy \vee Pbafy)$$

Para $y = a$ tenemos:

$$Paafa = Paaa \in P^{\mathcal{I}}$$

Concluimos que $\mathcal{M} \models_\sigma \forall x(Qxfx \wedge \exists x(Payfx \vee Pbyfx))$ donde $\sigma(y) = a$

b) Define un estado σ tal que $\mathcal{M} \not\models_\sigma \exists x(\neg Qxx) \wedge Pyfyffy \to Qxy \vee Qyfx$
Primero hacemos una observación:

$$\exists x(\neg Qxx) \wedge Pyfyffy \to Qxy \vee Qyfx \equiv \exists x(\neg Qxx) \wedge Pyfyffy \to Qzy \vee Qyfz$$

Entonces hay que ver un estado σ tal que $\mathcal{M} \not\models_\sigma \exists x(\neg Qxx) \wedge Pyfyffy \to Qzy \vee Qyfz$
σ es tal que:

$$\sigma(y) = a$$
$$\sigma(z) = b$$

Así el antecedente de la implicación es verdadero pero el lado derecho es falso. Veamos por que:

$$\mathcal{M} \models_\sigma \exists x(\neg Qxx)$$

Pues para $x = b$ tenemos $Qbb \notin Q^{\mathcal{I}}$.

$$\mathcal{M} \models_\sigma Pyfyffy$$

$$Pyfyffy =_\sigma Pafaffa = Paafa = Paaa \in P^{\mathcal{I}}.$$

Y ahora veremos que

$$\mathcal{M} \not\models_\sigma Qzy \vee Qyfz$$

$$Qzy =_\sigma Qba \notin Q^{\mathcal{I}}$$
$$Qyfz =_\sigma Qafb = Qac \notin Q^{\mathcal{I}}$$

Entonces para la σ dada $\mathcal{M} \not\models_\sigma \exists x(\neg Qxx) \wedge Pyfyffy \to Qxy \vee Qyfx$

c) Decide si $\mathcal{M} \models \exists x(Pxxx \wedge Qxfx \wedge \neg Qxx)$
Es cierto pues para $x = b$ tenemos:

$$\begin{aligned} Pbbb &\in P^{\mathcal{I}} \\ Qbfb = Qbc &\in Q^{\mathcal{I}} \\ Qbb &\notin Q^{\mathcal{I}} \end{aligned}$$

9. Considera el siguiente programa lógico.

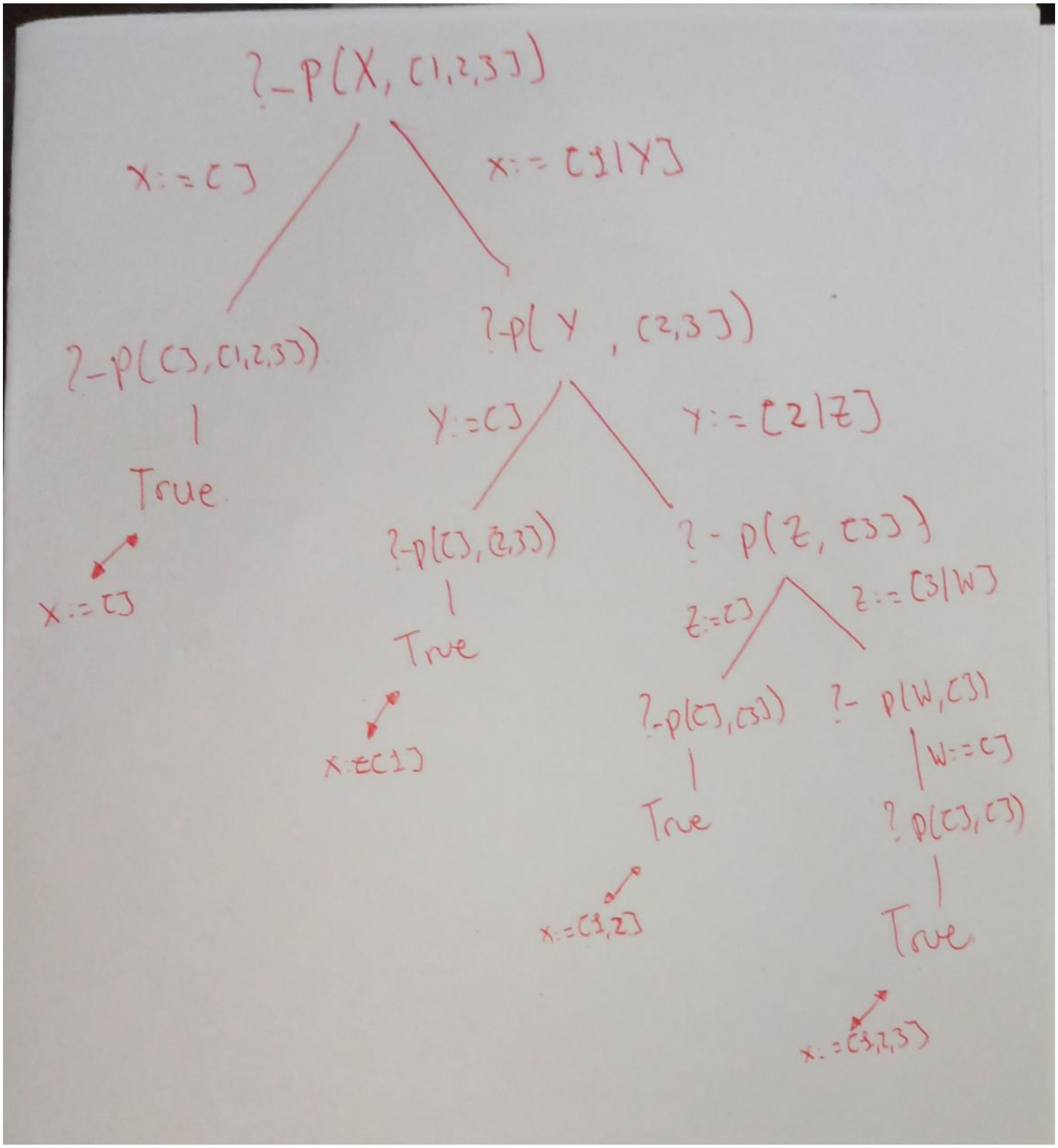
```
p ([ ],R).
p ([H|T1], [H|T2]) :- p (T1,T2).
```

a) Da una especificación informal del predicado `p`.
`p` solo regresa `true` si se itera sobre la lista izquierda hasta llegar a la lista vacía.
Y por el segundo enunciado sabemos que `p` solo continua iterando ambas listas si la cabeza de ambas listas son iguales.

`p` indica si toda la primera lista recibida es sublista de la segunda lista recibida.

b) Construye el árbol de búsqueda para la meta `?- p(X,[1,2,3]`.

No se poner las sustituciones en las aristas, imagen en la siguiente pagina


?-p(X, [1,2,3])
X:=[]
X:=[1|Y]
?-p([],[1,2,3])
True
X:=[]
?-p(Y , [2,3])
Y:=[]
Y:=[2|Z]
?-p([],[2,3])
True
X:=[1]
?-p(Z, [3])
Z:=[]
Z:=[3|W]
?-p([],[3])
True
X:=[1,2]
?-p(W,[])
W:=[]
?-p([],[])
True
X:=[1,2,3]