# Práctico 1: Cálculo proposicional y Expresiones cuantificadas

Algoritmos y Estructuras de Datos I
1$^{\text{er}}$ cuatrimestre 2016

*Un objetivo de esta guía es retomar la práctica del cálculo proposicional y de predicados, e introducirnos al cálculo con cuantificadores generales. Otro objetivo es recuperar la práctica en la definición de funciones recursivas y demostraciones por inducción. Lograr familiaridad con los axiomas y teoremas del cálculo, y habilidad en las demostraciones es necesario para poder encarar la tarea de derivación y demostración de programas que encararemos de aquí en adelante.*

1. Demostrá los siguientes teoremas del cálculo proposicional utilizando únicamente los axiomas del digesto. Cuando el axioma se aplique en una subfórmula, subrayala para ayudar a distinguir la corrección del paso aplicado. Por ejemplo, la siguiente es una demostración de $(p \not\equiv (q \not\equiv r)) \equiv ((p \not\equiv q) \not\equiv r)$:

$$
\begin{array}{ll}
 & (p \not\equiv (q \not\equiv r)) \\
\equiv & \{ \text{ Definición de } \not\equiv \ \} \\
 & \neg(p \equiv \underline{(q \not\equiv r)}) \\
\equiv & \{ \text{ Definición de } \not\equiv \ \} \\
 & \neg(p \equiv \underline{\neg(q \equiv r)}) \\
\equiv & \{ \text{ Definición de } \neg \ \} \\
 & \neg(p \equiv \neg q \equiv r) \\
\equiv & \{ \text{ Definición de } \not\equiv \ \} \\
 & \underline{(p \equiv \neg q)} \not\equiv r \\
\equiv & \{ \text{ Definición de } \not\equiv \ \} \\
 & \underline{\neg(p \equiv q)} \not\equiv r \\
\equiv & \{ \text{ Definición de } \not\equiv \ \} \\
 & (p \not\equiv q) \not\equiv r
\end{array}
$$

   a) *Idempotencia de la conjunción*: $p \wedge p \equiv p$.

   b) *Neutro de la conjunción*: $p \wedge True \equiv p$.

   c) *Absorción de la conjunción*: $p \wedge (p \vee q) \equiv p$

2. Los siguientes teoremas van a ser de mucha utilidad a lo largo de toda la materia, por lo tanto es útil recordarlos! Demostralos utilizando los axiomas y teoremas del cálculo proposicional listados en el digesto (y cualquier otro teorema que se te ocurra y demuestres, claro). Tené en cuenta las siguientes ayudas que pueden serte útiles:

   - Si hay que demostrar una equivalencia, puede ser mejor comenzar con el término más *complejo* e intentar llegar al termino más simple.
   - La Regla Dorada siempre es útil cuando el teorema incluye la conjunción, ya que permite obtener una fórmula que incluye la disjunción, para la cual existen mas reglas.
   - Las reglas de distributividad ayudan a eliminar paréntesis.
   - Para resolver un teorema se pueden utilizar los demás teoremas ya demostrados, sobre todo si tienen la misma *pinta*.

   a) *Absorción de la disyunción*: $p \vee (p \wedge q) \equiv p$

   b) *Debilitamiento para* $\wedge$: $p \wedge q \Rightarrow p$.

   c) *Debilitamiento para* $\vee$: $p \Rightarrow p \vee q$.

   d) *Relación entre* $\Rightarrow$ *y* $\vee$: $p \Rightarrow q \equiv \neg p \vee q$.

   e) *Contrarrecíproca*: $p \Rightarrow q \equiv \neg q \Rightarrow \neg p$.

   f) *De Morgan para* $\wedge$: $\neg(p \wedge q) \equiv \neg p \vee \neg q$

*g*) *De Morgan para* $\vee$: $\neg(p \vee q) \equiv \neg p \wedge \neg q$

*h*) *Distributividad de* $\vee$ *con* $\wedge$: $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

*i*) *Distributividad de* $\wedge$ *con* $\vee$: $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

*j*) *Intercambio para* $\Rightarrow$: $p \Rightarrow (q \Rightarrow r) \equiv p \wedge q \Rightarrow r$.

*k*) *Implicación de la disyunción*: $p \vee q \Rightarrow r \equiv (p \Rightarrow r) \wedge (q \Rightarrow r)$.

*l*) *Distributividad de* $\Rightarrow$ *con respecto a* $\wedge$: $p \Rightarrow (q \wedge r) \equiv (p \Rightarrow q) \wedge (p \Rightarrow r)$.

*m*) *Relación de* $\equiv$ *con* $\wedge$ *y* $\vee$: $p \equiv q \equiv (p \wedge q) \vee (\neg p \wedge \neg q)$

3. Definí recursivamente la función $\Notin$: $A \to [A] \to Bool$ que determina si un elemento pertenece a una lista, es decir, $x \Notin xs$ devuelve $True$ si y sólo si el valor $x$ ocurre al menos una vez dentro de la lista $xs$.

4. Para cada una de las siguientes fórmulas, describí su significado utilizando el lenguaje natural, y escribí fórmulas equivalentes utilizando la función $\Notin$ del ejercicio anterior.

   *a*) $\langle \forall\, i \ : 0 \leq i < \#xs : \ xs.i > 0 \,\rangle$

   *b*) $\langle \exists\, i \ : 0 \leq i < \#xs : \ xs.i = x \,\rangle$

   *c*) $\langle \forall\, i \ : 0 \leq i < \#xs : \ \langle \exists\, j \ : 0 \leq j < \#ys : \ xs.i = ys.j \,\rangle \,\rangle$

   *d*) $\langle \forall\, i \ : 0 \leq i < \#xs - 1 : \ xs.i = xs.(i+1) \,\rangle$

5. Definí recursivamente una función $paratodo : [Bool] \to Bool$ que verifica que todos los elementos de una lista son $True$, es decir, que satisface la siguiente especificación:

$$paratodo.xs \equiv \langle \forall\, i \ : 0 \leq i < \#xs : \ xs.i \,\rangle$$

6. Suponiendo que $f : A \to Bool$ es una función fija cualquiera, y $xs : [A]$, caracterizá con una fórmula la siguiente función recursiva:

$$existe : [A] \to Bool$$

$$existe.[\,] = False$$
$$existe.(x \triangleright xs) = f.x \vee exists.xs$$

7. Enunciá los axiomas de *rango vacío*, *rango unitario*, *partición de rango*, *término*, *término constante*, *distributividad*, y *anidado* para cada uno de los siguientes cuantificadores:

   *a*) Cuantificador universal ($\forall$)
   *b*) Cuantificador existencia ($\exists$)
   *c*) Sumatoria ($\sum$)
   *d*) Productoria ($\prod$)
   *e*) Máximo (Max)
   *f*) Mínimo (Min)
   *g*) Intesección ($\bigcap$)
   *h*) Unión ($\bigcup$)

8. Considerá una función $sumatoria : [Num] \to Num$ que suma todos los elementos de una lista. Por un lado, escribí una especificación que caraterice esta función, utilizando el cuantificador $\Sigma$. Por otro lado, escribí un programa recursivo que compute la función.

9. Suponé que *sumatoria* es la función del ejercicio anterior. Aunque pueda parecer extraño en principio, **no es cierto** que

$$sumatoria.xs = \langle \sum x \ : x \Notin xs : \ x \,\rangle$$

   ¿Se te ocurre un contraejemplo que demuestre tal afirmación?

10. Suponiendo que $T : A \to Bool$ es un predicado cualquiera, y $xs : [A]$, demostrá por inducción en $xs$ que

$$\langle \forall\, i \ : 0 \leq i < \#xs : \ T.(xs.i) \,\rangle \equiv \langle \forall\, x \ : x \Notin xs : \ T.x \,\rangle$$

    Utilizá los axiomas de $\forall$ del digesto, y la definición de $\Notin$ del ejercicio 3.

11. Demostra los siguientes teoremas sobre $\forall$, utilizando los axiomas y teoremas del digesto:

    *a*) *Instanciación de* $\forall$: $\langle \forall i \ : : \ T.i \rangle \Rightarrow T.x$, cuando $x$ no está cuantificada.
    ¿Como sería la regla de instanciación para $\exists$? Enunciala y demostrala.

    *b*) *Intercambio para* $\forall$ *(generalizada)*: $\langle \forall i \ : \ R.i \wedge S.i \ : \ T.i \rangle \equiv \langle \forall i \ : \ R.i \ : \ S.i \Rightarrow T.i \rangle$

12. Podemos definir un cuantificador de *conteo* $N$ utilizando la sumatoria:

$$\langle \mathrm{N}\, i \ : R.i : \ T.i \,\rangle \doteq \langle \sum i \ : R.i \wedge T.i : \ 1 \,\rangle$$

   *a*) Enunciá y demostrá las reglas de *rango vacío*, *rango unitario* y *partición de rango* para $N$.

   *b*) Demostrá que $\langle \sum i \ : \ R.i \wedge T.i \ : \ k \rangle = \langle \mathrm{N} i \ : \ R.i \ : \ T.i \rangle * k$

13. Demostrá la siguiente relación entre los cuantificadores de máximo y el mínimo:

$$z = \langle \mathrm{Min} \ i \ : \ R.i \ : \ -T.i \rangle \equiv z = -\langle \mathrm{Max} \ i \ : \ R.i \ : \ T.i \rangle$$

14. Definí recursivamente las siguientes funciones.

   *a*) La función *paratodo′*, que dada una lista de valores $xs : [A]$ y un predicado $T : A \to Bool$, determina si todos los elementos en $xs$ hacen verdadero el predicado $T$, es decir:

$$paratodo' : [A] \to (A \to Bool) \to Bool$$

$$paratodo'.xs.T \equiv \langle \forall\, i \ : 0 \leq i < \#xs : \ T.(xs.i) \,\rangle$$

   Puede ser de ayuda recordar la función del ejercicio 5.

   *b*) La función *existe′*, que dada una lista de valores $xs : [A]$ y un predicado $T : A \to Bool$, determina si algún elemento en $xs$ hace verdadero el predicado $T$, es decir:

$$existe' : [A] \to (A \to Bool) \to Bool$$

$$existe'.xs.T \equiv \langle \exists\, i \ : 0 \leq i < \#xs : \ T.(xs.i) \,\rangle$$

   Puede ser de ayuda recordar la función del ejercicio 6.

   *c*) La función *sumatoria*, que dada una lista de valores $xs : [A]$ y una función $T : A \to Num$ (toma elementos de $A$ y devuelve números), calcula la suma de la aplicación de $T$ a los elementos en $xs$ es decir:

$$sumatoria' : [A] \to (A \to Num) \to Num$$

$$sumatoria'.xs.T \equiv \langle \sum i \ : 0 \leq i < \#xs : \ T.(xs.i) \,\rangle$$

   Puede ser de ayuda recordar la función del ejercicio 8.

   *d*) La función *productoria*, que dada una lista de valores $xs : [A]$ y una función $T : A \to Num$, calcula el producto de la aplicación de $T$ a los elementos de $xs$, es decir:

$$productoria : [A] \to (A \to Bool) \to Bool$$

$$productoria.xs.T \equiv \langle \prod i \ : 0 \leq i < \#xs : \ T.(xs.i) \,\rangle$$

15. Demostrá los teoremas de Separación de Término para un cuantificador arbitrario:

   *a*) $\langle \bigoplus i \ : 0 \leq i < n+1 : \ T.i. \,\rangle = \langle \bigoplus i \ : 0 \leq i < n : \ T.i \,\rangle \oplus T.n$

   *b*) $\langle \bigoplus i \ : 0 \leq i < n+1 : \ T.i \,\rangle = T.0 \oplus \langle \bigoplus i \ : 0 \leq i < n : \ T.(i+1) \,\rangle$

16. Todas las funciones del ejercicio 14 son similares entre sí: cada una aplica la función término $T$ a todos los elementos de una lista, y luego aplica algún operador entre todos ellos, obteniendose así el resultado final. Para el caso de la lista vacía, se devuelve el elemento neutro.

   Guiándote por esta observación, definí de manera recursiva la función *cuantGen* (denota la cuantificación generalizada) que tomando como argumento un operador, su elemento neutro, una lista de elementos y una función término, aplica el operador a los elementos de la lista, transformados por la función término:

$$cuantGen : (B \to B \to B) \to B \to [A] \to (A \to B) \to B$$

$$cuantGen.\oplus.z.rs.T = \langle \bigoplus x \ : x \in rs : \ T.x \,\rangle$$