FUNDAMENTOS DE COMPUTACIÓN
Repartido: Bool Parte 2 - La matemática de los booleanos

1. Las afirmaciones del Lenguaje Matemático

Desde la introducción, este curso se ha propuesto como un curso de Matemática, en particular de la Matemática de los programas.
¿Cuál es entonces la Matemática de los booleanos? ¿Qué teoremas interesantes podemos producir?
Al estudiar propiedades de los booleanos y de los programas que operan con ellos, vamos a estar sobre todo interesados en **demostrar afirmaciones** de la forma: **Para todo booleano b se cumple P(b)**, donde **P** es una propiedad concreta. En adelante, esto será escrito así (**∀b :: Bool) P(b)**.

La primera observación importante es que las afirmaciones a considerar **no son parte del cálculo lambda** que estamos desarrollando. Los tipos y objetos de este lenguaje son, obviamente, mencionados en esas afirmaciones, pero ellas son formuladas en el **metalenguaje o lenguaje matemático**. Es decir, así como venimos desde el comienzo hablando en general, acerca del cálculo lambda, lo continuaremos haciendo, pero ahora formulando y demostrando propiedades de sus objetos con rigurosidad matemática.

El cuerpo de la afirmación típica a considerar, es decir **P(b),** será una afirmación que menciona a la variable **b**. Posibles ejemplos de estas propiedades podrían ser:
not (not b) = b o **b && False = False**.

O sea, P(b) afirma algo acerca del booleano genérico b. Podemos entonces separar P y b simplemente: todo lo que queda de P(b) cuando omitimos b, es la propiedad P.
En los ejemplos precedentes:
- Si P(b) es **not (not b) = b**, entonces P es **not (not ●) = ●**
- Si P(b) es **b && False = False**, entonces P es **● && False = False**,

donde hemos dejado un hueco en el lugar del booleano genérico b.

En otras palabras, P es una expresión de nuestro lenguaje matemático que espera ser completada (saturada) con un individuo (en este caso, cualquier expresión booleana) para convertirse en una afirmación.
Este tipo de expresiones se llaman propiedades o predicados. Un ejemplo de predicado en el contexto de la aritmética podría ser la siguiente expresión: **● es par.**
Cuando el hueco es rellenado por un individuo (en este caso, un entero), se obtiene una afirmación, como por ejemplo **5 es par**, que puede ser verdadera o falsa. Si es verdadera, decimos que el individuo **5** cumple la propiedad (en este caso sabemos que no se cumple).

Lo importante acerca de estas afirmaciones de la forma P(b) es que debemos ser capaces de **sustituir en ellas la variable b por otras expresiones**, según nos convenga para proceder con el razonamiento.
Por suerte, ya estamos entrenados en estos menesteres, gracias al estudio de la sustitución que hemos hecho para las expresiones del cálculo lambda .

2. Demostraciones por casos

Ahora bien, cuando adjuntamos a P(b) el prefijo (∀b :: Bool) estamos haciendo otra afirmación: **P(b) es verdadera para todos los casos posibles de b :: Bool,** o, en otras palabras, que todos los booleanos cumplen la propiedad P.
Recordemos que hay infinitas expresiones booleanas, por ejemplo: True, not True, not(not True), not(not(not True)), etc... Pero por la definición dada del tipo Bool, sabemos que cualquier expresión booleana va a ser por definición igual a alguno de los valores False o True (y no hay otra opción!).

Esto justifica el siguiente método de demostración de propiedades de los booleanos:

Para demostrar que todos los booleanos cumplen una cierta propiedad P, alcanza claramente con demostrar:

- **el caso False:** o sea, demostrar **P(False)**
- **el caso True:** o sea, demostrar **P(True).**

Aclaremos esto con un ejemplo de una propiedad a demostrar por el método recién presentado. Demostraremos el un lema sobre la función **not**.

Copiamos aquí la definición de **not** dada para tenerla cerca:

not :: Bool → Bool
not = λb→ case b of {False → True ; True → False }

La propiedad a demostrar se escribe como el siguiente lema:

$Lema_{not}$: (∀b :: Bool) not (not b) = b.

Observar que la propiedad P a demostrar es **not (not ●) = ●.**
Los casos a demostrar se verán en la siguiente demostración del lema.

Demostración: Por casos en b:: Bool.
Caso False: Hay que demostrar not (not False) = False *{o sea, P(False)}*

Dem. not (not False)
= (def. not)
not ((λb→case b of {False → True ; True → False }) False)
= (regla β)
not (case False of {False → True ; True → False })
= (case False)
not True
= (def. not, regla β, case True)
False.

Caso True: Hay que demostrar not (not True) = True *{o sea, P(True)}*

Dem. not (not True)
= (def. not, regla β, case True)
not False
= (def. not, regla β, case False)
True.

□

Con estos dos casos acabamos de demostrar que para cualquier booleano b se cumple que not (not b) = b.

Cuando debamos demostrar una igualdad entre dos expresiones e1 y e2, vamos a utilizar la forma de demostración **ecuacional directa**. O sea, partimos de e1, y paso a paso vamos obteniendo expresiones intermedias mediante igualdades debidamente justificadas hasta llegar a e2. Como la igualdad es una relación **transitiva**, si tenemos una cadena de expresiones iguales, se cumple que la primera (e1) es igual a la última (e2).

Con el mismo método podemos demostrar la segunda propiedad enunciada al principio:

Lema1$_{\&\&}$: (∀b :: Bool) b && False = False.

Acá la propiedad P a demostrar es ● **&& False = False**.

Recordemos la definición de la función (&&):

(&&) :: Bool → Bool → Bool
(&&) = λb1→ λb2→ case b1 of { False → False ; True → b2 }

Procedemos a demostrar este lema por casos:

Demostración: Por casos en b:: Bool.
Caso False: False && False =$^{?}$ False

Dem. False && False
= (def. &&)
(λb1→λb2→ case b1 of {False → False ; True → b2}) False False
= (regla β x 2)
case False of {False → False ; True → False}
= (case False)
False.

Caso True:True && False =$^{?}$ False

Dem. True && False
= (def. &&, regla β x 2)
case True of {False → False ; True → False}
= (case True)
False.

□

Observar que no todas las demostraciones deben hacerse por casos en el booleano.
Por ejemplo, el siguiente lema se demuestra directamente, sin necesidad de analizar el valor de b:

Lema2$_{\&\&}$: (∀b :: Bool) False && b = False

Esta igualdad se va a cumplir aplicando directamente la definición de &&, por lo tanto, para demostrar (∀b :: Bool) P(b) en este caso, tomaremos un **b :: Bool genérico** y demostraremos la igualdad:

Demostración: Sea b :: Bool cualquiera *{se elige un b genérico}*
False && b
= (def. &&, regla β x 2)
case False of {False → False ; True → b}
= (case False)
False.
□

Del mismo modo que demostramos los lemas que dicen cómo se comporta el && cuando uno de sus argumentos es False, se pueden demostrar también los siguientes resultados para cuando uno de los argumentos es True:

Lema3$_{\&\&}$: (∀b :: Bool) b && True = b

Lema4$_{\&\&}$: (∀b :: Bool) True && b = b

Dejamos estas demostraciones como *ejercicio*.
Una de ellas sale directamente y la otra requiere una demostración por casos en b. ¿Se puede ver fácilmente cuál es cuál y por qué?

Vamos ahora a demostrar un resultado más interesante, como es el caso de la conmutatividad del &&:

Conm$_{\&\&}$: (∀b1 :: Bool)(∀b2 :: Bool) b1 && b2 = b2 && b1.
La demostración se hará por casos en alguno de los booleanos b1 o b2.
La haremos por casos en b1, pero también podríamos hacerlo con b2.

Observar que al hacer casos en b1, la propiedad P que estamos demostrando es la siguiente: P = (**∀b2 :: Bool) ● && b2 = b2 && ●.**

Veamos cómo sería la demostración:

<u>Demostración</u>: Por casos en **b1:: Bool**.
Caso False: $(\forall$ b2 :: Bool) False && b2 $=^?$ b2 && False

Acá podemos tomar un b2 cualquiera, y sabemos que (por los Lemas 1 y 2) ambas expresiones son iguales a False.
Una forma de escribir la demostración de este caso es entonces, del siguiente modo:

Dem. Sea b2 :: Bool cualquiera.
False && b2
= (Lema2$_{\&\&}$)
False
= (Lema1$_{\&\&}$)
b2 && False.

Observar que estamos utilizando el Lema1$_{\&\&}$ "al revés", o sea de la forma
b2 && False = False.
Esto lo podemos hacer porque la igualdad es una relación **simétrica**, o sea, si **e1 = e2** entonces también se cumple que **e2 = e1**.

Otra forma de demostrar una igualdad es utilizando las propiedades que ya mencionamos de la igualdad (transitividad y simetría), y razonar **por reducción a la misma expresión.** Esto es: para demostrar que **e1 = e2**, demostramos que **existe una tercera expresión e3** tal que **e1 = e3** y **e2 = e3**. De esta última igualdad obtenemos que e3 = e2 (por simetría), y finalmente, como e1 = e3 y e3 = e2, obtenemos e1 = e2 por transitividad.

Vamos a realizar de este modo la demostración del caso **True** de la demostración de la conmutatividad del &&. Dividimos la hoja en dos, y avanzamos en cada lado mediante igualdades directas, hasta llegar a una misma expresión:

Caso True : $(\forall$ b2 :: Bool) True && b2 $=^?$ b2 && True
Dem. Sea b2 :: Bool cualquiera

True && b2	b2 && True
= (Lema4$_{\&\&}$)	= (Lema3$_{\&\&}$)
b2	b2

Ambas expresiones son iguales por ser iguales a la misma expresión b2.

□