Ideas de teoría de tipos

Mario Román

<2016-01-08 Fri 15:13>

Una recopilación de algunas ideas y enlaces después de haber empezado a leer sobre teoría de tipos.

Los sistemas de tipos tienen su utilidad en las matemáticas. Sirven para modelar una fundamentación de las matemáticas distinta de la usual fundamentación conjuntista; y tienen varias aplicaciones interesantes en lenguajes funcionales y asistentes de demostración. En particular, sobre los tipos se puede definir un álgebra y se pueden representar sistemas lógicos. Vamos a tratar esas aplicaciones referenciando en cada caso artículos donde se exponen en profundidad.

Inducción estructural

La inducción estructural es una generalización de la inducción usual sobre los naturales que la extiende a otras estructuras representables como tipos de un lenguaje funcional. Sobre la inducción estructural hemos escrito previamente en el blog una introducción:

- Inducción Estructural - Blog LibreIM

En ese post se escriben ejemplos sobre los naturales y los árboles. El artículo sobre el que se basa es:

- When can we do induction? - math.blogoverflow

Ejemplos y más detalles sobre inducción estructural y sus usos pueden encontrarse en:

- Some notes on Structural Induction - Michael Erdmann
- Structural Induction Principles for Functional Programmers - James Caldwell

Y varias demostraciones por inducción estructural implementadas en Coq en este repositorio sobre **recorridos en árboles**.

Álgebra de tipos

En un post anterior del blog de **LibreIM** hemos tratado el álgebra de tipos. Ese post se basó sobre otros tres publicados en el blog de Chris Taylor:

- Álgebra de tipos - Blog LibreIM
- The algebra of algebraic data types, Part I - Chris Taylor
- The algebra of algebraic data types, Part II - Chris Taylor
- The algebra of algebraic data types, Part III - Chris Taylor

En el segundo de los artículos se usan funciones generadoras para probar resultados sobre los números de Catalan y los árboles binarios. La teoría de funciones generadoras necesaria para entender el tratamiento de los árboles binarios la explica Mike Spivey en *The catalan numbers from their generating function*.

Además, existe un resultado de **Fiore y Leinster** que afirma que si demostramos una relacion polinómica para números complejos, también será válida para cualquier semianillo. Y por tanto, para los tipos. Esto quiere decir que, en la mayoría de las ocasiones, podemos usar la resta o la división de tipos como si existieran. La demostración excluye algunos casos particulares y se expone aquí:

- Objects of categories as complex numbers - Marcelo Fiore y Tom Leinster

Sobre el uso de las derivadas en el álgebra de tipos existe un resultado de **Conor McBride** que relaciona las derivadas parciales con los *zippers* de Haskell usados para representar contextos. Puede leerse aquí:

- The derivative of a regular type is its type of one-hole contexts - Conor McBride.

Lógica con tipos

La aplicación de los tipos a la lógica y las demostraciones parte del isomorfismo de Curry-Howard, que relaciona los sistemas de tipos con sistemas lógicos. El sistema más simple donde puede apreciarse el isomorfismo es el **cálculo lambda tipado**, que es isomorfo a la **deducción natural**. La deducción natural es un ejemplo de lógica intuicionista, lo que en la práctica quiere decir que *no* (!) se tienen el *tercio excluso* y la *doble negación* como axiomas:

$$A \vee \neg A$$

$$\neg\neg A \implies A$$

Los apuntes sobre **Curry-Howard** de los repositorios del doble grado explican el isomorfismo sobre la deducción natural y el cálculo lambda tipado. El código fuente acompañando los apuntes está escrito en Coq y Haskell.

La idea de tratar las proposiciones como tipos la expone **Philip Wadler** en los dos siguientes artículos: primero de manera didáctica, con una introducción histórica y sobre el sistema de la deducción natural, y luego de forma más compleja, exponiendo el isomorfismo sobre el sistema de tipos de Haskell.

- Propositions as Types - Philip Wadler
- The Girard-Reynolds Isomorphism - Philip Wadler

Parametricidad

La parametricidad limita las instancias posibles de los tipos de la forma `forall a. p(a)`, y nos permite obtener teoremas sobre todas las instancias de esos tipos. Se explica a nivel intuitivo en el siguiente post de Bartosz Milewski y más formalmente en este paper de Philip Wadler:

- Parametricity: Money for Nothing and Theorems for Free - Bartosz Milewski
- Theorems for free! - Philip Wadler

Teoría de tipos

Los tipos pueden usarse para fundamentar las matemáticas, del mismo modo que lo hacen los conjuntos (en sistemas axiomáticos como ZFC) o las categorías (en sistemas como ETCS). En el siguiente artículo se discuten las diferencias de ambos con la **teoría de tipos dependientes de Martin-Löf**, que se expone por completo en las notas de Nordström, Petersson y Smith:

- From Set Theory to Type Theory - The n-Category Café
- Martin-Löf Type Theory - B. Nordström, K. Petersson, J.M. Smith

Esto nos da una fundamentación de las matemáticas con una interpretación computacional clara.

Una refinación de esas teorías para producir una fundamentación también constructivista de las matemáticas es el Cálculo de Construcciones (Calculus of constructions, COC) desarrollado por **Thierry Coquand** y **Gérard Huet**, que finalmente dará lugar al asistente de demostraciones **COQ**, desarrollado por el INRIA. El **λ-cubo** es un diagrama para exponer cómo este sistema amplía al *cálculo lambda tipado* y al *Sistema* F_ω que usa Haskell.

- Calculus of Constructions - T. Coquand, G. Huet.
- The Coq proof assistant - INRIA