Propuesta de Tesina
Licenciatura en Ciencias de la Computación

16 de marzo de 2023

Postulante Agustín Settimo – S-5289/2

Director Cecilia Manzino

Codirector Cristian Sottile

1. Situación del Postulante

El postulante ya ha aprobado la totalidad de las asignaturas de la carrera.

Actualmente el postulante se encuentra trabajando como desarrollador de software con una dedicación de 40hs semanales. Se espera dedicar 20hs semanales a la realización de la Tesina.

2. Título

Formalización de Sistema I con tipo Unit.

3. Introducción

Tanto los sistemas de tipos como los sistemas de pruebas distinguen elementos que tienen diferente forma aunque tengan el mismo significado, como pueden ser las pruebas de las conjunciones $A \wedge B$ y $B \vee A$, por lo cual una prueba de una no constituye una prueba de la otra, a pesar de que se puede demostrar mediante la existencia de un isomorfismo que dichas proposiciones son equivalentes. Sistema I es un cálculo lambda simplemente tipado con pares, extendido con una teoría ecuacional obtenida a partir de los isomorfismos de tipos existentes entre los tipos simples con pares, de forma tal que las proposiciones con mismo significado son equivalentes.

Este calculo tiene aplicaciones interesantes. Por ejemplo, desde el punto de vista de los programas, nos permite construir expresiones de formas que antes eran erróneas, un termino $f : (A \wedge B) \Rightarrow C$ puede ser combinado como $f\langle a, b\rangle$ ó $f\ a\ b$, ya que por el isomorfismo de curry el termino f también tiene tipo $A \Rightarrow B \Rightarrow C$. Por otro lado, los isomorfismos hacen que las pruebas sean más naturales, por ejemplo, para probar, $(A \wedge A \Rightarrow B) \Rightarrow B$, deberíamos primero introducir la hipótesis $(A \wedge A \Rightarrow B)$, y luego descomponerla en A y $A \Rightarrow B$, en cambio utilizando el isomorfismo de curry transformamos el objetivo en $(A \Rightarrow A \Rightarrow B) \Rightarrow B$ y luego introducimos directamente las hipótesis A y $A \Rightarrow B$.

4. Fundamentos y estado del arte

El primer paso es considerar dos proposiciones A y B como isomorfas si existen dos pruebas $A \Rightarrow B$ y $B \Rightarrow A$, tal que al componerlas, en ambos sentidos, obtenemos como resultado la identidad. Di Cosmo [1] caracterizó los conjuntos mínimos de isomorfismos que permiten construir todos los demás.

La primera versión de Sistema I introducida en [2] define una relación de equivalencia entre los términos, inducida por los isomorfismos de tipos. Si bien este sistema es fuertemente normalizante, no posee la propiedad de introducción (todo termino cerrado en forma normal es una introducción).

Luego, Sistema I^{η} [3] añade la regla de η-expansión a la relación de reducción lo cual permite desatascar los términos que impedían obtener la propiedad de introducción en Sistema I.

En términos de expresividad, un siguiente paso natural para un lenguaje de programación simplemente tipado consiste en la adición de polimorfismo. Desde el punto de vista lógico, esto significa llevarlo de ser un lenguaje capaz de expresar especificaciones formales en primer orden, a uno capaz de expresar especificaciones en segundo orden. Justamente esto propone SIP [4] (Sistema I Polimórfico).

Si bien existe una implementación en Haskell de Sistema I [5], no se hayan formalizaciones en asistentes de pruebas como *Agda* o *Coq*. Por otro lado, no se han explorado los isomorfismo que surgen de la inclusión del tipo Unit. Por ejemplo, el isomorfismo $\top \equiv \top \Rightarrow \top$ permite tipar el termino $\Omega = (\lambda x^{\top}.xx)(\lambda x^{\top}.xx) : \top$. Una de las preguntas que surgen a

5. Objetivos específicos

Proponemos extender Sistema I agregando el tipo $\top$. Para esto, añadiremos el tipo $\top$ y las reglas de tipado. Luego consideraremos tres nuevos isomorfismos correspondientes a dicho tipo, aplicando la misma técnica utilizada en SIP para internalizar los isomorfismos extenderemos la relación de equivalencia entre términos.

Luego realizaremos una formalización de toda la teoría en Agda, lo cual incluye la definición de los tipos y términos, las reglas de tipado, la relación de isomorfismo entre tipos, la relación de equivalencia entre términos, casos de ejemplos y la demostración de propiedades. Analizaremos los diferentes estilos de formalización para elegir el más adecuado tomando como referencia formalizaciones de STLC preexistentes.

Evaluaremos también la posibilidad de formalizar la prueba de normalización fuerte para Sistema I con $\top$.

6. Metodología y Plan de Trabajo

Para alcanzar los objetivos planteados se propone el siguiente programa tentativo de trabajo.

- Estudio de Sistema I y revisión del estado del arte. ($\sim$ 2 semanas)
- Búsqueda de formalizaciones de STLC en Agda que sirvan de base. ($\sim$ 1 semanas)

- Extensión de Sistema I con tipo Unit y los isomorfismos correspondientes. ($\sim$ 1 semanas)
- Formalización en Agda de Sistema I. ($\sim$ 3 semanas)
- Demostración de propiedades en Agda. ($\sim$ 2 semanas)
- Síntesis de los resultados obtenidos y escritura de la tesina. ($\sim$ 6 semanas)

Referencias

[1] Roberto Di Cosmo. Isomorphisms of types: from λ-calculus to information retrieval and language design. Progress in Theoretical Computer Science. Birkhauser, 1995.

[2] Alejandro Díaz-Caro y Gilles Dowek. Proof normalisation in a logic identifying isomorphic propositions. Schloss Dagstuhl-LeibnizZentrum fuer Informatik, 2019.

[3] Alejandro Díaz-Caro y Gilles Dowek. Extensional proofs in a propositional logic modulo isomorphisms, 2020.

[4] Sottile, Cristian. Agregando polimorfismo a una lógica que identifica proposiciones isomorfas, 2020.

[5] Alejandro Díaz-Caro y Pablo E. Martínez Líopez. Isomorphisms considered as equalities: Projecting functions and enhancing partial application through an implementation of $\lambda+$, 2015.