Cocientando programas por utilidad

Cristian Sottile

Becario doctoral

Alejandro Díaz-Caro

Director

Fidel

Codirector

4to Día de la Investigación en Ciencias de la Computación

18 de marzo de 2022

Introducción

Utilidad de un programa

Introducción

Utilidad de un programa

$$suma : (int \times int) \to int$$

$$suma\ \langle x, y \rangle \stackrel{\text{def}}{=} x + y$$

$$suma\ \langle 1, 2 \rangle \ \to^* \ 3$$

$$suma' : int \to int \to int$$

$$suma'\ x\ y \stackrel{\text{def}}{=} x + y$$

$$suma'\ 1\ 2 \ \to^* \ 3$$

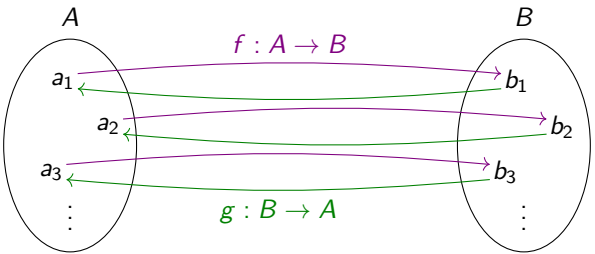
Introducción

Utilidad de un programa

$$suma : (int \times int) \to int$$
$$suma\ \langle x, y\rangle \stackrel{\text{def}}{=} x + y$$
$$suma\ \langle 1, 2\rangle \ \to^*\ 3$$

$$suma' : int \to int \to int$$
$$suma'\ x\ y \stackrel{\text{def}}{=} x + y$$
$$suma'\ 1\ 2 \ \to^*\ 3$$

Teoría de tipos isomorfos



- Tipos isomorfos: misma información, misma utilidad
- Caracterización de tipos isomorfos [DiCosmo '95]

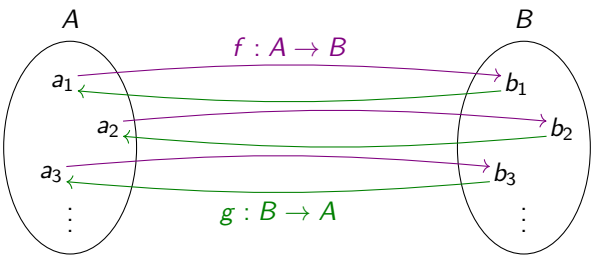
Introducción

Utilidad de un programa

$$suma : (int \times int) \to int \qquad suma' : int \to int \to int$$
$$suma\ \langle x, y\rangle \stackrel{\text{def}}{=} x + y \qquad suma'\ x\ y \stackrel{\text{def}}{=} x + y$$

$suma\ \langle 1, 2\rangle \to^* 3$ $\quad suma\ 1\ 2$ $\quad suma'\ 1\ 2 \to^* 3$ $\quad suma'\ \langle 1, 2\rangle$

Teoría de tipos isomorfos



- Tipos isomorfos: misma información, misma utilidad
- Caracterización de tipos isomorfos [DiCosmo '95]

Lambda cálculo módulo isomorfismos

[Díaz-Caro y Dowek, LSFA '12]

Características

- Equivalencia entre tipos $(A \times B) \to C \equiv A \to B \to C$
- Equivalencia entre términos $f\ \langle a, b\rangle \rightleftarrows f\ a\ b$
- Adaptación de reglas de tipado y reducción $(\lambda x^A.\ t)u \to [x := u]t \quad \text{si } u : A$

Lambda cálculo módulo isomorfismos

[Díaz-Caro y Dowek, LSFA '12]

Características

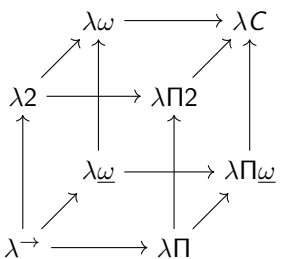
- Equivalencia entre tipos $(A \times B) \to C \equiv A \to B \to C$
- Equivalencia entre términos $f\ \langle a, b\rangle \rightleftarrows f\ a\ b$
- Adaptación de reglas de tipado y reducción $(\lambda x^A.\ t)u \to [x := u]t \quad \text{si } u : A$

Trabajos en esta línea

- Implementación en Haskell [Díaz-Caro y Martínez López, IFL '15]
- Normalización fuerte [Díaz-Caro y Dowek, FSCD '19]
- Extensión a η [Accattoli y Díaz-Caro, FLOPS '20]
- **Extensión a polimorfismo [Sottile, Díaz-Caro y Martínez López, IFL '20]**
- Normalización fuerte para η [Díaz-Caro y Dowek, en revisión]

Objetivo

- Largo plazo: cocientar tipos isomorfos en lenguajes prácticos (Haskell, Ocaml, Coq, Agda)
- Corto-mediano plazo: avanzar cocientando tipos isomorfos en el λ-cubo



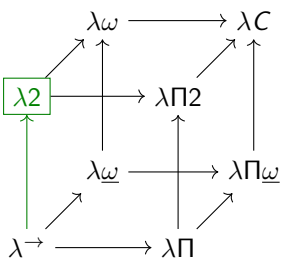
$\uparrow$ Agrega términos dependiendo de tipos (polimorfismo)

$\nearrow$ Agrega tipos dependiendo de tipos (constructores de tipos)

$\rightarrow$ Agrega tipos dependiendo de términos (tipos dependientes)

Objetivo

- Largo plazo: cocientar tipos isomorfos en lenguajes prácticos (Haskell, Ocaml, Coq, Agda)
- Corto-mediano plazo: avanzar cocientando tipos isomorfos en el λ-cubo



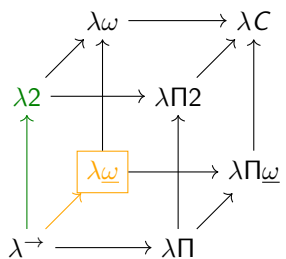
$\uparrow$ Agrega términos dependiendo de tipos (polimorfismo)

$\nearrow$ Agrega tipos dependiendo de tipos (constructores de tipos)

$\rightarrow$ Agrega tipos dependiendo de términos (tipos dependientes)

Objetivo

- Largo plazo: cocientar tipos isomorfos en lenguajes prácticos (Haskell, Ocaml, Coq, Agda)
- Corto-mediano plazo: avanzar cocientando tipos isomorfos en el λ-cubo



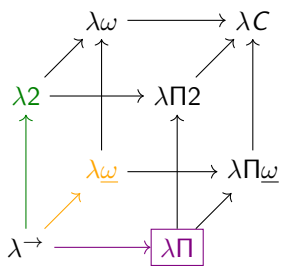
$\uparrow$ Agrega términos dependiendo de tipos (polimorfismo)

$\nearrow$ Agrega tipos dependiendo de tipos (constructores de tipos)

$\rightarrow$ Agrega tipos dependiendo de términos (tipos dependientes)

Objetivo

- Largo plazo: cocientar tipos isomorfos en lenguajes prácticos (Haskell, Ocaml, Coq, Agda)
- Corto-mediano plazo: avanzar cocientando tipos isomorfos en el λ-cubo



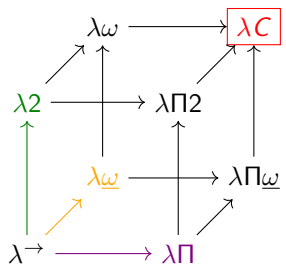
$\uparrow$ Agrega términos dependiendo de tipos (polimorfismo)

$\nearrow$ Agrega tipos dependiendo de tipos (constructores de tipos)

$\rightarrow$ Agrega tipos dependiendo de términos (tipos dependientes)

Objetivo

- Largo plazo: cocientar tipos isomorfos en lenguajes prácticos (Haskell, Ocaml, Coq, Agda)
- Corto-mediano plazo: avanzar cocientando tipos isomorfos en el λ-cubo



$\uparrow$ Agrega términos dependiendo de tipos (polimorfismo)

$\nearrow$ Agrega tipos dependiendo de tipos (constructores de tipos)

$\rightarrow$ Agrega tipos dependiendo de términos (tipos dependientes)

Resumen

- Utilidad de programas
- Teoría de tipos isomorfos
- Cocientar por utilidad