Formalización de la propiedad de Progreso para el $\lambda-$cálculo simplemente tipado

Cristian Sottile

14 de septiembre de 2023

Abstract

La propuesta es formalizar la propiedad de Progreso del $\lambda-$cálculo simplemente tipado. El objetivo del trabajo es comprender en profundidad las maneras de formalizar el $\lambda-$cálculo simplemente tipado y sus propiedades principales. El asistente a usar será Agda por el mismo motivo, comprender en profundidad esa herramienta en particular.

1 La propiedad

En el $\lambda-$cálculo simplemente tipado definimos una relación entre términos que llamamos reducción y expresa la idea de computación. Las formas normales son los términos que "no reducen". Los valores son los términos que se construyen (dependiendo la variante del sistema puede que sea únicamente) con los constructores de introducción de su tipo. Por ejemplo:

- el constructor de introducción de las funciones es λ
- el constructor de eliminación de las funciones es la aplicación (yuxtaposición)
- el constructor de introducción de los pares es $(\_, \_)$
- los constructores de eliminación de los pares son las proyecciones fst y snd
- los constructores de introducción de los números son zero y suc

Lo esperable de un lenguaje de programación es que todo término se corresponda vía reducción con un valor de su tipo. Por ejemplo $(\lambda x.x)$ zero se corresponde con el valor zero. Si nuestro lenguaje no contara con la regla β, el término $(\lambda x.x)$ zero estaría en forma normal sin corresponderse con un valor. Llamamos a estos términos "trabados". La propiedad de Progreso establece que no existen términos trabados en el $\lambda-$cálculo simplemente tipado.

2 La formalización

Tenemos dos maneras de expresar progreso en Agda: mediante una definición inductiva de las pruebas de una proposición, y otra mediante el uso de conectivos lógicos. Abordaremos ambas, las compararemos y probaremos que son equivalentes. Previamente deberemos definir varias funciones y lemas referentes al $\lambda-$cálculo simplemente tipado, así como la propia representación del cálculo.

2.1 Definición inductiva

Definimos la relación Progress, que establece cuándo un término progresa, de la siguiente manera:

```
data Progress (M : Term) : Set where
  step : forall {N} → (M → N) → Progress M
  done :              Value M → Progress M
```

Con el primer constructor indicamos que si hay un término N al que reduzca M, entonces cumple progreso, y con el segundo que si es un valor, entonces cumple progreso.

2.2 Conectivos lógicos

Utilizamos los conectivos lógicos (que previamente definiremos) de disyunción y cuantificación existencial para enunciar el teorema como una proposición que establece que siempre que un término tenga tipo bajo el contexto cerrado, ese término es un valor o bien tiene un reducto.

$$\mathsf{progress} : \forall M, A.\ \emptyset \vdash M : A \to \mathsf{Value}\ M \uplus \exists N (M \longrightarrow N)$$