Inferencia de Tipos

Paradigmas de Lenguajes de Programación

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

18 de septiembre de 2018
A 12 años de la desaparición de Jorge Julio Lopez

Tema del día

Ejemplo

Dada la expresión $\lambda x{:}Nat.\ \text{isZero}(x)$ ¿Qué tipo tiene?

Tema del día

Ejemplo

Dada la expresión $\lambda x{:}Nat.\ \mathrm{isZero}(x)$ ¿Qué tipo tiene?

$$\emptyset \rhd \lambda x{:}Nat.\ \mathrm{isZero}(x) : Nat \to Bool$$

¿Cómo hicimos? ¿Se puede automatizar? ¿Podríamos no escribir el $: Nat$?

Tema del día

Ejemplo

Dada la expresión $\lambda x{:}Nat.\ \mathrm{isZero}(x)$ ¿Qué tipo tiene?

$$\emptyset \rhd \lambda x{:}Nat.\ \mathrm{isZero}(x) : Nat \to Bool$$

¿Cómo hicimos? ¿Se puede automatizar? ¿Podríamos no escribir el $: Nat$? ¿Y para $\lambda x.\ \lambda y.\ (x\ y)\ (\lambda z.\ x)$?

Tema del día

Ejemplo

Dada la expresión $\lambda x{:}Nat.\ \text{isZero}(x)$ ¿Qué tipo tiene?

$$\emptyset \triangleright \lambda x{:}Nat.\ \text{isZero}(x) : Nat \rightarrow Bool$$

¿Cómo hicimos? ¿Se puede automatizar? ¿Podríamos no escribir el $: Nat$? ¿Y para $\lambda x.\ \lambda y.\ (x\ y)\ (\lambda z.\ x)$?

Inferencia

- Dada una expresión, ¿tiene tipo? ¿cuál es este tipo? ¿es el más general?
- Árbol Sintáctico
- Algoritmo de inferencia.
  unificación, sustituciones, variables de tipos, contextos, etc.

Más ejemplos *a ojo*

¿Qué tipo tiene?

- $\lambda x.\ \mathsf{succ}(x)$

Más ejemplos *a ojo*

¿Qué tipo tiene?

- $\lambda x.\ \mathrm{succ}(x) : \mathit{Nat} \to \mathit{Nat}$

Más ejemplos *a ojo*

¿Qué tipo tiene?

- $\lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$

Más ejemplos *a ojo*

¿Qué tipo tiene?

- $\lambda x{:}\mathit{Nat}.\ \mathsf{succ}(x) : \mathit{Nat} \to \mathit{Nat}$
- $\lambda x.\ \mathsf{succ}(y)$

Más ejemplos *a ojo*

¿Qué tipo tiene?

- $\lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\lambda x{:}\Box.\ \mathrm{succ}(y) : \Box \to Nat$

Más ejemplos *a ojo*

¿Qué tipo tiene?

- $\emptyset \rhd \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \rhd \lambda x{:}\square.\ \mathrm{succ}(y) : \square \to Nat$

Más ejemplos *a ojo*

¿Qué tipo tiene? ¿En qué contexto?

- $\emptyset \rhd \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \rhd \lambda x{:}\square.\ \mathrm{succ}(y) : \square \to Nat$

Más ejemplos *a ojo*

¿Qué tipo tiene? ¿En qué contexto?

- $\emptyset \rhd \lambda x{:}Nat.\ \mathsf{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \rhd \lambda x{:}\square.\ \mathsf{succ}(y) : \square \to Nat$
- $(\lambda x.\ \mathsf{isZero}(x))\ true$

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?

- $\emptyset \triangleright \lambda x{:}\mathit{Nat}.\ \mathrm{succ}(x) : \mathit{Nat} \to \mathit{Nat}$
- $\{y{:}\mathit{Nat}\} \triangleright \lambda x{:}\square.\ \mathrm{succ}(y) : \square \to \mathit{Nat}$
- $(\lambda x.\ \mathrm{isZero}(x))\ \mathit{true}$ no tiene tipo.

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?

- $\emptyset \triangleright \lambda x{:}Nat.\ \mathsf{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \triangleright \lambda x{:}\square.\ \mathsf{succ}(y) : \square \to Nat$
- $(\lambda x.\ \mathsf{isZero}(x))\ true$ no tiene tipo.
- $\lambda x.\ x$

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?

- $\emptyset \triangleright \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \rightarrow Nat$
- $\{y{:}Nat\} \triangleright \lambda x{:}\Box.\ \mathrm{succ}(y) : \Box \rightarrow Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ true$ no tiene tipo.
- $\emptyset \triangleright \lambda x : Nat.\ x : Nat \rightarrow Nat$

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?¿Es lo más general posible?

- $\emptyset \triangleright \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \rightarrow Nat$
- $\{y{:}Nat\} \triangleright \lambda x{:}\square.\ \mathrm{succ}(y) : \square \rightarrow Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ true$ no tiene tipo.
- $\emptyset \triangleright \lambda x : Nat.\ x : Nat \rightarrow Nat$ no es lo más general.

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?¿Es lo más general posible?

- $\emptyset \rhd \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \rhd \lambda x{:}t.\ \mathrm{succ}(y) : t \to Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ true$ no tiene tipo.
- $\emptyset \rhd \lambda x : Nat.\ x : Nat \to Nat$ no es lo más general.
- $\emptyset \rhd \lambda x{:}t.\ x : t \to t$ es lo más general.

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?¿Es lo más general posible?

- $\emptyset \rhd \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \rhd \lambda x{:}t.\ \mathrm{succ}(y) : t \to Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ true$ no tiene tipo.
- $\emptyset \rhd \lambda x : Nat.\ x : Nat \to Nat$ no es lo más general.
- $\emptyset \rhd \lambda x{:}t.\ x : t \to t$ es lo más general.
- $\lambda f.\ \lambda x.\ f\ (f\ x)$

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?¿Es lo más general posible?

- $\emptyset \triangleright \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \triangleright \lambda x{:}t.\ \mathrm{succ}(y) : t \to Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ \mathit{true}$ no tiene tipo.
- $\emptyset \triangleright \lambda x : Nat.\ x : Nat \to Nat$ no es lo más general.
- $\emptyset \triangleright \lambda x{:}t.\ x : t \to t$ es lo más general.
- $\emptyset \triangleright \lambda f{:}t \to t.\ \lambda x{:}t.\ f\ (f\ x) : (t \to t) \to t \to t$

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?¿Es lo más general posible?

- $\emptyset \triangleright \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \triangleright \lambda x{:}t.\ \mathrm{succ}(y) : t \to Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ true$ no tiene tipo.
- $\emptyset \triangleright \lambda x : Nat.\ x : Nat \to Nat$ no es lo más general.
- $\emptyset \triangleright \lambda x{:}t.\ x : t \to t$ es lo más general.
- $\emptyset \triangleright \lambda f{:}t \to t.\ \lambda x{:}t.\ f\ (f\ x) : (t \to t) \to t \to t$
- $\lambda x.\ \lambda y.\ f\ y\ x$

Más ejemplos *a ojo*

¿Tiene tipo? ¿Qué tipo tiene? ¿En qué contexto?¿Es lo más general posible?

- $\emptyset \triangleright \lambda x{:}Nat.\ \mathrm{succ}(x) : Nat \to Nat$
- $\{y{:}Nat\} \triangleright \lambda x{:}t.\ \mathrm{succ}(y) : t \to Nat$
- $(\lambda x.\ \mathrm{isZero}(x))\ true$ no tiene tipo.
- $\emptyset \triangleright \lambda x : Nat.\ x : Nat \to Nat$ no es lo más general.
- $\emptyset \triangleright \lambda x{:}t.\ x : t \to t$ es lo más general.
- $\emptyset \triangleright \lambda f{:}t \to t.\ \lambda x{:}t.\ f\ (f\ x) : (t \to t) \to t \to t$
- $\{f{:}t \to s \to u\} \triangleright \lambda x{:}s.\ \lambda y{:}t.\ f\ y\ x : s \to t \to u$

Generalidad

Dijimos que queremos el juicio *más general* ¿Qué significa ser el más general?

Generalidad

Dijimos que queremos el juicio *más general* ¿Qué significa ser el más general?

- $\emptyset \triangleright \lambda x{:}t.\ x : t \rightarrow t$ captura todos los juicios derivables para $\lambda x.\ x$. Por ejemplo, ¿cuáles?

Generalidad

Dijimos que queremos el juicio *más general* ¿Qué significa ser el más general?

- $\emptyset \triangleright \lambda x{:}t.\ x : t \to t$ captura todos los juicios derivables para $\lambda x.\ x$. Por ejemplo, ¿cuáles?
- $\emptyset \triangleright \lambda x{:}s.\ \lambda y{:}t.\ y : s \to t \to t$ captura todos los juicios derivables para $\lambda x.\ \lambda y.\ y$. Por ejemplo, ¿cuáles?

Generalidad

Dijimos que queremos el juicio *más general* ¿Qué significa ser el más general?

- $\emptyset \triangleright \lambda x{:}t.\ x : t \rightarrow t$ captura todos los juicios derivables para $\lambda x.\ x$. Por ejemplo, ¿cuáles?
- $\emptyset \triangleright \lambda x{:}s.\ \lambda y{:}t.\ y : s \rightarrow t \rightarrow t$ captura todos los juicios derivables para $\lambda x.\ \lambda y.\ y$. Por ejemplo, ¿cuáles?

¿Y por qué directamente no decimos $\emptyset \triangleright \lambda x{:}\sigma.\ x : \sigma \rightarrow \sigma$?

Generalidad

Dijimos que queremos el juicio *más general* ¿Qué significa ser el más general?

- $\emptyset \triangleright \lambda x{:}t.\ x : t \to t$ captura todos los juicios derivables para $\lambda x.\ x$. Por ejemplo, ¿cuáles?
- $\emptyset \triangleright \lambda x{:}s.\ \lambda y{:}t.\ y : s \to t \to t$ captura todos los juicios derivables para $\lambda x.\ \lambda y.\ y$. Por ejemplo, ¿cuáles?

¿Y por qué directamente no decimos $\emptyset \triangleright \lambda x{:}\sigma.\ x : \sigma \to \sigma$?
Recordemos que cuando usamos σ, τ, ... NO son variables de tipo, es una manera que usamos en el *metalenguaje* de la materia para hacer referencia a un tipo sin explicitar cuál es.
Las *variables de tipo* son tipos concretos y especiales que agregamos a nuestro conjunto de tipos, que tienen la particularidad de que pueden ser afectados por sustituciones.

Algoritmo de inferencia (W)

Dado un término U sin anotaciones de tipos, hallar un término M tipable y anotado, un contexto Γ y un tipo σ, lo más generales posibles, tales que:

1. $\Gamma \triangleright M : \sigma$
2. $\texttt{Erase}(M) = U$

si U es tipable, o informar que no lo es en caso contrario.
Obs: El resultado del algoritmo, cuando tiene éxito, es

Inferencia

Algoritmo de inferencia (W)

Dado un término U sin anotaciones de tipos, hallar un término M tipable y anotado, un contexto Γ y un tipo σ, lo más generales posibles, tales que:

1. $\Gamma \triangleright M : \sigma$
2. $\texttt{Erase}(M) = U$

si U es tipable, o informar que no lo es en caso contrario.
Obs: El resultado del algoritmo, cuando tiene éxito, es un **juicio** de tipado.
Obs':

Inferencia

Algoritmo de inferencia (W)

Dado un término U sin anotaciones de tipos, hallar un término M tipable y anotado, un contexto Γ y un tipo σ, lo más generales posibles, tales que:

1. $\Gamma \triangleright M : \sigma$
2. $\texttt{Erase}(M) = U$

si U es tipable, o informar que no lo es en caso contrario.
Obs: El resultado del algoritmo, cuando tiene éxito, es un **juicio** de tipado.
Obs': Inferencia NO es chequeo de tipos. ¿Por qué?

Ejercicios

Veamos cómo aplicar el algoritmo (y cómo surge cada componente)

Ejercicios

- $y\ (\lambda x.\ x)\ \mathit{true}$

Ejercicios

Veamos cómo aplicar el algoritmo (y cómo surge cada componente)

Ejercicios

- $y\ (\lambda x.\ x)\ \mathit{true}$
- $\text{if}\ x\ \text{then}\ (\lambda x.\ x)\ \ y\ \text{else}\ \text{succ}(y)$

Ejercicios

Veamos cómo aplicar el algoritmo (y cómo surge cada componente)

Ejercicios

- $y\ (\lambda x.\ x)\ \mathit{true}$
- $\text{if } x \text{ then } (\lambda x.\ x)\ y \text{ else succ}(y)$
- $\lambda x.\ \lambda y.\ (x\ y)\ (\lambda z.\ x)$

Machete

$$\mathbb{W}(\mathit{true}) \quad \xrightarrow{\texttt{ret}} \quad \emptyset \triangleright \mathit{true} : \mathit{Bool}$$

Machete

$$
\begin{aligned}
\mathbb{W}(\mathit{true}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{true} : \mathit{Bool} \\
\mathbb{W}(\mathit{false}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{false} : \mathit{Bool}
\end{aligned}
$$

Machete

$$
\begin{aligned}
\mathbb{W}(\mathit{true}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{true} : \mathit{Bool} \\
\mathbb{W}(\mathit{false}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{false} : \mathit{Bool} \\
\mathbb{W}(\mathit{0}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{0} : \mathit{Nat}
\end{aligned}
$$

Machete

$$\begin{aligned} \mathbb{W}(\mathit{true}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{true} : \mathit{Bool} \\ \mathbb{W}(\mathit{false}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{false} : \mathit{Bool} \\ \mathbb{W}(\mathit{0}) &\xrightarrow{\texttt{ret}} \emptyset \rhd \mathit{0} : \mathit{Nat} \\ \mathbb{W}(x) &\xrightarrow{\texttt{ret}} \{x : s\} \rhd x : s, \ \ s \text{ variable fresca} \end{aligned}$$

Machete

$\mathbb{W}(\mathsf{succ}(U)) \xrightarrow{\texttt{ret}} S\Gamma \triangleright S(\mathsf{succ}(M)) : Nat$ donde

- $\Gamma \triangleright M : \sigma \leftarrow \mathbb{W}(U)$
- $S \leftarrow MGU\{\sigma \doteq Nat\}$

Machete

$\mathbb{W}(\mathrm{succ}(U)) \xrightarrow{\texttt{ret}} S\Gamma \rhd S(\mathrm{succ}(M)) : Nat$ donde

- $\Gamma \rhd M : \sigma \leftarrow \mathbb{W}(U)$
- $S \leftarrow MGU\{\sigma \doteq Nat\}$

$\mathbb{W}(\lambda x.\ U) \xrightarrow{\texttt{ret}} \Gamma' \rhd \lambda x{:}\tau'.\ M : \tau' \to \rho$ donde

- $\Gamma \rhd M : \rho \leftarrow \mathbb{W}(U)$
- $\tau' \leftarrow \begin{cases} \alpha \text{ si } x{:}\alpha \in \Gamma \\ s \text{ variable fresca en otro caso} \end{cases}$
- $\Gamma' \leftarrow \Gamma \ominus \{x\}$

Machete

$\mathbb{W}(\text{succ}(U)) \xrightarrow{\texttt{ret}} S\Gamma \rhd S(\text{succ}(M)) : Nat$ donde

- $\Gamma \rhd M : \sigma \leftarrow \mathbb{W}(U)$
- $S \leftarrow MGU\{\sigma \doteq Nat\}$

$\mathbb{W}(\lambda x.\ U) \xrightarrow{\texttt{ret}} \Gamma' \rhd \lambda x{:}\tau'.\ M : \tau' \to \rho$ donde

- $\Gamma \rhd M : \rho \leftarrow \mathbb{W}(U)$
- $\tau' \leftarrow \begin{cases} \alpha \text{ si } x{:}\alpha \in \Gamma \\ s \text{ variable fresca en otro caso} \end{cases}$
- $\Gamma' \leftarrow \Gamma \ominus \{x\}$

$\mathbb{W}(U\ V) \xrightarrow{\texttt{ret}} S\Gamma_1 \cup S\Gamma_2 \rhd S(M\ N) : St$ donde

- $\Gamma_1 \rhd M : \tau \leftarrow \mathbb{W}(U)$
- $\Gamma_2 \rhd N : \rho \leftarrow \mathbb{W}(V)$
- t variable fresca
- $S \leftarrow MGU\{\tau \doteq \rho \to t\} \cup \{\sigma_1 \doteq \sigma_2 \mid x : \sigma_1 \in \Gamma_1, x : \sigma_2 \in \Gamma_2\}$

Ejercicios

Una modificación

Supongamos que no queremos que sean válidas abstracciones en las cuales la variable ligada no aparezca en el cuerpo. Para eso, modificamos la siguiente regla de tipado:

$$\frac{\Gamma \cup \{x : \sigma\} \rhd M : \tau \quad x \in FV(M)}{\Gamma \rhd \lambda x : \sigma.\ M : \sigma \to \tau} \text{ T-Abs}$$

¿Cómo debe modificarse el algoritmo de inferencia para incorporar esta variante?

Método del Árbol: paso por paso

Método del Árbol: paso por paso

1. Convencerse de que tipa (...o de que no tipa).

Método del Árbol: paso por paso

1. Convencerse de que tipa (...o de que no tipa).
2. Construir el árbol de análisis sintáctico.

Método del Árbol: paso por paso

1. Convencerse de que tipa (...o de que no tipa).
2. Construir el árbol de análisis sintáctico.
3. Aplicar las reglas sobre las hojas, indicando qué reglas, unificaciones y sustituciones se aplican en cada paso.

Método del Árbol: paso por paso

1. Convencerse de que tipa (...o de que no tipa).
2. Construir el árbol de análisis sintáctico.
3. Aplicar las reglas sobre las hojas, indicando qué reglas, unificaciones y sustituciones se aplican en cada paso.
4. Hacer lo mismo para los nodos internos a medida que sus hijos queden resueltos.
5. Si se pudo tipar la raíz, listo. Si no, indicar qué fue lo que falló.

Fin de la clase

¿? ¿? ¿? ¿? ¿? ¿? ¿? ¿? ¿?