Typage

Coinduction

2018-2019
Giovanni Bernardi, gioXYZirif.fr
http://www.irif.fr/~gio/index.xhtml
Université Paris Diderot

Plan

1. Questions
2. Mini historical remarks
3. More fixed points
4. Deciding type equivalence
5. A type system with recursive types

Questions questions questions . . .

1. In which year was the first paper on types published ?

Questions questions questions . . .

1. In which year was the first paper on types published ?
2. What does Kleene fixed point theorem state ?

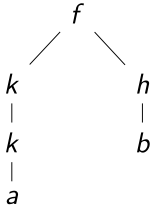
Questions questions questions . . .

1. In which year was the first paper on types published ?
2. What does Kleene fixed point theorem state ?
3. What is a tree ?

Questions questions questions . . .

$$\Sigma = \{a, b, f, k, h\}$$

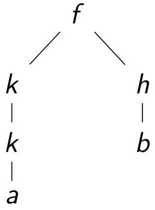
$$\begin{aligned} s(\varepsilon) &= f \\ s(1) = s(11) &= k \quad & s(111) = a \\ s(2) &= h & s(21) = b \end{aligned}$$



Questions questions questions . . .

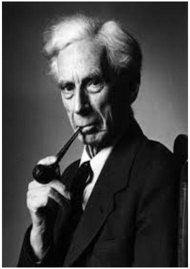
$$\Sigma = \{a, b, f, k, h\}$$

$$
\begin{array}{ll}
s(\varepsilon) = f & \\
s(1) = s(11) = k & s(111) = a \\
s(2) = h & s(21) = b
\end{array}
$$



what about the arities?

1908, Russell



A *type* is defined as the range of significance of a propositional function, *i.e.* as the collection of arguments for which that said function has values.

1968, Morris

[. . .] types and type declarations are often described as communications to a compiler to aid it in allocating storage, etc.

What was the problem again?

$$A \;::=\; \mathcal{T} \mid \underline{x} \mid \underline{\mu x.A} \mid A \times A \mid A \to A$$

- $\mu x.T$ binds x in T, free and bound variables as expected
- μ-types are closed and <u>contractive</u> terms

when are two types equal ?

$$\mu y.y \stackrel{?}{=} \mu x.z$$

$$\mu y.y \stackrel{?}{=} \mu x.x$$

$$\mu x.(int \times x) \stackrel{?}{=} int \times \mu x.(int \times x)$$

$$\mu x.x \to x \stackrel{?}{=} (\mu x.x \to x) \to (\mu x.x \to x)$$

What was the problem again?

$$A \ ::= \ \mathcal{T} \mid \underline{x} \mid \underline{\mu x.A} \mid A \times A \mid A \to A$$

- $\mu x.T$ binds x in T, free and bound variables as expected
- μ-types are closed and <u>contractive</u> terms

A *<u>contractive</u>* if for any subexpression of A of the form

$$\mu x.\mu x_1.\mu x_2.\dots\mu x_n.B$$

the term B is not x.

- not contractive: $\mu x.x$
- contractive: $\mu x.y$ — but not closed
- not contractive: $int \to \mu x.x$
- contractive: $\mu x.x \to x$

Type equivalence semantic approach

$\Sigma = \mathcal{T} \cup \{\times, \rightarrow\}$

$$
\begin{array}{lcll}
\mathit{treeof}(c)(\varepsilon) & = & c & \text{where } c \in \mathcal{T} \\
\mathit{treeof}(A_1 \rightarrow A_2)(\varepsilon) & = & \rightarrow & \\
\mathit{treeof}(A_1 \rightarrow A_2)(i\pi) & = & \mathit{treeof}(A_i)(\pi) & \\
\vdots & & & \\
\mathit{treeof}(\mu x.A)(\pi) & = & \mathit{treeof}(A\{x/\mu x.A\})(\pi) &
\end{array}
$$

Lemma

For every μ-type A the $\mathit{treeof}(A)$ is defined. **Why ?** □

Let $A \stackrel{ext}{=} B$ whenever $\mathit{treeof}(A) \stackrel{ext}{=} \mathit{treeof}(B)$

How to decide $\stackrel{ext}{=}$?

More on fixed points

Theorem (Knaster 1928 - Tarski 1955)

If $\langle L, \leq \rangle$ *complete lattice,* $f : L \to L$ *monotone function then*

- $\mu f = \bigsqcap \{ x \mid f(x) \leq x \}$
- $\nu f = \bigsqcup \{ x \mid x \leq f(x) \}$ $\square$

More on fixed points

A poset $\langle L, \leq \rangle$ is a *complete lattice* if

- $L \neq \emptyset$, and
- for every $S \in parts(L)$. $\bigsqcup S$ and $\bigsqcap S$ exist

Lemma

Every complete lattice is a CPO.

Theorem (Knaster 1928 - Tarski 1955)

If $\langle L, \leq \rangle$ complete lattice, $f : L \to L$ monotone function then

- $\mu f = \bigsqcap \{ x \mid f(x) \leq x \}$
- $\nu f = \bigsqcup \{ x \mid x \leq f(x) \}$ □

More on fixed points

A poset $\langle L, \leq \rangle$ is a *complete lattice* if

- $L \neq \emptyset$, and
- for every $S \in parts(L)$. $\bigsqcup S$ and $\bigsqcap S$ exist

Lemma
Every complete lattice is a CPO.

Theorem (Knaster 1928 - Tarski 1955)

If $\langle L, \leq \rangle$ complete lattice, $f : L \to L$ monotone function then

- $\mu f = \bigsqcap \{ x \mid f(x) \leq x \}$
- $\nu f = \bigsqcup \{ x \mid x \leq f(x) \}$ *coinduction* □

Type equivalence syntactic approach

$$F \quad : \quad \mathit{parts}(\text{Types}^2_\mu) \to \mathit{parts}(\text{Types}^2_\mu)$$

$$\begin{aligned} F(\mathcal{R}) \triangleq\ & \{\, (c,c) \mid c \in \mathcal{T}\,\} \\ & \cup \{\, (A_1 \times A_2, B_1 \times B_2) \mid \forall i \in \{1,2\}.\, A_i \mathrel{\mathcal{R}} B_i \,\} \\ & \cup \{\, (A_1 \to A_2, B_1 \to B_2) \mid B_1 \mathrel{\mathcal{R}} A_1, A_2 \mathrel{\mathcal{R}} B_2 \,\} \\ & \cup \{\, (A, \mu x.B) \mid A \mathrel{\mathcal{R}} B\{x/\mu x.B\} \,\} \\ & \cup \{\, (\mu x.A, B) \mid A\{x/\mu x.A\} \mathrel{\mathcal{R}} B \,\} \end{aligned}$$

- $\langle \mathit{parts}(\text{Types}^2_\mu), \subseteq \rangle$ complete lattice, F monotone
- νF exists by Knaster-Tarski
- Let

$$\begin{aligned} \leq: \ &\triangleq\ \nu F \\ \approx \ &\triangleq\ \leq: \cap \leq:^{-1} \end{aligned}$$

Type equivalence

Syntactic definition justified by semantic one

$$\approx \, = \, \stackrel{ext}{=}$$

How to show $A \approx B$? Show $A <: B$ and $B <: A$ no brainer

Coinductive proof method

How to show $A <: B$?

1. By definition $<: \, = \nu F$
2. By Knaster-Tarski $<: \, = \bigcup \{ \mathcal{R} \mid \mathcal{R} \subseteq F(\mathcal{R}) \}$
3. It suffices to define relation $\mathcal{R}$ such that

$$A \, \mathcal{R} \, B, \qquad \mathcal{R} \subseteq F(\mathcal{R})$$

Example

Let $A = \mu x.x \to x$, why $A \approx A \to A$?

Let

$$\mathcal{R} \;=\; \{(A, A \to A) \qquad\qquad\qquad \}$$

1. By definition $A \,\mathcal{R}\, A \to A$
2. Routine work shows that $\mathcal{R} \subseteq F(\mathcal{R})$ and $\mathcal{R}^{-1} \subseteq F(\mathcal{R}^{-1})$,

$$A <: A \to A, \quad A \to A <: A$$

Example

Let $A = \mu x.x \to x$, why $A \approx A \to A$?

Let

$$\mathcal{R} \;=\; \{(A, A \to A), (A \to A, A \to A), \\ \}$$

1. By definition $A \, \mathcal{R} \, A \to A$
2. Routine work shows that $\mathcal{R} \subseteq F(\mathcal{R})$ and $\mathcal{R}^{-1} \subseteq F(\mathcal{R}^{-1})$,

$$A <: A \to A, \quad A \to A <: A$$

Example

Let $A = \mu x.x \to x$, why $A \approx A \to A$?

Let

$$\mathcal{R} \;=\; \{(A, A \to A), (A \to A, A \to A), \\ (A, A)\}$$

1. By definition $A \,\mathcal{R}\, A \to A$
2. Routine work shows that $\mathcal{R} \subseteq F(\mathcal{R})$ and $\mathcal{R}^{-1} \subseteq F(\mathcal{R}^{-1})$,

$$A <: A \to A, \quad A \to A <: A$$

Example

Let $A = \mu x.x \to x$, why $A \approx A \to A$?

Let

$$\mathcal{R} \;=\; \{(A, A \to A), (A \to A, A \to A), \\ (A \to A, A), (A, A)\}$$

1. By definition $A \,\mathcal{R}\, A \to A$
2. Routine work shows that $\mathcal{R} \subseteq F(\mathcal{R})$ and $\mathcal{R}^{-1} \subseteq F(\mathcal{R}^{-1})$,

$$A <: A \to A, \quad A \to A <: A$$

Example

Let $A = \mu x.x \to x$, why $A \approx A \to A$?

Let

$$\mathcal{R} \;=\; \{(A, A \to A), (A \to A, A \to A), \\ (A \to A, A), (A, A)\}$$

1. By definition $A \, \mathcal{R} \, A \to A$
2. Routine work shows that $\mathcal{R} \subseteq F(\mathcal{R})$ and $\mathcal{R}^{-1} \subseteq F(\mathcal{R}^{-1})$,

$$A <: A \to A, \quad A \to A <: A$$

Write a decision procedure for $\approx$

λ-calculus

typing rules from [Cardone and Coppo, 1991]

$$M, N ::= x \mid c \mid MN \mid \lambda x.M$$

An equi-recursive system

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{}{\Gamma, c : A \vdash c : A_c(c)}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

$$\frac{\Gamma \vdash M : B}{\Gamma \vdash M : A}\ A \approx B$$

Example

Let $A = \mu x.((x \to x) \to x)$

$$
\dfrac{
  \dfrac{
    \dfrac{}{x : A \to A \vdash x : A \to A}
    \qquad
    \dfrac{\dfrac{}{x : A \to A \vdash x : A \to A}}{x : A \to A \vdash x : A}\;(\approx)
  }{
    \dfrac{x : A \to A \vdash xx : (A \to A) \to A}{x : A \to A \vdash xx : A}\;(\approx)
  }
}{\vdash \lambda x.xx : (A \to A) \to A}
$$

$$
\dfrac{
  \dfrac{
    \dfrac{
      \dfrac{\dfrac{}{x : A \vdash x : A}}{x : A \vdash x : A \to ((A \to A) \to A)}\;(\approx)
      \qquad
      \dfrac{}{x : A \vdash x : A}\;(\approx)
    }{x : A \vdash xx : (A \to A) \to A}
  }{\vdash \lambda x.xx : A \to ((A \to A) \to A)}
  \qquad
  \dfrac{\vdots}{\vdash \lambda x.xx : A}
}{\vdash (\lambda x.xx)(\lambda x.xx) : (A \to A) \to A}
$$

λ-calculus

typing rules from [Cardone and Coppo, 1991]

An equi-recursive system

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{}{\Gamma, c : A \vdash c : A_c(c)}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

$$\frac{\Gamma \vdash M : B}{\Gamma \vdash M : A}\ A \approx B$$

- Strong Normalisation is false! $\vdash (\lambda x.(xx))(\lambda x.(xx))$

Implement

- *treeof*
- decision procedure for $\approx$
- Work on the project