Constraint based Termination

Frédéric Blanqui[1] Colin Riba[2]

[1]INRIA & LORIA

[2]INPL & LORIA

TYPES 2006 University of Nottingham

Outline

Outline

- Simply typed λ-calculus with let (λlet).
- Constants defined via rewrite rules

$$f\vec{l} \quad \longrightarrow \quad r \qquad \text{with} \qquad \vec{l} \quad \text{algebraic}$$

- Rewrite relation

$$f\vec{l}\sigma \qquad \longrightarrow \qquad r\sigma$$

- Sufficient condition for strong normalization of well-typed terms.

Example

$$\begin{array}{rcl} \mathsf{minus}, \mathsf{div} & : & \mathsf{N} \Rightarrow \mathsf{N} \Rightarrow \mathsf{N} \\ \mathsf{s} & : & \mathsf{N} \Rightarrow \mathsf{N} \end{array}$$

$$\mathsf{div}\,(\mathsf{s}\,x)\,y \quad \rightarrow \quad \mathsf{s}(\mathsf{div}\,(\mathsf{minus}\,x\,y)\,y)$$

$$\mathsf{s}\,x \quad \overset{?}{>} \quad \mathsf{minus}\,x\,y$$

Outline

Example

$\mathsf{N}(\alpha)$: terms headed by less than n constructor symbols s.

$$\begin{array}{lcl} \mathsf{minus}, \mathsf{div} & : & \mathsf{N}(\alpha) \Rightarrow \mathsf{N} \Rightarrow \mathsf{N}(\alpha) \\ \mathsf{s} & : & \mathsf{N}(\alpha) \Rightarrow \mathsf{N}(\alpha + 1) \end{array}$$

$$\mathsf{div}\,(\mathsf{s}x)\,y \quad \rightarrow \quad \mathsf{s}(\mathsf{div}\,(\mathsf{minus}\,x\,y)\,y)$$

$$x : \mathsf{N}(\alpha) \quad \vdash \quad \mathsf{s}x : \mathsf{N}(\alpha + 1) \quad , \quad \mathsf{minus}\,x\,y : \mathsf{N}(\alpha)$$

$$\alpha + 1 \quad > \quad \alpha$$

Example

$\mathsf{N}(\alpha)$: terms headed by less than n constructor symbols s.

$$\begin{array}{lcl} \mathsf{minus}, \mathsf{div} & : & \mathsf{N}(\alpha) \Rightarrow \mathsf{N} \Rightarrow \mathsf{N}(\alpha) \\ \mathsf{s} & : & \mathsf{N}(\alpha) \Rightarrow \mathsf{N}(\alpha + 1) \end{array}$$

$$\mathsf{div}\,(\mathsf{s}x)\,y \quad \rightarrow \quad \mathsf{s}(\mathsf{div}\,(\mathsf{minus}\,x\,y)\,y)$$

$$x : \mathsf{N}(\alpha) \quad \vdash \quad \mathsf{s}x : \mathsf{N}(\alpha + 1) \quad , \quad \mathsf{minus}\,x\,y : \mathsf{N}(\alpha)$$

$$\alpha + 1 \quad > \quad \alpha$$

Previous work

- Hughes, Pareto & Sabry, 1996
- Giménez 1998
- Amadio & Coupet-Grimal, 1998
- Abel, 2002
- Barthe, Frade, Giménez, Pinto & Uustalu 2000-2004
- Blanqui 2004,2005

Outline

$$
\begin{array}{lcl}
\mathsf{qsort}\, l & \longrightarrow & \mathsf{qsapp}\, l\, [\,] \\
 & & \\
\mathsf{qsapp}\, [\,]\, l' & \longrightarrow & l' \\
\mathsf{qsapp}\, (x :: l)\, l' & \longrightarrow & \mathsf{let} \quad y = \mathsf{pivot}\, x\, l \\
 & & \mathsf{in} \quad \mathsf{qsapp}\, (\pi_1\, y)\, (x :: \mathsf{qsapp}\, (\pi_2\, y)\, l')
\end{array}
$$

$$
\begin{array}{rcl}
\mathsf{qsort} & : & \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\alpha) \\
\mathsf{qsapp} & : & \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\beta) \Rightarrow \mathsf{L}(\alpha + \beta) \\
 & & \\
\mathsf{pivot} & : & \mathsf{N} \Rightarrow \mathsf{L}(?) \Rightarrow \mathsf{L}(?) \times \mathsf{L}(?)
\end{array}
$$

$$
\begin{array}{lcl}
\mathsf{qsort}\, l & \longrightarrow & \mathsf{qsapp}\, l\, [\,] \\
\\
\mathsf{qsapp}\, [\,]\, l' & \longrightarrow & l' \\
\mathsf{qsapp}\, (x :: l)\, l' & \longrightarrow & \mathsf{let} \quad y = \mathsf{pivot}\, x\, l \\
 & & \mathsf{in} \quad \mathsf{qsapp}\, (\pi_1\, y)\, (x :: \mathsf{qsapp}\, (\pi_2\, y)\, l')
\end{array}
$$

$$
\begin{array}{rcl}
\mathsf{qsort} & : & \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\alpha) \\
\mathsf{qsapp} & : & \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\beta) \Rightarrow \mathsf{L}(\alpha + \beta) \\
\\
\mathsf{pivot} & : & \mathsf{N} \Rightarrow \mathsf{L}(?) \Rightarrow \mathsf{L}(?) \times \mathsf{L}(?)
\end{array}
$$

$$\begin{array}{lll} \mathsf{qsapp}\,(x :: l)\,l' & \rightarrow & \mathsf{let} \quad y = \mathsf{pivot}\,x\,l \\ & & \mathsf{in} \quad \mathsf{qsapp}\,(\pi_1\,y)\,(x :: \mathsf{qsapp}\,(\pi_2\,y)\,l') \end{array}$$

$$\begin{array}{llll} \text{If} & \mathsf{pivot} & : & \mathsf{N} \Rightarrow \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\alpha) \times \mathsf{L}(\alpha) \\ \text{then} & \mathsf{qsort} & : & \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\infty) \end{array}$$

We would like

$$\begin{array}{lll} \mathsf{pivot} & : & \mathsf{N} \Rightarrow \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\gamma) \times \mathsf{L}(\delta) \\ & & \text{with } \alpha = \gamma + \delta \end{array}$$

Constrained types

$$\begin{array}{lll} & & \mathsf{L}(\alpha) \quad : \quad \text{lists of length } \alpha \\ \mathsf{pivot} & : & \mathsf{N} \Rightarrow \forall\alpha\,\mathsf{L}(\alpha) \Rightarrow \exists\gamma\exists\delta\,(\alpha = \gamma + \delta)\,\mathsf{L}(\gamma) \times \mathsf{L}(\delta) \end{array}$$

$$\begin{array}{lll} \mathsf{qsapp}\,(x :: l)\,l' & \rightarrow & \mathsf{let} \quad y = \mathsf{pivot}\,x\,l \\ & & \mathsf{in} \quad \mathsf{qsapp}\,(\pi_1\,y)\,(x :: \mathsf{qsapp}\,(\pi_2\,y)\,l') \end{array}$$

$$\begin{array}{llll} \text{If} & \mathsf{pivot} & : & \mathsf{N} \Rightarrow \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\alpha) \times \mathsf{L}(\alpha) \\ \text{then} & \mathsf{qsort} & : & \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\infty) \end{array}$$

We would like

$$\begin{array}{lll} \mathsf{pivot} & : & \mathsf{N} \Rightarrow \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\gamma) \times \mathsf{L}(\delta) \\ & & \text{with } \alpha = \gamma + \delta \end{array}$$

Constrained types

$$\begin{array}{lll} & & \mathsf{L}(\alpha) \quad : \quad \text{lists of length } \alpha \\ \mathsf{pivot} & : & \mathsf{N} \Rightarrow \forall\alpha\,\mathsf{L}(\alpha) \Rightarrow \exists\gamma\exists\delta\,(\alpha = \gamma + \delta)\,\mathsf{L}(\gamma) \times \mathsf{L}(\delta) \end{array}$$

$$\text{qsapp}\,(x :: l)\,l' \;\rightarrow\; \text{let}\quad y = \text{pivot}\,x\,l \quad \text{in}\quad \text{qsapp}\,(\pi_1\,y)\,(x :: \text{qsapp}\,(\pi_2\,y)\,l')$$

If pivot : $\mathsf{N} \Rightarrow \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\alpha) \times \mathsf{L}(\alpha)$

then qsort : $\mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\infty)$

We would like

pivot : $\mathsf{N} \Rightarrow \mathsf{L}(\alpha) \Rightarrow \mathsf{L}(\gamma) \times \mathsf{L}(\delta)$
with $\alpha = \gamma + \delta$

Constrained types

$\mathsf{L}(\alpha)$: lists of length α

pivot : $\mathsf{N} \Rightarrow \forall\alpha\,\mathsf{L}(\alpha) \Rightarrow \exists\gamma\exists\delta\,(\alpha = \gamma + \delta)\,\mathsf{L}(\gamma) \times \mathsf{L}(\delta)$

- Xi's Dependent ML 1998 - 2002 ($ML^{\Pi,\Sigma}$).
- Types constructors : product types and sum types. Reflected at the term level: $\lambda\alpha.t$, $\langle\alpha, t\rangle$ (with appropriate destructors)
- Two-level approach : ML, $ML^{\Pi,\Sigma}$
  - Elaboration process : $ML \longrightarrow ML^{\Pi,\Sigma}$.
  - Normalization proved in $ML^{\Pi,\Sigma}$.
  - Erasure : $ML^{\Pi,\Sigma} \longrightarrow ML$ that preserves normalization.

- Xi's Dependent ML 1998 - 2002 ($ML^{\Pi,\Sigma}$).
- Types constructors : product types and sum types. Reflected at the term level: $\lambda\alpha.t$, $\langle\alpha, t\rangle$ (with appropriate destructors)
- Two-level approach : ML, $ML^{\Pi,\Sigma}$
  - Elaboration process : $ML \longrightarrow ML^{\Pi,\Sigma}$.
  - Normalization proved in $ML^{\Pi,\Sigma}$.
  - Erasure : $ML^{\Pi,\Sigma} \longrightarrow ML$ that preserves normalization.

Contributions

- We present a type system $\lambda\mathcal{C}$ with constrained simple types.
  We use it in a criterion for Strong Normalization of rewriting (plus β-reduction).
- An alternative to Xi's type system: intersection and union instead of type constructors (product, sum)
- The systems λlet and $\lambda\mathcal{C}$ use the same terms.
- If rules are typable in $\lambda\mathcal{C}$ with some constrains on the type of constants symbols, then every term typable in λlet is $\mathcal{SN}$.

Contributions

- We present a type system $\lambda\mathcal{C}$ with constrained simple types.
  We use it in a criterion for Strong Normalization of rewriting (plus β-reduction).
- An alternative to Xi's type system: intersection and union instead of type constructors (product, sum)
- The systems λlet and $\lambda\mathcal{C}$ use the same terms.
- If rules are typable in $\lambda\mathcal{C}$ with some constrains on the type of constants symbols, then every term typable in λlet is $\mathcal{SN}$.

Contributions

- We present a type system $\lambda\mathcal{C}$ with constrained simple types.
  We use it in a criterion for Strong Normalization of rewriting (plus β-reduction).
- An alternative to Xi's type system: intersection and union instead of type constructors (product, sum)
- The systems λlet and $\lambda\mathcal{C}$ use the same terms.
- If rules are typable in $\lambda\mathcal{C}$ with some constrains on the type of constants symbols, then every term typable in λlet is $\mathcal{SN}$.

Outline

Constraints

- First order language with logical connectives $\wedge$, $\supset$, $\forall$, $\exists$.
- Notations
  - $\mathcal{V}$: variables
  - P, Q : constraints
  - C : conjunction of constraints
- Satisfiability in $\mathbb{N}$,
- $C \vdash P$ iff for all $\mu : \mathcal{V} \longrightarrow \mathbb{N}$,

$$\mu \models C \supset P \ .$$

- Types

$$T, U \in \mathcal{T} ::= \mathsf{B}(\alpha) \mid T \Rightarrow U \mid T \times U \mid (\forall \alpha P)T \mid (\exists \alpha P)T$$

- $\mathsf{B} \quad =_{\mathsf{def}} \quad \exists \alpha \mathsf{B}(\alpha).$
- Inductive types: B with comes constructors

$$\mathsf{c} \quad : \quad \vec{T} \quad \Rightarrow \quad \vec{\mathsf{B}}(\vec{\alpha}) \quad \Rightarrow \quad \mathsf{B}(max(\vec{\alpha}) + 1)$$

where $\mathsf{B} \notin \vec{T}$

- Terms

$$t, u \in \Lambda \ ::= \ x \ \mid \ \mathsf{f} \ \mid \ \mathsf{c} \ \mid \ \lambda x.t \ \mid \ t\,u \ \mid \ \text{let } x = t \text{ in } u$$

- Rewrite rules

$$\mathsf{f}\vec{l} \quad \rightarrow \quad r \qquad \text{with} \qquad \vec{l} \qquad \text{algebraic constructor terms}$$

- Reductions

Rewriting	$\mathsf{f}\vec{l}\sigma$	$\rightarrow$	$r\sigma$
β-reduction	$(\lambda x.t)u$	$\rightarrow$	$t[u/x]$
let -reduction	$\text{let } x = t \text{ in } u$	$\rightarrow$	$u[t/x]$

Outline

$$(\forall I)\quad \frac{C \wedge P \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T \qquad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P)T}$$

$$(\forall E)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P)T \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha]}$$

$$(\exists I)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P)T}$$

$$(\exists E)\frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P)T \quad C \wedge P \;;\; \Gamma, x : T \vdash_{\lambda\mathcal{C}} u : U \quad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} \text{let } x = t \text{ in } u : U}$$

$$(\forall I)\quad \frac{C \wedge P \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T \qquad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P) T}$$

$$(\forall E)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P) T \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha]}$$

$$(\exists I)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P) T}$$

$$(\exists E)\frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P) T \quad C \wedge P \;;\; \Gamma, x : T \vdash_{\lambda\mathcal{C}} u : U \quad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} \mathsf{let}\; x = t \;\mathsf{in}\; u : U}$$

$$(\forall I)\quad \frac{C \wedge P \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T \qquad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P)T}$$

$$(\forall E)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P)T \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha]}$$

$$(\exists I)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P)T}$$

$$(\exists E)\frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P)T \quad C \wedge P \;;\; \Gamma, x : T \vdash_{\lambda\mathcal{C}} u : U \quad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} \text{let } x = t \text{ in } u : U}$$

$$(\forall I)\quad \frac{C \wedge P \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T \qquad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P) T}$$

$$(\forall E)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\forall \alpha P) T \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha]}$$

$$(\exists I)\quad \frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P) T}$$

$$(\exists E)\frac{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : (\exists \alpha P) T \quad C \wedge P \;;\; \Gamma, x : T \vdash_{\lambda\mathcal{C}} u : U \quad C \vdash \exists \alpha P}{C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} \mathsf{let}\; x = t \;\mathsf{in}\; u : U}$$

Outline

Subtyping

Subtyping relation $C \vdash T \leq U$ defined via constraints $(\!|T \leq U|\!)$:

$$C \vdash T \leq U \qquad \text{iff} \qquad C \supset (\!|T \leq U|\!)$$

$(\!| T \leq U |\!)$

$$(\!| \mathsf{B}(\mathfrak{a}) \leq \mathsf{B}(\mathfrak{b}) |\!) \quad =_{\text{def}} \quad (\mathfrak{a} = \mathfrak{b})$$

$$(\!| T \leq (\exists \alpha P) U |\!) \quad =_{\text{def}} \quad \exists \alpha \, [\, P \wedge (\!| T \leq U |\!) \,]$$

$$(\!| (\exists \alpha P) T \leq U |\!) \quad =_{\text{def}} \quad \exists \alpha P \quad \wedge \quad \forall \alpha \, [\, P \supset (\!| T \leq U |\!) \,]$$

$$(\!| T \leq (\forall \alpha P) U |\!) \quad =_{\text{def}} \quad \exists \alpha P \quad \wedge \quad \forall \alpha \, [\, P \supset (\!| T \leq U |\!) \,]$$

$$(\!| (\forall \alpha P) T \leq U |\!) \quad =_{\text{def}} \quad \exists \alpha \, [\, P \wedge (\!| T \leq U |\!) \,]$$

$(\!| T \leq U |\!)$

$$(\!| \mathsf{B}(\mathfrak{a}) \leq \mathsf{B}(\mathfrak{b}) |\!) \quad =_{\mathsf{def}} \quad (\mathfrak{a} = \mathfrak{b})$$

$$(\!| T \leq (\exists \alpha P) U |\!) \quad =_{\mathsf{def}} \quad \exists \alpha \, [\, P \ \wedge \ (\!| T \leq U |\!) \,]$$

$$(\!| (\exists \alpha P) T \leq U |\!) \quad =_{\mathsf{def}} \quad \exists \alpha P \quad \wedge \quad \forall \alpha \, [\, P \ \supset \ (\!| T \leq U |\!) \,]$$

$$(\!| T \leq (\forall \alpha P) U |\!) \quad =_{\mathsf{def}} \quad \exists \alpha P \quad \wedge \quad \forall \alpha \, [\, P \ \supset \ (\!| T \leq U |\!) \,]$$

$$(\!| (\forall \alpha P) T \leq U |\!) \quad =_{\mathsf{def}} \quad \exists \alpha \, [\, P \ \wedge \ (\!| T \leq U |\!) \,]$$

$(\!| T \leq U |\!)$

$$(\!| \mathsf{B}(\mathfrak{a}) \leq \mathsf{B}(\mathfrak{b}) |\!) \quad =_{\text{def}} \quad (\mathfrak{a} = \mathfrak{b})$$

$$(\!| T \leq (\exists \alpha P) U |\!) \quad =_{\text{def}} \quad \exists \alpha \, [\, P \, \wedge \, (\!| T \leq U |\!) \,]$$

$$(\!| (\exists \alpha P) T \leq U |\!) \quad =_{\text{def}} \quad \exists \alpha P \quad \wedge \quad \forall \alpha \, [\, P \, \supset \, (\!| T \leq U |\!) \,]$$

$$(\!| T \leq (\forall \alpha P) U |\!) \quad =_{\text{def}} \quad \exists \alpha P \quad \wedge \quad \forall \alpha \, [\, P \, \supset \, (\!| T \leq U |\!) \,]$$

$$(\!| (\forall \alpha P) T \leq U |\!) \quad =_{\text{def}} \quad \exists \alpha \, [\, P \, \wedge \, (\!| T \leq U |\!) \,]$$

Outline

- No type annotation under abstractions. (Because we want terms of $\lambda\mathcal{C}$ be terms of λ).
- Incomplete bidirectional type checking algorithm.
- Constraints generation

  $C \; ; \; \Gamma \vdash t \uparrow T$
  given Γ and t, generates C and T such that $C \; ; \; \Gamma \vdash_{\lambda\mathcal{C}} t : T$.

  $C \; ; \; \Gamma \vdash t \downarrow T$
  given Γ, t and T, generates C such that $C \; ; \; \Gamma \vdash_{\lambda\mathcal{C}} t : T$.

- Constraints in Presburger arithmetic.

- No type annotation under abstractions. (Because we want terms of $\lambda\mathcal{C}$ be terms of λ).
- Incomplete bidirectional type checking algorithm.
- Constraints generation

  $C \;;\; \Gamma \vdash t \uparrow T$
  given Γ and t, generates C and T such that $C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T$.

  $C \;;\; \Gamma \vdash t \downarrow T$
  given Γ, t and T, generates C such that $C \;;\; \Gamma \vdash_{\lambda\mathcal{C}} t : T$.
- Constraints in Presburger arithmetic.

Outline

- Types interpreted as candidates of reducibility:

  if $T \in \mathcal{T}$ and $\mu : \mathcal{V} \longrightarrow \mathbb{N}$ then $[\![T]\!]\mu \in \mathcal{CR}$

- Soundness :

  if $C\ ;\ \Gamma \vdash_{\lambda\mathcal{C}} t : T$ and $\mu, \theta \models C\ ,\ \Gamma$ then $t\theta \in [\![T]\!]\mu$

- We let

$$[\![(\forall \alpha P)T]\!]\mu =_{\text{def}} \bigcap\{\ [\![T]\!]\mu[\mathfrak{a}/\alpha]\ \ ;\ \ \mu[\mathfrak{a}/\alpha] \models P\ \}$$

$$[\![(\exists \alpha P)T]\!]\mu =_{\text{def}} \bigcup\{\ [\![T]\!]\mu[\mathfrak{a}/\alpha]\ \ ;\ \ \mu[\mathfrak{a}/\alpha] \models P\ \}$$

- Types interpreted as candidates of reducibility:

$$\text{if} \quad T \in \mathcal{T} \quad \text{and} \quad \mu : \mathcal{V} \longrightarrow \mathbb{N} \quad \text{then} \quad [\![T]\!]\mu \in \mathcal{CR}$$

- Soundness :

$$\text{if} \quad C \,;\, \Gamma \vdash_{\lambda\mathcal{C}} t : T \quad \text{and} \quad \mu, \theta \models C \,,\, \Gamma \quad \text{then} \quad t\theta \in [\![T]\!]\mu$$

- We let

$$[\![(\forall \alpha P)T]\!]\mu \quad =_{\text{def}} \quad \bigcap\{\, [\![T]\!]\mu[\mathfrak{a}/\alpha] \quad ; \quad \mu[\mathfrak{a}/\alpha] \models P \,\}$$

$$[\![(\exists \alpha P)T]\!]\mu \quad =_{\text{def}} \quad \bigcup\{\, [\![T]\!]\mu[\mathfrak{a}/\alpha] \quad ; \quad \mu[\mathfrak{a}/\alpha] \models P \,\}$$

Interpretation of base types

- Base types B interpreted as

$$[\![\mathsf{B}]\!] \quad : \quad \mathbb{N} \quad \longrightarrow \quad \mathcal{P}(\Lambda)$$

- Singleton interpretation
- Example :

$$\mathsf{s}^i\ 0 \in [\![\mathsf{N}]\!](j) \qquad \text{iff} \qquad i = j$$

- It follows that

$$[\![\mathsf{N}]\!](i) \quad \not\subseteq \quad [\![\mathsf{N}]\!](i+1)$$

Some cases of $(\!|T \leq U|\!)$

- Subtyping : $(\!|T \leq U|\!)$ is such that

$$\text{if} \quad \mu \models (\!|T \leq U|\!) \quad \text{then} \quad [\![T]\!]\mu \subseteq [\![U]\!]\mu$$

- $(\!|\mathsf{B}(\mathfrak{a}) \leq \mathsf{B}(\mathfrak{b})|\!) \quad =_{\text{def}} \quad \mathfrak{a} = \mathfrak{b}$

$$\text{if} \quad \mathfrak{a} = \mathfrak{b} \quad \text{then} \quad [\![\mathsf{B}]\!](\mathfrak{a}) \subseteq [\![\mathsf{B}]\!](\mathfrak{b})$$

- $(\!|T \leq (\exists \alpha P)U|\!) \quad =_{\text{def}} \quad \exists \alpha\, [\, P \wedge (\!|T \leq U|\!)\,] \qquad (\alpha \notin T)$

$$\text{if} \quad \mu \models \exists \alpha\, [\, P \wedge (\!|T \leq U|\!)\,] \quad \text{then}$$

$$[\![T]\!]\mu \quad \subseteq \quad \bigcup \{\, [\![U]\!]\mu[\mathfrak{a}/\alpha] \quad ; \quad \mu[\mathfrak{a}/\alpha] \models P \,\}$$

Some cases of $(\!| T \leq U |\!)$

- Subtyping : $(\!| T \leq U |\!)$ is such that

$$\text{if} \quad \mu \models (\!| T \leq U |\!) \quad \text{then} \quad [\![T]\!]\mu \subseteq [\![U]\!]\mu$$

- $(\!| \mathsf{B}(\mathfrak{a}) \leq \mathsf{B}(\mathfrak{b}) |\!) \quad =_{\text{def}} \quad \mathfrak{a} = \mathfrak{b}$

$$\text{if} \quad \mathfrak{a} = \mathfrak{b} \quad \text{then} \quad [\![\mathsf{B}]\!](\mathfrak{a}) \subseteq [\![\mathsf{B}]\!](\mathfrak{b})$$

- $(\!| T \leq (\exists \alpha P) U |\!) \quad =_{\text{def}} \quad \exists \alpha \, [\, P \wedge (\!| T \leq U |\!) \,] \qquad (\alpha \notin T)$

$$\text{if} \quad \mu \models \exists \alpha \, [\, P \wedge (\!| T \leq U |\!) \,] \quad \text{then}$$

$$[\![T]\!]\mu \quad \subseteq \quad \bigcup \{\, [\![U]\!]\mu[\mathfrak{a}/\alpha] \quad ; \quad \mu[\mathfrak{a}/\alpha] \models P \,\}$$

Outline

- For all rule $f\vec{l} \longrightarrow r$, if

$$f \;:\; \vec{V} \;\Rightarrow\; \forall\vec{\alpha}\ \vec{T} \;\Rightarrow\; U$$

and

$$C; \Gamma \vdash_{\lambda\mathcal{C}} f\vec{l} \;:\; U[\vec{\mathfrak{a}}/\vec{\alpha}]$$

and

$$C; \Gamma \vdash_{\lambda\mathcal{C}} r \;:\; U[\vec{\mathfrak{a}}/\vec{\alpha}]$$

using

$$f \;:\; \vec{V} \;\Rightarrow\; \forall\vec{\alpha}\ (\vec{\alpha} < \vec{\mathfrak{a}})\ \vec{T} \;\Rightarrow\; U$$

then we get soundness of $[\![\cdot]\!]$; hence $\mathcal{SN}$ of terms typable in $\lambda\mathcal{C}$.

- If constants have type of the form

$$f \;:\; \vec{V} \;\Rightarrow\; \forall\vec{\alpha}\ \vec{B}(\vec{\alpha}) \;\Rightarrow\; \mathcal{Q}\vec{\beta}\ \bigotimes \vec{B}(\vec{\beta})$$

then every term typable in λlet is $\mathcal{SN}$.

- For all rule $\mathsf{f}\vec{l} \longrightarrow r$, if

$$\mathsf{f} \;:\; \vec{V} \;\Rightarrow\; \forall\vec{\alpha}\ \vec{T} \;\Rightarrow\; U$$

and

$$C; \Gamma \vdash_{\lambda\mathcal{C}} \mathsf{f}\vec{l} \;:\; U[\vec{\mathfrak{a}}/\vec{\alpha}]$$

and

$$C; \Gamma \vdash_{\lambda\mathcal{C}} r \;:\; U[\vec{\mathfrak{a}}/\vec{\alpha}]$$

using $\mathsf{f} \;:\; \vec{V} \;\Rightarrow\; \forall\vec{\alpha}\ (\vec{\alpha} < \vec{\mathfrak{a}})\ \vec{T} \;\Rightarrow\; U$

then we get soundness of $[\![\cdot]\!]$; hence $\mathcal{SN}$ of terms typable in $\lambda\mathcal{C}$.

- If constants have type of the form

$$\mathsf{f} \;:\; \vec{V} \;\Rightarrow\; \forall\vec{\alpha}\ \vec{\mathsf{B}}(\vec{\alpha}) \;\Rightarrow\; \mathcal{Q}\vec{\beta}\ \bigotimes \vec{\mathsf{B}}(\vec{\beta})$$

then every term typable in λlet is $\mathcal{SN}$.

Conditional rewriting

- McCarthy's 91 function:

$$\leq\ x\ 100 \quad = \quad \text{true} \quad \supset \quad \mathsf{f}\,x \quad \rightarrow \quad \text{let} \quad y = \mathsf{f}\,(\text{plus}\,x\ 11) \text{ in } \mathsf{f}\,y$$

$$\leq\ x\ 100 \quad = \quad \text{false} \quad \supset \quad \mathsf{f}\,x \quad \rightarrow \quad \text{minus}\,x\ 10$$

- with types

$$\text{minus} \ : \ \forall\alpha\,\beta\ \mathsf{N}(\alpha) \quad \Rightarrow \quad \mathsf{N}(\beta) \quad \Rightarrow \quad \exists\delta\,P\ \mathsf{N}(\delta)$$
$$P \ =_{\text{def}} \ (\alpha \leq \beta \wedge \delta = 0) \vee (\alpha > \beta \wedge \alpha = \beta + \delta)$$

$$\mathsf{f} \ : \ \forall\alpha \mathsf{N}(\alpha) \quad \Rightarrow \quad \exists\beta\,Q\ \mathsf{N}(\beta)$$
$$Q \ =_{\text{def}} \ (\alpha \leq 100 \wedge \beta = 91) \vee (\alpha > 100 \wedge \alpha = \beta + 10)$$

Conditional rewriting

- McCarthy's 91 function:

$$\leq\ x\ 100 \quad = \quad \text{true} \quad \supset \quad \mathsf{f}\,x \quad \rightarrow \quad \begin{array}{ll}\text{let} & y = \mathsf{f}\,(\text{plus}\,x\ 11)\\ \text{in} & \mathsf{f}\,y\end{array}$$

$$\leq\ x\ 100 \quad = \quad \text{false} \quad \supset \quad \mathsf{f}\,x \quad \rightarrow \quad \text{minus}\,x\ 10$$

- with types

$$\begin{array}{rcl}\text{minus} & : & \forall\alpha\,\beta\ \mathsf{N}(\alpha) \quad \Rightarrow \quad \mathsf{N}(\beta) \quad \Rightarrow \quad \exists\delta\,P\ \mathsf{N}(\delta)\\ P & =_{\text{def}} & (\alpha \leq \beta \wedge \delta = 0) \vee (\alpha > \beta \wedge \alpha = \beta + \delta)\end{array}$$

$$\begin{array}{rcl}\mathsf{f} & : & \forall\alpha \mathsf{N}(\alpha) \quad \Rightarrow \quad \exists\beta\,Q\ \mathsf{N}(\beta)\\ Q & =_{\text{def}} & (\alpha \leq 100 \wedge \beta = 91) \vee (\alpha > 100 \wedge \alpha = \beta + 10)\end{array}$$

Conclusion

- We proposed a constraint type system. In which we can express size relations between input and output of functions.
- No elaboration, but some automation may be interesting for the insertion of let .
- The constraints allows to express relationship between input and output of functions.
- Conditional rewriting.

Further work

- Constraints solving.
  Simplification of constraints
- Higher-order types.
- Dependent types and polymorphism.
- Other properties than termination.

Example with Higher-order types

$$\begin{array}{lcl} c & : & ((C \Rightarrow N) \Rightarrow N) \Rightarrow C \\ ex & : & C \Rightarrow N \end{array}$$

$$ex\,(c\,f) \quad \longrightarrow \quad f\,ex$$

$$c\,f \quad \overset{?}{>}$$

Example with Higher-order types

$$\begin{array}{rcl} \mathsf{c} & : & ((\mathsf{C}(\alpha) \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}) \Rightarrow \mathsf{C}(\alpha+1) \\ \mathsf{ex} & : & \mathsf{C}(\alpha) \Rightarrow \mathsf{N} \end{array}$$

$$\mathsf{ex}\,(\mathsf{c}f) \;\longrightarrow\; f\;\mathsf{ex}$$

$$f : (\mathsf{C}(\alpha) \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N} \;\vdash\; \mathsf{c}f : \mathsf{C}(\alpha+1)$$

$$\mathsf{ex} : \mathsf{C}(\alpha+1) \Rightarrow \mathsf{N}$$

$$\mathsf{ex} : \mathsf{C}(\alpha) \Rightarrow \mathsf{N}$$

$$\alpha + 1 \;>\; \alpha$$

Example with Higher-order types

$$\begin{array}{rcl} \mathsf{c} & : & ((\mathsf{C}(\alpha) \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}) \Rightarrow \mathsf{C}(\alpha+1) \\ \mathsf{ex} & : & \mathsf{C}(\alpha) \Rightarrow \mathsf{N} \end{array}$$

$$\mathsf{ex}\,(\mathsf{c}f) \quad \rightarrow \quad f\ \mathsf{ex}$$

$$f : (\mathsf{C}(\alpha) \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N} \quad \vdash \quad \mathsf{c}f : \mathsf{C}(\alpha+1)$$

$$\mathsf{ex} : \mathsf{C}(\alpha+1) \Rightarrow \mathsf{N} \qquad \mathsf{ex} : \mathsf{C}(\alpha) \Rightarrow \mathsf{N}$$

$$\alpha + 1 \quad > \quad \alpha$$

Union types vs Sum types

- let -reduction makes fail subject reduction
- Union types:

$$(\exists E)\frac{C;\Gamma \vdash t : (\exists \alpha P)T \quad C \wedge P;\Gamma, x : T \vdash u : U \quad C \vdash \exists \alpha P}{C;\Gamma \vdash u[t/x] : U}$$

- Sum types:

$$(\exists I)\frac{C;\Gamma \vdash t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C;\Gamma \vdash \langle t \rangle : (\exists \alpha P)T}$$

$$\text{let } x = \langle t \rangle \text{ in } u \quad \rightarrow \quad u[t/x]$$

$$[\![(\exists \alpha P)T]\!] =_{\text{def}} \{t \in \mathcal{SN}; t \rightarrow^* \langle u \rangle \Rightarrow u \in \bigcup \{[\![T]\!]\mu[\mathfrak{a}/\alpha]; \ldots\}\}$$

Union types vs Sum types

- let -reduction makes fail subject reduction
- Union types:

$$(\exists E)\frac{C;\Gamma \vdash t : (\exists \alpha P)T \quad C \wedge P;\Gamma, x : T \vdash u : U \quad C \vdash \exists \alpha P}{C;\Gamma \vdash u[t/x] : U}$$

- Sum types:

$$(\exists I)\frac{C;\Gamma \vdash t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C;\Gamma \vdash \langle t \rangle : (\exists \alpha P)T}$$

$$\text{let } x = \langle t \rangle \text{ in } u \quad \rightarrow \quad u[t/x]$$

$$[\![(\exists \alpha P)T]\!] =_{\text{def}} \{t \in \mathcal{SN}; t \rightarrow^* \langle u \rangle \Rightarrow u \in \bigcup \{[\![T]\!]\mu[\mathfrak{a}/\alpha]; \ldots\}\}$$

Union types vs Sum types

- let -reduction makes fail subject reduction
- Union types:

$$(\exists E)\frac{C;\Gamma \vdash t : (\exists \alpha P)T \quad C \wedge P;\Gamma, x : T \vdash u : U \quad C \vdash \exists \alpha P}{C;\Gamma \vdash u[t/x] : U}$$

- Sum types:

$$(\exists I)\frac{C;\Gamma \vdash t : T[\mathfrak{a}/\alpha] \qquad C \vdash P[\mathfrak{a}/\alpha]}{C;\Gamma \vdash \langle t \rangle : (\exists \alpha P)T}$$

$$\text{let } x = \langle t \rangle \text{ in } u \quad \rightarrow \quad u[t/x]$$

$$[\![(\exists \alpha P)T]\!] =_{\text{def}} \{t \in \mathcal{SN}; t \rightarrow^* \langle u \rangle \Rightarrow u \in \bigcup \{[\![T]\!]\mu[\mathfrak{a}/\alpha]; \ldots\}\}$$