A calculus of coercions proving the strong normalization of ML^F

Giulio Manzonetto

g.manzonetto@cs.ru.nl
Intelligent Systems
Radboud University – Nijmegen

Paolo Tranquilli

paolo.tranquilli@ens-lyon.fr
LIP
École Normale Supérieure de Lyon

5th International Workshop on Higher-Order Rewriting

Edinburgh, UK, 14th of July 2010

Type inference vs. type polymorphism

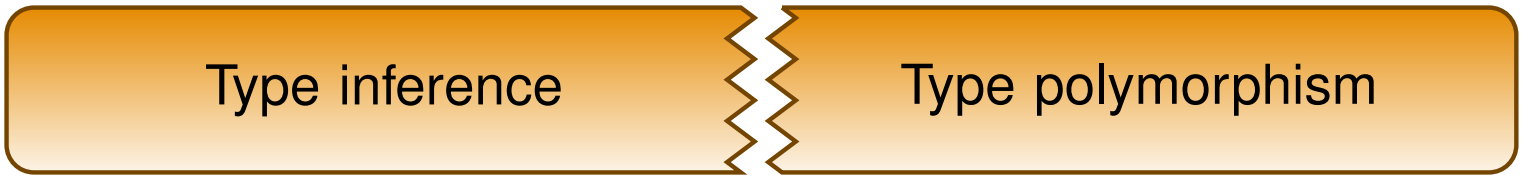
- Two issues:

Type inference	Type polymorphism
• programmer thinks about code only • machine detects “untypable” code	• expressiveness • flexibility • code reuse

- System F type inference (and even checking!) is undecidable (Wells 1999)

Type inference vs. type polymorphism

- Two issues:



  - programmer thinks about code only
  - machine detects “untypable” code

  - expressiveness
  - flexibility
  - code reuse

- System F type inference (and even checking!) is undecidable (Wells 1999)

The ML solution

- Restrict polymorphism to second class one:
  - No polymorphic abstraction, e.g. $\lambda x.xx$ is untypable
  - Polymorphism only for named variables $\texttt{let}\ x{=}M\ \texttt{in}\ N$
- Hindley-Milner algorithm gives type inference

- So, everybody happy?
- The programmer cannot write anything with first class polymorphism, even if he knows how to give such types
- One programs in a golden cage... a cage nonetheless

The ML solution

- Restrict polymorphism to second class one:
  - No polymorphic abstraction, e.g. $\lambda x.xx$ is untypable
  - Polymorphism only for named variables $\texttt{let}\ x{=}M\ \texttt{in}\ N$
- Hindley-Milner algorithm gives type inference

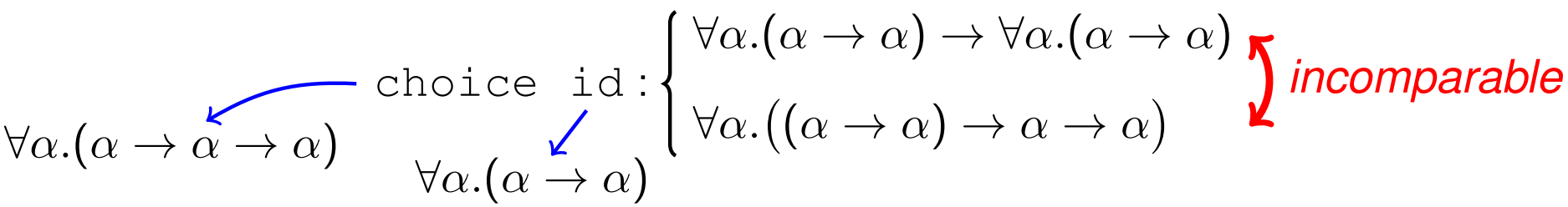
- So, everybody happy?
- The programmer cannot write anything with first class polymorphism, even if he knows how to give such types
- One programs in a golden cage... a cage nonetheless

Recovering first class polymorphism

- ML^F (Le Botlan and Rémy 2003) is an extension of ML which
  - uses partial type annotations
  - allows writing all system F programs
  - is conservative: ML programs are automatically typed without annotations

Flexible quantification

Problem: no principal types in system F



ML^F solution: $\forall(\alpha \geq \sigma)\tau$, i.e. lower bounds in quantification

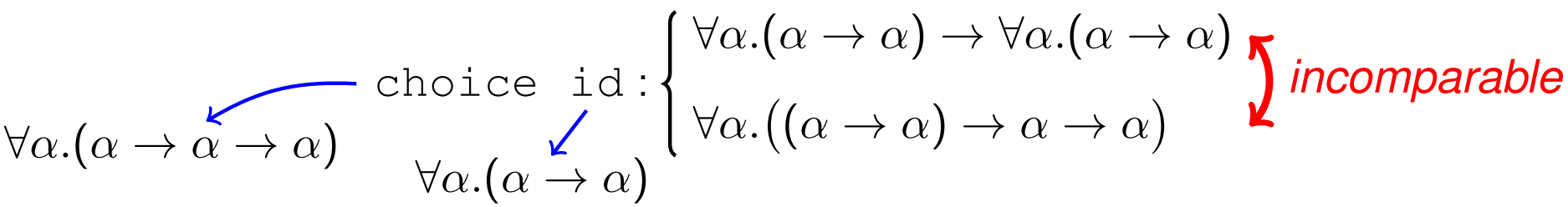
$$\texttt{choice id} : \forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta)$$

$\forall(\alpha \geq \sigma)\tau \equiv$ "τ for all α instances of σ"

Problem: instance relation $\leq$ is non-trivial. . .

Flexible quantification

Problem: no principal types in system F



ML^F solution: $\forall(\alpha \geq \sigma)\tau$, i.e. lower bounds in quantification

$$\texttt{choice id} : \forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta)$$

$\forall(\alpha \geq \sigma)\tau \equiv$ "τ for all α instances of σ"

Problem: instance relation $\leq$ is non-trivial...

The issue

- We know $F \subseteq ML^F$
- What about $ML^F \overset{?}{\subseteq} F$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial
- So let's get down to details... such as what is ML^F?
- Different versions

$$ML^F \begin{cases} ML^F & \text{no type annotation} \rightsquigarrow \text{undecidable} \\ eML^F & \text{"just enough" annotations} \rightsquigarrow \text{type inference} \\ & \text{full type annotations} \rightsquigarrow \text{internal language} \end{cases}$$

- We will present xML^F only (Rémy and Yakobowski 2009)

The issue

- We know $F \subseteq ML^F$
- What about $ML^F \overset{?}{\subseteq} F$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial

- So let's get down to details... such as what is ML^F?
- Different versions

$$ML^F \begin{cases} ML^F & \text{no type annotation} \rightsquigarrow \text{undecidable} \\ eML^F & \text{"just enough" annotations} \rightsquigarrow \text{type inference} \\ & \text{full type annotations} \rightsquigarrow \text{internal language} \end{cases}$$

- We will present xML^F only (Rémy and Yakobowski 2009)

The issue

- We know $\mathrm{F} \subseteq \mathrm{ML}^\mathrm{F}$
- What about $\mathrm{ML}^\mathrm{F} \overset{?}{\subseteq} \mathrm{F}$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial

- So let's get down to details. . . such as what is ML^F?
- Different versions

$$\mathrm{ML}^\mathrm{F} \begin{cases} \mathrm{iML}^\mathrm{F} & \text{no type annotation} \rightsquigarrow \text{undecidable} \\ \mathrm{eML}^\mathrm{F} & \text{“just enough” annotations} \rightsquigarrow \text{type inference} \\ \mathrm{xML}^\mathrm{F} & \text{full type annotations} \rightsquigarrow \text{internal language} \end{cases}$$

- We will present xML^F only (Rémy and Yakobowski 2009)

The issue

- We know $F \subseteq ML^F$
- What about $ML^F \stackrel{?}{\subseteq} F$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial

- So let's get down to details. . . such as what is ML^F?
- Different versions

ML^F {
- iML^F no type annotation $\rightsquigarrow$ undecidable
- eML^F "just enough" annotations $\rightsquigarrow$ type inference
- xML^F full type annotations $\rightsquigarrow$ internal language
}

- We will present xML^F only (Rémy and Yakobowski 2009)

The issue

- We know $\mathrm{F} \subseteq \mathrm{ML}^\mathrm{F}$
- What about $\mathrm{ML}^\mathrm{F} \stackrel{?}{\subseteq} \mathrm{F}$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial

- So let's get down to details. . . such as what is ML^F?
- Different versions

$$\mathrm{ML}^\mathrm{F} \begin{cases} \mathrm{iML}^\mathrm{F} & \text{no type annotation} & \rightsquigarrow & \text{undecidable} \\ \mathrm{eML}^\mathrm{F} & \text{“just enough” annotations} & \rightsquigarrow & \text{type inference} \\ \mathrm{xML}^\mathrm{F} & \text{full type annotations} & \rightsquigarrow & \text{internal language} \end{cases}$$

- We will present xML^F only (Rémy and Yakobowski 2009)

The issue

- We know $\mathrm{F} \subset \mathrm{ML}^\mathrm{F}$
- What about $\mathrm{ML}^\mathrm{F} \stackrel{?}{\subset} \mathrm{F}$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial

- So let's get down to details... such as what is ML^F?
- Different versions

$$\mathrm{ML}^\mathrm{F} \begin{cases} \mathrm{iML}^\mathrm{F} & \text{no type annotation} \rightsquigarrow \text{undecidable} \\ \mathrm{eML}^\mathrm{F} & \text{“just enough” annotations} \rightsquigarrow \text{type inference} \\ \mathrm{xML}^\mathrm{F} & \text{full type annotations} \rightsquigarrow \text{internal language} \end{cases}$$

- We will present xML^F only (Rémy and Yakobowski 2009)

The issue

- We know $\mathsf{F} \subset \mathsf{ML}^\mathsf{F}$
- What about $\mathsf{ML}^\mathsf{F} \overset{?}{\subset} \mathsf{F}$? Conjectured false
- $\Longrightarrow$ strong normalization result non-trivial

- So let's get down to details. . . such as what is ML^F?
- Different versions

$$\mathsf{ML}^\mathsf{F} \begin{cases} \mathsf{iML}^\mathsf{F} & \text{no type annotation} & \rightsquigarrow & \text{undecidable} \\ \mathsf{eML}^\mathsf{F} & \text{“just enough” annotations} & \rightsquigarrow & \text{type inference} \\ \mathsf{xML}^\mathsf{F} & \text{full type annotations} & \rightsquigarrow & \text{internal language} \end{cases}$$

- We will present xML^F only (Rémy and Yakobowski 2009)

Types and terms

- Types: $\sigma, \tau ::= \alpha \mid \sigma \to \tau \mid \forall(\alpha \geq \sigma)\tau \mid \bot$
- $\bot \equiv \forall\alpha.\alpha$, $\forall\alpha.\tau \equiv \forall(\alpha \geq \bot)\tau$
- Terms: $a, b ::= x \mid \lambda(x : \sigma)a \mid ab \mid \Lambda(\alpha \geq \sigma)a \mid a\phi$
- Idea: everything explicit $\rightsquigarrow$ explicit witnesses ϕ of non-trivial instance relations (defined next)

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}\ \textsc{Var}$$

$$\frac{\Gamma, x : \tau \vdash a : \sigma}{\Gamma \vdash \lambda(x : \tau)a : \tau \to \sigma}\ \textsc{Abs} \qquad \frac{\Gamma \vdash a : \sigma \to \tau \qquad \Gamma \vdash b : \sigma}{\Gamma \vdash ab : \tau}\ \textsc{App}$$

$$\frac{\Gamma, \alpha \geq \sigma \vdash a : \tau \qquad \alpha \notin \mathrm{ftv}(\Gamma)}{\Gamma \vdash \Lambda(\alpha \geq \sigma)a : \forall(\alpha \geq \sigma)\tau}\ \textsc{TAbs} \qquad \frac{\Gamma \vdash a : \tau \qquad \Gamma \vdash \phi : \tau \leq \sigma}{\Gamma \vdash a\phi : \sigma}\ \textsc{TApp}$$

- Type erasure: $\lceil \lambda(x : \sigma)a \rceil = \lambda x \lceil a \rceil$, $\lceil \Lambda(\alpha \geq \sigma)a \rceil = \lceil a\phi \rceil = \lceil a \rceil$

Types and terms

- Types: $\sigma, \tau ::= \alpha \mid \sigma \to \tau \mid \forall(\alpha \geq \sigma)\tau \mid \bot$
- $\bot \equiv \forall\alpha.\alpha$, $\forall\alpha.\tau \equiv \forall(\alpha \geq \bot)\tau$
- Terms: $a, b ::= x \mid \lambda(x : \sigma)a \mid ab \mid \Lambda(\alpha \geq \sigma)a \mid a\phi$
- Idea: everything explicit $\rightsquigarrow$ explicit witnesses ϕ of non-trivial instance relations (defined next)

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}\,\textsc{Var}$$

$$\frac{\Gamma, x : \tau \vdash a : \sigma}{\Gamma \vdash \lambda(x : \tau)a : \tau \to \sigma}\,\textsc{Abs} \qquad \frac{\Gamma \vdash a : \sigma \to \tau \qquad \Gamma \vdash b : \sigma}{\Gamma \vdash ab : \tau}\,\textsc{App}$$

$$\frac{\Gamma, \alpha \geq \sigma \vdash a : \tau \qquad \alpha \notin \mathrm{ftv}(\Gamma)}{\Gamma \vdash \Lambda(\alpha \geq \sigma)a : \forall(\alpha \geq \sigma)\tau}\,\textsc{TAbs} \qquad \frac{\Gamma \vdash a : \tau \qquad \Gamma \vdash \phi : \tau \leq \sigma}{\Gamma \vdash a\phi : \sigma}\,\textsc{TApp}$$

- Type erasure: $\lceil \lambda(x : \sigma)a \rceil = \lambda x \lceil a \rceil$, $\lceil \Lambda(\alpha \geq \sigma)a \rceil = \lceil a\phi \rceil = \lceil a \rceil$

Type instantiations

- Instances:

$$
\begin{array}{rcll}
\phi, \psi & ::= & \mathbf{1} & \text{identity} \\
& \mid & \tau & \text{bottom} \\
& \mid & !\alpha & \text{abstract} \\
& \mid & \mathcal{Y} & \text{introduction} \\
& \mid & \phi; \psi & \text{composition} \\
& \mid & \forall(\geq \phi) & \text{inside} \\
& \mid & \forall(\alpha \geq)\phi & \text{under} \\
& \mid & \& & \text{elimination}
\end{array}
$$

$$\frac{}{\Gamma \vdash \mathbf{1} : \tau \leq \tau}\ \textsc{IId} \qquad \frac{\Gamma \vdash \phi : \tau_1 \leq \tau_2 \qquad \Gamma \vdash \psi : \tau_2 \leq \tau_3}{\Gamma \vdash \phi; \psi : \tau_1 \leq \tau_3}\ \textsc{IComp}$$

$$\frac{}{\Gamma \vdash \tau : \bot \leq \tau}\ \textsc{IBot} \qquad \frac{\Gamma \vdash \phi : \tau_1 \leq \tau_2}{\Gamma \vdash \forall(\geq \phi) : \forall(\alpha \geq \tau_1)\tau \leq \forall(\alpha \geq \tau_2)\tau}\ \textsc{IInside}$$

$$\frac{\alpha \geq \tau \in \Gamma}{\Gamma \vdash !\alpha : \tau \leq \alpha}\ \textsc{IAbs} \qquad \frac{\Gamma, \alpha \geq \tau \vdash \phi : \tau_1 \leq \tau_2}{\Gamma \vdash \forall(\alpha \geq)\phi : \forall(\alpha \geq \tau)\tau_1 \leq \forall(\alpha \geq \tau)\tau_2}\ \textsc{IUnder}$$

$$\frac{\alpha \notin \mathrm{ftv}(\tau)}{\Gamma \vdash \mathcal{Y} : \tau \leq \forall(\alpha \geq \bot)\tau}\ \textsc{IIntro} \qquad \frac{}{\Gamma \vdash \& : \forall(\alpha \geq \sigma)\tau \leq \sigma\{\tau/\alpha\}}\ \textsc{IElim}$$

Type instantiations

- Instances:

$$
\begin{array}{rcll|rcl}
\phi, \psi & ::= & \mathbf{1} & \text{identity} & \mid & \phi; \psi & \text{composition} \\
 & \mid & \tau & \text{bottom} & \mid & \forall(\geq \phi) & \text{inside} \\
 & \mid & !\alpha & \text{abstract} & \mid & \forall(\alpha \geq)\phi & \text{under} \\
 & \mid & \mathcal{Y} & \text{introduction} & \mid & \& & \text{elimination}
\end{array}
$$

$$\frac{}{\Gamma \vdash \mathbf{1} : \tau \leq \tau}\ \textsc{IId} \qquad \frac{\Gamma \vdash \phi : \tau_1 \leq \tau_2 \qquad \Gamma \vdash \psi : \tau_2 \leq \tau_3}{\Gamma \vdash \phi; \psi : \tau_1 \leq \tau_3}\ \textsc{IComp}$$

$$\frac{}{\Gamma \vdash \tau : \bot \leq \tau}\ \textsc{IBot} \qquad \frac{\Gamma \vdash \phi : \tau_1 \leq \tau_2}{\Gamma \vdash \forall(\geq \phi) : \forall(\alpha \geq \tau_1)\tau \leq \forall(\alpha \geq \tau_2)\tau}\ \textsc{IInside}$$

$$\frac{\alpha \geq \tau \in \Gamma}{\Gamma \vdash !\alpha : \tau \leq \alpha}\ \textsc{IAbs} \qquad \frac{\Gamma, \alpha \geq \tau \vdash \phi : \tau_1 \leq \tau_2}{\Gamma \vdash \forall(\alpha \geq)\phi : \forall(\alpha \geq \tau)\tau_1 \leq \forall(\alpha \geq \tau)\tau_2}\ \textsc{IUnder}$$

$$\frac{\alpha \notin \mathrm{ftv}(\tau)}{\Gamma \vdash \mathcal{Y} : \tau \leq \forall(\alpha \geq \bot)\tau}\ \textsc{IIntro} \qquad \frac{}{\Gamma \vdash \& : \forall(\alpha \geq \sigma)\tau \leq \sigma\{\tau/\alpha\}}\ \textsc{IElim}$$

Examples

- Recovering system F type and instances:

$$\langle\tau\rangle := \forall(\geq \tau); \& : \forall\alpha\sigma := \forall(\alpha \geq \bot)\sigma \leq \sigma\{\tau/\sigma\}$$

- The example at the beginning of the talk:

$$\&:$$

$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \big(\forall\alpha(\alpha \to \alpha)\big) \to \forall\alpha(\alpha \to \alpha)$$

$$\mathfrak{X}; \forall(\gamma \geq)(\forall(\geq \langle\gamma\rangle); \&):$$

$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \forall\gamma\big((\gamma \to \gamma) \to \gamma \to \gamma\big)$$

(scary...)

Examples

- Recovering system F type and instances:

$$\langle\tau\rangle := \forall(\geq \tau); \& : \forall\alpha\sigma := \forall(\alpha \geq \bot)\sigma \leq \sigma\{\tau/\sigma\}$$

- The example at the beginning of the talk:

$$\&:$$

$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \big(\forall\alpha(\alpha \to \alpha)\big) \to \forall\alpha(\alpha \to \alpha)$$

$$\mathscr{X}; \forall(\gamma \geq)(\forall(\geq \langle\gamma\rangle); \&):$$

$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \forall\gamma\big((\gamma \to \gamma) \to \gamma \to \gamma\big)$$

(scary...)

Examples

- Recovering system F type and instances:

$$\langle\tau\rangle := \forall(\geq \tau); \& : \forall\alpha\sigma := \forall(\alpha \geq \bot)\sigma \leq \sigma\{\tau/\sigma\}$$

- The example at the beginning of the talk:

$$\&:$$

$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \big(\forall\alpha(\alpha \to \alpha)\big) \to \forall\alpha(\alpha \to \alpha)$$

$$\mathscr{X}; \forall(\gamma \geq)(\forall(\geq \langle\gamma\rangle); \&):$$

$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \forall\gamma\big((\gamma \to \gamma) \to \gamma \to \gamma\big)$$

(scary...)

Examples

- Recovering system F type and instances:

$$\langle\tau\rangle := \forall(\geq \tau); \& : \forall\alpha\sigma := \forall(\alpha \geq \bot)\sigma \leq \sigma\{\tau/\sigma\}$$

- The example at the beginning of the talk:

$$\&:$$
$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \big(\forall\alpha(\alpha \to \alpha)\big) \to \forall\alpha(\alpha \to \alpha)$$

$$\mathscr{X}; \forall(\gamma \geq)(\forall(\geq \langle\gamma\rangle); \&):$$
$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \forall\gamma\big((\gamma \to \gamma) \to \gamma \to \gamma\big)$$

(scary...)

Examples

- Recovering system F type and instances:

$$\langle\tau\rangle := \forall(\geq \tau); \& : \forall\alpha\sigma := \forall(\alpha \geq \bot)\sigma \leq \sigma\{\tau/\sigma\}$$

- The example at the beginning of the talk:

$$\&:$$
$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \big(\forall\alpha(\alpha \to \alpha)\big) \to \forall\alpha(\alpha \to \alpha)$$

$$\mathfrak{X}; \forall(\gamma \geq)(\forall(\geq \langle\gamma\rangle); \&):$$
$$\forall\big(\beta \geq \forall\alpha(\alpha \to \alpha)\big)(\beta \to \beta) \leq \forall\gamma\big((\gamma \to \gamma) \to \gamma \to \gamma\big)$$

(scary...)

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$
\begin{aligned}
(\lambda(x:\tau)a)b &\to_\beta a\{b/x\} \\
a\mathbf{1} &\to_\iota a \\
a(\phi;\psi) &\to_\iota (a\phi)\psi \\
a^{\forall} &\to_\iota \Lambda(\alpha \geq \bot)a, \quad \alpha \notin \mathsf{ftv}(\tau) \\
(\Lambda(\alpha \geq \tau)a)\& &\to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\} \\
(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) &\to_\iota \Lambda(\alpha \geq \tau)(a\phi) \\
(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) &\to_\iota \Lambda(\alpha \geq \sigma)a\{\phi; !\alpha/!\alpha\} \\
& \qquad \text{(with } \phi : \tau \leq \sigma\text{)}
\end{aligned}
$$

- Non-trivial type reduction (size can increase)
- "Abstraction" $!\alpha$ behaves as variable

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$(\lambda(x:\tau)a)b \to_\beta a\{b/x\}$$

$$a\mathbf{1} \to_\iota a$$

$$a(\phi;\psi) \to_\iota (a\phi)\psi$$

identity does nothing, composition composes

$$(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) \to_\iota \Lambda(\alpha \geq \tau)(a\phi)$$

$$(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) \to_\iota \Lambda(\alpha \geq \sigma)a\{\phi; !\alpha/!\alpha\}$$

(with $\phi : \tau \leq \sigma$)

- Non-trivial type reduction (size can increase)
- "Abstraction" $!\alpha$ behaves as variable

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$(\lambda(x:\tau)a)b \to_\beta a\{b/x\}$$
$$a\mathbf{1} \to_\iota a$$
$$a(\phi;\psi) \to_\iota (a\phi)\psi$$
$$a^{\forall} \to_\iota \Lambda(\alpha \geq \bot)a, \quad \alpha \notin \mathsf{ftv}(\tau)$$

introduction of an empty type abstraction

$$(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) \to_\iota \Lambda(\alpha \geq \sigma)a\{\phi; !\alpha/!\alpha\}$$
(with $\phi : \tau \leq \sigma$)

- Non-trivial type reduction (size can increase)
- "Abstraction" $!\alpha$ behaves as variable

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$(\lambda(x:\tau)a)b \to_\beta a\{b/x\}$$
$$a\mathbf{1} \to_\iota a$$
$$a(\phi;\psi) \to_\iota (a\phi)\psi$$
$$a^{\mathcal{Y}} \to_\iota \Lambda(\alpha \geq \bot)a, \quad \alpha \notin \mathrm{ftv}(\tau)$$
$$(\Lambda(\alpha \geq \tau)a)\& \to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\}$$

elimination substitutes type for variable
but what about $!\alpha : \tau \leq \alpha$ in a?
replaced with $\mathbf{1} : \tau \leq \tau = \alpha\{\tau/\alpha\}$

- Non-trivial type reduction (size can increase)
- "Abstraction" $!\alpha$ behaves as variable

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$(\lambda(x:\tau)a)b \to_\beta a\{b/x\}$$
$$a\mathbf{1} \to_\iota a$$
$$a(\phi;\psi) \to_\iota (a\phi)\psi$$
$$a^{\mathcal{Y}} \to_\iota \Lambda(\alpha \geq \bot)a, \quad \alpha \notin \mathrm{ftv}(\tau)$$
$$(\Lambda(\alpha \geq \tau)a)\& \to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\}$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) \to_\iota \Lambda(\alpha \geq \tau)(a\phi)$$

under rule passes instantiation underneath (notice variable unification)

- Non-trivial type reduction (size can increase)
- "Abstraction" $!\alpha$ behaves as variable

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$(\lambda(x:\tau)a)b \to_\beta a\{b/x\}$$
$$a\mathbf{1} \to_\iota a$$
$$a(\phi;\psi) \to_\iota (a\phi)\psi$$
$$a^{\mathcal{Y}} \to_\iota \Lambda(\alpha \geq \bot)a, \quad \alpha \notin \mathrm{ftv}(\tau)$$
$$(\Lambda(\alpha \geq \tau)a)\& \to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\}$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) \to_\iota \Lambda(\alpha \geq \tau)(a\phi)$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) \to_\iota \Lambda(\alpha \geq \sigma)a\{\phi; !\alpha/!\alpha\}$$
(with $\phi : \tau \leq \sigma$)

- Non-trivi
- "Abstrac

inside rule changes bound
but what about $!\alpha : \tau \leq \alpha$?
replaced with $\phi; !\alpha : \tau \leq \sigma \leq \alpha$

Reduction

Main contribution of xML^F: evolution of instance witnesses

$$
\begin{aligned}
(\lambda(x:\tau)a)b &\to_\beta a\{b/x\} \\
a\mathbf{1} &\to_\iota a \\
a(\phi;\psi) &\to_\iota (a\phi)\psi \\
a^{\mathcal{Y}} &\to_\iota \Lambda(\alpha \geq \bot)a, \quad \alpha \notin \mathsf{ftv}(\tau) \\
(\Lambda(\alpha \geq \tau)a)\& &\to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\} \\
(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) &\to_\iota \Lambda(\alpha \geq \tau)(a\phi) \\
(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) &\to_\iota \Lambda(\alpha \geq \sigma)a\{\phi; !\alpha/!\alpha\} \\
& \qquad \text{(with } \phi : \tau \leq \sigma\text{)}
\end{aligned}
$$

- Non-trivial type reduction (size can increase)
- "Abstraction" $!\alpha$ behaves as variable

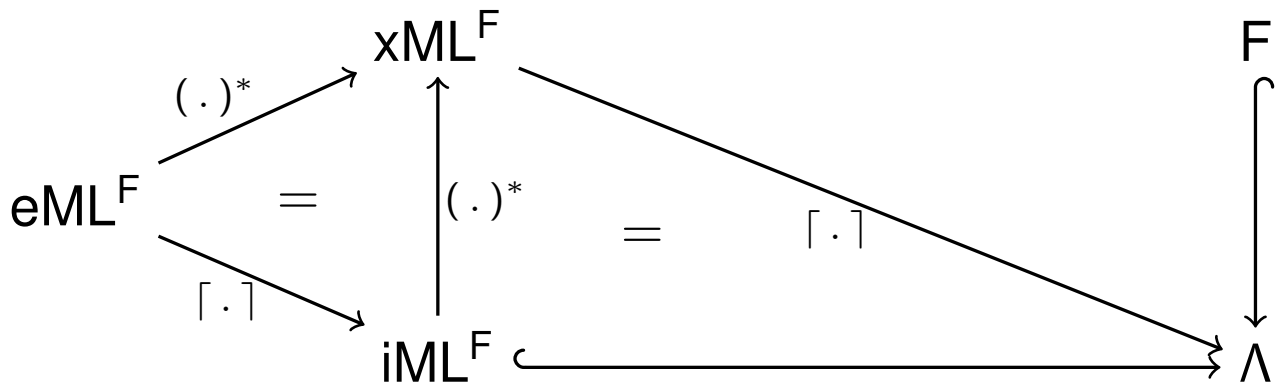
Intermezzo: what about candidates of reducibility?

- Our proof will be a simulation in system F
- We started working with candidates, failing to find a good interpretation for iML$^{\mathsf{F}}$
- When we drifted to xML$^{\mathsf{F}}$, we found the simulation
- However in a way the naïve interpretation works for xML$^{\mathsf{F}}$ (but not directly for other versions)!

$$[\![\forall(\alpha \geq \sigma)\tau]\!]_\Sigma := \bigcap_{S \supseteq [\![\sigma]\!]_\Sigma} [\![\tau]\!]_{\Sigma[\alpha \mapsto S]}$$

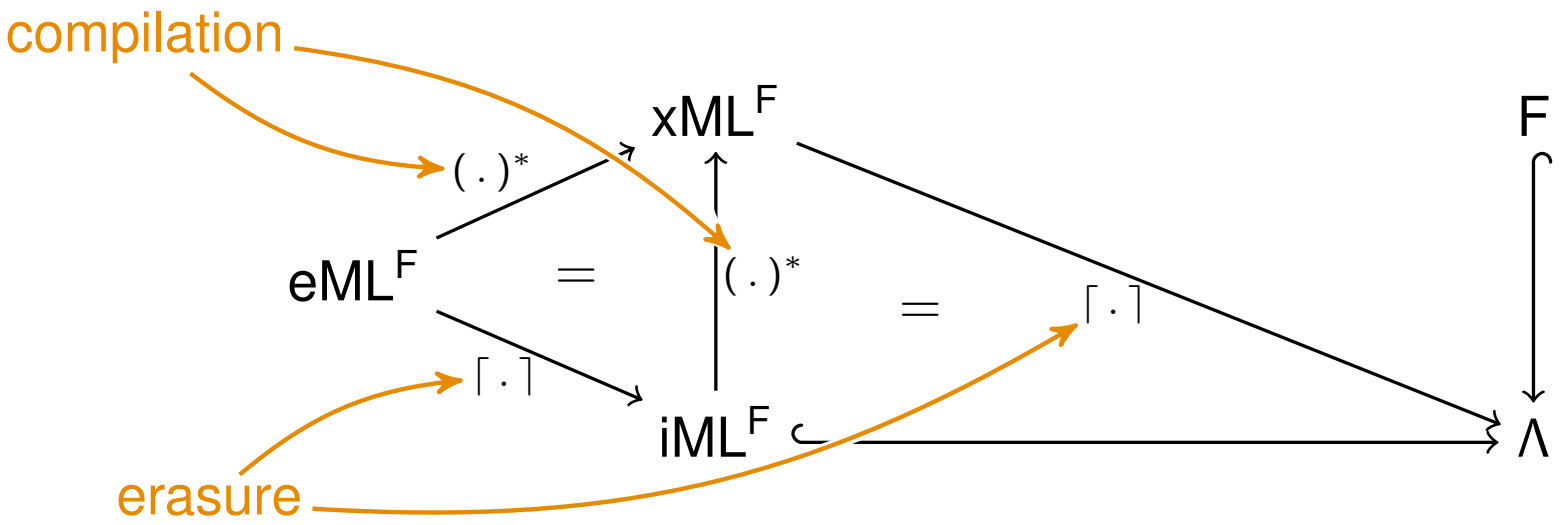
- But beware: they prove SN of type erasure $\lceil a \rceil$, with non-trivial type reduction left out

The big picture



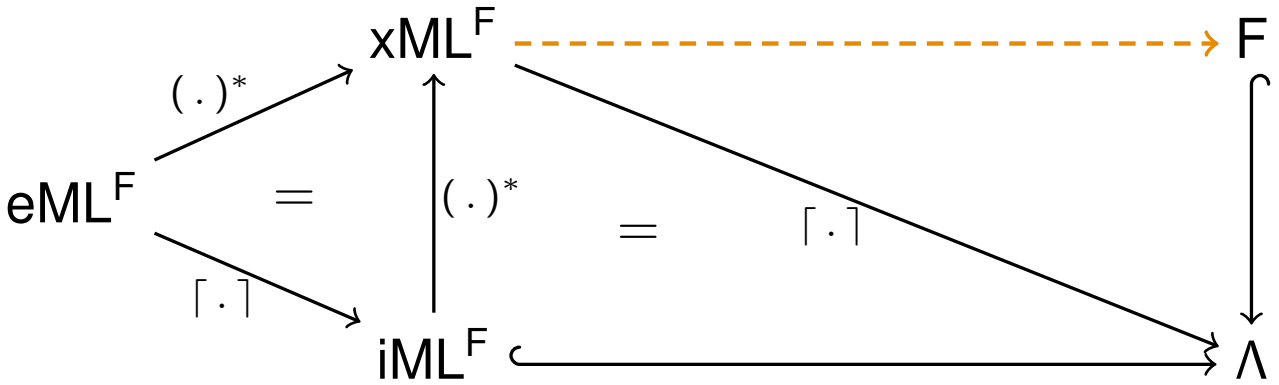
- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

The big picture



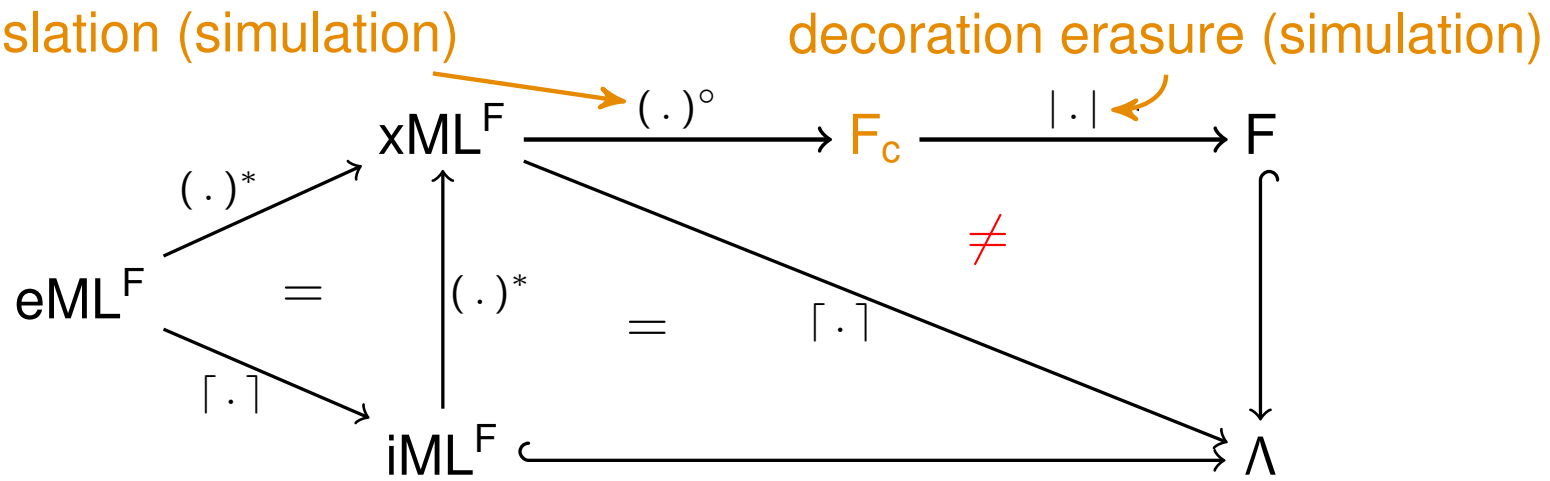
- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

The big picture



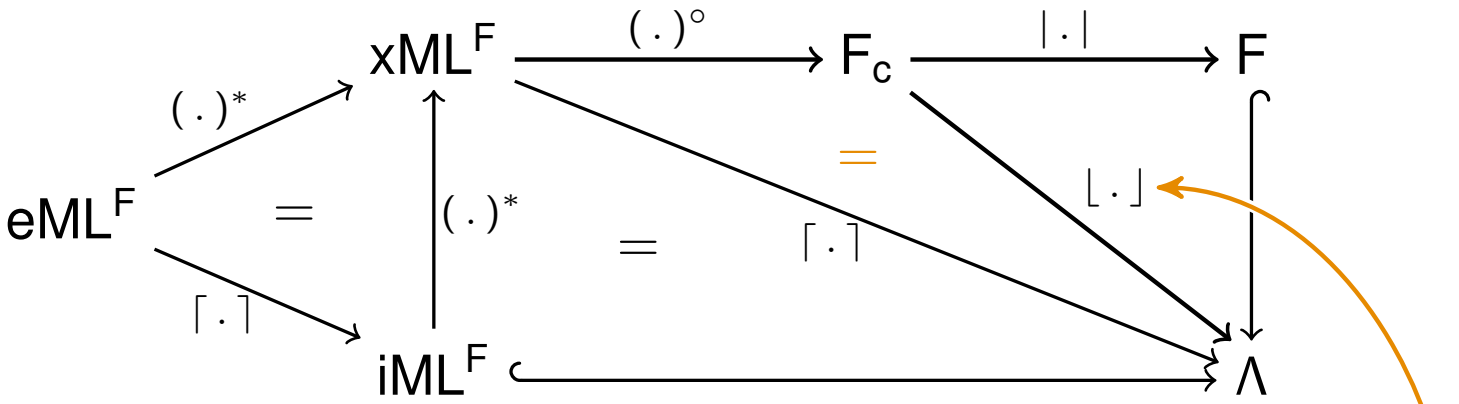
- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

The big picture



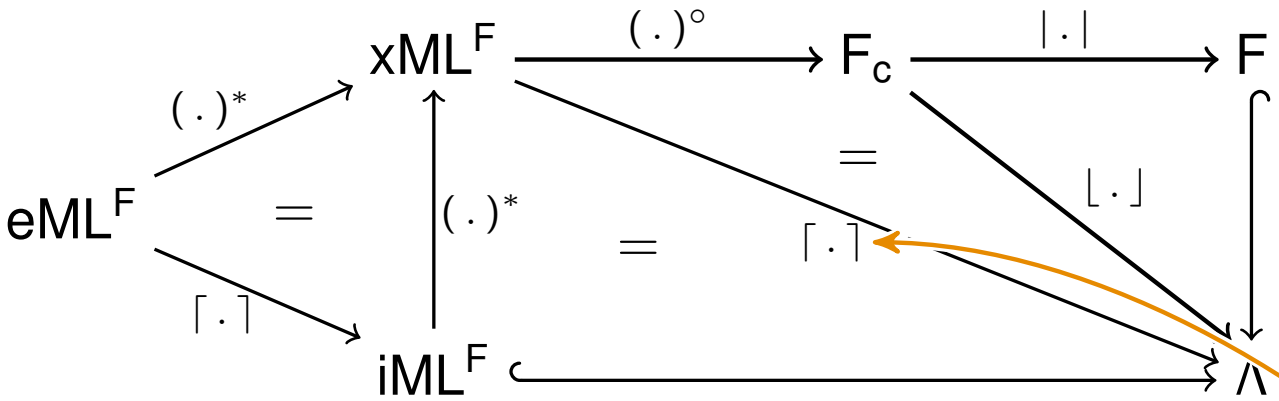
- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

The big picture



- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

The big picture



- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

A disclaimer

F_c is tailored down to obtain the main result, but is potentially more general

The syntax of F_c

- Term types: $\sigma, \tau ::= \alpha \mid \sigma \to \tau \mid \forall \alpha.\sigma \mid \kappa \to \tau$
- Coercion types: $\kappa ::= \sigma \multimap \tau$

- 3 arrows $\rightsquigarrow$ 3 abstraction/application pairs (just a decoration!)
- Terms: $a, b ::= x \mid \lambda x.a \mid ab \mid \underline{\lambda}x.a \mid a \triangleleft b \mid \underline{\lambda}x.a \mid a \triangleright b$

- $\underline{\lambda}x.a$ (paired with $a \triangleleft b$) expects a coercion (is typed $\kappa \to \sigma$)
- $\underline{\lambda}x.a$ (paired with $a \triangleright b$) builds a coercion (is typed $\sigma \multimap \tau$)

Typing rules

- Linear arrow for coercions not casual: F_c is a decoration of dual intuitionistic linear logic (Barber, Plotkin 1997)
- Judgments: $\Gamma; L \vdash a : \zeta$, with $\#\,\mathrm{dom}(L) \leq 1$
- Regular $+$ linear contexts, restricting L to at most one variable

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{tc} y : \zeta}\,\textsc{Ax} \qquad \frac{\Gamma, x : \tau; \vdash_t a : \sigma}{\Gamma; \vdash_t \lambda x.a : \tau \to \sigma}\,\textsc{Abs} \qquad \frac{\Gamma; \vdash_t a : \sigma \to \tau \qquad \Gamma; \vdash_t b : \sigma}{\Gamma; \vdash_t ab : \tau}\,\textsc{App}$$

$$\frac{}{\Gamma; z : \tau \vdash_\ell z : \tau}\,\textsc{LAx} \qquad \frac{\Gamma; z : \tau \vdash_\ell a : \sigma}{\Gamma; \vdash_c \underline{\lambda} z.a : \tau \multimap \sigma}\,\textsc{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{t\ell} a : \sigma}{\Gamma; L \vdash_{t\ell} \underline{\lambda} x.a : \kappa \to \sigma}\,\textsc{CAbs}$$

$$\frac{\Gamma; \vdash_c a : \sigma_1 \multimap \sigma_2 \qquad \Gamma; L \vdash_{t\ell} b : \sigma_1}{\Gamma; L \vdash_{t\ell} a \triangleright b : \sigma_2}\,\textsc{LApp} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \kappa \to \sigma \qquad \Gamma \vdash_c b : \kappa}{\Gamma; L \vdash_{t\ell} a \triangleleft b : \sigma}\,\textsc{CApp}$$

$$\frac{\Gamma; L \vdash_{t\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}\,\textsc{Gen} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{t\ell} a : \sigma\{\tau'/\alpha\}}\,\textsc{Inst}$$

Typing rules

subscript of $\vdash$ is just convenience:

no linear context, regular type $\rightsquigarrow \Gamma; \vdash_{\mathrm{t}} a : \sigma$, terms

no linear context, coercion type $\rightsquigarrow \Gamma; \vdash_{\mathrm{c}} a : \kappa$, coercions

linear context, regular type $\rightsquigarrow \Gamma; x : \tau \vdash_{\ell} a : \sigma$, in progress

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{\mathrm{tc}} y : \zeta}\,\textsc{Ax} \qquad \frac{\Gamma, x : \tau; \vdash_{\mathrm{t}} a : \sigma}{\Gamma; \vdash_{\mathrm{t}} \lambda x.a : \tau \to \sigma}\,\textsc{Abs} \qquad \frac{\Gamma; \vdash_{\mathrm{t}} a : \sigma \to \tau \quad \Gamma; \vdash_{\mathrm{t}} b : \sigma}{\Gamma; \vdash_{\mathrm{t}} ab : \tau}\,\textsc{App}$$

$$\frac{}{\Gamma; z : \tau \vdash_{\ell} z : \tau}\,\textsc{LAx} \qquad \frac{\Gamma; z : \tau \vdash_{\ell} a : \sigma}{\Gamma; \vdash_{\mathrm{c}} \underline{\lambda} z.a : \tau \multimap \sigma}\,\textsc{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{\mathrm{t}\ell} a : \sigma}{\Gamma; L \vdash_{\mathrm{t}\ell} \underline{\Lambda} x.a : \kappa \to \sigma}\,\textsc{CAbs}$$

$$\frac{\Gamma; \vdash_{\mathrm{c}} a : \sigma_1 \multimap \sigma_2 \quad \Gamma; L \vdash_{\mathrm{t}\ell} b : \sigma_1}{\Gamma; L \vdash_{\mathrm{t}\ell} a \triangleright b : \sigma_2}\,\textsc{LApp} \qquad \frac{\Gamma; L \vdash_{\mathrm{t}\ell} a : \kappa \to \sigma \quad \Gamma \vdash_{\mathrm{c}} b : \kappa}{\Gamma; L \vdash_{\mathrm{t}\ell} a \triangleleft b : \sigma}\,\textsc{CApp}$$

$$\frac{\Gamma; L \vdash_{\mathrm{t}\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{\mathrm{t}\ell} a : \forall \alpha.\sigma}\,\textsc{Gen} \qquad \frac{\Gamma; L \vdash_{\mathrm{t}\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{\mathrm{t}\ell} a : \sigma\{\tau'/\alpha\}}\,\textsc{Inst}$$

Typing rules

- Linear arrow for coercions not casual: F_c is a decoration of dual intuitionistic linear logic (Barber, Plotkin 1997)
- Judgments: $\Gamma; L \vdash_{tc} a : \zeta$ with $\#\mathrm{dom}(L) \leq 1$
- Regular and linear axioms as expected, no linear context in regular axiom

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{tc} y : \zeta}\,\mathrm{Ax} \qquad \frac{\Gamma, x : \tau; \vdash_t a : \sigma}{\Gamma; \vdash_t \lambda x.a : \tau \to \sigma}\,\mathrm{Abs} \qquad \frac{\Gamma; \vdash_t a : \sigma \to \tau \quad \Gamma; \vdash_t b : \sigma}{\Gamma; \vdash_t ab : \tau}\,\mathrm{App}$$

$$\frac{}{\Gamma; z : \tau \vdash_\ell z : \tau}\,\mathrm{LAx} \qquad \frac{\Gamma; z : \tau \vdash_\ell a : \sigma}{\Gamma; \vdash_c \underline{\lambda} z.a : \tau \multimap \sigma}\,\mathrm{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{t\ell} a : \sigma}{\Gamma; L \vdash_{t\ell} \underline{\lambda} x.a : \kappa \to \sigma}\,\mathrm{CAbs}$$

$$\frac{\Gamma; \vdash_c a : \sigma_1 \multimap \sigma_2 \quad \Gamma; L \vdash_{t\ell} b : \sigma_1}{\Gamma; L \vdash_{t\ell} a \triangleright b : \sigma_2}\,\mathrm{LApp} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \kappa \to \sigma \quad \Gamma \vdash_c b : \kappa}{\Gamma; L \vdash_{t\ell} a \triangleleft b : \sigma}\,\mathrm{CApp}$$

$$\frac{\Gamma; L \vdash_{t\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}\,\mathrm{Gen} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{t\ell} a : \sigma\{\tau'/\alpha\}}\,\mathrm{Inst}$$

Typing rules

- Linear arrow for coercions not casual: F_c is a decoration of dual intuitionistic linear logic (Barber, Plotkin 1997)
- Judgments: $\Gamma; L \vdash a : \zeta$, with $\#\,\mathrm{dom}(L) \leq 1$
- Regu… …ariable

regular operations available for regular terms only

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{tc} y : \zeta}\,\textsc{Ax} \qquad \frac{\Gamma, x : \tau; \vdash_t a : \sigma}{\Gamma; \vdash_t \lambda x.a : \tau \to \sigma}\,\textsc{Abs} \qquad \frac{\Gamma; \vdash_t a : \sigma \to \tau \qquad \Gamma; \vdash_t b : \sigma}{\Gamma; \vdash_t ab : \tau}\,\textsc{App}$$

$$\frac{}{\Gamma; z : \tau \vdash_\ell z : \tau}\,\textsc{LAx} \qquad \frac{\Gamma; z : \tau \vdash_\ell a : \sigma}{\Gamma; \vdash_c \underline{\lambda} z.a : \tau \multimap \sigma}\,\textsc{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{t\ell} a : \sigma}{\Gamma; L \vdash_{t\ell} \underline{\lambda} x.a : \kappa \to \sigma}\,\textsc{CAbs}$$

$$\frac{\Gamma; \vdash_c a : \sigma_1 \multimap \sigma_2 \qquad \Gamma; L \vdash_{t\ell} b : \sigma_1}{\Gamma; L \vdash_{t\ell} a \triangleright b : \sigma_2}\,\textsc{LApp} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \kappa \to \sigma \qquad \Gamma \vdash_c b : \kappa}{\Gamma; L \vdash_{t\ell} a \triangleleft b : \sigma}\,\textsc{CApp}$$

$$\frac{\Gamma; L \vdash_{t\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}\,\textsc{Gen} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{t\ell} a : \sigma\{\tau'/\alpha\}}\,\textsc{Inst}$$

Typing rules

- Linear arrow for coercions not casual: F_c is a decoration of dual intuitionistic linear logic (Barber, Plotkin 1997)
- Judgments: $\Gamma; L \vdash a : \zeta$, with $\#\,\mathrm{dom}(L) \leq 1$
- Regular + linear contexts, restricting L to at most one variable

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{tc} y : \zeta}\,\textsc{Ax} \qquad \frac{\Gamma, x : \tau; \vdash_t a : \sigma}{\Gamma; \vdash_t \lambda x.a : \tau \to \sigma}\,\textsc{Abs} \qquad \frac{\Gamma; \vdash_t a : \sigma \to \tau \quad \Gamma; \vdash_t b : \sigma}{\Gamma; \vdash_t ab : \tau}\,\textsc{App}$$

creation of a coercion

$$\frac{}{\Gamma; z : \tau \vdash_\ell z : \tau}\,\textsc{LAx} \qquad \frac{\Gamma; z : \tau \vdash_\ell a : \sigma}{\Gamma; \vdash_c \underline{\lambda} z.a : \tau \multimap \sigma}\,\textsc{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{t\ell} a : \sigma}{\Gamma; L \vdash_{t\ell} \underline{\lambda} x.a : \kappa \to \sigma}\,\textsc{CAbs}$$

$$\frac{\Gamma; \vdash_c a : \sigma_1 \multimap \sigma_2 \quad \Gamma; L \vdash_{t\ell} b : \sigma_1}{\Gamma; L \vdash_{t\ell} a \triangleright b : \sigma_2}\,\textsc{LApp} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \kappa \to \sigma \quad \Gamma \vdash_c b : \kappa}{\Gamma; L \vdash_{t\ell} a \triangleleft b : \sigma}\,\textsc{CApp}$$

$$\frac{\Gamma; L \vdash_{t\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}\,\textsc{Gen} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{t\ell} a : \sigma\{\tau'/\alpha\}}\,\textsc{Inst}$$

Typing rules

- Linear arrow for coercions not casual: F_c is a decoration of dual intuitionistic linear logic (Barber, Plotkin 1997)
- Judgments: $\Gamma; L \vdash a : \zeta$, with $\#\,\mathrm{dom}(L) \leq 1$
- Regular + linear contexts, restricting L to at most one variable

other coercion operation allowed
both terms and coercion in progress

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{tc} y : \zeta}\,\textsc{Ax} \qquad \frac{\Gamma; \vdash_t b : \sigma}{\ldots}\,\textsc{App}$$

$$\frac{}{\Gamma; z : \tau \vdash_\ell z : \tau}\,\textsc{LAx} \qquad \frac{\Gamma; z : \tau \vdash_\ell a : \sigma}{\Gamma; \vdash_c \underline{\lambda} z.a : \tau \multimap \sigma}\,\textsc{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{t\ell} a : \sigma}{\Gamma; L \vdash_{t\ell} \underline{\lambda} x.a : \kappa \to \sigma}\,\textsc{CAbs}$$

$$\frac{\Gamma; \vdash_c a : \sigma_1 \multimap \sigma_2 \quad \Gamma; L \vdash_{t\ell} b : \sigma_1}{\Gamma; L \vdash_{t\ell} a \triangleright b : \sigma_2}\,\textsc{LApp} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \kappa \to \sigma \quad \Gamma \vdash_c b : \kappa}{\Gamma; L \vdash_{t\ell} a \triangleleft b : \sigma}\,\textsc{CApp}$$

$$\frac{\Gamma; L \vdash_{t\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}\,\textsc{Gen} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{t\ell} a : \sigma\{\tau'/\alpha\}}\,\textsc{Inst}$$

Typing rules

- Linear arrow for coercions not casual: F_c is a decoration of dual intuitionistic linear logic (Barber, Plotkin 1997)
- Judgments: $\Gamma; L \vdash a : \zeta$, with $\#\,\mathrm{dom}(L) \leq 1$
- Regular $+$ linear contexts, restricting L to at most one variable

$$\frac{\Gamma(y) = \zeta}{\Gamma; \vdash_{tc} y : \zeta}\,\textsc{Ax} \qquad \frac{\Gamma, x : \tau; \vdash_t a : \sigma}{\Gamma; \vdash_t \lambda x.a : \tau \to \sigma}\,\textsc{Abs} \qquad \frac{\Gamma; \vdash_t a : \sigma \to \tau \qquad \Gamma; \vdash_t b : \sigma}{\Gamma; \vdash_t ab : \tau}\,\textsc{App}$$

$$\frac{}{\Gamma; z : \tau \vdash_\ell z : \tau}\,\textsc{LAx} \qquad \frac{\Gamma; z : \tau \vdash_\ell a : \sigma}{\Gamma; \vdash_c \underline{\lambda} z.a : \tau \multimap \sigma}\,\textsc{LAbs} \qquad \frac{\Gamma, x : \kappa; L \vdash_{t\ell} a : \sigma}{\Gamma; L \vdash_{t\ell} \underline{\lambda} x.a : \kappa \to \sigma}\,\textsc{CAbs}$$

$$\frac{\Gamma; \vdash_c a : \sigma_1 \multimap \sigma_2 \qquad \Gamma; L \vdash_{t\ell} b : \sigma_1}{\Gamma; L \vdash_{t\ell} a \triangleright b : \sigma_2}\,\textsc{LApp} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \kappa \to \sigma \qquad \Gamma \vdash_c b : \kappa}{\Gamma; L \vdash_{t\ell} a \triangleleft b : \sigma}\,\textsc{CApp}$$

$$\frac{\Gamma; L \vdash_{t\ell} a : \sigma \qquad \alpha \notin \mathrm{ftv}(\Gamma; L)}{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}\,\textsc{Gen} \qquad \frac{\Gamma; L \vdash_{t\ell} a : \forall \alpha.\sigma}{\Gamma; L \vdash_{t\ell} a : \sigma\{\tau'/\alpha\}}\,\textsc{Inst}$$

Reduction

Not much to say:

$$(\lambda x.a)b \to_\beta a\{b/x\}$$
$$(\underline{\lambda}x.a) \triangleleft b \to_c a\{b/x\}$$
$$(\underline{\lambda}x.a) \triangleright b \to_c a\{b/x\}$$

First results

Theorem

F_c *enjoys subject reduction and is confluent*

Theorem

F_c *is strongly normalizing*

Proof. If $a \to_{\beta c} b$ then $|a| \to |b|$, where $|\,.\,|$ is straightforward collapsing of all to regular system F □

Lemma

$$\begin{array}{ccc} a & \xrightarrow{\beta} & b_2 \\ {\scriptstyle c*}\downarrow & & \vdots{\scriptstyle c*} \\ b_1 & \dashrightarrow_{\beta} & c \end{array}$$

Coercion erasure

- How to recover actual semantics of term?
- Coercion erasure, defined on regular terms:

$$\lfloor x \rfloor := x, \quad \lfloor \lambda x.a \rfloor := \lambda x.\lfloor a \rfloor, \quad \lfloor ab \rfloor := \lfloor a \rfloor \lfloor b \rfloor,$$
$$\lfloor \underline{\lambda} x.a \rfloor := \lfloor a \rfloor, \quad \lfloor a \triangleleft b \rfloor := \lfloor a \rfloor, \quad \lfloor a \triangleright b \rfloor := \lfloor b \rfloor.$$

Lemma

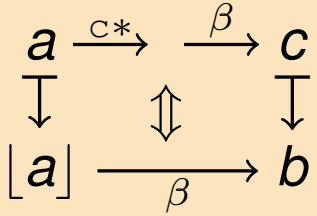
1. *If* $a \to_\beta b$ *then* $\lfloor a \rfloor \to \lfloor b \rfloor$
2. *If* $a \to_c b$ *then* $\lfloor a \rfloor = \lfloor b \rfloor$

Quest for bisimulation

We want the following ($\to_{\mathrm{C}}$ are silent actions)

Theorem (weak bisimulation)

$\lfloor a \rfloor \to_\beta b$ *iff* $a \overset{*}{\to}_{\mathrm{C}} \to_\beta c$ *with* $\lfloor c \rfloor = b$



With what I presented, it fails: $\lfloor \, . \, \rfloor$ is surjective over Λ, using context $i : o \multimap (o \to o), p : (o \to o) \multimap o$. Coercion variables may block regular redexes

We impose a condition on contexts:

$$x : \kappa \in \Gamma \implies \kappa = \sigma \multimap \alpha$$

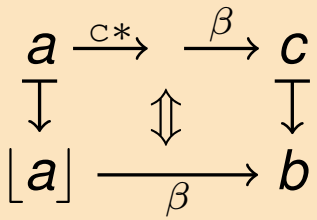
This condition suffices to have the result!

Quest for bisimulation

We want the following ($\to_{\mathrm{c}}$ are silent actions)

Theorem (weak bisimulation)

$\lfloor a \rfloor \to_\beta b$ *iff* $a \overset{*}{\to}_{\mathrm{c}} \to_\beta c$ *with* $\lfloor c \rfloor = b$



With what I presented, it fails: $\lfloor \, . \, \rfloor$ is surjective over Λ, using context $i : o \multimap (o \to o), p : (o \to o) \multimap o$. Coercion variables may block regular redexes

We impose a condition on contexts:

$$x : \kappa \in \Gamma \implies \kappa = \sigma \multimap \alpha$$

This condition suffices to have the result!

Back to xML^F: translating types and contexts

- Time to translate!
- Idea: quantification $\forall(\alpha \geq \sigma)\tau$ expects instantiation of σ in α taking place of $!\alpha$
- On types:

$$\alpha^\bullet := \alpha, \qquad (\sigma \to \tau)^\bullet := \sigma^\bullet \to \tau^\bullet,$$
$$\bot^\bullet := \forall\alpha.\alpha, \quad (\forall(\alpha \geq \sigma)\tau)^\bullet := \forall\alpha.(\sigma^\bullet \multimap \alpha) \to \tau^\bullet,$$

- On contexts:

$$(x : \tau)^\bullet := x : \tau^\bullet, \qquad (\alpha \geq \tau)^\bullet := i_\alpha : \tau^\bullet \multimap \alpha$$

- Idea originally in (Leijen 2007), but no dynamic property studied

Translating terms and instantiations

- On terms:

$$x^\circ := x, \qquad (\lambda(x:\tau)a)^\circ := \lambda x.a^\circ, \qquad (ab)^\circ := a^\circ b^\circ,$$
$$(\Lambda(\alpha \geq \tau)a)^\circ := \underline{\lambda} i_\alpha.a^\circ, \qquad (a\phi)^\circ := \phi^\circ \triangleright a^\circ$$

- On instantiations:

$$\begin{aligned}
(\mathbf{1})^\circ &:= \underline{\lambda} z.z \\
(\phi;\psi)^\circ &:= \underline{\lambda} z.\psi^\circ \triangleright (\phi^\circ \triangleright z) \\
\tau^\circ &:= \underline{\lambda} x.x \\
(!\alpha)^\circ &:= i_\alpha \\
(\parr)^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.x \\
(\&)^\circ &:= \underline{\lambda} x.x \triangleleft \underline{\lambda} z.z \\
(\forall(\geq \phi))^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.x \triangleleft (\underline{\lambda} z.i_\alpha \triangleright (\phi^\circ \triangleright z)) \\
(\forall(\alpha \geq)\phi)^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.\phi^\circ \triangleright (x \triangleleft i_\alpha)
\end{aligned}$$

$$a\mathbf{1} \to_\iota a$$
$$a(\phi;\psi) \to_\iota (a\phi)\psi$$
$$a\parr \to_\iota \Lambda(\alpha \geq \bot)a$$
$$(\Lambda(\alpha \geq \tau)a)\& \to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\}$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) \to_\iota \Lambda(\alpha \geq \sigma)a\{\phi;!\alpha/!\alpha\}$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) \to_\iota \Lambda(\alpha \geq \tau)(a\phi)$$

Translating terms and instantiations

- On terms:

$$x^\circ := x, \qquad (\lambda(x:\tau)a)^\circ := \lambda x.a^\circ, \qquad (ab)^\circ := a^\circ b^\circ,$$
$$(\Lambda(\alpha \geq \tau)a)^\circ := \underline{\lambda} i_\alpha.a^\circ, \qquad (a\phi)^\circ := \phi^\circ \triangleright a^\circ$$

- On instantiations:

$$\begin{aligned}
(\mathbf{1})^\circ &:= \underline{\lambda} z.z \\
(\phi;\psi)^\circ &:= \underline{\lambda} z.\psi^\circ \triangleright (\phi^\circ \triangleright z) \\
\tau^\circ &:= \underline{\lambda} x.x \\
(!\alpha)^\circ &:= i_\alpha \\
(\parr)^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.x \\
(\&)^\circ &:= \underline{\lambda} x.x \triangleleft \mathbf{1}^\circ \\
(\forall(\geq \phi))^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.x \triangleleft (\phi;!\alpha)^\circ \\
(\forall(\alpha \geq)\phi)^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.\phi^\circ \triangleright (x \triangleleft i_\alpha)
\end{aligned}$$

$$a\mathbf{1} \to_\iota a$$
$$a(\phi;\psi) \to_\iota (a\phi)\psi$$
$$a\parr \to_\iota \Lambda(\alpha \geq \bot)a$$
$$(\Lambda(\alpha \geq \tau)a)\& \to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\}$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) \to_\iota \Lambda(\alpha \geq \sigma)a\{\phi;!\alpha/!\alpha\}$$
$$(\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) \to_\iota \Lambda(\alpha \geq \tau)(a\phi)$$

Translating terms and instantiations

- On terms:

$$x^\circ := x, \qquad (\lambda(x:\tau)a)^\circ := \lambda x.a^\circ, \qquad (ab)^\circ := a^\circ b^\circ,$$
$$(\Lambda(\alpha \geq \tau)a)^\circ := \underline{\lambda} i_\alpha.a^\circ, \qquad (a\phi)^\circ := \phi^\circ \triangleright a^\circ$$

- On instantiations:

$$\begin{aligned}
(\mathbf{1})^\circ &:= \underline{\lambda} z.z & a\mathbf{1} &\to_\iota a \\
(\phi;\psi)^\circ &:= \underline{\lambda} z.\psi^\circ \triangleright (\phi^\circ \triangleright z) & a(\phi;\psi) &\to_\iota (a\phi)\psi \\
\tau^\circ &:= \underline{\lambda} x.x \\
(!\alpha)^\circ &:= i_\alpha \\
(\parr)^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.x & a\parr &\to_\iota \Lambda(\alpha \geq \bot)a \\
(\&)^\circ &:= \underline{\lambda} x.x \triangleleft \mathbf{1}^\circ & (\Lambda(\alpha \geq \tau)a)\& &\to_\iota a\{\mathbf{1}/!\alpha\}\{\tau/\alpha\} \\
(\forall(\geq \phi))^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.x \triangleleft (\phi;!\alpha)^\circ \\
& & (\Lambda(\alpha \geq \tau)a)(\forall(\geq \phi)) &\to_\iota \Lambda(\alpha \geq \sigma)a\{\phi;!\alpha/!\alpha\} \\
(\forall(\alpha \geq)\phi)^\circ &:= \underline{\lambda} x.\underline{\lambda} i_\alpha.\phi^\circ \triangleright (x \triangleleft i_\alpha) \\
& & (\Lambda(\alpha \geq \tau)a)(\forall(\alpha \geq)\phi) &\to_\iota \Lambda(\alpha \geq \tau)(a\phi)
\end{aligned}$$

Properties

Lemma (soundness)

If $\Gamma \vdash a : \sigma$ *in* $\mathsf{xML}^{\mathsf{F}}$ *then* $\Gamma^{\bullet}; \vdash_{\mathtt{t}} a^{\circ} : \sigma^{\bullet}$ *in* F_{c}

Lemma (simulation)

1. *If* $a \to_{\beta} b$ *then* $a^{\circ} \to_{\beta} b^{\circ}$
2. *if* $a \to_{\iota} b$ *then* $a^{\circ} \to_{\mathsf{c}}^{+} b^{\circ}$

Corollary

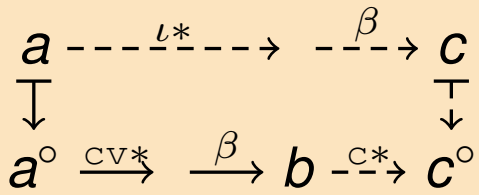
$\mathsf{xML}^{\mathsf{F}}$ *is strongly normalizing*

Lemma

Translation and coercion erasure agree with $\mathsf{xML}^{\mathsf{F}}$*'s type erasure:* $\lfloor a^{\circ} \rfloor = \lceil a \rceil$

Obtaining bisimulation for xML$^{\mathsf{F}}$

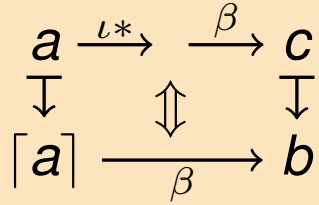
Lemma (lifting)



($\rightarrow_{\mathrm{cv}} \subseteq \rightarrow_{\mathrm{c}}$ is a particular reduction for which bisimulation still holds)

Theorem (weak bisimulation)

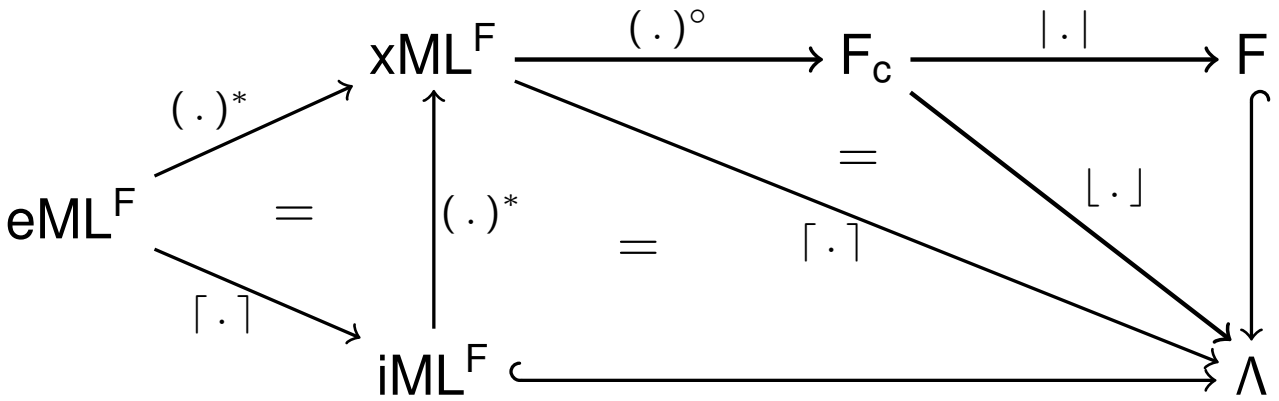
$\lceil a \rceil \rightarrow_\beta b$ *iff* $a \stackrel{*}{\rightarrow}_\iota \rightarrow_\beta c$ *with* $\lceil c \rceil = b$



Corollary

iML$^{\mathsf{F}}$ *and* eML$^{\mathsf{F}}$ *are strongly normalizing*

The big picture



- Enter coercion calculus
- A decoration of system F abstracting uses of type coercions...
- ...having a coercion erasure which is a weak bisimulation (which implies SN of xML^F)...
- ...and implying that xML^F's erasure is a weak bisimulation too (which implies SN of iML^F and eML^F too)

Future work

- Lift unnecessary constraints of F_c
- Handle variant/contravariant arrow coercion
- Check F_c with existing calculi with coercions
- use F_c to prove other conjectures, on ML^F, for example that it types more terms than system F
- can ideas from F_c generalize ML^F?

Thanks

Questions?