A Glimpse at Intersection Types

Pierre VIAL
Équipe Gallinette
Inria - LS2N

September 13, 2019

Non-Idempotent
Gardner 94 - de Carvalho 07

Intersection
Coppo-Dezani 80

Type Theory

Where we lie

Non-Idempotent

Gardner 94 - de Carvalho 07

Intersection

Coppo-Dezani 80

Type Theory → Curry-Howard correspondence

Non-Idempotent

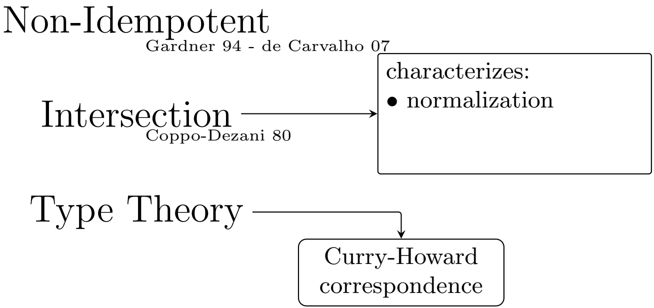
Gardner 94 - de Carvalho 07

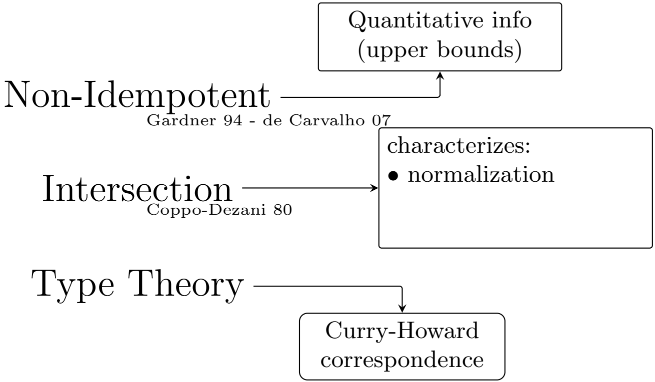
Intersection → characterizes:

- normalization
- complexity classes
- MSO-sat.

Coppo-Dezani 80

Type Theory → Curry-Howard correspondence


Non-Idempotent
Gardner 94 - de Carvalho 07
Intersection
Coppo-Dezani 80
characterizes:
• normalization
Type Theory
Curry-Howard
correspondence


Quantitative info
(upper bounds)
Non-Idempotent
Gardner 94 - de Carvalho 07
characterizes:
• normalization
Intersection
Coppo-Dezani 80
Type Theory
Curry-Howard
correspondence

Quantitative info (upper bounds)

Non-Idempotent → Quantitative info (upper bounds)

Gardner 94 - de Carvalho 07

Intersection → characterizes:

- normalization

Coppo-Dezani 80

Type Theory

Plan

Intersection types (overview)

- Introduced by **Coppo-Dezani** (78-80) to "interpret more terms"
  - Charac. of Weak Norm. for λI-terms (no erasing β-step).
  - Extended later for λ-terms, head, weak or strong normalization...
  - Filter models
- Model-checking
  - *Ong 06:* monadic second order (MSO) logic is decidable for higher-order recursion schemes (HORS)
  - *Kobayashi-Ong 09:* MSO is decidable for higher-order programs
    + using intersection types to simplify Ong's algorithm.
  - Refined by *Grellois-Melliès 14-15*
- Complexity:
  - Upper bounds for reduction sequences (*Gardner 94, de Carvalho 07*) or exact bounds (*Bernadet-Lengrand 11, Accattoli-Lengrand-Kesner, ICFP'18*).
  - *Terui 06:* upper bounds for terms in a red. sequence
  - *De Benedetti-Ronchi della Roccha 16*: characterization of FPTIME

Terminal states and execution/reduction strategies

$$\underbrace{2 + 3 \times 5} \longrightarrow \underbrace{2 + 15} \longrightarrow 17$$

Reducible (non-terminal) states

Terminal state

$$\underbrace{2 + 3 \times 5} \quad \longrightarrow \quad \underbrace{2 + 15} \quad \longrightarrow \quad 17$$

Reducible (non-terminal) states

Terminal state

- Let $f(x) = x \times x \times x$. What is the value of $f(3+4)$?

Terminal states and execution/reduction strategies

$$\underbrace{2 + 3 \times 5} \longrightarrow \underbrace{2 + 15} \longrightarrow 17$$

Reducible (non-terminal) states

Terminal state

- Let $f(x) = x \times x \times x$. What is the value of $f(3+4)$?

Kim (smart)

$$\begin{array}{lll} f(3+4) & \to & f(7) \\ & \to & 7 \times 7 \times 7 \\ & \to & 49 \times 7 \\ & \to & 343 \end{array}$$

Lee (not so)

$$\begin{array}{lll} f(3+4) & \to & (3+4) \times (3+4) \times (3+4) \\ & \to & 7 \times (3+4) \times (3+4) \\ & \to & 7 \times 7 \times (3+4) \\ & \to & 7 \times 7 \times 7 \\ & \to & 49 \times 7 \\ & \to & 343 \end{array}$$

Thurston (don't be Thurston)

$$\begin{array}{lll} f(3+4) & \to & (3+4) \times (3+4) \times (3+4) \\ & \to & 3 \times (3+4) \times (3+4) + 4 \times (3+4) \times (3+4) \\ & \to & \text{dozens of computation steps} \\ & \dots & \dots\dots\dots\dots\dots\dots \\ & \to & 343 \end{array}$$

Terminal states and execution/reduction strategies

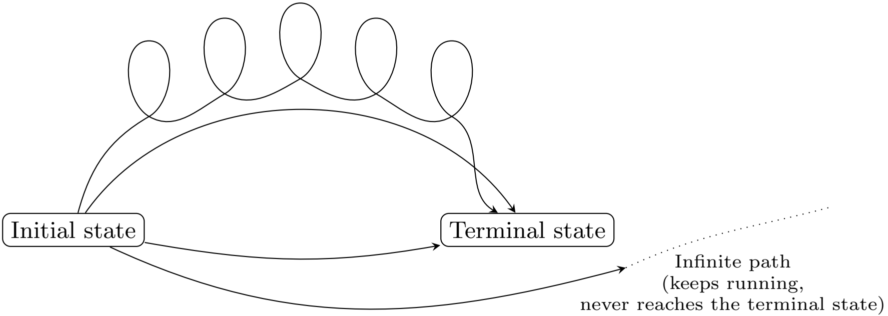
$$\underbrace{2 + 3 \times 5} \longrightarrow \underbrace{2 + 15} \longrightarrow 17$$

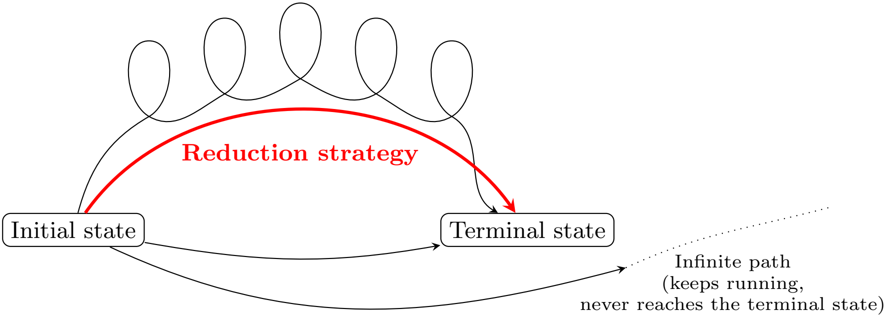
Reducible (non-terminal) states

Terminal state

Terminal states and execution/reduction strategies

Terminal states and execution/reduction strategies





Reduction strategy

- **Choice** of a reduction path.
- Can be **complete** (w.r.t. termin.).
- Must be **certified**.

Intersections types (Coppo, Dezani, 1980)

Goal

Equivalences of the form

"the program t is typable iff it can reach a terminal state"

Idea: **several** certificates to a same subprogram (next slides).

Intersections types (Coppo, Dezani, 1980)

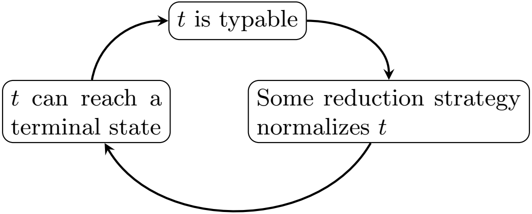
Goal

Equivalences of the form

"the program t is typable iff it can reach a terminal state"

Idea: **several** certificates to a same subprogram (next slides).

Proof: by the "circular" implications:



Intersections types (Coppo, Dezani, 1980)

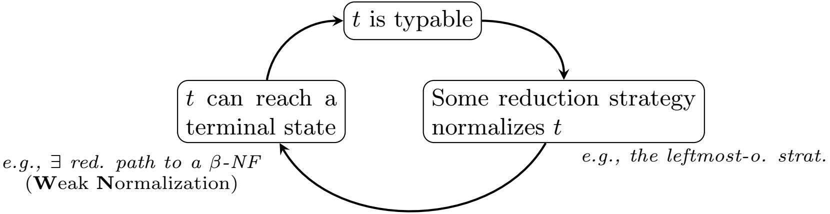
Goal

Equivalences of the form

"the program t is typable iff it can reach a terminal state"

Idea: **several** certificates to a same subprogram (next slides).

Proof: by the "circular" implications:



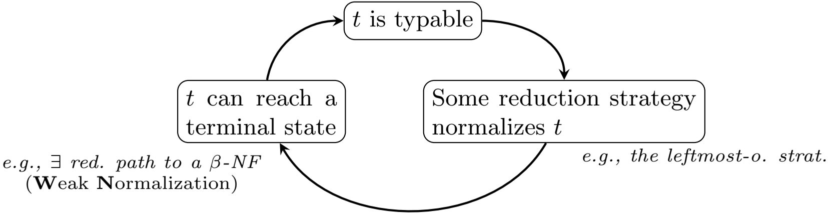
Goal

Equivalences of the form

"the program t is typable iff it can reach a terminal state"

Idea: **several** certificates to a same subprogram (next slides).

Proof: by the "circular" implications:



t is WN iff the leftmost-o. stategy terminates on t

nothing to do with types

Intersections types (Coppo, Dezani, 1980)

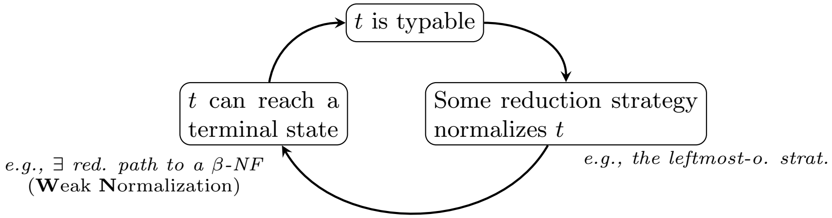
Goal

Equivalences of the form

"the program t is typable iff it can reach a terminal state"

Idea: **several** certificates to a same subprogram (next slides).

Proof: by the "circular" implications:



Intersections types (Coppo, Dezani, 1980)

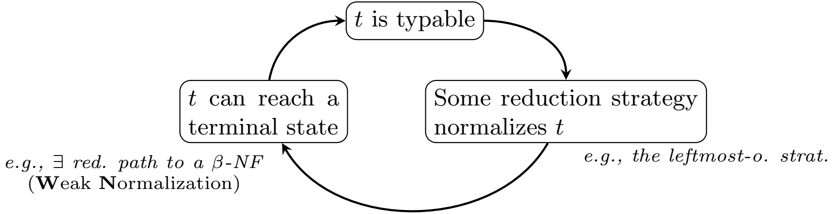
Goal

Equivalences of the form

"the program t is typable iff it can reach a terminal state"

Idea: **several** certificates to a same subprogram (next slides).

Proof: by the "circular" implications:



Intersection types

- Perhaps too expressive...
- ...but certify **reduction strategies**!

- Naively, $A \wedge B$ stands for $A \cap B$:

t is of type $A \wedge B$ if t can be typed with A as well as B.

$$\frac{I : A \to A \qquad I : (A \to B) \to (A \to B)}{I : (A \to A) \wedge ((A \to B) \to (A \to B))} \wedge-\texttt{intro} \qquad \textit{(with } I = \lambda x.x\textit{)}$$

Intuitions (syntax)

- Naively, $A \wedge B$ stands for $A \cap B$:

$$\textit{t is of type } A \wedge B \textit{ if } t \textit{ can be typed with } A \textit{ as well as } B.$$

$$\frac{I : A \to A \qquad I : (A \to B) \to (A \to B)}{I : (A \to A) \wedge ((A \to B) \to (A \to B))} \wedge-\texttt{intro} \quad \textit{(with } I = \lambda x.x\textit{)}$$

- Intersection = kind of *finite polymorphism*.

$$(A \to A) \wedge ((A \to B) \to (A \to B)) = \textbf{double} \text{ instance of } \forall X.X \to X$$

$$\textit{(with } X = A \textit{ and } X = A \to B\textit{)}$$

Intuitions (syntax)

- Naively, $A \wedge B$ stands for $A \cap B$:

  t is of type $A \wedge B$ if t can be typed with A as well as B.

$$\frac{I : A \to A \qquad I : (A \to B) \to (A \to B)}{I : (A \to A) \wedge ((A \to B) \to (A \to B))} \wedge\!-\!\texttt{intro} \qquad \textit{(with } I = \lambda x.x\textit{)}$$

- Intersection = kind of *finite polymorphism.*

$$(A \to A) \wedge ((A \to B) \to (A \to B)) = \textbf{double} \text{ instance of } \forall X.X \to X$$

(with $X = A$ and $X = A \to B$)

- But *less constrained*:

  assigning $x : o \wedge (o \to o') \wedge (o \to o) \to o$ is legal.

  *(**not** an instance of a polymorphic type except $\forall X.X :=$ `False`!)*

A good intersection type system should enjoy:

Subject Reduction (SR):
Typing is stable under reduction.

Subject Expansion (SE):
Typing is stable under anti-reduction.

SE is usually not verified by simple or polymorphic type systems

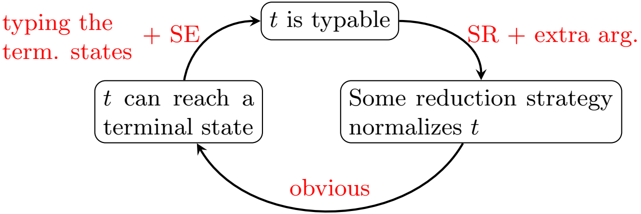
Subject Reduction and Subject Expansion

A good intersection type system should enjoy:

Subject Reduction (SR):
Typing is stable under reduction.

Subject Expansion (SE):
Typing is stable under anti-reduction.

SE is usually not verified by simple or polymorphic type systems

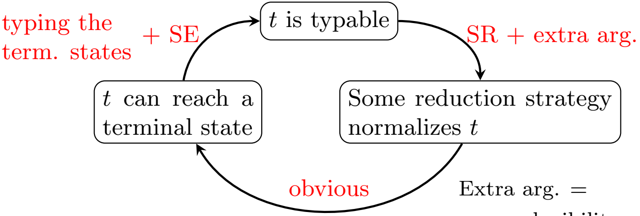


A good intersection type system should enjoy:

Subject Reduction (SR):
Typing is stable under reduction.

Subject Expansion (SE):
Typing is stable under anti-reduction.

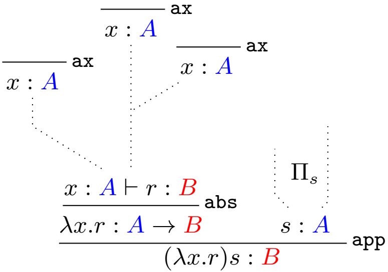
SE is usually not verified by simple or polymorphic type systems



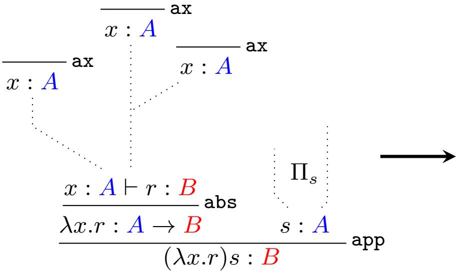
Extra arg. =

- reducibility cand.
- non-trivial well-founded order.
- can it be simpler?

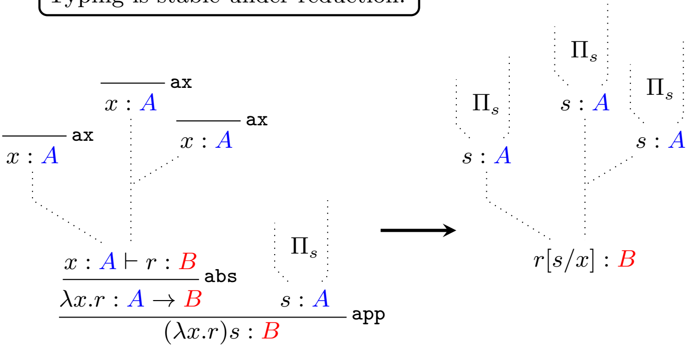
Subject Reduction (SR):
Typing is stable under reduction.



Subject Reduction (SR):
Typing is stable under reduction.

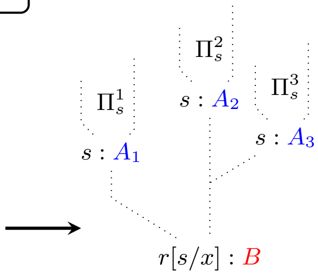


Subject Reduction (SR):
Typing is stable under reduction.

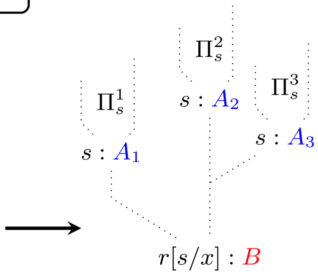


Ensuring Subject Expansion

Subject Expansion (SE):
Typing is stable under anti-reduction.



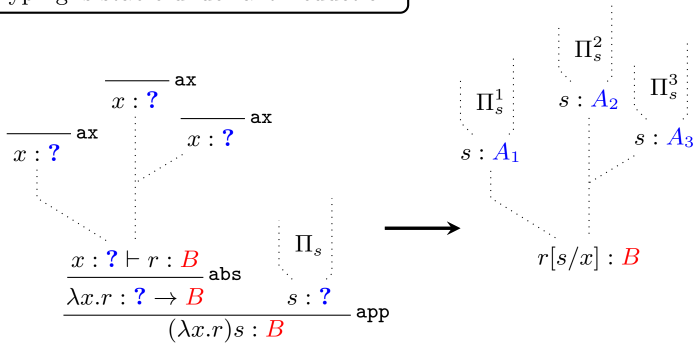
Subject Expansion (SE):
Typing is stable under anti-reduction.



think of $(\lambda x.x\,x)I \to_\beta I\,I$

- Left occ. of I: $(A \to A) \to (A \to A)$
- Right occ. of I: $A \to A$

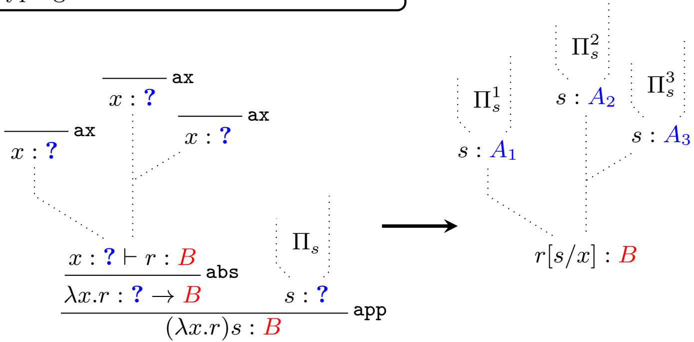
Subject Expansion (SE):
Typing is stable under anti-reduction.



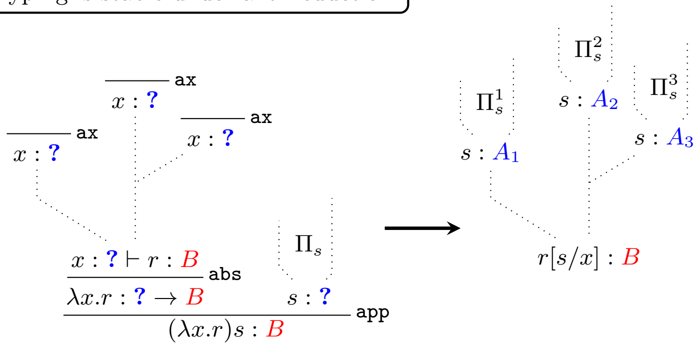
think of $(\lambda x.x\,x)I \to_\beta I\,I$

- Left occ. of I: $(A{\to}A){\to}(A{\to}A)$
- Right occ. of I: $A{\to}A$

Subject Expansion (SE):
Typing is stable under anti-reduction.



Subject Expansion (SE):
Typing is stable under anti-reduction.

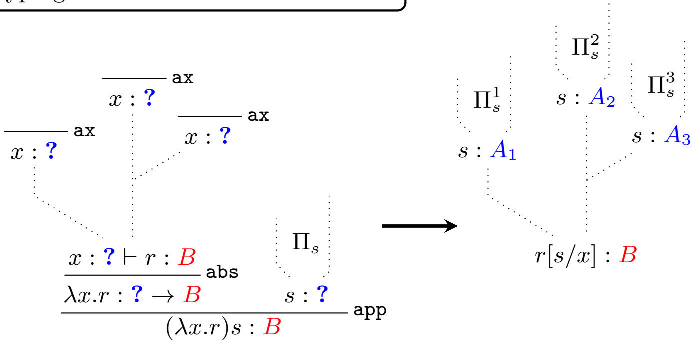


Solution:

- Allow several type assignments for a same variable/subterm

$x : A_1 \wedge A_2 \wedge A_3$

Subject Expansion (SE):
Typing is stable under anti-reduction.



Solution:

- Allow several type assignments for a same variable/subterm

$$x : A_1 \wedge A_2 \wedge A_3 \vdash x : A_i \quad (i = 1, 2, 3)$$

Typing every normal form

- Consider $(y(x\,(\lambda z.z)))\,(x\,(\lambda z.z\,c))$

Typing every normal form

- Consider $(y(x\,(\lambda z.z)))\,(x\,(\lambda z.z\,c))$
- We want $x : E \to F$

Typing every normal form

- Consider $(y(x\,(\lambda z.z)))\,(x\,(\lambda z.z\,c))$
- We want $x : E \to F$
- $\lambda z.z : A \to A$ *vs.* $\lambda z.z\,c : (C \to D) \to D$

Typing every normal form

- Consider $(y(x\,(\lambda z.z)))\,(x\,(\lambda z.z\,c))$
- We want $x : E \to F$
- $\lambda z.z : A \to A$ *vs.* $\lambda z.z\,c : (C \to D) \to D$

$$E = A \to A \text{ or } E = (C \to D) \to D?$$

Typing every normal form

- Consider $(y(x\,(\lambda z.z)))\,(x\,(\lambda z.z\,c))$
- We want $x : E \to F$
- $\lambda z.z : A \to A$ *vs.* $\lambda z.z\,c : (C \to D) \to D$

$$E = A \to A \text{ or } E = (C \to D) \to D?$$

Solution:

- Allow several type assignments for a same variable/subterm

Typing every normal form

- Consider $(y(x\,(\lambda z.z)))\,(x\,(\lambda z.z\,c))$
- We want $x : E \to F$
- $\lambda z.z : A \to A$ *vs.* $\lambda z.z\,c : (C \to D) \to D$

$$E = A \to A \text{ or } E = (C \to D) \to D?$$

Solution:

- Allow several type assignments for a same variable/subterm

- Typing normal form: just structural induction (no clash).

Non-idempotency

Computation causes **duplication**.

Computation causes **duplication**.

Non-idempotent intersection types

Disallow duplication for typing certificates.

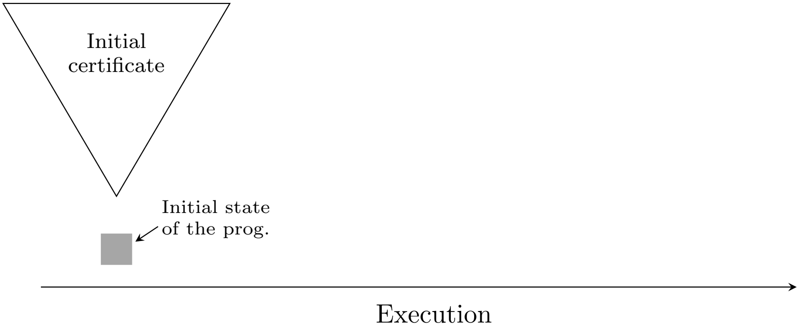
- $\rightsquigarrow$ Possibly many certificates (subderivations) for a subprogram.
- $\rightsquigarrow$ Size of certificates decreases.

Computation causes **duplication**.

Non-idempotent intersection types

Disallow duplication for typing certificates.

- ⇝ Possibly many certificates (subderivations) for a subprogram.
- ⇝ Size of certificates decreases.

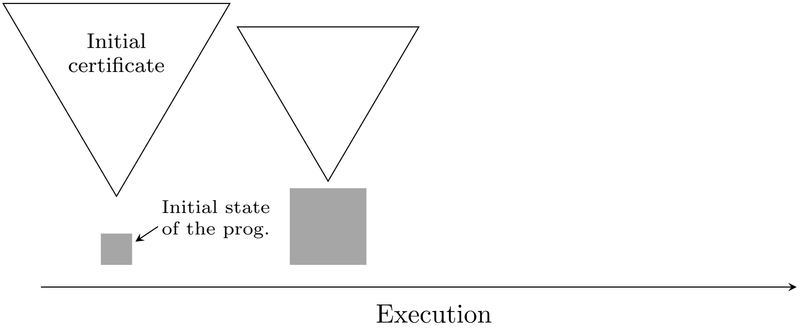


Computation causes **duplication**.

Non-idempotent intersection types

Disallow duplication for typing certificates.

- ⇝ Possibly many certificates (subderivations) for a subprogram.
- ⇝ Size of certificates decreases.

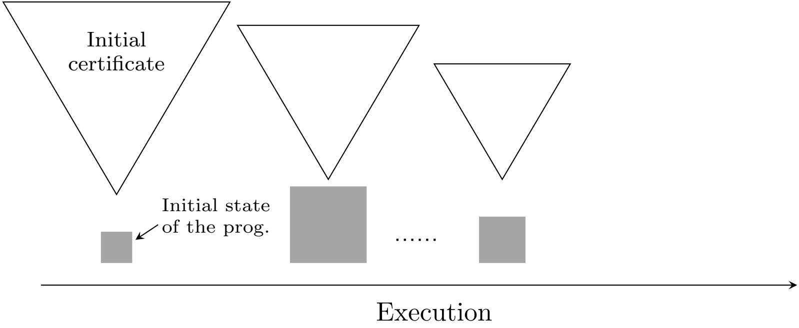


Computation causes **duplication**.

Non-idempotent intersection types

Disallow duplication for typing certificates.

- ⇝ Possibly many certificates (subderivations) for a subprogram.
- ⇝ Size of certificates decreases.

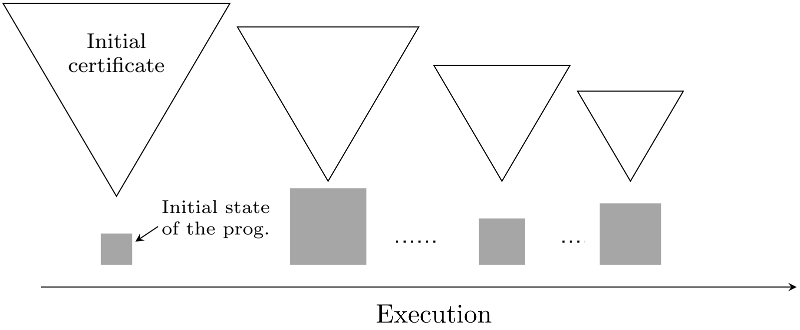


Computation causes **duplication**.

Non-idempotent intersection types

Disallow duplication for typing certificates.

- ⇝ Possibly many certificates (subderivations) for a subprogram.
- ⇝ Size of certificates decreases.

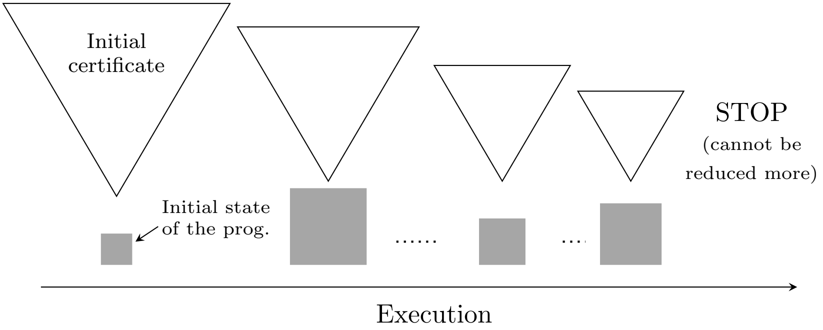


Computation causes **duplication**.

Non-idempotent intersection types

Disallow duplication for typing certificates.

- ⇝ Possibly many certificates (subderivations) for a subprogram.
- ⇝ Size of certificates decreases.

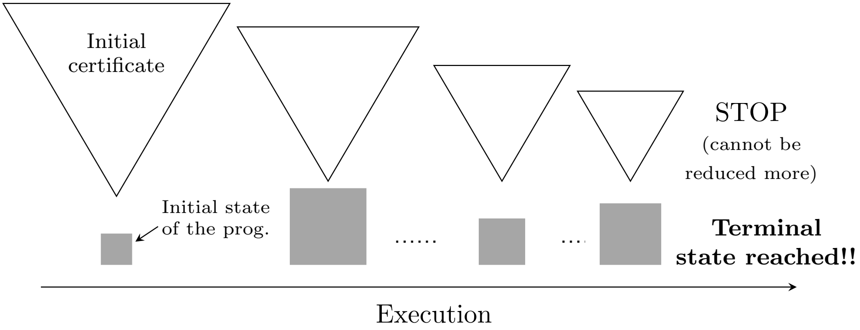


Computation causes **duplication**.

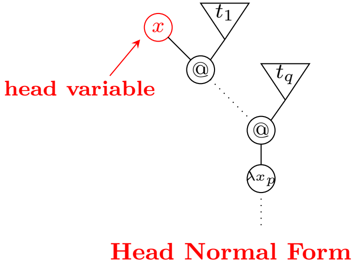
Non-idempotent intersection types

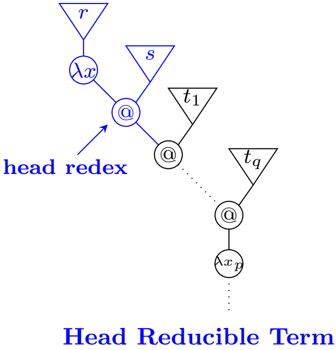
Disallow duplication for typing certificates.

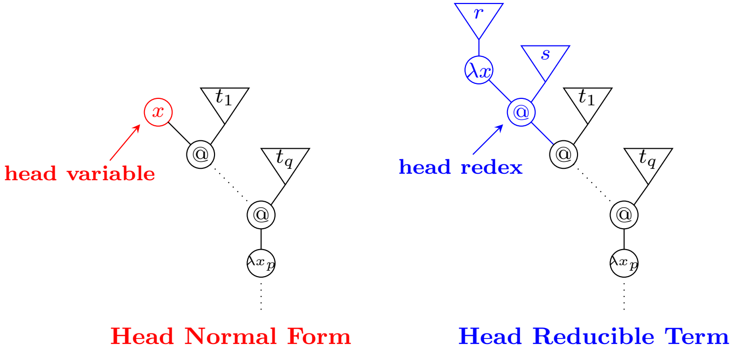
- ⇝ Possibly many certificates (subderivations) for a subprogram.
- ⇝ Size of certificates decreases.



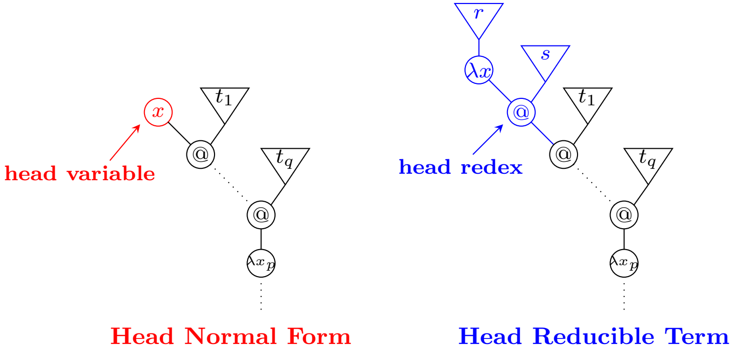
Plan


x
t_1
@
t_q
@
λx_p
head variable
Head Normal Form


r
λx
s
@
t_1
@
t_q
@
λx_p
head redex
Head Reducible Term



- t is **head normalizing (HN)** if $\exists$ reduction path from t to a HNF.

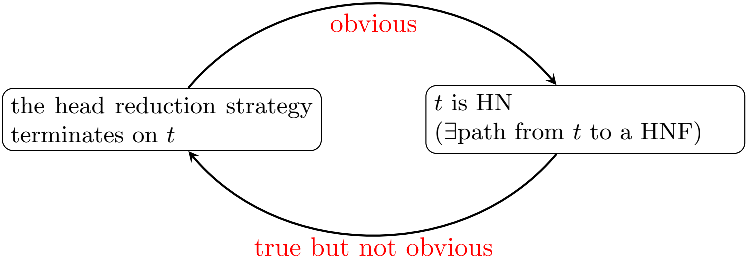


- t is **head normalizing (HN)** if ∃ reduction path from t to a HNF.
- The **head reduction strategy**: reducing head redexes while it is possible.

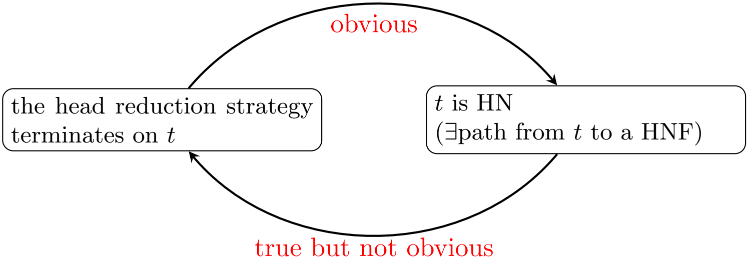
Head Normalization (λ)

- t is **head normalizing (HN)** if $\exists$ reduction path from t to a HNF.
- The **head reduction strategy**: reducing head redexes while it is possible.

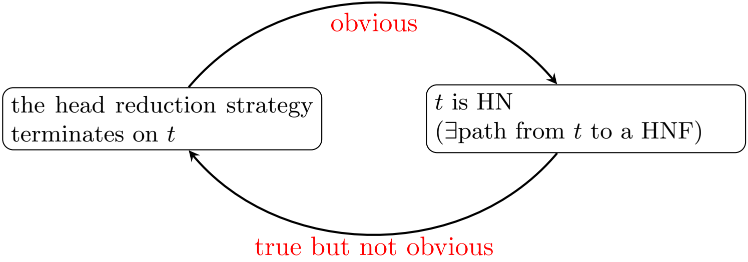
Head Normalization (λ)



- t is **head normalizing (HN)** if $\exists$ reduction path from t to a HNF.
- The **head reduction strategy**: reducing head redexes while it is possible.



- The **head reduction strategy**: reducing head redexes while it is possible.



Intersection types come to help!

- The **head reduction strategy**: reducing head redexes while it is possible.

Intersection types (Coppo-Dezani 80)

- Type constructors: $o \in \mathscr{O}$, $\to$ and $\wedge$ (intersection).

Intersection types (Coppo-Dezani 80)

- Type constructors: $o \in \mathscr{O}$, $\rightarrow$ and $\wedge$ (intersection).

- **Strict types**:
  no inter. on the *right* h.s. of $\rightarrow$, *e.g.*, $(A \wedge B) \rightarrow A$, not $A \rightarrow (B \wedge C)$

  $\rightsquigarrow$ *no intro/elim. rules for* $\wedge$

Intersection types (Coppo-Dezani 80)

- Type constructors: $o \in \mathscr{O}$, $\rightarrow$ and $\wedge$ (intersection).

- **Strict types**:
  no inter. on the *right* h.s. of $\rightarrow$, *e.g.*, $(A \wedge B) \rightarrow A$, not $A \rightarrow (B \wedge C)$

  $\rightsquigarrow$ *no intro/elim. rules for* $\wedge$

Assoc.: $(A \wedge B) \wedge C \sim A \wedge (B \wedge C)$

Comm.: $A \wedge B \sim B \wedge A$

Intersection types (Coppo-Dezani 80)

- Type constructors: $o \in \mathscr{O}$, $\rightarrow$ and $\wedge$ (intersection).

- **Strict types**:
  no inter. on the *right* h.s. of $\rightarrow$, *e.g.*, $(A \wedge B) \rightarrow A$, not $A \rightarrow (B \wedge C)$

  $\rightsquigarrow$ *no intro/elim. rules for* $\wedge$

Assoc.: $(A \wedge B) \wedge C \sim A \wedge (B \wedge C)$

Comm.: $A \wedge B \sim B \wedge A$

Idempotency? $A \wedge A \sim A$

Intersection types (Coppo-Dezani 80)

- Type constructors: $o \in \mathscr{O}$, $\rightarrow$ and $\wedge$ (intersection).

- **Strict types**:
  no inter. on the *right* h.s. of $\rightarrow$, *e.g.*, $(A \wedge B) \rightarrow A$, not $A \rightarrow (B \wedge C)$

  $\rightsquigarrow$ *no intro/elim. rules for* $\wedge$

Assoc.: $(A \wedge B) \wedge C \sim A \wedge (B \wedge C)$

Comm.: $A \wedge B \sim B \wedge A$

Idempotency? $A \wedge A \sim A$

Yes → Coppo-Dezani 80: Typing= *qualitative* info.

No → Gardner 94 - de Carvalho 07: Typing= *quantitative* info.

Intersection types (Coppo-Dezani 80)

- Type constructors: $o \in \mathscr{O}$, $\rightarrow$ and $\wedge$ (intersection).

- **Strict types**:
  no inter. on the *right* h.s. of $\rightarrow$, *e.g.*, $(A \wedge B) \rightarrow A$, not $A \rightarrow (B \wedge C)$

  $\rightsquigarrow$ *no intro/elim. rules for* $\wedge$

Assoc.: $(A \wedge B) \wedge C \sim A \wedge (B \wedge C)$

Comm.: $A \wedge B \sim B \wedge A$

Idempotency? $A \wedge A \sim A$

Yes — Coppo-Dezani 80: Typing= *qualitative* info.

No — Gardner 94 - de Carvalho 07: Typing= *quantitative* info.

- Collapsing $A \wedge B \wedge C$ into $[A, B, C]$ (**multiset**) $\rightsquigarrow$ no need for perm rules etc.

$$A \wedge B \wedge A := [A, B, A] = [A, A, B] \neq [A, B] \qquad [A, B, A] = [A, B] + [A]$$

$$\text{Types:} \quad \tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i\in I} \to \tau$$

- **intersection** = **multiset** of types $[\sigma_i]_{i\in I}$
- only on the left-h.s of $\to$ (*strictness*)

Types: $\tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i \in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i \in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma;\, x : [\sigma_i]_{i \in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i \in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i \in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i \in I}}{\Gamma +_{i \in I} \Gamma_i \vdash t\, u : \tau}\ \texttt{app}$$

System $\mathscr{R}_0$ (Gardner 94-de Carvalho 07)

Types: $\tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i \in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i \in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i \in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i \in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i \in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i \in I}}{\Gamma +_{i \in I} \Gamma_i \vdash t\,u : \tau}\ \texttt{app}$$

Remark

- **Relevant** system (no weakening, *cf.* `ax`-rule)

System $\mathscr{R}_0$ (Gardner 94-de Carvalho 07)

$$\text{Types:} \qquad \tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i\in I} \to \tau$$

- **intersection** = **multiset** of types $[\sigma_i]_{i\in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i\in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i\in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i\in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i\in I}}{\Gamma +_{i\in I} \Gamma_i \vdash t\,u : \tau}\ \texttt{app}$$

Remark

- **Relevant** system (no weakening, *cf.* ax-rule)
- **Non-idempotency ($\sigma \wedge \sigma \neq \sigma$):**
  in app-rule, pointwise multiset sum *e.g.*,

$$(x : [\sigma]; y : [\tau]) + (x : [\sigma, \tau]) = x : [\sigma, \sigma, \tau]; y : [\tau]$$

System $\mathscr{R}_0$ (Gardner 94-de Carvalho 07)

Types: $\tau, \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i \in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i \in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\,\texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i \in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i \in I} \to \tau}\,\texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i \in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i \in I}}{\Gamma +_{i \in I} \Gamma_i \vdash t\,u : \tau}\,\texttt{app}$$

Example

$$\dfrac{\dfrac{}{f : [o] \to o}\,\texttt{ax} \qquad \dfrac{\dfrac{}{f : [o] \to o}\,\texttt{ax} \qquad \dfrac{}{x : o}\,\texttt{ax}}{f\,x : o}\,\texttt{app}}{f(f\,x) : o}\,\texttt{app}$$

System $\mathscr{R}_0$ (Gardner 94-de Carvalho 07)

Types: $\tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i\in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i\in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i\in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i\in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i\in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i\in I}}{\Gamma +_{i\in I} \Gamma_i \vdash t\,u : \tau}\ \texttt{app}$$

Example

$$\dfrac{\dfrac{}{f : [o] \to o}\ \texttt{ax} \qquad \dfrac{\dfrac{}{f : [o] \to o}\ \texttt{ax} \qquad \dfrac{}{x : o}\ \texttt{ax}}{f\,x : o}\ \texttt{app}}{f : [[o] \to o, [o] \to o], x : [o] \vdash f(f\,x) : o}\ \texttt{app}$$

System $\mathscr{R}_0$ (Gardner 94-de Carvalho 07)

Types: $\tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i \in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i \in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i \in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i \in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i \in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i \in I}}{\Gamma +_{i \in I} \Gamma_i \vdash t\, u : \tau}\ \texttt{app}$$

Types: $\tau,\ \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i\in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i\in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i\in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i\in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i\in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i\in I}}{\Gamma +_{i\in I} \Gamma_i \vdash t\,u : \tau}\ \texttt{app}$$

Head redexes always typed!

System $\mathscr{R}_0$ (Gardner 94-de Carvalho 07)

Types: $\quad \tau, \sigma \quad ::= \quad o \quad | \quad [\sigma_i]_{i \in I} \to \tau$

- **intersection** = **multiset** of types $[\sigma_i]_{i \in I}$
- only on the left-h.s of $\to$ (*strictness*)

$$\frac{}{x : [\tau] \vdash x : \tau}\ \texttt{ax} \qquad \frac{\Gamma; x : [\sigma_i]_{i \in I} \vdash t : \tau}{\Gamma \vdash \lambda x.t : [\sigma_i]_{i \in I} \to \tau}\ \texttt{abs}$$

$$\frac{\Gamma \vdash t : [\sigma_i]_{i \in I} \to \tau \qquad (\Gamma_i \vdash u : \sigma_i)_{i \in I}}{\Gamma +_{i \in I} \Gamma_i \vdash t\, u : \tau}\ \texttt{app}$$

Head redexes always typed!

but an arg. may be typed 0 time

- **Weighted Subject Reduction**
  - Reduction preserves types and environments, and...
  - ... *head* reduction strictly decreases the number of nodes of the deriv. tree (`size`).
    (actually, holds for any typed redex)

- **Subject Expansion**
  - Anti-reduction preserves types and environments.

Theorem (de Carvalho)

Let t be a λ-term. Then equivalence between:

1. *t is typable (in $\mathscr{R}_0$)*
2. *t is HN*
3. *the head reduction strategy terminates on t ($\rightsquigarrow$ **certification!**)*

Bonus (quantitative information)

If Π types t, then $\texttt{size}(\Pi)$ bounds the number of **steps** of the head red. strategy on t

Head vs Weak and Strong Normalization

Let t be a λ-term.

- **Head normalization (HN):**
  there is a path from t to a head normal form.

- **Weak normalization (WN):**
  there is *at least one path* from t to a β-**Normal Form** (NF)

- **Strong normalization (SN):**
  there is *no infinite path* starting at t.

$$\text{SN} \Rightarrow \text{WN} \Rightarrow \text{HN}$$

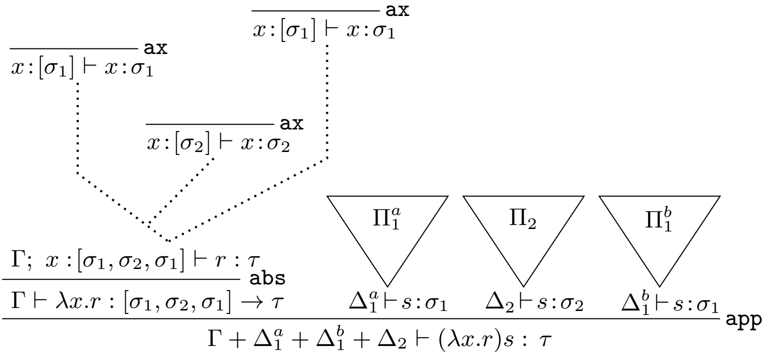
$y\,\Omega$ HNF but not WN

$(\lambda x.y)\Omega$ WN but not SN

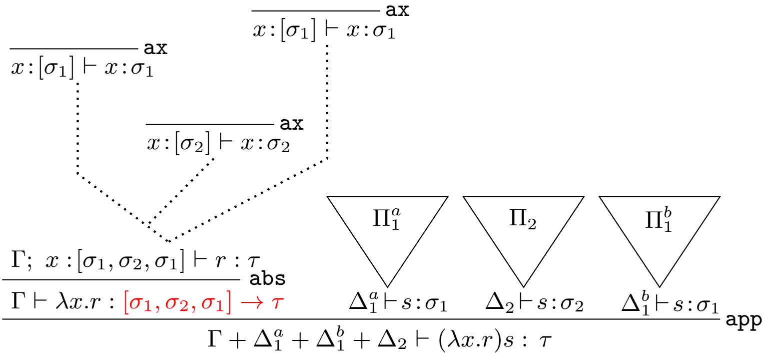
Characterizing Weak and Strong Normalization

HN	System $\mathscr{R}_0$ *any* arg. can be left *untyped*	$\mathtt{sz}(\Pi)$ bounds the number of *head* reduction steps
WN	System $\mathscr{R}_0$ + **unforgetfulness criterion** *non-erasable* args must be typed	$\mathtt{sz}(\Pi)$ bounds the number of leftmost-outermost red. steps (and more)
SN	Modify system $\mathscr{R}_0$ with **choice operator** *all* args must be typed	$\mathtt{sz}(\Pi)$ bounds the length of *any* reduction path

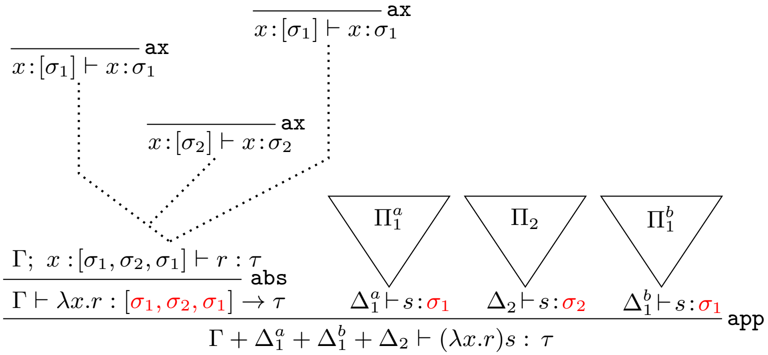
From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$



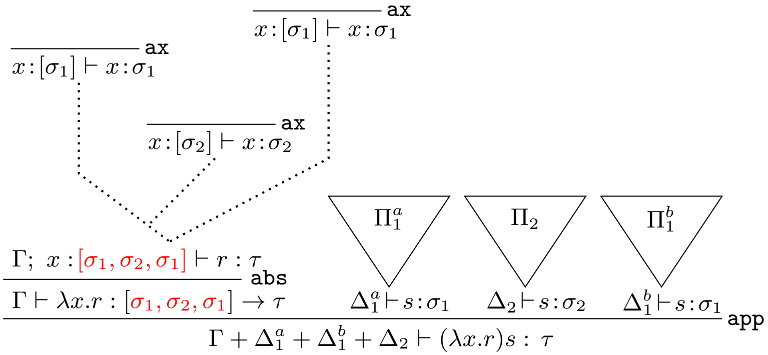
From a typing of $(\lambda x.r)s$ … to a typing of $r[s/x]$



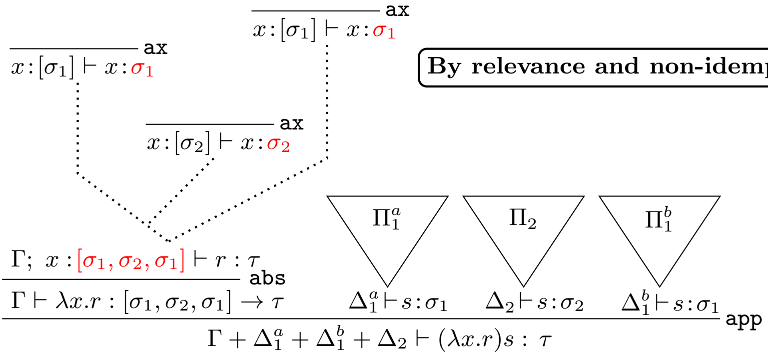
From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$



From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$

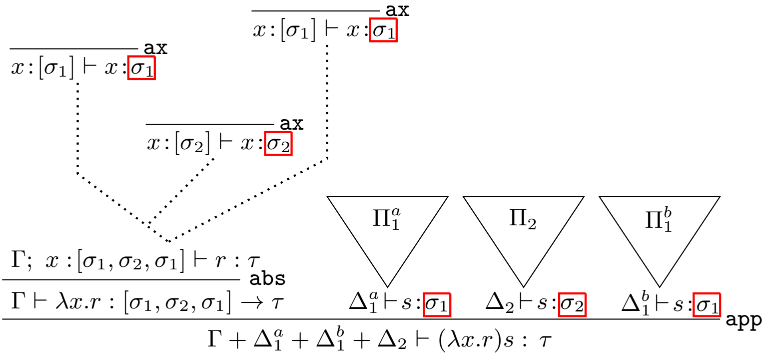


From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$

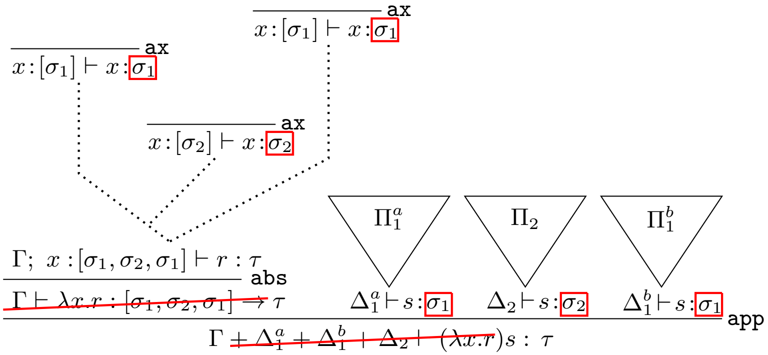


By relevance and non-idempotency!

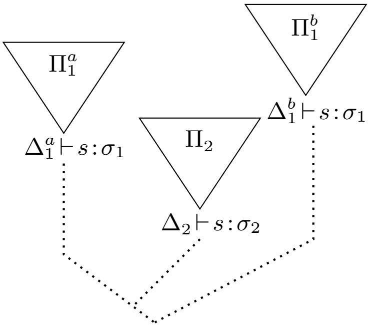
From a typing of $(\lambda x.r)s$ … to a typing of $r[s/x]$



From a typing of $(\lambda x.r)s$ … to a typing of $r[s/x]$

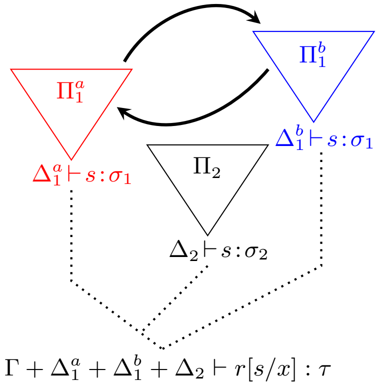


From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$



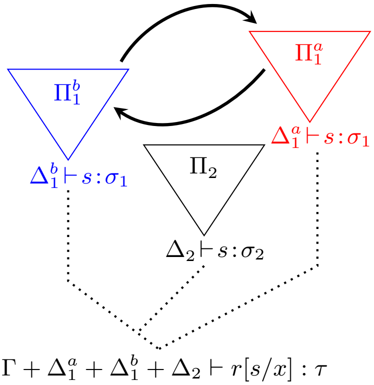
$\Gamma + \Delta_1^a + \Delta_1^b + \Delta_2 \vdash r[s/x] : \tau$

From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$



Non-determinism of SR

From a typing of $(\lambda x.r)s$ ... to a typing of $r[s/x]$

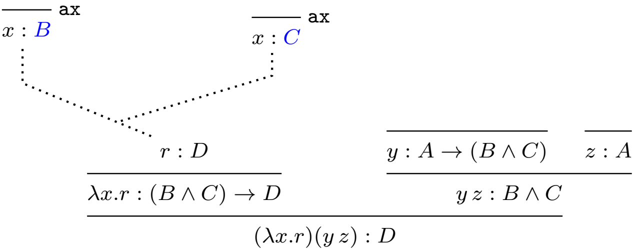


Non-determinism of SR

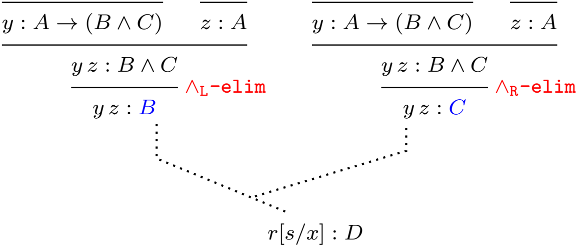
Plan

Non-strictness

$$\dfrac{\dfrac{\begin{array}{cc}\dfrac{}{x : {\color{blue}B}}\,\texttt{ax} & \dfrac{}{x : {\color{blue}C}}\,\texttt{ax} \\ \vdots & \vdots \\ \end{array} \atop r : D}{\lambda x.r : (B \wedge C) \to D} \qquad \dfrac{\dfrac{}{y : A \to (B \wedge C)} \quad \dfrac{}{z : A}}{y\,z : B \wedge C}}{(\lambda x.r)(y\,z) : D}$$


ax
$x : B$
ax
$x : C$
$r : D$
$y : A \to (B \wedge C)$
$z : A$
$\lambda x.r : (B \wedge C) \to D$
$y\,z : B \wedge C$
$(\lambda x.r)(y\,z) : D$

gives


$y : A \to (B \wedge C)$
$z : A$
$y\,z : B \wedge C$
$\wedge_{\mathrm{L}}$-elim
$y\,z : B$
$y : A \to (B \wedge C)$
$z : A$
$y\,z : B \wedge C$
$\wedge_{\mathrm{R}}$-elim
$y\,z : C$
$r[s/x] : D$

Strictness + Relevance + Idempotence

- Two possibles applications rules:

$$\frac{\Gamma \vdash t : \{A_i\}_{i \in I} \to B \quad (\Delta_i \vdash u : A_i)_{i \in I}}{\Gamma \cup (\cup_{i \in I} \Delta_i) \vdash t\,u : B}\,\texttt{app}$$

Arg. redundancy allowed

Strictness + Relevance + Idempotence

- Two possibles applications rules:

$$\frac{\Gamma \vdash t : \{A_i\}_{i\in I} \rightarrow B \quad (\Delta_i \vdash u : A_i)_{i\in I}}{\Gamma \cup (\cup_{i\in I} \Delta_i) \vdash t\,u : B}\ \texttt{app}$$

Arg. redundancy allowed

- Leads to:

$$\dfrac{\dfrac{\begin{array}{c}\dfrac{}{x : \{A\} \vdash x : A}\ \texttt{ax} \\ \vdots \\ \Gamma;\ x : \{A\} \vdash r : B\end{array}}{\Gamma \vdash \lambda x.r : \{A\} \rightarrow B}\ \texttt{abs} \qquad \begin{array}{c}\Pi^a \\ \vdots \\ \Delta^a \vdash s : A\end{array} \qquad \begin{array}{c}\Pi^b \\ \vdots \\ \Delta^b \vdash s : A\end{array}}{\Gamma \cup \Delta^a \cup \Delta^b \vdash (\lambda x.r)s : B}$$

How do we reduce this?

Strictness + Relevance + Idempotence

- Two possibles applications rules:

$$\frac{\Gamma \vdash t : \{A_i\}_{i\in I} \to B \quad (\Delta_i \vdash u : A_i)_{i \in I}}{\Gamma \cup (\cup_{i\in I}\Delta_i) \vdash t\,u : B}\ \texttt{app} \qquad \frac{\Gamma \vdash t : \{A_i\}^{\neq}_{i\in I} \to B \quad (\Delta_i \vdash u : A_i)_{i \in I}}{\Gamma \cup (\cup_{i\in I}\Delta_i) \vdash t\,u : B}\ \texttt{app}_{\neq}$$

Arg. redundancy allowed ...... disallowed

- Leads to:

$$\frac{}{x : \{A\} \vdash x : A}\ \texttt{ax}$$

$$\frac{\dfrac{\Gamma;\ x : \{A\} \vdash r : B}{\Gamma \vdash \lambda x.r : \{A\} \to B}\ \texttt{abs} \qquad \begin{matrix}\Pi^a \\ \vdots \\ \Delta^a \vdash s : A\end{matrix} \qquad \begin{matrix}\Pi^b \\ \vdots \\ \Delta^b \vdash s : A\end{matrix}}{\Gamma \cup \Delta^a \cup \Delta^b \vdash (\lambda x.r)s : B}$$

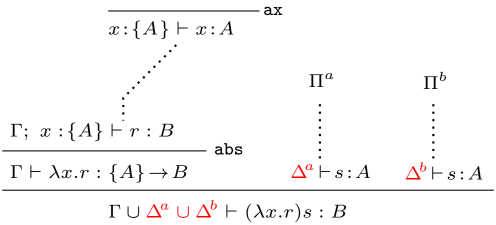
How do we reduce this?

- Two possibles applications rules:

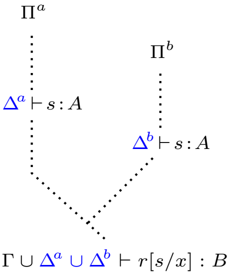
$$\frac{\Gamma \vdash t : \{A_i\}_{i\in I} \to B \quad (\Delta_i \vdash u : A_i)_{i\in I}}{\Gamma \cup (\cup_{i\in I}\Delta_i) \vdash t\,u : B}\ \texttt{app} \qquad \frac{\Gamma \vdash t : \{A_i\}^{\neq}_{i\in I} \to B \quad (\Delta_i \vdash u : A_i)_{i\in I}}{\Gamma \cup (\cup_{i\in I}\Delta_i) \vdash t\,u : B}\ \texttt{app}_{\neq}$$

Arg. redundancy allowed ...... disallowed

- Leads to:



How do we reduce this?



How do we expand this?

Plan

Intersection types *via* Grothendieck construction
[Mazza,Pellissier,V., POPL2018]

- Categorical generalization of ITS. *à la* Melliès-Zeilberger.
- Type systems = 2-operads (see below).

Type systems as 2-operads

- Level 1: $\Gamma \vdash t : B$ — t = *multimorphism* from Γ to B.
- Level 2: if $\Gamma \vdash t : B \overset{\text{SR}}{\rightsquigarrow} \Gamma \vdash t' : B$,
  $t \rightsquigarrow t'$ = *2-morphism* from t to t'.

- Construction of an i.t.s. via a Grothendieck construction (pullbacks).
- **Modularity:** retrieving automatically
  e.g., *e.g.*, Coppo-Dezani, Gardner, $\mathscr{R}_0$, call-by-value + $\mathcal{H}_{\lambda\mu}$ (use *cyclic* 2-operads)

Intersection types **characterize**
various **semantic** properties

+ bring info. **on operational semantics!**

Intersection types **characterize**
various **semantic** properties

+ bring info. **on operational semantics!**

Non-idempotency:
forbid duplication of typing deriv.

Intersection types **characterize**
various **semantic** properties

+ bring info. **on operational semantics!**

Non-idempotency:
forbid duplication of typing deriv.

↓

Simple proof of termination.

typing brings quali. and quanti. info.

Intersection types **characterize**
various **semantic** properties

\+ bring info. **on operational semantics!**

Non-idempotency:
forbid duplication of typing deriv.

Simple proof of termination.

typing brings quali. and quanti. info.

Very simple
operational semantics

Intersection types **characterize**
various **semantic** properties

+ bring info. **on operational semantics!**

Non-idempotency:
forbid duplication of typing deriv.

Simple proof of termination.

typing brings quali. and quanti. info.

Very simple
operational semantics

Adapts to other higher-order calculi
e.g., feat. classical logic

Kesner-V., FSCD'17

Doggy bag

Intersection types **characterize**
various **semantic** properties

+ bring info. **on operational semantics!**

Non-idempotency:
forbid duplication of typing deriv.

→ Simple proof of termination.

typing brings quali. and quanti. info.

→ Very simple
operational semantics

Adapts to other higher-order calculi
e.g., feat. classical logic

Kesner-V., FSCD'17

Adapts to the infinitary calculus

V., LiCS'17

Doggy bag

Intersection types **characterize** various **semantic** properties

+ bring info. **on operational semantics!**

Non-idempotency: forbid duplication of typing deriv.

Simple proof of termination.

typing brings quali. and quanti. info.

Very simple operational semantics

Adapts to other higher-order calculi *e.g., feat. classical logic*

Kesner-V., FSCD'17

Adapts to the infinitary calculus

V., LiCS'17

Upper bounds refine into exact bounds!

Accattoli-Lengrand-Kesner, ICFP'18

Thank you for your attention!

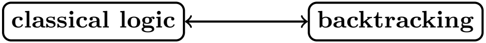
The Lambda-Mu Calculus

- Intuit. logic + Peirce's Law $((A \to B) \to A) \to A$
  gives classical logic.

The Lambda-Mu Calculus

- Intuit. logic + Peirce's Law $((A \to B) \to A) \to A$
  gives classical logic.

- **Griffin 90**: `call–cc` and Felleisen's $\mathcal{C}$-operator typable with Peirce's Law $((A \to B) \to A) \to A$
  $\rightsquigarrow$ the **Curry-Howard** iso extends to classical logic



The Lambda-Mu Calculus

- Intuit. logic + Peirce's Law $((A \to B) \to A) \to A$
  gives classical logic.

- **Griffin 90**: `call–cc` and Felleisen's $\mathcal{C}$-operator typable with Peirce's Law $((A \to B) \to A) \to A$
  $\rightsquigarrow$ the **Curry-Howard** iso extends to classical logic

classical logic ⟷ **backtracking**

- **Parigot 92**: $\lambda\mu$-calculus = computational interpretation of classical *natural deduction* (*e.g., vs.* $\bar{\lambda}\mu\tilde{\mu}$).
  judg. of the form $A, A \to B \vdash A \mid B, C$

Peirce's Law in Classical Natural Deduction

$$\dfrac{\dfrac{}{(A \to B) \to A \vdash (A \to B) \to A} \qquad \dfrac{\dfrac{}{A \vdash A, B}}{\vdash A \to B, A}}{\dfrac{\dfrac{(A \to B) \to A \vdash A, A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A}}$$

Standard Style

$$\dfrac{\dfrac{}{(A \to B) \to A \vdash (A \to B) \to A} \qquad \dfrac{\dfrac{}{A \vdash A, \color{red}{B}}}{\vdash A \to B, A}}{\dfrac{(A \to B) \to A \vdash A, A}{\dfrac{(A \to B) \to A \vdash A}{\vdash ((A \to B) \to A) \to A}}}$$

Standard Style

Peirce's Law in Classical Natural Deduction

Peirce's Law in Classical Natural Deduction

$$\dfrac{\dfrac{}{(A \to B) \to A \vdash (A \to B) \to A \mid} \qquad \dfrac{\dfrac{\dfrac{}{A \vdash A \mid B}}{A \vdash B \mid A}\ \texttt{act}}{\vdash A \to B \mid A}}{\dfrac{(A \to B) \to A \vdash A \mid A}{\dfrac{(A \to B) \to A \vdash A \mid}{\vdash ((A \to B) \to A) \to A \mid}}}$$

Focussed Style

In the right hand-side of $\Gamma \vdash F \mid \Delta$

- 1 active formula F
- inactive formulas Δ

$$\dfrac{\dfrac{\dfrac{\dfrac{}{(A \to B) \to A \vdash (A \to B) \to A \mid} \qquad \dfrac{\dfrac{\dfrac{}{A \vdash A \mid {\color{red}B}}}{A \vdash {\color{red}B} \mid A}\,{\color{red}\texttt{act}}}{\vdash A \to B \mid A}}{(A \to B) \to A \vdash A \mid A}}{(A \to B) \to A \vdash A \mid}}{\vdash ((A \to B) \to A) \to A \mid}$$

Focussed Style

In the right hand-side of $\Gamma \vdash {\color{red}F} \mid {\color{blue}\Delta}$

- 1 active formula F
- inactive formulas Δ

The $\lambda\mu$-calculus

- **Syntax:** λ-calculus

- **Syntax:** λ-calculus
  + names α, β, γ (store inactive formulas)

$$x_1 : D, y : E \vdash t : C \mid \alpha : A, \beta : B$$

- **Syntax:** λ-calculus

  + names α, β, γ (store inactive formulas)

$$x_1 : D, y : E \vdash t : C \mid \alpha : A, \beta : B$$

  + two constructors $[\alpha]t$ (naming) and $\mu\alpha$ (μ-abs.)

  de/activation

The $\lambda\mu$-calculus

- **Syntax:** λ-calculus

  + names α, β, γ (store inactive formulas)

$$x_1 : D, y : E \vdash t : C \mid \alpha : A, \beta : B$$

  + two constructors $[\alpha]t$ (naming) and $\mu\alpha$ (μ-abs.)

  de/activation

- Typed and untyped version

  Simply typable $\Rightarrow$ *SN*

THE $\lambda\mu$-CALCULUS

- **Syntax:** λ-calculus

  + names α, β, γ (store inactive formulas)

  $$x_1 : D, y : E \vdash t : C \mid \alpha : A, \beta : B$$

  + two constructors $[\alpha]t$ (naming) and $\mu\alpha$ (μ-abs.)

  de/activation

- Typed and untyped version

  Simply typable $\Rightarrow$ *SN*

- `call–cc` $:= \lambda y.\mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x)$:

The $\lambda\mu$-calculus

- **Syntax:** λ-calculus

  + names α, β, γ (store inactive formulas)

  $$x_1 : D, y : E \vdash t : C \mid \alpha : A, \beta : B$$

  + two constructors $[\alpha]t$ (naming) and $\mu\alpha$ (μ-abs.)

  de/activation

- Typed and untyped version

  Simply typable $\Rightarrow$ *SN*

- `call–cc` $:= \lambda y.\mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x) : ((A \to B) \to A) \to A$

- **Syntax:** λ-calculus

  $+$ names α, β, γ (store inactive formulas)

  $$x_1 : D, y : E \vdash t : C \mid \alpha : A, \beta : B$$

  $+$ two constructors $[\alpha]t$ (naming) and $\mu\alpha$ (μ-abs.)

  de/activation

- Typed and untyped version

  Simply typable $\Rightarrow$ *SN*

- `call–cc` $:= \lambda y.\mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x) : ((A \to B) \to A) \to A$

How do we adapt the non-idempotent machinery to $\lambda\mu$?

Intersection: $\mathcal{I}, \mathcal{J} := [\mathcal{U}_k]_{k \in K}$

$\mathcal{U}, \mathcal{V} =: \langle \sigma_k \rangle_{k \in K}$: **Union**

Intersection: $\mathcal{I}, \mathcal{J} := [\mathcal{U}_k]_{k \in K}$

$\mathcal{U}, \mathcal{V} =: \langle \sigma_k \rangle_{k \in K}$: **Union**

$$x : [\mathcal{U}_1, \mathcal{U}_2];\ y : [\mathcal{V}] \vdash t : \mathcal{U} \mid \alpha : \langle \sigma_1, \sigma_2 \rangle, \beta : \langle \tau_1, \tau_2, \tau_3 \rangle$$

The Typing System

Intersection: $\mathcal{I}, \mathcal{J} := [\mathcal{U}_k]_{k \in K}$ $\quad\quad$ $\mathcal{U}, \mathcal{V} =: \langle \sigma_k \rangle_{k \in K}$: **Union**

$$x : [\mathcal{U}_1, \mathcal{U}_2];\ y : [\mathcal{V}] \vdash t : \mathcal{U} \mid \alpha : \langle \sigma_1, \sigma_2 \rangle, \beta : \langle \tau_1, \tau_2, \tau_3 \rangle$$

Features

Syntax-direction, relevance, multiplicative rules, **accumulation of typing information**.

Intersection: $\mathcal{I}, \mathcal{J} := [\mathcal{U}_k]_{k\in K}$ $\qquad$ $\mathcal{U}, \mathcal{V} =: \langle \sigma_k \rangle_{k\in K}$: **Union**

$$x : [\mathcal{U}_1, \mathcal{U}_2];\ y : [\mathcal{V}] \vdash t : \mathcal{U} \mid \alpha : \langle \sigma_1, \sigma_2 \rangle, \beta : \langle \tau_1, \tau_2, \tau_3 \rangle$$

Features

Syntax-direction, relevance, multiplicative rules, **accumulation of typing information**.

- app-rule based upon the *admissible* rule of ND:

$$\frac{A_1 \to B_1 \vee \ldots \vee A_k \to B_k \qquad A_1 \wedge \ldots \wedge A_k}{B_1 \vee \ldots \vee B_k} \qquad \left(vs. \frac{\overset{*}{\overline{A \to B \quad A}}}{} B \right)$$

The Typing System

Intersection: $\mathcal{I}, \mathcal{J} := [\mathcal{U}_k]_{k\in K}$ $\qquad$ $\mathcal{U}, \mathcal{V} =: \langle \sigma_k \rangle_{k\in K}$: **Union**

$$x : [\mathcal{U}_1, \mathcal{U}_2];\ y : [\mathcal{V}] \vdash t : \mathcal{U} \mid \alpha : \langle \sigma_1, \sigma_2 \rangle, \beta : \langle \tau_1, \tau_2, \tau_3 \rangle$$

Features

Syntax-direction, relevance, multiplicative rules, **accumulation of typing information**.

- app-rule based upon the *admissible* rule of ND:

$$\frac{A_1 \to B_1 \vee \ldots \vee A_k \to B_k \qquad A_1 \wedge \ldots \wedge A_k}{B_1 \vee \ldots \vee B_k} \qquad \left(vs. \frac{\dfrac{*}{A \to B} \quad A}{B} \right)$$

$$\texttt{call-cc} : [[[A] \to B] \to A] \to \langle A, A \rangle \qquad \text{vs.} \qquad ((A \to B) \to A) \to A$$

The Typing System

Intersection: $\mathcal{I}, \mathcal{J} := [\mathcal{U}_k]_{k \in K}$ $\qquad$ $\mathcal{U}, \mathcal{V} =: \langle \sigma_k \rangle_{k \in K}$: **Union**

$$x : [\mathcal{U}_1, \mathcal{U}_2];\ y : [\mathcal{V}] \vdash t : \mathcal{U} \mid \alpha : \langle \sigma_1, \sigma_2 \rangle, \beta : \langle \tau_1, \tau_2, \tau_3 \rangle$$

Features

Syntax-direction, relevance, multiplicative rules, **accumulation of typing information**.

- app-rule based upon the *admissible* rule of ND:

$$\frac{A_1 \to B_1 \vee \ldots \vee A_k \to B_k \qquad A_1 \wedge \ldots \wedge A_k}{B_1 \vee \ldots \vee B_k} \qquad \left(vs. \frac{\dfrac{*}{A \to B} \quad A}{B} \right)$$

$$\texttt{call-cc} : [[[A] \to B] \to A] \to \langle A, A \rangle \qquad \text{vs.} \qquad ((A \to B) \to A) \to A$$

System $\mathcal{H}_{\lambda\mu}$ (Head Normalization)

- **Weighted Subject Reduction + Subject Expansion**

$$\texttt{size}(\Pi) = \begin{cases} \text{number of nodes of } \Pi \, + \\ \text{size of the } \textbf{type arities} \text{ of all the names of commands } + \\ \textbf{multiplicities} \text{ of arguments in all the } \textbf{app. nodes} \end{cases}$$

- **Weighted Subject Reduction + Subject Expansion**

$$\texttt{size}(\Pi) = \begin{cases} \text{number of nodes of } \Pi + \\ \text{size of the } \textbf{type arities} \text{ of all the names of commands } + \\ \textbf{multiplicities} \text{ of arguments in all the } \textbf{app. nodes} \end{cases}$$

- Characterizes **Head Normalization**

*adaptable to **Strong Normalization***

Theorem [Kesner,V.,FSCD17]:
Let t be a $\lambda\mu$-term. Equiv. between:

- t is $\mathcal{H}_{\lambda\mu}$-typable
- t is HN
- The head red. strategy terminates on t

+ **quantitative info**.

- **Weighted Subject Reduction + Subject Expansion**

$$\texttt{size}(\Pi) = \begin{cases} \text{number of nodes of } \Pi \, + \\ \text{size of the } \textbf{type arities} \text{ of all the names of commands } + \\ \textbf{multiplicities} \text{ of arguments in all the } \textbf{app. nodes} \end{cases}$$

- Characterizes **Head Normalization**

adaptable to ***Strong Normalization***

Theorem [Kesner,V.,FSCD17]:
Let t be a $\lambda\mu$-term. Equiv. between:

- t is $\mathcal{H}_{\lambda\mu}$-typable
- t is HN
- The head red. strategy terminates on t

+ **quantitative info**.

- Small-step version.

Infinitary calculi

- Infinitary λ-trees provide various semantics to the λ-calculus.

  Böhm t. [68 or later], Lévy-Longo t. [77,83], Berarducci t. [96].

Infinitary calculi

- Infinitary λ-trees provide various semantics to the λ-calculus.

  Böhm t. [68 or later], Lévy-Longo t. [77,83], Berarducci t. [96].

- Infinite λ-calculi *Kennaway, Klop, Sleep and de Vries [97]*
  - **7 variants**
  - only **3** have a **good behavior** (partial infinitary confluence), respectively recovering Böhm, L-L and Berar. trees as infinite NF.

- Infinitary λ-trees provide various semantics to the λ-calculus.
  Böhm t. [68 or later], Lévy-Longo t. [77,83], Berarducci t. [96].

- Infinite λ-calculi *Kennaway, Klop, Sleep and de Vries [97]*
  - **7 variants**
  - only **3** have a **good behavior** (partial infinitary confluence),
    respectively recovering Böhm, L-L and Berar. trees as infinite NF.

- Main idea:

Productive terms

- may not terminate…
- …but keep on outputting info. (*e.g.*, sub-HNF)
- *sound* infinite red. sequence

vs.

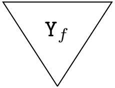
Meaningless terms

- do not output any info. ever (even a head variable)
- unsound infinite red. sequences

Productive vs. Unproductive Reduction

Productive reduction: $\Delta_f := \lambda x.f(xx)$ $\quad$ $\mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

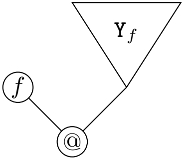
$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^\infty f^\omega$



Productive vs. Unproductive Reduction

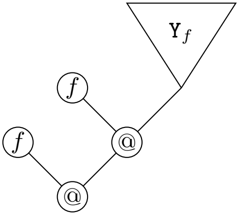
Productive reduction: $\Delta_f := \lambda x.f(xx)$ $\mathtt{Y}_f := \Delta_f\Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^\infty f^\omega$$



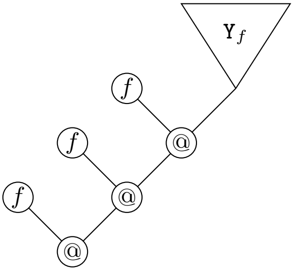
Productive reduction: $\Delta_f := \lambda x.f(xx)$ $\quad$ $\mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^{\infty} f^{\omega}$$



Productive reduction: $\Delta_f := \lambda x.f(xx)$ $\mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

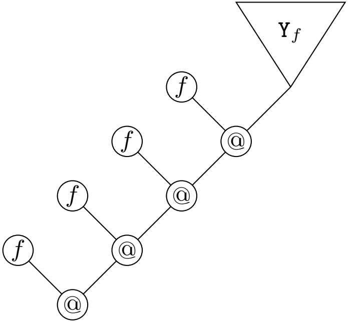
$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^\infty f^\omega$$



Productive vs. Unproductive Reduction

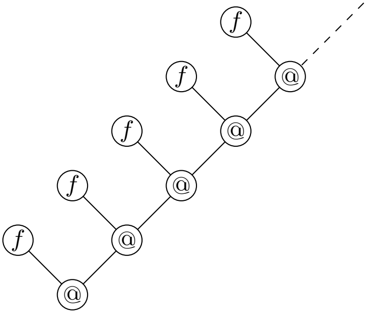
Productive reduction: $\Delta_f := \lambda x. f(xx) \qquad \mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^\infty f^\omega$$



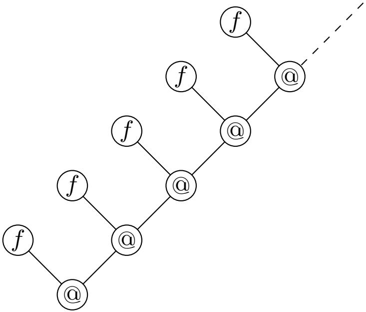
Productive reduction: $\Delta_f := \lambda x.f(xx)$ $\quad\mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^\infty f^\omega$$



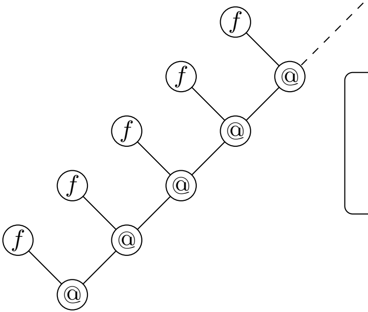
Productive reduction: $\Delta_f := \lambda x.f(xx)$ $\quad$ $\mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^{\infty} f^{\omega}$$



Productive reduction: $\Delta_f := \lambda x.f(xx) \qquad \mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^{\infty} f^{\omega}$$

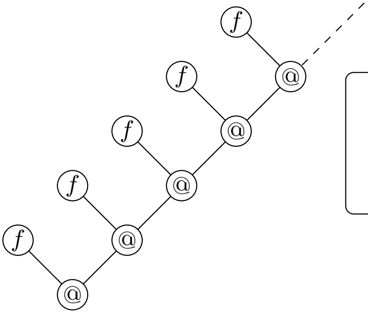


- $\mathtt{Y}_f$ not WN
- $\mathtt{Y}_f$ is ∞-WN
- ∞-NF: $f^{\omega} = f(f^{\omega})$

(Böhm tree)

Productive reduction: $\Delta_f := \lambda x. f(xx)$ $\quad \mathtt{Y}_f := \Delta_f \Delta_f$ "Curry f"

$$\mathtt{Y}_f \to f(\mathtt{Y}_f) \to f^2(\mathtt{Y}_f) \to f^3(\mathtt{Y}_f) \to f^4(\mathtt{Y}_f) \to \ldots \to f^n(\mathtt{Y}_f) \to \ldots \to^\infty f^\omega$$



- $\mathtt{Y}_f$ not WN
- $\mathtt{Y}_f$ is ∞-WN
- ∞-NF: $f^\omega = f(f^\omega)$ *(Böhm tree)*

Unproductive reduction: let $\Delta = \lambda x.x\,x$, $\Omega = \Delta\,\Delta$

$$\Omega \to \Omega \to \Omega \to \Omega \to \Omega \to \Omega \to \ldots$$

- **Klop's Problem:** characterizing ∞-WN with inter. types

- **Klop's Problem:** characterizing ∞-WN with inter. types

  - **Tatsuta [07]:** an inductive ITS cannot do it.
  - Can a coinductive ITS characterize the set of ∞-WN terms?

- **Klop's Problem:** characterizing ∞-WN with inter. types

> - **Tatsuta [07]:** an inductive ITS cannot do it.
> - Can a coinductive ITS characterize the set of ∞-WN terms?

Multiset intersection:

- $\oplus$ syntax-direction
- $\ominus$ non-determinism of proof red.
- $\ominus$ lack **tracking**: $[\sigma, \tau, \sigma] = [\underset{?}{\sigma}, \tau] + [\underset{?}{\sigma}]$.

Klop's Problem and System S

- **Klop's Problem:** characterizing ∞-WN with inter. types

> - **Tatsuta [07]:** an inductive ITS cannot do it.
> - Can a coinductive ITS characterize the set of ∞-WN terms?

Multiset intersection:

- $\oplus$ syntax-direction
- $\ominus$ non-determinism of proof red.
- $\ominus$ lack **tracking**:
  $[\sigma, \tau, \sigma] = [\underset{?}{\sigma}, \tau] + [\underset{?}{\sigma}]$.

Retrieving soundness

- coind. type grammars
  $\rightsquigarrow$ **unsoundness** (Ω typable)
- using a validity criterion
  $\rightsquigarrow$ Need for tracking

Klop's Problem and System S

- **Klop's Problem:** characterizing ∞-WN with inter. types

> - **Tatsuta [07]:** an inductive ITS cannot do it.
> - Can a coinductive ITS characterize the set of ∞-WN terms?

Multiset intersection:

- $\oplus$ syntax-direction
- $\ominus$ non-determinism of proof red.
- $\ominus$ lack **tracking**: $[\sigma, \tau, \sigma] = [\underset{?}{\sigma}, \tau] + [\underset{?}{\sigma}]$.

Retrieving soundness

- coind. type grammars $\rightsquigarrow$ **unsoundness** (Ω typable)
- using a validity criterion $\rightsquigarrow$ Need for tracking

- Solution: **sequential** intersection

> **System S**
> $\rightsquigarrow$ replace $[\sigma_i]_{i\in I}$ with $(k \cdot \sigma_k)_{k\in K}$

- **Tracking:** $(3 \cdot \sigma, 5 \cdot \tau, 9 \cdot \sigma) = (3 \cdot \sigma, 5 \cdot \tau) \uplus (9 \cdot \sigma)$

Proposition

In System $\mathtt{S}$:

- Validity (aka *approximability*) can be defined.
- *SR:* typing is stable by productive ∞-reduction.
- *SE:* *approximable* typing stable by productive ∞-expansion.

Theorem (V,LiCS'17)

- *A ∞-term t is ∞-WN iff t is unforgetfully typable by means of an approximable derivation* $\rightsquigarrow$ *Klop's Problem solved*
- *The hereditary head reduction strategy is complete for infinitary weak normalization.*

Characterization of infinitary WN

Proposition

In System S:

- Validity (aka *approximability*) can be defined.
- *SR:* typing is stable by productive ∞-reduction.
- *SE: approximable* typing stable by productive ∞-expansion.

Theorem (V,LiCS'17)

- *A ∞-term t is ∞-WN iff t is unforgetfully typable by means of an approximable derivation* $\rightsquigarrow$ *Klop's Problem solved*
- *The hereditary head reduction strategy is complete for infinitary weak normalization.*

Bonus: positive answer to TLCA Problem #20

System S also provides a type-theoretic characterization of the **hereditary permutations** (not possible in the inductive case, Tatsuta [LiCS'07]).

Confluence in the infinitary calculi

- In the infinitary calculi:

confluence
only up to the collapsing of the meaningless terms

Confluence in the infinitary calculi

- In the infinitary calculi:

confluence
only up to the collapsing of the meaningless terms

- Let $\mathtt{Y}_I = (\lambda x.I(x\,x))(\lambda x.I(x\,x))$

$$\begin{array}{ccccccccc} \mathtt{Y}_I & \to & I(\mathtt{Y}_I) & \to & \dots & \to & I^n(\mathtt{Y}_I) & \to^\infty & I^\omega \\ \downarrow_2 \\ \Omega \end{array}$$

- In the infinitary calculi:

confluence
only up to the collapsing of the meaningless terms

- Let $\mathtt{Y}_I = (\lambda x.I(x\,x))(\lambda x.I(x\,x))$

$$\begin{array}{ccccccccc} \mathtt{Y}_I & \to & I(\mathtt{Y}_I) & \to & \dots & \to & I^n(\mathtt{Y}_I) & \to^\infty & I^\omega \\ \downarrow_2 & & & & & & & & \\ \Omega & & & & & & & & \end{array}$$

- Structure of proofs *Kennaway et al. 96, Czjaka 14*
    - Using an intermediary calculi ε satisfying confluence.

Confluence in the infinitary calculi

- In the infinitary calculi:

confluence
only up to the collapsing of the meaningless terms

- Let $\mathtt{Y}_I = (\lambda x.I(x\,x))(\lambda x.I(x\,x))$

$$\begin{array}{ccccccccc} \mathtt{Y}_I & \to & I(\mathtt{Y}_I) & \to & \ldots & \to & I^n(\mathtt{Y}_I) & \to^\infty & I^\omega \\ \downarrow_2 & & & & & & & & \\ \Omega & & & & & & & & \end{array}$$

- Structure of proofs *Kennaway et al. 96, Czjaka 14*
  - Using an intermediary calculi ε satisfying confluence.
  - Translating the red. sequences of the ∞-calculi into the ε-calc *via* technical lemmas of the form:

Lemma: if $t \to_\infty t'$ HNF, then $t \to_{\mathtt{h}}^* t'_0$ HNF (finite sequence)

- In the infinitary calculi:

confluence
only up to the collapsing of the meaningless terms

- Let $\texttt{Y}_I = (\lambda x.I(x\,x))(\lambda x.I(x\,x))$

$$\begin{array}{ccccccccccc} \texttt{Y}_I & \to & I(\texttt{Y}_I) & \to & \ldots & \to & I^n(\texttt{Y}_I) & \to^\infty & I^\omega \\ \downarrow_2 & & & & & & & & \\ \Omega & & & & & & & & \end{array}$$

- Structure of proofs *Kennaway et al. 96, Czjaka 14*
  - Using an intermediary calculi ε satisfying confluence.
  - Translating the red. sequences of the ∞-calculi into the ε-calc *via* technical lemmas of the form:

Lemma: if $t \to_\infty t'$ HNF, then $t \to_{\mathtt{h}}^* t'_0$ HNF (finite sequence)

Can *inductive* non-idem. inter. type systems help simplify proofs of infinitary confluence?