Intuitive proofs of termination

Jean-Jacques Lévy

Iscas, 2014-04-06

Plan

- Playing with multiset orderings
- First-order termination
- Higher-order typed λ-calculus
- Girard's proof for strong normalization
- Redex creation and strong normalization
- Finite developments
- Todo list

今天，小菜一碟

.. joint work with Giulio Manzonetto ..

Termination

By Rob Knies
August 16, 2006

”Turing proved that, in general, proving program termination is ‘undecidable,’ ” Cook says. “However, this result does not preclude the existence of future program-termination proof tools that work 99.9 percent of the time on programs written by humans. This is the sort of tool that we’re aiming to make.”

Byron Cook’ Terminator program

[SLAM/Static Driver Verifier (SDV) project]

Well-founded orderings

- no infinite strictly decreasing chains

$(\mathbf{N}, <)$

$(\mathbf{N} \times \mathbf{N},\ <_{lex})$

$(\mathbf{N} \times \mathbf{N} \times \mathbf{N},\ <_{lex})$

$(\ 2^{\mathbf{N}} = \mathsf{set}(\mathbf{N})\ ,\ <_{mset})$

$(\ \mathbf{N}^{\mathbf{N}} = \mathsf{multiset}(\mathbf{N})\ ,\ <_{mset})$

Well-quasi orderings

- any infinite sequence $\langle x_1, x_2, \ldots x_i, \ldots \rangle$ contains $x_i \leq x_j$ for some $i < j$

$(\ \mathbf{F}\ ,\ =)$ where $\mathbf{F}$ is finite set

Multiset ordering (1/2)

$$\{\ 3,\ 5,\ 7,\ 4,\ 1,\ 6,\ 5\ \}$$

$>_{mset}$

$$\{\ 3,\ 5,\ 7,\ 3,\ 2,\ 3,\ 1,\ 6,\ 5\ \}$$

$>_{mset}$

$$\{\ 3,\ 5,\ 7,\ 3,\ 2,\ 3,\ 1,\ 1,\ 4,\ 1,\ 5,\ 5\ \}$$

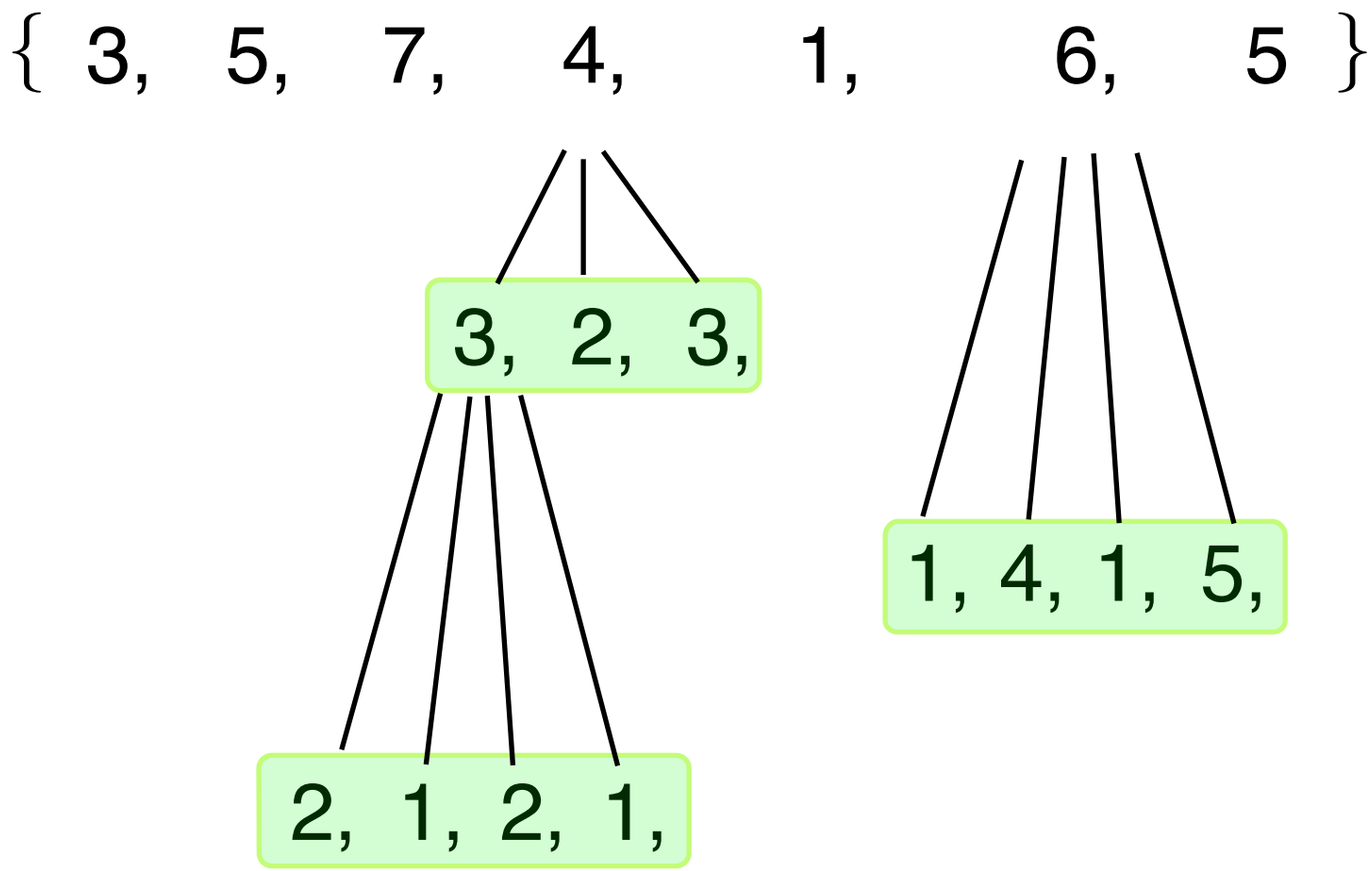
$>_{mset}$

$$\{\ 3,\ 5,\ 7,\ 3,\ 2,\ 3,\ 1,\ 1,\ 4,\ 1,\ 5\ \}$$

$>_{mset}$

$$\{\ 3,\ 5,\ 7,\ 2,\ 1,\ 2,\ 1,\ 2,\ 3,\ 1,\ 1,\ 4,\ 1,\ 5\ \}$$

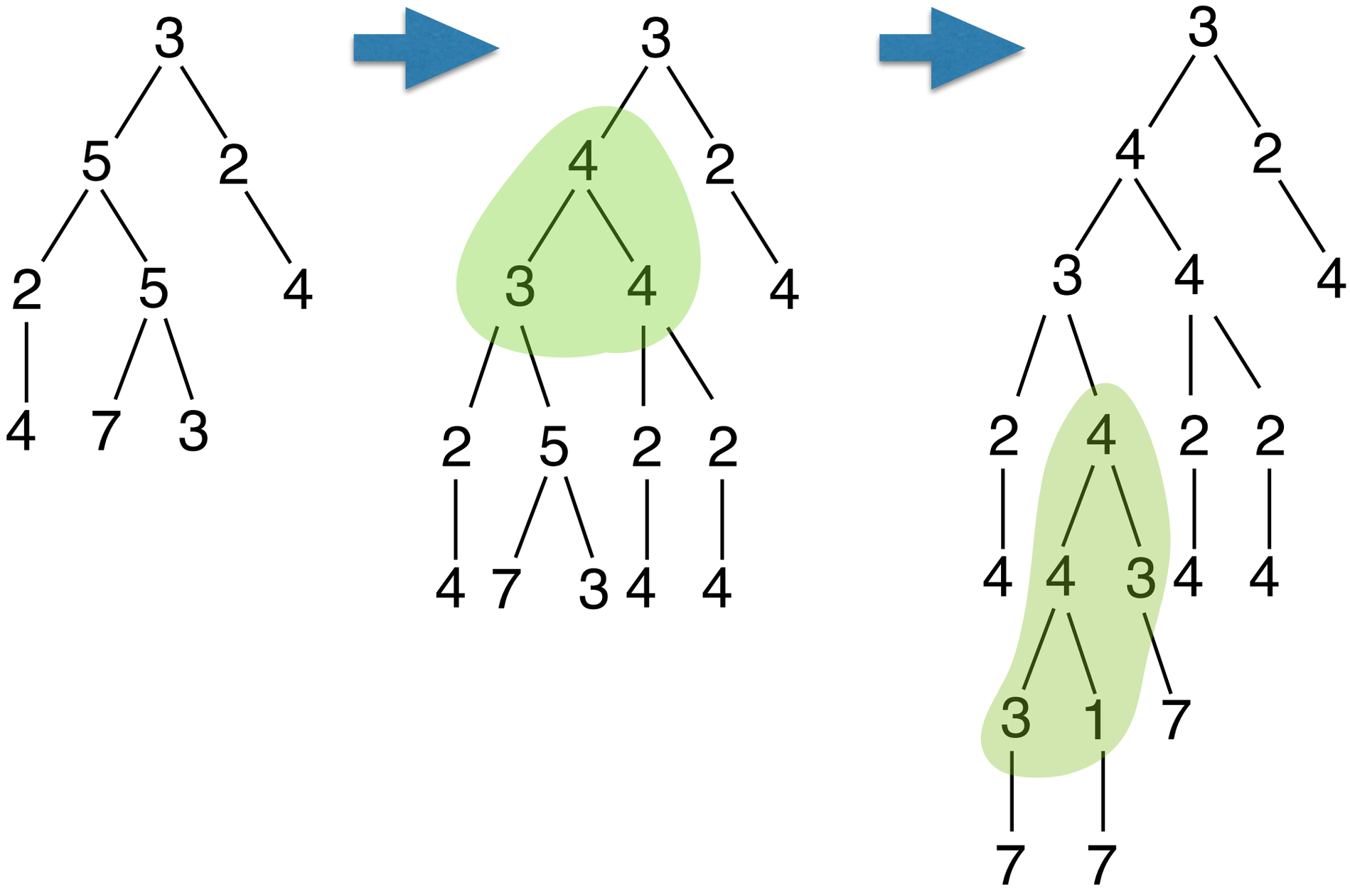
Multiset ordering (2/2)



- well founded by Koenig's lemma

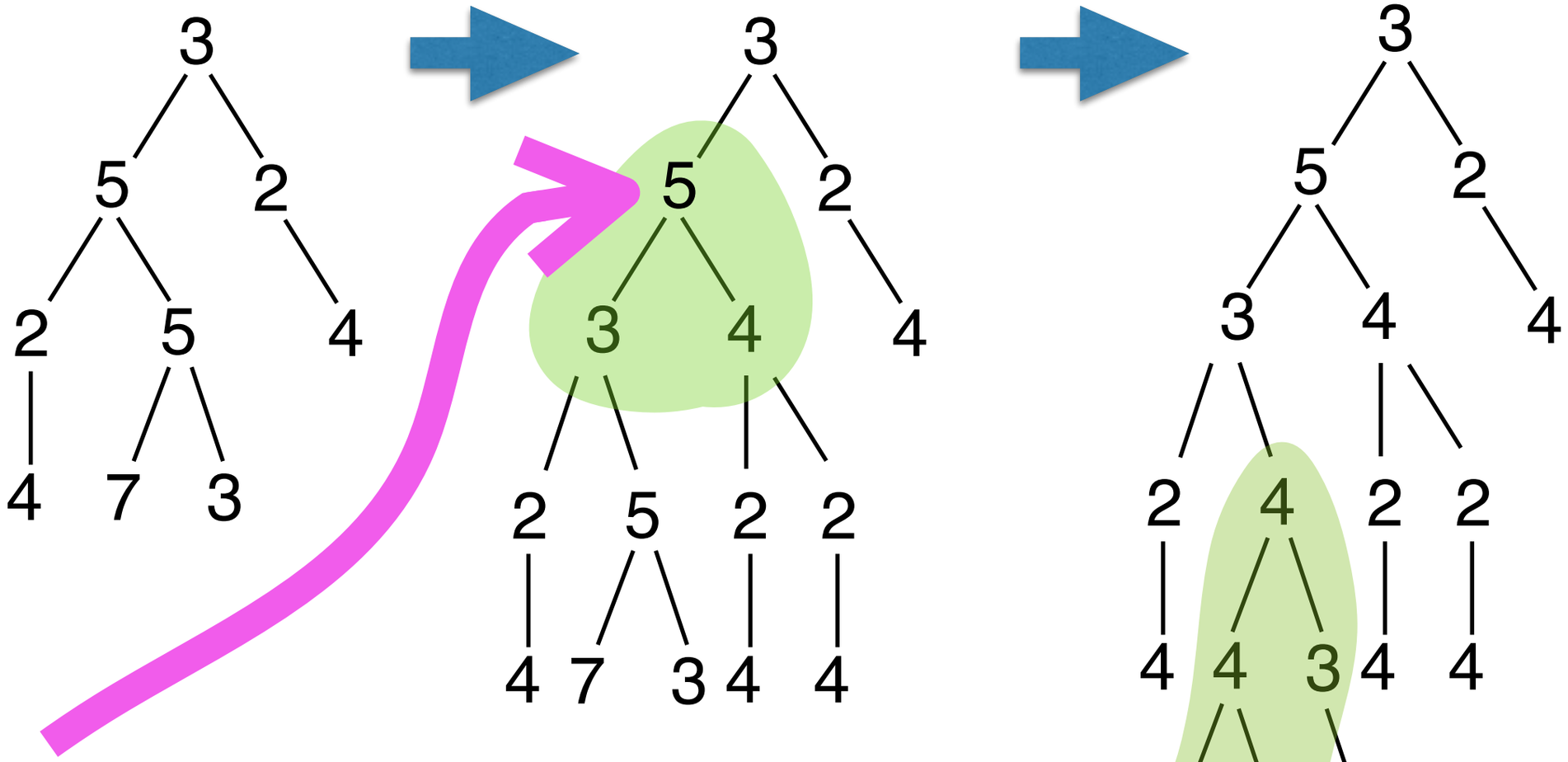
Recursive path orderings (1/5)

- we now order trees



Recursive path orderings (2/5)

- even with replacing nodes with same weight



extra condition about
decreasing of subtrees

Recursive path orderings (3/5)

[Dershowitz, 81]

$t = f(t_1, t_2, \ldots t_n) > g(u_1, u_2, \ldots u_m) = u$

iff

(1) $\#f > \#g$ and $t > u_j$ forall j

(2) $t_i \geq u$ for some i

(3) $\#f = \#g$ and $(t_1, t_2, \ldots t_n) >_{lex,mset} (u_1, u_2, \ldots u_m)$
and $t > u_j$ forall j

Recursive path orderings (4/5)

```
fact (n) = if n = 0 then 1 else n * fact (n - 1)
```

$\#\mathsf{fact} = 1$
$\#\mathsf{if} = \#(-) = \#(*) = \#(=) = 0$

```
ack (m, n) = if m = 0 then n+1 else
             if n = 0 then ack (m-1, 1) else
             ack (m-1, ack(m, n-1))
```

$\#\mathsf{ack} = 1$
$\#\mathsf{if} = \#(-) = \#(=) = 0$

- termination whatever is the evaluation strategy

  (Strong Normalization)

Recursive path orderings (5/5)

```
fact (n) -> if n = 0 then 1 else n * fact (n - 1)
n+1 - 1 = n
n*m -> p
```
where $p = m \times n$
```
if true then P else Q -> P
if false then P else Q -> Q
n+1 = 0 -> false
0 = 0 -> true
```

$\#\mathsf{fact} = 2$
$\#\mathit{if} = \#(*) = \#(= 0) = 1$
$\#n = \#\mathit{true} = \#\mathit{false} = 0$

Recap

- replace **old** objects by finitely **new** smaller ones
- if none new, old ones finish by disappear
- we did not see **proofs**
- r.p.o is particular case of '**simplification** orderings'
- elegant proof uses **Kruskal's** embedding theorem

Higher order

Functional languages

- functions as arguments or result of functions
- the simplest paradigm is Church' λ-calculus

$$
\begin{array}{lcll}
M, N & ::= & x & \text{variable} \\
& | & \lambda x.M & \text{abstraction} \\
& | & MN & \text{application}
\end{array}
$$

1st&2nd-order typing rules

(variable) $$\frac{}{\Gamma, x{:}\tau \vdash x{:}\tau}$$

(application) $$\frac{\Gamma \vdash M : \sigma \to \tau \quad \Gamma \vdash N{:}\sigma}{\Gamma \vdash MN : \tau}$$

(abstraction) $$\frac{\Gamma, x{:}\sigma \vdash M{:}\tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$

(1st-order)

$$\frac{\Gamma \vdash M : \forall\alpha.\tau}{\Gamma \vdash M : \tau\{\alpha := \sigma\}}$$

$$\frac{\Gamma \vdash M{:}\tau \quad \alpha \notin \mathsf{TVar}(\Gamma)}{\Gamma \vdash M : \forall\alpha.\tau}$$

(2nd-order)

Typed functional languages

- at 1st-order:

  $\lambda x.x \;:\; \alpha \to \alpha$

  $\lambda x.\lambda y.x \;:\; \alpha \to \beta \to \alpha$

  $\lambda f.\lambda g.\lambda x.\, g(f\, x) \;:\; (\alpha \to \beta) \to (\beta \to \gamma) \to \alpha \to \gamma$

  $\lambda x.\, x\, x$ is not typed

- at 2nd-order:

  $\lambda x.\, x\, x \;:\; (\forall \alpha.\, \alpha \to \alpha) \to (\forall \alpha.\, \alpha \to \alpha)$

  $(\lambda x.\, x\, x)(\lambda x.\, x\, x)$ is not typed

Higher-order rules

$$\text{(var)}\quad \frac{\Gamma \vdash A : s}{\Gamma, x{:}A \vdash x{:}A}$$

$$\text{(weak)}\quad \frac{\Gamma \vdash A : B \quad \Gamma \vdash C : s}{\Gamma,\ x{:}C \vdash A : B}$$

$$\text{(app)}\quad \frac{\Gamma \vdash M : \Pi x{:}A.\ B \quad \Gamma \vdash a{:}A}{\Gamma \vdash Ma : B\{x := a\}}$$

$$\text{(abs)}\quad \frac{\Gamma, x{:}A \vdash b{:}B \quad \Gamma \vdash \Pi x{:}A.\ B : s}{\Gamma \vdash \lambda x{:}A.\ b : \Pi x{:}A.\ B}$$

$$\text{(const)}\quad \vdash * : \Box$$

$$\text{(prod)}\quad \frac{\Gamma \vdash A{:}s_1 \quad \Gamma,\ x{:}A \vdash B{:}s_2}{\Gamma \vdash \Pi x{:}A.\ B : s_2}$$

$$\text{(conv)}\quad \frac{\Gamma \vdash A{:}B \quad B =_\beta B' \quad \Gamma \vdash B'{:}s}{\Gamma \vdash A{:}B'}$$

Usual sorts for PTS

where (s_1, s_2) possible values are:

$\lambda\!\!\rightarrow$	$(*,*)$			
$\lambda 2$	$(*,*)$	$(\Box,*)$		
$\lambda \mathrm{P}$	$(*,*)$		$(*,\Box)$	
$\lambda \mathrm{P}2$	$(*,*)$	$(\Box,*)$	$(*,\Box)$	
$\lambda\underline{\omega}$	$(*,*)$			$(\Box,\Box)$
$\lambda\omega$	$(*,*)$	$(\Box,*)$		$(\Box,\Box)$
$\lambda \mathrm{P}\underline{\omega}$	$(*,*)$		$(*,\Box)$	$(\Box,\Box)$
$\lambda \mathrm{P}\omega{=}\lambda \mathrm{C}$	$(*,*)$	$(\Box,*)$	$(*,\Box)$	$(\Box,\Box)$

and usual abbrevs

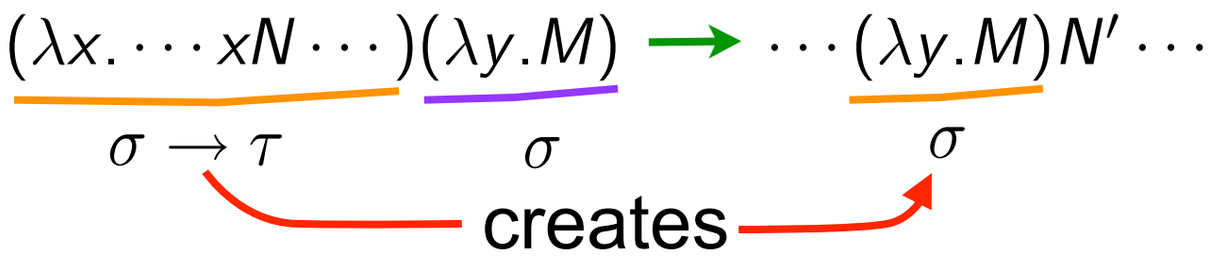
$$\forall\alpha.A \equiv \Pi\alpha{:}*\,.\,A$$

$$\Lambda\alpha.M \equiv \lambda\alpha{:}*\,.\,M$$

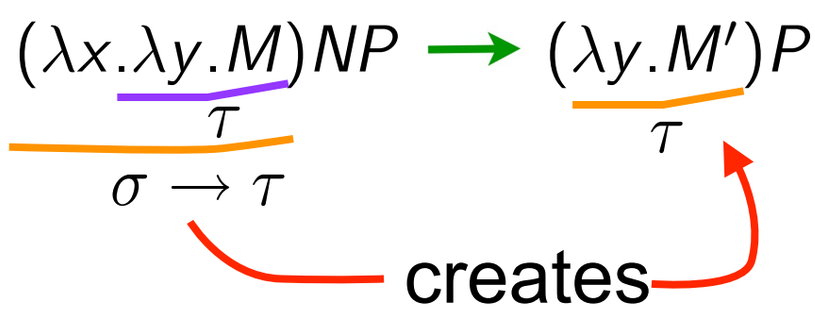
$$A \to B \equiv \Pi x{:}A.\,B \quad \textbf{when} \quad x \notin \mathsf{FVar}(B)$$

Strong Normalisation (1st order)

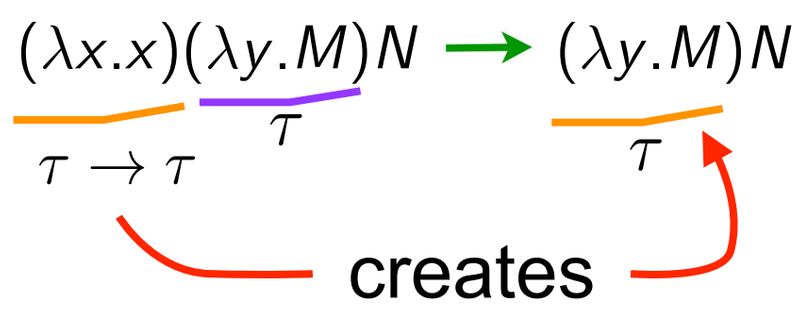
- why 1st-order typed calculus normalizes ?



creation downward



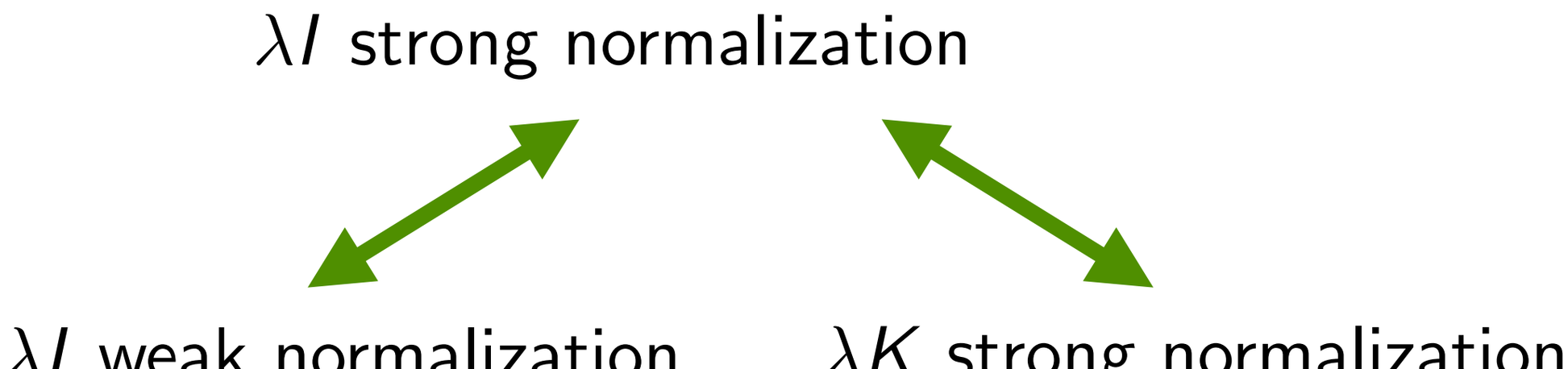
creation upward



- degree of a redex is type of its function part

- degree strictly decreases with creation

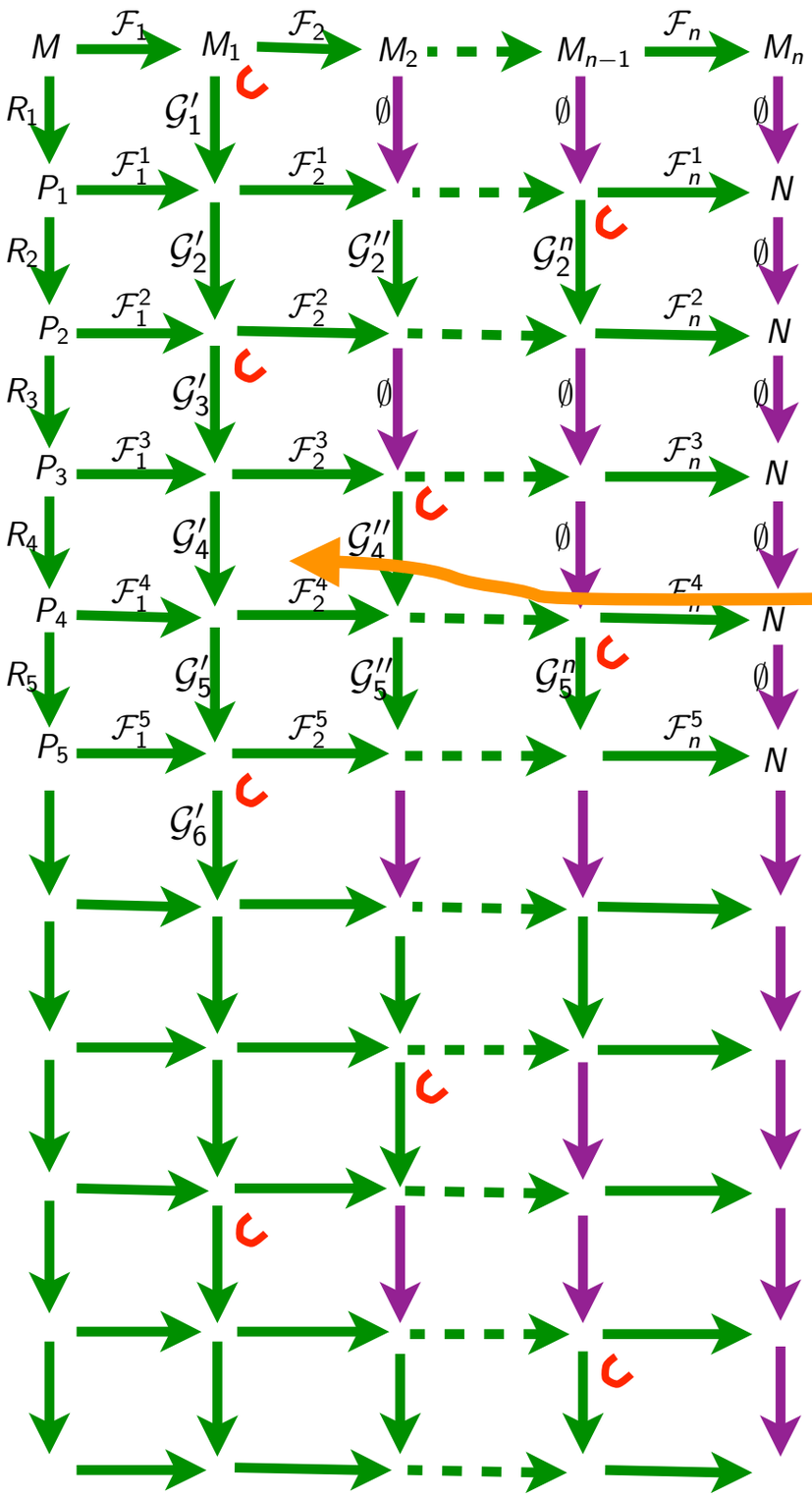
Weak vs Strong Normalisation



- true in any PTS lambda system

[conjecture Barendregt / Geuvers]

Weak vs Strong Normalisation



any reduction to nf

parallel moves

λI-calculus

finite developments

Weak normalization in lambda-I

- innermost reduction clearly terminates (in lambda-K fst order)

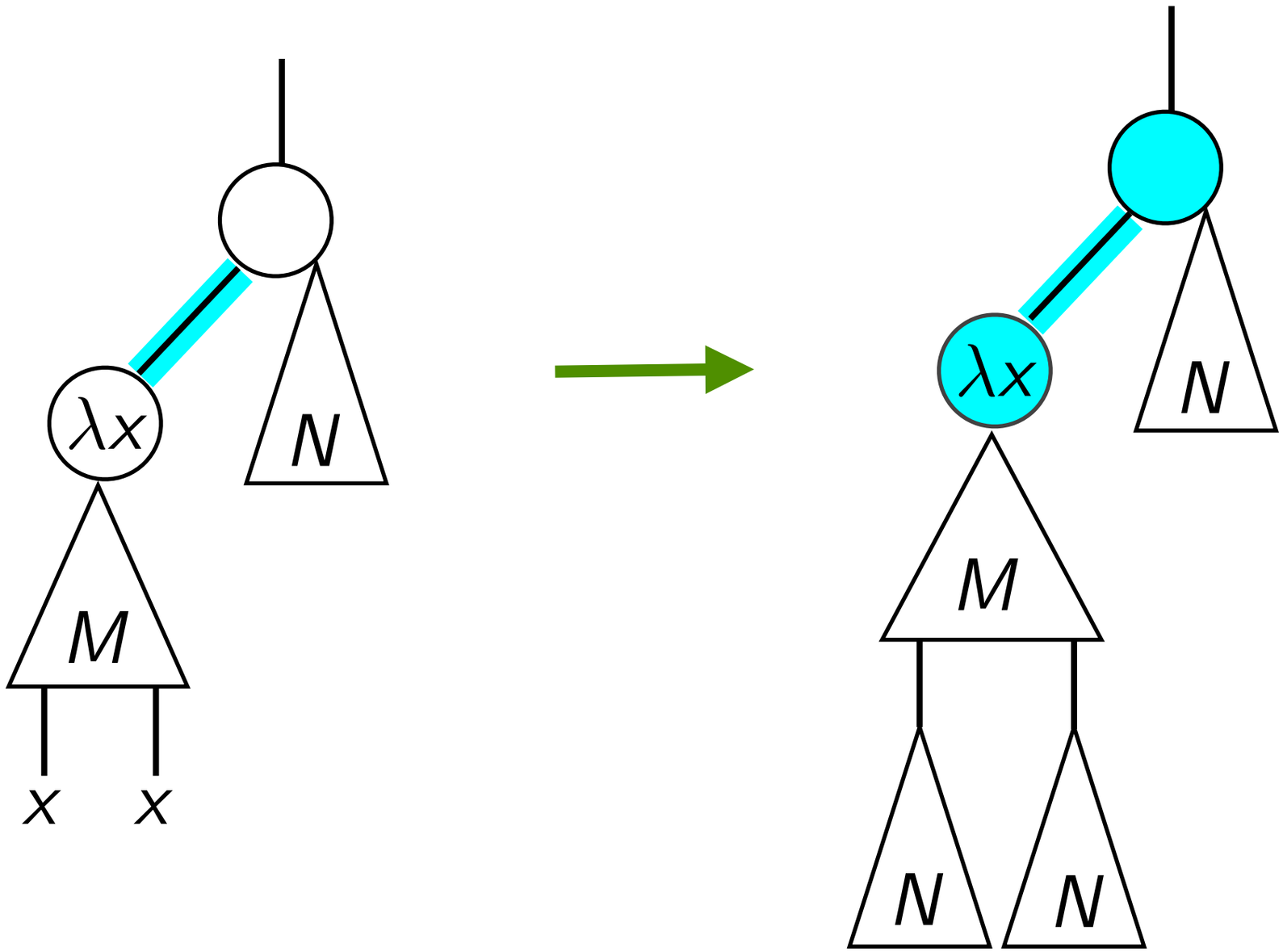
(take multiset ordering on degrees of redexes)

- weak implies strong in lambda-I

(take same argument as for standardization proof: finite developments + cube lemma)

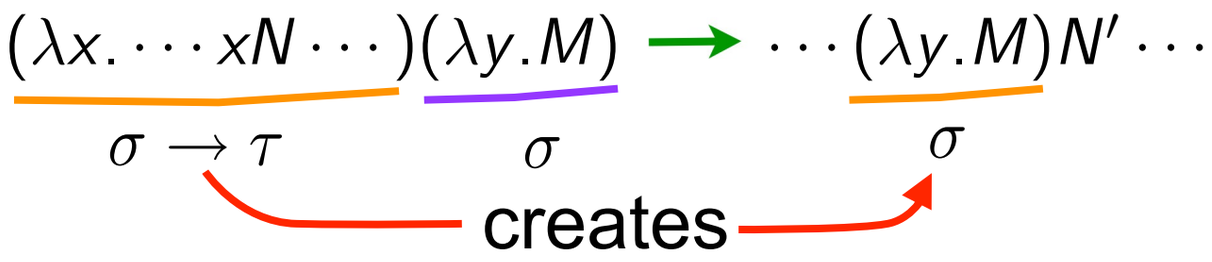
Weak vs Strong Normalisation

- Nederpelt[72], Klop[80], Sorensen[?]

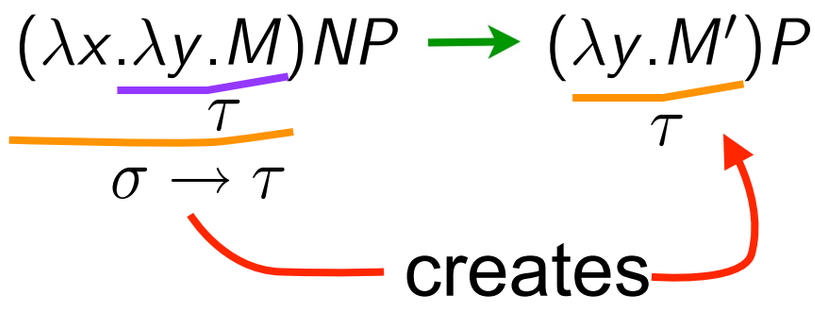


Strong Normalisation (1st order)

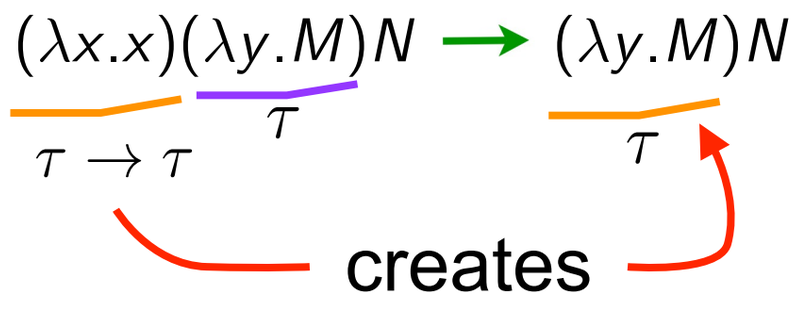
- why typed 1st-order calculus normalizes ?



creation downward



creation upward

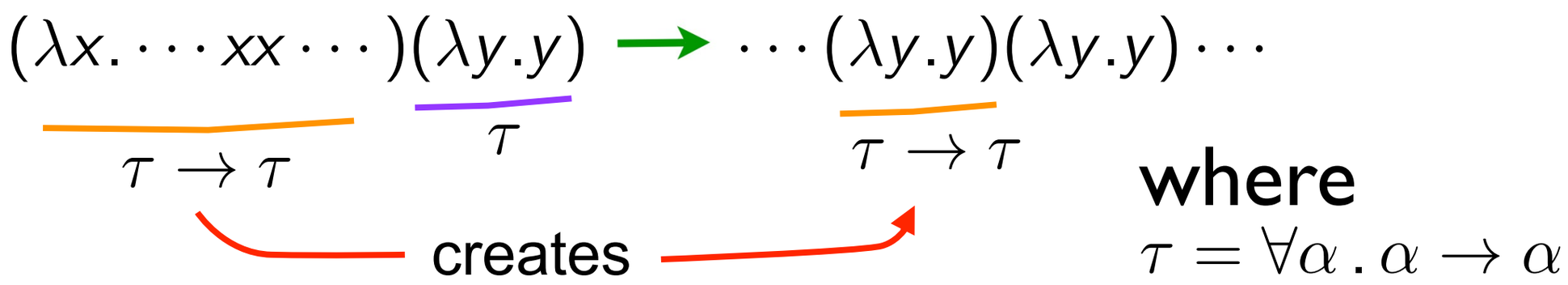


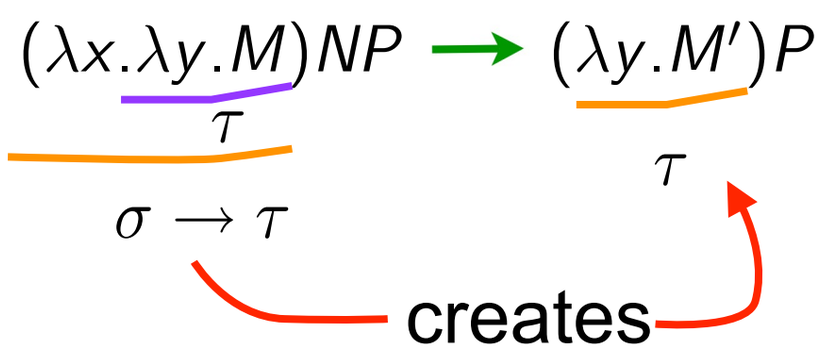
- degree of a redex is type of its function part

- degree strictly decreases with creation

Strong Normalisation(2nd order)

- why system F normalizes ?





Strong Normalisation(2nd order)

- looking more closely at system F

$$(\lambda x.\cdots xx\cdots)(\lambda y.y) \rightarrow \cdots(\lambda y.y)(\lambda y.y)\cdots$$

$\tau \to \tau$ $\quad$ τ $\quad$ $\tau \to \tau$

creates

where
$\tau = \forall\alpha\,.\,\alpha \to \alpha$

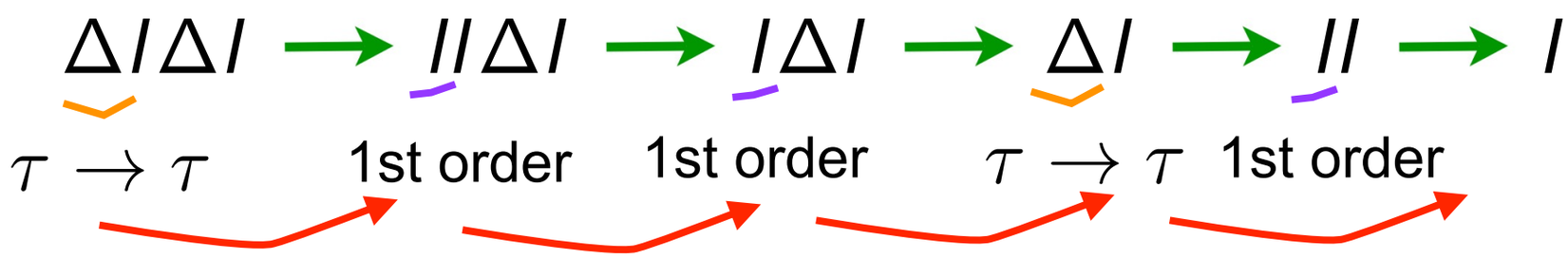
$$(\lambda x.\cdots xx\cdots)(\lambda y.y) \rightarrow \cdots(\lambda y.y)(\lambda y.y)\cdots$$

2nd order

fst order !

also typable with
$\forall\alpha\,.\,\tau' \to \tau'$ **where**
$\tau' = \alpha \to \alpha$

Strong Normalisation(2nd order)



where

$\tau = \forall\alpha\,.\,\alpha \to \alpha$

$\Delta = \lambda x.xx$

$I = \lambda x.x$

Semantics

Girard - Tait - Krivine proof

- **Definition** (**saturated sets**) $X \in \mathrm{SAT}$ iff

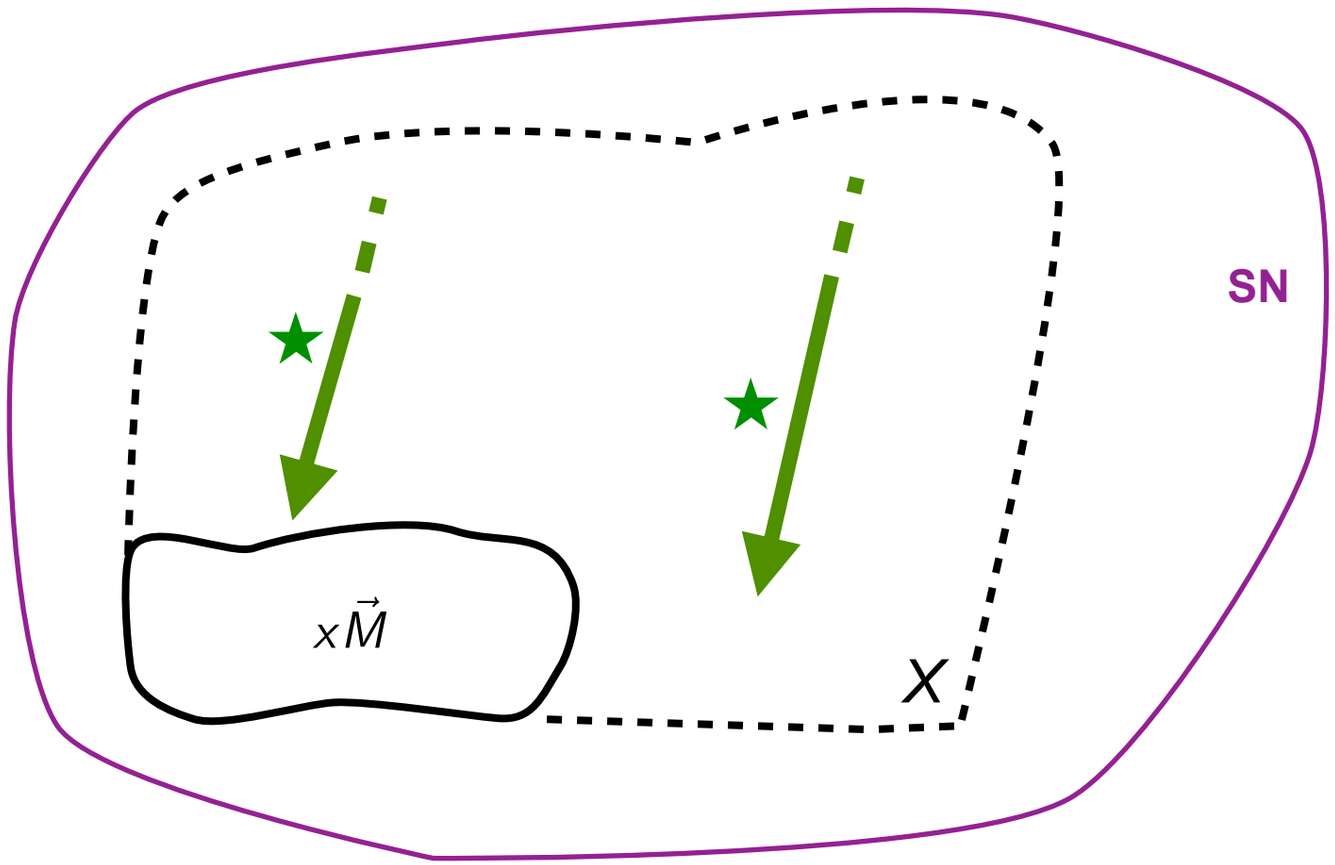
(0) $X \subset \mathcal{SN}$

(1) $x\vec{M} \in X$

(2) $M\{x := N\}\vec{P} \in X$ implies $(\lambda x.M)N\vec{P} \in X$

(1) = non emptyness **(2) = closed by SN-head-beta-expansion**

A saturated set



Girard - Tait - Krivine proof

- **Definition** (**saturated sets**) $X \in \mathrm{SAT}$ iff

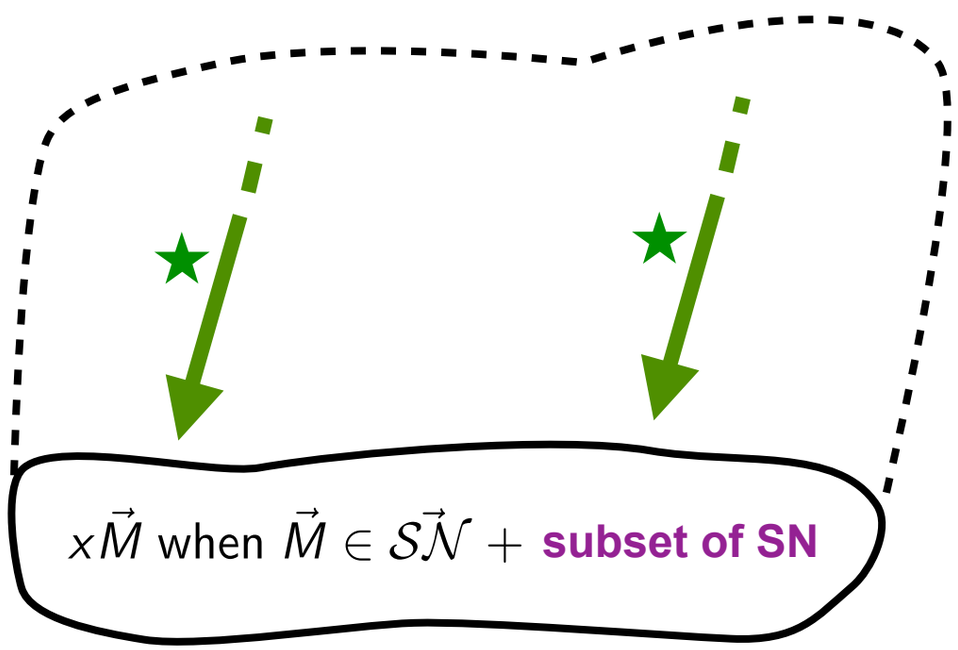
(0) $X \subset \mathcal{SN}$

(1) $x\vec{M} \in X$ iff $\vec{M} \in \mathcal{S}\vec{\mathcal{N}}$

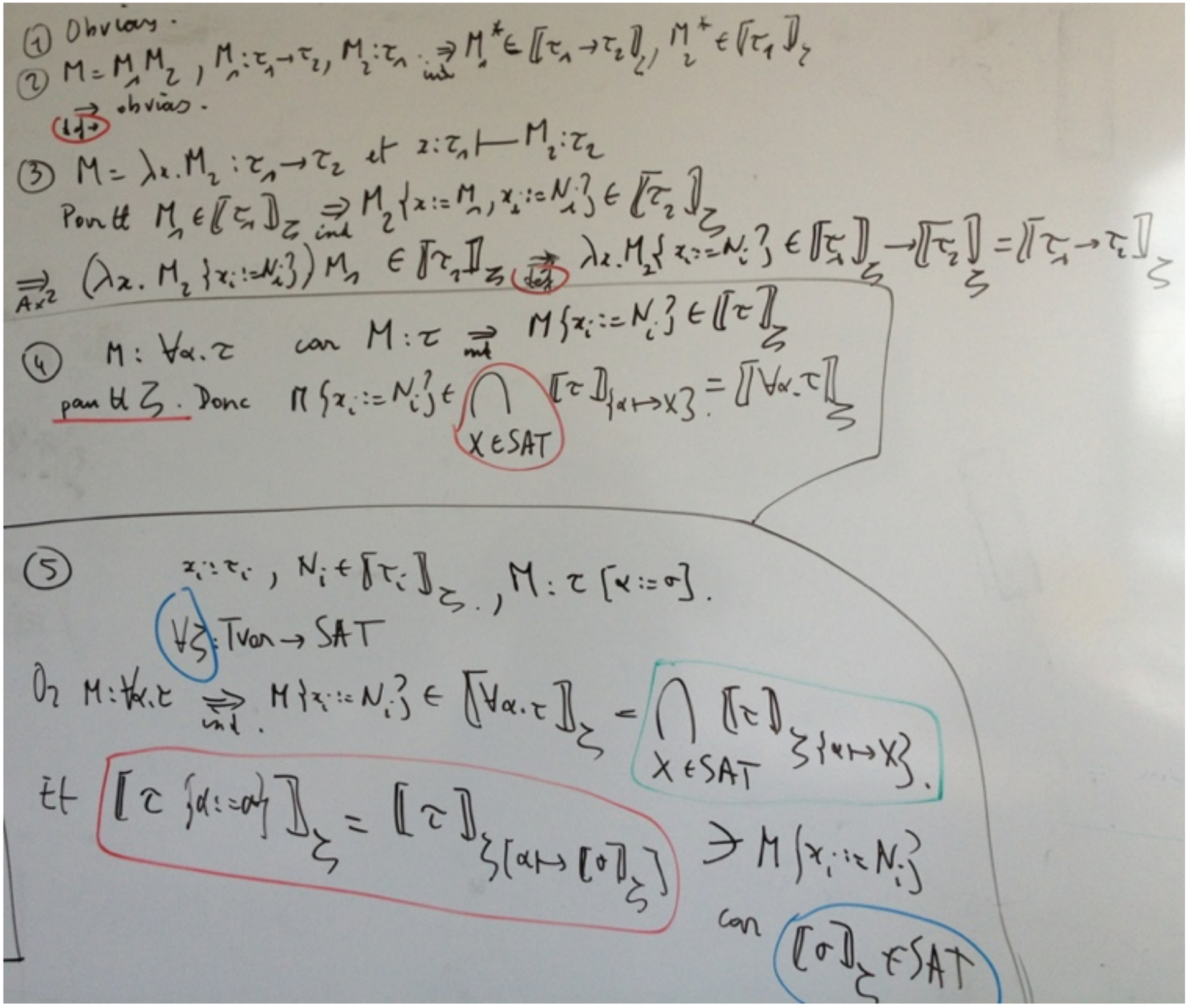
(2) $M\{x := N\}\vec{P} \in X$ implies $(\lambda x.M)N\vec{P} \in X$ when $N \in \mathcal{SN}$

(1) = non emptyness **(2) = closed by SN-head-beta-expansion**

A saturated set



Proofs


① Obvious.
② M = M1 M2 , M1 : τ1 → τ2 , M2 : τ1 ⇒ M1* ∈ [[τ1 → τ2]]ζ , M2* ∈ [[τ1]]ζ
obvious.
③ M = λx.M2 : τ1 → τ2 et x : τ1 ⊢ M2 : τ2
Pour tt M1 ∈ [[τ1]]ζ ⇒ M2{x := M1, xi := Ni} ∈ [[τ2]]ζ
⇒ (λx. M2{xi := Ni}) M1 ∈ [[τ1]]ζ ⇒ λx.M2{xi := Ni} ∈ [[τ1]]ζ → [[τ2]]ζ = [[τ1 → τ2]]ζ
④ M : ∀α.τ car M : τ ⇒ M{xi := Ni} ∈ [[τ]]ζ
pour tt ζ. Donc M{xi := Ni} ∈ ⋂ X∈SAT [[τ]]ζ{α↦X} = [[∀α.τ]]ζ
⑤ xi : τi , Ni ∈ [[τi]]ζ , M : τ [x := σ].
∀ζ : Tvar → SAT
Or M : ∀α.τ ⇒ M{xi := Ni} ∈ [[∀α.τ]]ζ = ⋂ X∈SAT [[τ]]ζ{α↦X}.
et [[τ{α := σ}]]ζ = [[τ]]ζ{α↦[[σ]]ζ} ⇒ M{xi := Ni}
car [[σ]]ζ ∈ SAT

Proofs

$$[\![\tau\{\alpha:=\sigma\}]\!]_\zeta = [\![\tau]\!]_{\zeta\{\alpha\mapsto[\![\sigma]\!]_\zeta\}}$$

Pose $\tau^* = \tau\{\alpha:=\sigma\}$

① $\tau=\alpha$ $\quad \tau^*=\sigma$ $\quad [\![\tau^*]\!]_\zeta = [\![\sigma]\!]_\zeta$

② $\tau=\beta$ $\quad \tau^*=\beta$

③ $\tau=\tau_1\to\tau_2$ $\quad [\![\tau^*]\!]_\zeta = [\![\tau_1^*\to\tau_2^*]\!]_\zeta \overset{\text{ind}}{=} [\![\tau_1^*]\!]_\zeta \to [\![\tau_2^*]\!]_\zeta$

mod + def

① $\tau=\forall\beta.\tau_1$ $\quad [\![\tau^*]\!]_\zeta = [\![\forall\beta.\tau_1^*]\!]_\zeta = \bigcap_{X\in SAT} [\![\tau_1^*]\!]_{\zeta\{\beta\mapsto X\}}$

$$\overset{\text{ind}}{=} \bigcap_{X\in SAT} [\![\tau_1]\!]_{\zeta\{\beta\mapsto X\}\{\alpha\mapsto[\![\sigma]\!]_\zeta\}}.$$

car $\forall\zeta$.

Or $\beta\in FV(\sigma)$

$$[\![\tau^*]\!]_\zeta = \bigcap_{X\in SAT} [\![\tau_1]\!]_{\zeta\{\alpha\mapsto[\![\sigma]\!]_\zeta\}\{\beta\mapsto X\}}.$$

$$= [\![\forall\alpha.\tau_1]\!]_{\zeta\{\alpha\mapsto[\![\sigma]\!]_\zeta\}}.$$

$\mathcal{N}_0 \subset \mathcal{SN} \to \mathcal{N}_0 \subset \mathcal{N}_0 \to \mathcal{SN} \subset \mathcal{SN}$

① $\vec{M}\in\mathcal{SN}^{\vec{}}$, $P\in\mathcal{SN} \Rightarrow x\vec{M}P\in\mathcal{N}_0$.

② $\mathcal{N}_0\subset\mathcal{SN}$

③ $M\in\mathcal{N}_0\to\mathcal{SN} \Rightarrow Mx\in\mathcal{SN}$ car $x\in\mathcal{N}_0$.

$\mathcal{SN}\in SAT$

① $\mathcal{N}_0\subset\mathcal{SN}$

② $M\{x:=N\}\vec{P}\in\mathcal{SN}$ et $N\in\mathcal{SN}$
$\Rightarrow (\lambda x.M)N\vec{P}\in\mathcal{SN}$.

$X,Y\in SAT \Rightarrow X\to Y\in SAT$

① $\vec{M}\in\mathcal{SN}^{\vec{}}$, $P\in X \Rightarrow x\vec{M}P\in Y$ car $P\in X\subset\mathcal{SN}$. (Lemme ①)

② $N\in\mathcal{SN}$, $M\{x:=N\}\vec{P}\in X\to Y$

$Q\in X \overset{IH}{\Rightarrow} M\{x:=N\}\vec{P}Q\in Y \overset{Ax2}{\Rightarrow} (\lambda x.M)N\vec{P}Q\in Y$

Par def $(\lambda x.M)N\vec{P}\in X\to Y$.

$X_i\in SAT \to \bigcap_i X_i\in SAT$

Obvious

Girard - Tait - Krivine proof

Let $\mathcal{N}_0 = \{x\vec{M} \mid \vec{M} \in \mathcal{S}\vec{\mathcal{N}}\}$

and $X \to Y = \{M \mid N \in X \Rightarrow MN \in Y\}$

- **Fact 1** $\mathcal{N}_0 \subset \mathcal{SN} \to \mathcal{N}_0 \subset \mathcal{N}_0 \to \mathcal{SN} \subset \mathcal{SN}$
- **Fact 2** $\mathcal{SN} \in \mathrm{SAT}$

- **Lemma 1** $X, Y \in \mathrm{SAT}$ implies $X \to Y \in \mathrm{SAT}$
- **Lemma 2** $X_i \in \mathrm{SAT}$ implies $\bigcap_{i \in I} X_i \in \mathrm{SAT}$

Girard - Tait - Krivine proof

- **Semantics of types** Let $\zeta \in \mathsf{TVar} \to \mathsf{SAT}$. Then $[\![\tau]\!]_\zeta$ is

$$[\![\alpha]\!]_\zeta = \zeta(\alpha)$$

$$[\![\sigma \to \tau]\!]_\zeta = [\![\sigma]\!]_\zeta \to [\![\tau]\!]_\zeta \qquad [\![\forall \alpha.\tau]\!]_\zeta = \bigcap_{X \in \mathsf{SAT}} [\![\tau]\!]_{\zeta\{\alpha \mapsto X\}}$$

- **Corollary (1-2)** $[\![\tau]\!]_\zeta \in \mathsf{SAT}$
- **Lemma 3 (subst)** $[\![\tau\{\alpha := \sigma\}]\!]_\zeta = [\![\tau]\!]_{\zeta\{\alpha \mapsto [\![\sigma]\!]_\zeta\}}$
- **Lemma 4** Let $x_1{:}\tau_1, \ldots x_n{:}\tau_n \vdash M{:}\tau$ and $N_1 \in [\![\tau_1]\!]_\zeta, \ldots N_n \in [\![\tau_n]\!]_\zeta$ Then $M\{x_1 := N_1, \ldots x_n := N_n\} \in [\![\tau]\!]_\zeta$
- **Corollary (4)** $\Gamma \vdash M{:}\tau$ implies $M \in \mathcal{SN}$

Girard - ... - Barendregt' style proof

- **Semantics of terms** Let $\rho \in \mathrm{Var} \to \Lambda$. Then

$$[\![M]\!]_\rho = M\{x_1 := \rho(x_1), \ldots x_n := \rho(x_n)\}$$

$$\rho, \zeta \models M{:}\tau \text{ iff } [\![M]\!]_\rho \in [\![\tau]\!]_\zeta$$

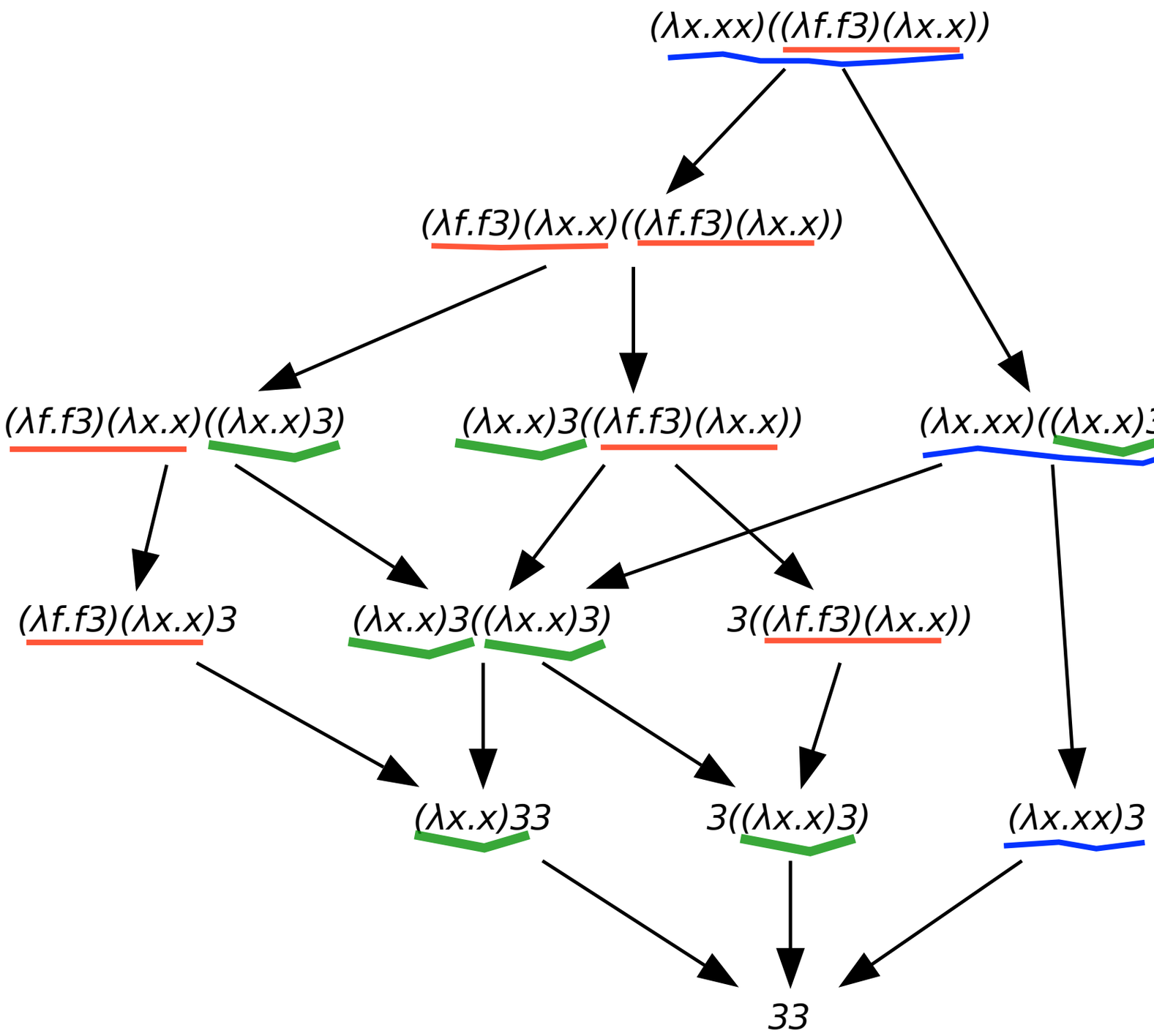
$$\rho, \zeta \models \Gamma \text{ iff } \rho, \zeta \models x{:}\tau \text{ for any } (x{:}\tau) \in \Gamma$$

$$\Gamma \models M{:}\tau \text{ iff } \forall \rho, \zeta \quad \rho, \zeta \models \Gamma \Rightarrow \rho, \zeta \models M{:}\tau$$

- **Lemma 3** (**subst**) $[\![\tau\{\alpha := \sigma\}]\!]_\zeta = [\![\tau]\!]_{\zeta\{\alpha \mapsto [\![\sigma]\!]_\zeta\}}$
- **Lemma 4** $\Gamma \vdash M{:}\tau$ implies $\Gamma \models M{:}\tau$
- **Corollary** $\Gamma \vdash M{:}\tau$ implies $M \in \mathcal{SN}$

Back to syntax

Redex families



- 3 redex families: **red**, **blue**, **green**.

Tracking redexes in untyped calculus

$$M, N, \ldots ::= x \mid MN \mid \lambda x\,.\,M \mid M^{\alpha}$$

$$(\lambda x\,.\,M)^{\alpha} N \longrightarrow M^{\lceil \alpha \rceil}\{x := N^{\lfloor \alpha \rfloor}\}$$

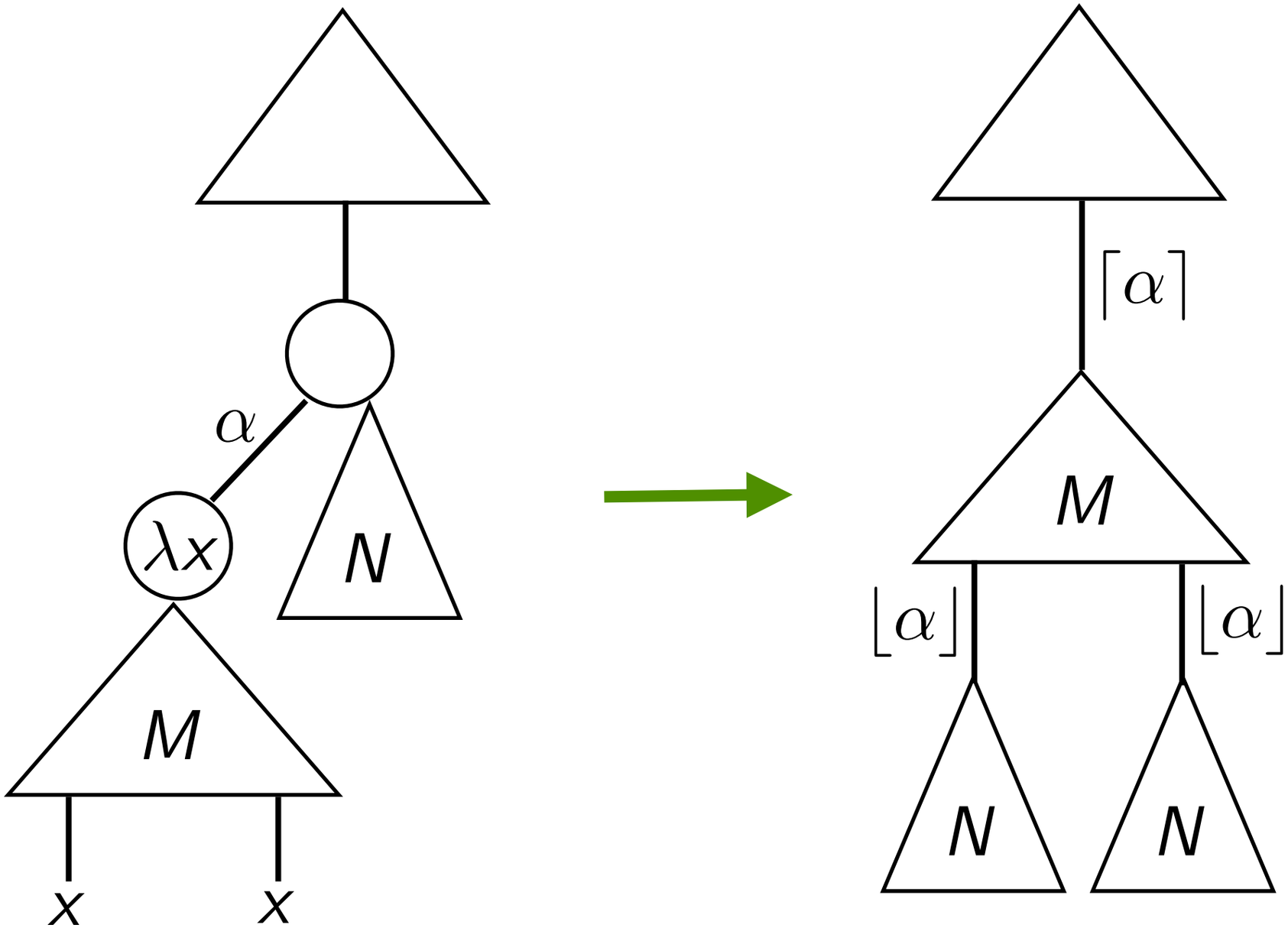
where

$$(M^{\alpha})^{\beta} = M^{\alpha\beta} \quad \textbf{and} \quad M^{\alpha}\{x := N\} = (M\{x := N\})^{\alpha}$$

The 2 theorems

- Confluence (consistent names of redexes)
- Created redexes contain names of creators

Graphically

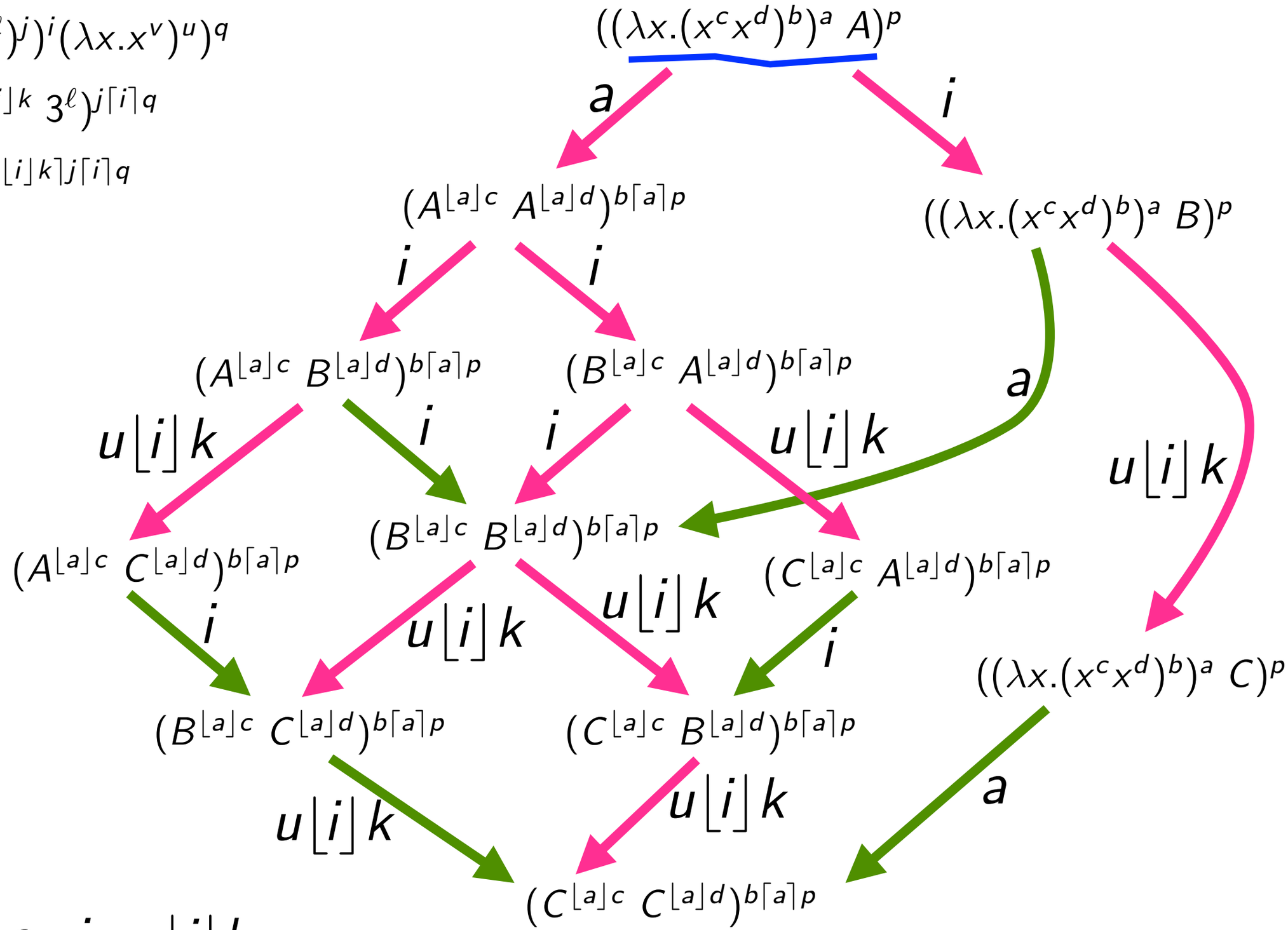


Redex families

$A = ((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^q$

$B = ((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q}$

$C = 3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q}$



- 3 families: a i $u\lfloor i\rfloor k$

$((\lambda x.(x^c x^d)^b)^a\ ((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^q)^p$

•3 families: a i $u\lfloor i\rfloor k$

a i

$(\ ((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^{q\lfloor a\rfloor c}$
$((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^{q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$((\lambda x.(x^c x^d)^b)^a((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q})^p$

i i

$(\ ((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^{q\lfloor a\rfloor c}$
$((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$(\ ((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q\lfloor a\rfloor c}$
$((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^{q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$u\lfloor i\rfloor k$ i i $u\lfloor i\rfloor k$ a $u\lfloor i\rfloor k$

$)^i(\lambda x.x^v)^u)^{q\lfloor a\rfloor c}$
$k\rceil j\lceil i\rceil q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$(\ ((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q\lfloor a\rfloor c}$
$((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$(\ 3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q\lfloor a\rfloor c}$
$((\lambda f.(f^k\ 3^\ell)^j)^i(\lambda x.x^v)^u)^{q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$((\lambda x.(x^c x^d)^b)^a$
$3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q})^p$

i $u\lfloor i\rfloor k$ $u\lfloor i\rfloor k$ i

$(\ ((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q\lfloor a\rfloor c}$
$3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$(\ 3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q\lfloor a\rfloor c}$
$((\lambda x.x^v)^{u\lfloor i\rfloor k}\ 3^\ell)^{j\lceil i\rceil q\lfloor a\rfloor d}\ )^{b\lceil a\rceil p}$

$u\lfloor i\rfloor k$ a

$u\lfloor i\rfloor k$

$(3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q\lfloor a\rfloor c}$
$3^{\ell\lfloor u\lfloor i\rfloor k\rfloor v\lceil u\lfloor i\rfloor k\rceil j\lceil i\rceil q\lfloor a\rfloor d})^{b\lceil a\rceil p}$

Finite and infinite reductions (1/3)

- **Definition** A **reduction relative** to a set $\mathcal{F}$ of redex families is any reduction contracting redexes in families of $\mathcal{F}$.

  A **development** of $\mathcal{F}$ is any maximal relative reduction.

- **Theorem** `[Finite Developments++, 76]`
  Let $\mathcal{F}$ be a finite set of redex families.

  (1) there are no infinite reductions relative to $\mathcal{F}$,
  (2) they all finish on same term N
  (3) All developments are equivalent by permutations.

Finite and infinite reductions (2/3)

- **Corollary** An **infinite reduction** contracts an **infinite set of redex families.**

- **Corollary** The first-order typed λ-calculus strongly terminates.

Proof In first-order typed λ-calculus:

(1) residuals $R' = (\lambda x.M')N'$ of $R = (\lambda x.M)N$ keep the degree

(2) new redexes have lower degree

Todo list

- Relate tracking of redexes to Girard's impredicative proof
- Find intuitive argument for SN in higher-order typed λ-calculus
- Find intuitive proof for SN in higher-order typed λ-calculus

[SN proof must always be in 3rd-order Peano logic]

The end