

Formalización de Sistema I con tipo Top

Agustin Francisco Settimo

Directora: Cecilia Manzino

Co-Director: Cristian Sottile

Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Departamento de Ciencias de la Computación



- 1 Marco teórico
 - Introducción a LCST
 - Introducción a Sistema I

- 2 Aportes
 - Formalización
 - Normalización fuerte

- 3 Conclusiones

- Creado para investigar los fundamentos de la matemática.
(principalmente, el concepto de recursión).

- Creado para investigar los fundamentos de la matemática.
(principalmente, el concepto de recursión).
- Es el lenguaje de programación funcional más simple que existe.

- Creado para investigar los fundamentos de la matemática. (principalmente, el concepto de recursión).
- Es el lenguaje de programación funcional más simple que existe.
 - Las funciones son anónimas:

$$\text{suma}(x, y) = x + y \quad \text{suma}(2, 3)$$

- Creado para investigar los fundamentos de la matemática. (principalmente, el concepto de recursión).
- Es el lenguaje de programación funcional más simple que existe.
 - Las funciones son anónimas:

$$\text{suma}(x, y) = x + y \quad \text{suma}(2, 3)$$

$$((x, y) \mapsto x + y) (2, 3)$$

- Creado para investigar los fundamentos de la matemática. (principalmente, el concepto de recursión).
- Es el lenguaje de programación funcional más simple que existe.
 - Las funciones son anónimas:

$$\text{suma}(x, y) = x + y \quad \text{suma}(2, 3)$$

$$((x, y) \mapsto x + y) (2, 3)$$

- Las funciones toman un solo argumento:

$$x \mapsto (y \mapsto x + y)$$

- Creado para investigar los fundamentos de la matemática. (principalmente, el concepto de recursión).
- Es el lenguaje de programación funcional más simple que existe.
 - Las funciones son anónimas:

$$\text{suma}(x, y) = x + y \quad \text{suma}(2, 3)$$

$$((x, y) \mapsto x + y) (2, 3)$$

- Las funciones toman un solo argumento:

$$x \mapsto (y \mapsto x + y)$$

$$(x \mapsto (y \mapsto x + y)) 2 \Rightarrow y \mapsto 2 + y$$

- Gramática de términos

- Gramática de términos

- Gramática de términos

$$t := x \mid \lambda x.t \mid t t$$

- Gramática de términos

$$t := x \mid \lambda x.t \mid t \ t \mid \langle t, t \rangle \mid \pi_1 t \mid \pi_2 t$$

- Gramática de términos

$$t := x \mid \lambda x.t \mid t \ t \mid \langle t, t \rangle \mid \pi_1 t \mid \pi_2 t \mid \star$$

- Computación (reglas de reescritura)

- Gramática de términos

$$t := x \mid \lambda x.t \mid t \ t \mid \langle t, t \rangle \mid \pi_1 t \mid \pi_2 t \mid \star$$

- Computación (reglas de reescritura)
 - Aplicación (β -reducción)

$$(\lambda x.t)s \hookrightarrow_{\beta} t[s/x]$$

- Gramática de términos

$$t := x \mid \lambda x.t \mid t \ t \mid \langle t, t \rangle \mid \pi_1 t \mid \pi_2 t \mid \star$$

- Computación (reglas de reescritura)
 - Aplicación (β -reducción)

$$(\lambda x.t)s \hookrightarrow_{\beta} t[s/x]$$

$t[s/x]$ reemplaza todas las variables x que aparecen en t por s .

- Gramática de términos

$$t := x \mid \lambda x.t \mid t \ t \mid \langle t, t \rangle \mid \pi_1 t \mid \pi_2 t \mid \star$$

- Computación (reglas de reescritura)
 - Aplicación (β -reducción)

$$(\lambda x.t)s \hookrightarrow_{\beta} t[s/x]$$

$t[s/x]$ reemplaza todas las variables x que aparecen en t por s .

- Proyección

$$\pi_1 \langle r, s \rangle \hookrightarrow_{\pi} r \quad \pi_2 \langle r, s \rangle \hookrightarrow_{\pi} s$$

Swap

$$\lambda p. \langle \pi_2(p), \pi_1(p) \rangle$$

Swap

$$\lambda p. \langle \pi_2(p), \pi_1(p) \rangle$$

$$(\lambda p. \langle \pi_2(p), \pi_1(p) \rangle) \langle a, b \rangle$$

Swap

$$\lambda p. \langle \pi_2(p), \pi_1(p) \rangle$$

$$(\lambda p. \langle \pi_2(p), \pi_1(p) \rangle) \langle a, b \rangle$$

$$\hookrightarrow_{\beta} \langle \pi_2 \langle a, b \rangle, \pi_1 \langle a, b \rangle \rangle$$

Swap

$$\lambda p. \langle \pi_2(p), \pi_1(p) \rangle$$

$$(\lambda p. \langle \pi_2(p), \pi_1(p) \rangle) \langle a, b \rangle$$

$$\hookrightarrow_{\beta} \langle \pi_2 \langle a, b \rangle, \pi_1 \langle a, b \rangle \rangle$$

$$\hookrightarrow_{\pi} \langle b, \pi_1 \langle a, b \rangle \rangle$$

Swap

$$\lambda p. \langle \pi_2(p), \pi_1(p) \rangle$$

$$(\lambda p. \langle \pi_2(p), \pi_1(p) \rangle) \langle a, b \rangle$$

$$\hookrightarrow_{\beta} \langle \pi_2 \langle a, b \rangle, \pi_1 \langle a, b \rangle \rangle$$

$$\hookrightarrow_{\pi} \langle b, \pi_1 \langle a, b \rangle \rangle$$

$$\hookrightarrow_{\pi} \langle b, a \rangle$$

Algunos términos no reducen (están atascados):

$$\langle a, b \rangle c$$

$$\pi_1(\lambda x.x)$$

Algunos términos no reducen (están atascados):

$$\langle a, b \rangle c$$

$$\pi_1(\lambda x.x)$$

Otros nunca terminan de reducir:

$$(\lambda x.xx)$$

Algunos términos no reducen (están atascados):

$$\langle a, b \rangle c$$

$$\pi_1(\lambda x.x)$$

Otros nunca terminan de reducir:

$$(\lambda x.xx)(\lambda x.xx)$$

Algunos términos no reducen (están atascados):

$$\langle a, b \rangle c$$

$$\pi_1(\lambda x.x)$$

Otros nunca terminan de reducir:

$$(\lambda x.xx)(\lambda x.xx) \hookrightarrow (\lambda x.xx)(\lambda x.xx)$$

Algunos términos no reducen (están atascados):

$$\langle a, b \rangle c$$

$$\pi_1(\lambda x.x)$$

Otros nunca terminan de reducir:

$$(\lambda x.xx)(\lambda x.xx) \hookrightarrow (\lambda x.xx)(\lambda x.xx) \hookrightarrow (\lambda x.xx)(\lambda x.xx) \dots$$

- Tipos

$$A := A \rightarrow A \mid A \times A \mid \top$$

- Tipos

$$A := A \rightarrow A \mid A \times A \mid \top$$

- Contextos de tipado

$$\Gamma := \emptyset \mid \Gamma, x : A$$

Reglas de tipado

$$\frac{\Gamma \ni x : A}{\Gamma \vdash x : A} (ax)$$

Reglas de tipado

$$\frac{\Gamma \ni x : A}{\Gamma \vdash x : A} (ax) \qquad \frac{}{\Gamma \vdash \star : \top} (\top_i)$$

Reglas de tipado

$$\frac{\Gamma \ni x : A}{\Gamma \vdash x : A} (ax) \quad \frac{}{\Gamma \vdash \star : \top} (\top_i)$$
$$\frac{\Gamma, x : A \vdash r : B}{\Gamma \vdash \lambda x. r : A \rightarrow B} (\Rightarrow_i) \quad \frac{\Gamma \vdash r : A \rightarrow B \quad \Gamma \vdash s : A}{\Gamma \vdash r s : B} (\Rightarrow_e)$$

Reglas de tipado

$$\frac{\Gamma \ni x : A}{\Gamma \vdash x : A} (ax) \quad \frac{}{\Gamma \vdash \star : \top} (\top_i)$$

$$\frac{\Gamma, x : A \vdash r : B}{\Gamma \vdash \lambda x. r : A \rightarrow B} (\Rightarrow_i) \quad \frac{\Gamma \vdash r : A \rightarrow B \quad \Gamma \vdash s : A}{\Gamma \vdash r s : B} (\Rightarrow_e)$$

$$\frac{\Gamma \vdash r : A \quad \Gamma \vdash s : B}{\Gamma \vdash \langle r, s \rangle : A \times B} (\times_i) \quad \frac{\Gamma \vdash r : A \times B}{\Gamma \vdash \pi_1(r) : A} (\times_{e1})$$
$$\frac{\Gamma \vdash r : A \times B}{\Gamma \vdash \pi_2(r) : B} (\times_{e2})$$

Definición (Forma normal)

t es normal si no puede reducirse. $t \not\rightarrow$

Definición (Forma normal)

t es normal si no puede reducirse. $t \not\rightarrow$

Progreso

Si $\Gamma \vdash t : A$ entonces t es normal o $t \hookrightarrow t'$.

Definición (Forma normal)

t es normal si no puede reducirse. $t \not\rightarrow$

Progreso

Si $\Gamma \vdash t : A$ entonces t es normal o $t \hookrightarrow t'$.

Normalización Fuerte

Si $\Gamma \vdash t : A$ entonces no existe ninguna secuencia de reducción infinita.

$t \hookrightarrow t_1 \hookrightarrow t_2 \hookrightarrow t_3 \hookrightarrow \dots$

- 1 Marco teórico
 - Introducción a LCST
 - Introducción a Sistema I
- 2 Aportes
 - Formalización
 - Normalización fuerte
- 3 Conclusiones

Los tipos indican la **forma** y la **clase de problema** de un programa:

Los tipos indican la **forma** y la **clase de problema** de un programa:

$$\textit{suma} : (\textit{int} \times \textit{int}) \rightarrow \textit{int}$$

Los tipos indican la **forma** y la **clase de problema** de un programa:

$$\textit{suma} : (\textit{int} \times \textit{int}) \rightarrow \textit{int}$$

Existen programas con tipos distintos para la misma clase de problema:

$$\textit{suma}' : \textit{int} \rightarrow \textit{int} \rightarrow \textit{int}$$

Los tipos indican la **forma** y la **clase de problema** de un programa:

$$\textit{suma} : (\textit{int} \times \textit{int}) \rightarrow \textit{int}$$

Existen programas con tipos distintos para la misma clase de problema:

$$\textit{suma}' : \textit{int} \rightarrow \textit{int} \rightarrow \textit{int}$$

Los programas se combinan según su forma:

$$\textit{suma} \langle 2, 3 \rangle \quad \textit{suma}' \ 2 \ 3$$

Los tipos indican la **forma** y la **clase de problema** de un programa:

$$\textit{suma} : (\textit{int} \times \textit{int}) \rightarrow \textit{int}$$

Existen programas con tipos distintos para la misma clase de problema:

$$\textit{suma}' : \textit{int} \rightarrow \textit{int} \rightarrow \textit{int}$$

Los programas se combinan según su forma:

$$\textit{suma} \langle 2, 3 \rangle \quad \textit{suma}' 2 3$$

Sería interesante poder combinarlos según su significado:

$$\textit{suma} 2 3 \quad \textit{suma}' \langle 2, 3 \rangle$$

Isomorfismos entre tipos

Dos tipos A y B son *isomorfos* si solo si existen:

$$f : A \rightarrow B \quad g : B \rightarrow A$$

Isomorfismos entre tipos

Dos tipos A y B son *isomorfos* si solo si existen:

$$f : A \rightarrow B \quad g : B \rightarrow A$$

Tales que:

$$f \circ g = id_A \quad g \circ f = id_B$$

Isomorfismos entre tipos

Dos tipos A y B son *isomorfos* si solo si existen:

$$f : A \rightarrow B \quad g : B \rightarrow A$$

Tales que:

$$f \circ g = id_A \quad g \circ f = id_B$$

Y lo notamos $A \equiv B$.

Isomorfismo de Curry

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$$

Isomorfismo de Curry

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$$

$$curry = \lambda f^{(A \times B) \rightarrow C}. \lambda a^A. \lambda b^B. f \langle a, b \rangle$$

Isomorfismo de Curry

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$$

$$curry = \lambda f^{(A \times B) \rightarrow C}. \lambda a^A. \lambda b^B. f \langle a, b \rangle$$

$$uncurry = \lambda f^{A \rightarrow B \rightarrow C}. \lambda y^{A \times B}. f(\pi_1 y)(\pi_2 y)$$

Isomorfismo de Curry

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$$

$$curry = \lambda f^{(A \times B) \rightarrow C}. \lambda a^A. \lambda b^B. f \langle a, b \rangle$$

$$uncurry = \lambda f^{A \rightarrow B \rightarrow C}. \lambda y^{A \times B}. f(\pi_1 y)(\pi_2 y)$$

$$curry \circ uncurry = id_{(A \times B) \rightarrow C} \quad uncurry \circ curry = id_{A \rightarrow B \rightarrow C}$$

$$A \times B \equiv B \times A \quad (\text{COMM})$$

$$A \times B \equiv B \times A$$

(COMM)

$$A \times (B \times C) \equiv (A \times B) \times C$$

(ASSO)

$$A \times B \equiv B \times A \quad (\text{COMM})$$

$$A \times (B \times C) \equiv (A \times B) \times C \quad (\text{ASSO})$$

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C) \quad (\text{CURRY})$$

$$A \times B \equiv B \times A \quad (\text{COMM})$$

$$A \times (B \times C) \equiv (A \times B) \times C \quad (\text{ASSO})$$

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C) \quad (\text{CURRY})$$

$$A \rightarrow (B \times C) \equiv (A \rightarrow B) \times (A \rightarrow C) \quad (\text{DIST})$$

$$A \times B \equiv B \times A \quad (\text{COMM})$$

$$A \times (B \times C) \equiv (A \times B) \times C \quad (\text{ASSO})$$

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C) \quad (\text{CURRY})$$

$$A \rightarrow (B \times C) \equiv (A \rightarrow B) \times (A \rightarrow C) \quad (\text{DIST})$$

$$A \times \top \equiv A \quad (\text{ID}_\times)$$

$$A \times B \equiv B \times A \quad (\text{COMM})$$

$$A \times (B \times C) \equiv (A \times B) \times C \quad (\text{ASSO})$$

$$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C) \quad (\text{CURRY})$$

$$A \rightarrow (B \times C) \equiv (A \rightarrow B) \times (A \rightarrow C) \quad (\text{DIST})$$

$$A \times \top \equiv A \quad (\text{ID}_{\times})$$

$$A \rightarrow \top \equiv \top \quad (\text{ABS})$$

$A \times B \equiv B \times A$	(COMM)
$A \times (B \times C) \equiv (A \times B) \times C$	(ASSO)
$(A \times B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$	(CURRY)
$A \rightarrow (B \times C) \equiv (A \rightarrow B) \times (A \rightarrow C)$	(DIST)
$A \times \top \equiv A$	(ID _×)
$A \rightarrow \top \equiv \top$	(ABS)
$\top \rightarrow A \equiv A$	(ID _⇒)

Se añade la regla \equiv (equiv)

$$\frac{A \equiv B \quad \Gamma \vdash r : A}{\Gamma \vdash r : B} (\equiv)$$

Internalización de isomorfismos

Se añade la regla \equiv (equiv)

$$\frac{A \equiv B \quad \Gamma \vdash r : A}{\Gamma \vdash r : B} (\equiv)$$

Esto permite usar $r : A$ en lugar de $r : B$, siempre y cuando $A \equiv B$

$$\frac{\frac{\Gamma \vdash r : A}{\Gamma \vdash \lambda x.r : C \rightarrow A} (\Rightarrow_i) \quad \frac{\Gamma \vdash s : B}{\Gamma \vdash \lambda x.s : C \rightarrow B} (\Rightarrow_i)}{\Gamma \vdash \langle \lambda x.r, \lambda x.s \rangle : (C \rightarrow A) \times (C \rightarrow B)} (\times_i) \quad (\equiv) \quad \frac{\Gamma \vdash \langle \lambda x.r, \lambda x.s \rangle : C \rightarrow (A \times B) \quad \Gamma \vdash t : C}{\Gamma \vdash \langle \lambda x.r, \lambda x.s \rangle t : A \times B} (\Rightarrow_e)$$

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

$$\lambda x^A. \langle r, s \rangle \Leftrightarrow \langle \lambda x^A. r, \lambda x^A. s \rangle \quad (\text{DIST}_\lambda)$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

$$\lambda x^A. \langle r, s \rangle \Leftrightarrow \langle \lambda x^A. r, \lambda x^A. s \rangle \quad (\text{DIST}_\lambda)$$

$$\langle r, s \rangle t \Leftrightarrow \langle r t, s t \rangle \quad (\text{DIST}_{app})$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

$$\lambda x^A. \langle r, s \rangle \Leftrightarrow \langle \lambda x^A. r, \lambda x^A. s \rangle \quad (\text{DIST}_\lambda)$$

$$\langle r, s \rangle t \Leftrightarrow \langle r t, s t \rangle \quad (\text{DIST}_{app})$$

$$r \langle s, t \rangle \Leftrightarrow r s t \quad (\text{CURRY})$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

$$\lambda x^A. \langle r, s \rangle \Leftrightarrow \langle \lambda x^A. r, \lambda x^A. s \rangle \quad (\text{DIST}_\lambda)$$

$$\langle r, s \rangle t \Leftrightarrow \langle r t, s t \rangle \quad (\text{DIST}_{app})$$

$$r \langle s, t \rangle \Leftrightarrow r s t \quad (\text{CURRY})$$

$$\langle \lambda x. r, \lambda x. s \rangle t$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

$$\lambda x^A. \langle r, s \rangle \Leftrightarrow \langle \lambda x^A. r, \lambda x^A. s \rangle \quad (\text{DIST}_\lambda)$$

$$\langle r, s \rangle t \Leftrightarrow \langle r t, s t \rangle \quad (\text{DIST}_{app})$$

$$r \langle s, t \rangle \Leftrightarrow r s t \quad (\text{CURRY})$$

$$\begin{aligned} & \langle \lambda x. r, \lambda x. s \rangle t \\ & \xrightarrow{\text{DIST}_\lambda} \\ & (\lambda x. \langle r, s \rangle) t \end{aligned}$$

Relación de equivalencia entre términos

$$\langle r, s \rangle \rightleftharpoons \langle s, r \rangle \quad (\text{COMM})$$

$$\langle r, \langle s, t \rangle \rangle \rightleftharpoons \langle \langle r, s \rangle, t \rangle \quad (\text{ASSO})$$

$$\lambda x^A. \langle r, s \rangle \rightleftharpoons \langle \lambda x^A. r, \lambda x^A. s \rangle \quad (\text{DIST}_\lambda)$$

$$\langle r, s \rangle t \rightleftharpoons \langle r t, s t \rangle \quad (\text{DIST}_{app})$$

$$r \langle s, t \rangle \rightleftharpoons r s t \quad (\text{CURRY})$$

$$\langle \lambda x. r, \lambda x. s \rangle t$$

$$\xrightarrow{\text{DIST}_\lambda}$$

$$(\lambda x. \langle r, s \rangle) t$$

$$\xrightarrow{\beta}$$

$$\langle r[t/x], s[t/x] \rangle$$

COMM permite cambiar el elemento de una proyección:

COMM permite cambiar el elemento de una proyección:

$$\begin{aligned}\pi_1 \langle r, s \rangle &\hookrightarrow_{\pi_1} r \\ \pi_1 \langle r, s \rangle &\xleftrightarrow{\text{COMM}} \pi_1 \langle s, r \rangle \hookrightarrow_{\pi_1} s\end{aligned}$$

COMM permite cambiar el elemento de una proyección:

$$\begin{aligned}\pi_1 \langle r, s \rangle &\hookrightarrow_{\pi_1} r \\ \pi_1 \langle r, s \rangle &\xleftrightarrow{\text{COMM}} \pi_1 \langle s, r \rangle \hookrightarrow_{\pi_1} s\end{aligned}$$

Si $r : A$ y $s : B$ esto puede ser un problema.

COMM permite cambiar el elemento de una proyección:

$$\begin{aligned}\pi_1 \langle r, s \rangle &\hookrightarrow_{\pi_1} r \\ \pi_1 \langle r, s \rangle &\xleftrightarrow{\text{COMM}} \pi_1 \langle s, r \rangle \hookrightarrow_{\pi_1} s\end{aligned}$$

Si $r : A$ y $s : B$ esto puede ser un problema.

La solución es acceder al elemento usando los tipos:

$$\text{si } r : A \quad \pi_A \langle r, s \rangle \hookrightarrow_{\pi} r$$

Se define la relación de reducción módulo isomorfismos:

$$\rightsquigarrow := \overset{*}{\longleftrightarrow} \circ \hookrightarrow \circ \overset{*}{\longleftrightarrow}$$

Se define la relación de reducción módulo isomorfismos:

$$\rightsquigarrow := \stackrel{*}{\longleftrightarrow} \circ \hookrightarrow \circ \stackrel{*}{\longleftrightarrow}$$

Se aplican todas las transformaciones necesarias para poder reducir el término correctamente.

- 1 Marco teórico
 - Introducción a LCST
 - Introducción a Sistema I
- 2 Aportes
 - Formalización
 - Normalización fuerte
- 3 Conclusiones

$$\lambda z.(\lambda y.y(\lambda x.x))(\lambda x.zx)$$

$$\lambda z. (\lambda y. y (\lambda x. x)) (\lambda x. zx)$$
$$\lambda (\lambda 0 (\lambda 0)) (\lambda 1 0)$$

$$\lambda z. (\lambda y. y (\lambda x. x)) (\lambda x. zx)$$

$$\lambda (\lambda 0 (\lambda 0)) (\lambda 1 0)$$

$$(\lambda x. \lambda y. zx (\lambda z. zx)) (\lambda x. wx)$$

$$\lambda z. (\lambda y. y (\lambda x. x)) (\lambda x. zx)$$

$$\lambda (\lambda 0 (\lambda 0)) (\lambda 1 0)$$

$$(\lambda x. \lambda y. zx (\lambda z. zx)) (\lambda x. wx)$$

$$(\lambda \lambda 2 1 (\lambda 0 2)) (\lambda 2 0)$$

$$\lambda z.(\lambda y.y(\lambda x.x))(\lambda x.zx)$$

$$\lambda(\lambda 0 (\lambda 0))(\lambda 1 0)$$

$$(\lambda x.\lambda y.zx(\lambda z.zx)) (\lambda x.wx)$$

$$(\lambda \lambda 2 1 (\lambda 0 2)) (\lambda 2 0) \hookrightarrow_{\beta} \lambda 1 \square (\lambda 0 \square)$$

$$\lambda z. (\lambda y. y (\lambda x. x)) (\lambda x. zx)$$

$$\lambda (\lambda 0 (\lambda 0)) (\lambda 1 0)$$

$$(\lambda x. \lambda y. zx (\lambda z. zx)) (\lambda x. wx)$$

$$(\lambda \lambda 2 1 (\lambda 0 2)) (\lambda 2 0) \hookrightarrow_{\beta} \lambda 1 (\lambda 3 0) (\lambda 0 (\lambda 4 0))$$

Substituciones explícitas

Usando índices, las substituciones son secuencias de términos:

$$\{0 \mapsto a, 1 \mapsto b, 2 \mapsto 0, 3 \mapsto 1, \dots\} = \{a, b, 0, 1, \dots\}$$

Usando índices, las substituciones son secuencias de términos:

$$\{0 \mapsto a, 1 \mapsto b, 2 \mapsto 0, 3 \mapsto 1, \dots\} = \{a, b, 0, 1, \dots\}$$

- id (substitución identidad) $\{i \mapsto i\} = \{0, 1, 2, \dots\}$.

Usando índices, las substituciones son secuencias de términos:

$$\{0 \mapsto a, 1 \mapsto b, 2 \mapsto 0, 3 \mapsto 1, \dots\} = \{a, b, 0, 1, \dots\}$$

- id (substitución identidad) $\{i \mapsto i\} = \{0, 1, 2, \dots\}$.
- \uparrow (shift) $\{i \mapsto i + 1\} = \{1, 2, 3, \dots\}$.

Usando índices, las substituciones son secuencias de términos:

$$\{0 \mapsto a, 1 \mapsto b, 2 \mapsto 0, 3 \mapsto 1, \dots\} = \{a, b, 0, 1, \dots\}$$

- id (substitución identidad) $\{i \mapsto i\} = \{0, 1, 2, \dots\}$.
- \uparrow (shift) $\{i \mapsto i + 1\} = \{1, 2, 3, \dots\}$.
- $a \bullet s$ (cons) $\{0 \mapsto a, i + 1 \mapsto s(i)\}$.

Substituciones explícitas

Usando índices, las substituciones son secuencias de términos:

$$\{0 \mapsto a, 1 \mapsto b, 2 \mapsto 0, 3 \mapsto 1, \dots\} = \{a, b, 0, 1, \dots\}$$

- id (substitución identidad) $\{i \mapsto i\} = \{0, 1, 2, \dots\}$.
- \uparrow (shift) $\{i \mapsto i + 1\} = \{1, 2, 3, \dots\}$.
- $a \bullet s$ (cons) $\{0 \mapsto a, i + 1 \mapsto s(i)\}$.

Substituciones explícitas

Usando índices, las substituciones son secuencias de términos:

$$\{0 \mapsto a, 1 \mapsto b, 2 \mapsto 0, 3 \mapsto 1, \dots\} = \{a, b, 0, 1, \dots\}$$

- id (substitución identidad) $\{i \mapsto i\} = \{0, 1, 2, \dots\}$.
- \uparrow (shift) $\{i \mapsto i + 1\} = \{1, 2, 3, \dots\}$.
- $a \bullet s$ (cons) $\{0 \mapsto a, i + 1 \mapsto s(i)\}$.

Por ejemplo $a \bullet id = \{0 \mapsto a, i + 1 \mapsto i\} = \{a, 0, 1, 2, \dots\}$

Usamos $\langle\langle s \rangle\rangle t$ para denotar la aplicación de la subst s sobre t .

Usamos $\langle\langle s \rangle\rangle t$ para denotar la aplicación de la subst s sobre t .

$$\langle\langle s \rangle\rangle n = s(n)$$

Usamos $\langle\langle s \rangle\rangle t$ para denotar la aplicación de la subst s sobre t .

$$\begin{aligned}\langle\langle s \rangle\rangle n &= s(n) \\ \langle\langle s \rangle\rangle (a \ b) &= \langle\langle s \rangle\rangle a \ \langle\langle s \rangle\rangle b\end{aligned}$$

Substituciones explícitas

Usamos $\langle\!\langle s \rangle\!\rangle t$ para denotar la aplicación de la subst s sobre t .

$$\langle\!\langle s \rangle\!\rangle n = s(n)$$

$$\langle\!\langle s \rangle\!\rangle (a \ b) = \langle\!\langle s \rangle\!\rangle a \ \langle\!\langle s \rangle\!\rangle b$$

$$\langle\!\langle s \rangle\!\rangle (\lambda a) = \lambda \langle\!\langle 0 \bullet (\uparrow \circ s) \rangle\!\rangle a$$

Usamos $\langle\!\langle s \rangle\!\rangle t$ para denotar la aplicación de la subst s sobre t .

$$\langle\!\langle s \rangle\!\rangle n = s(n)$$

$$\langle\!\langle s \rangle\!\rangle (a \ b) = \langle\!\langle s \rangle\!\rangle a \ \langle\!\langle s \rangle\!\rangle b$$

$$\langle\!\langle s \rangle\!\rangle (\lambda a) = \lambda \langle\!\langle 0 \bullet (\uparrow \circ s) \rangle\!\rangle a$$

Implementación de la β -reducción usando esta notación:

$$(\lambda t) \ a \hookrightarrow_{\beta} t[a] = \langle\!\langle a \bullet id \rangle\!\rangle t$$

$$A := \top \mid A \rightarrow A \mid A \times A$$

```
data Type : Set where
   $\top$       : Type
   $\_ \Rightarrow \_$  : Type  $\rightarrow$  Type  $\rightarrow$  Type
   $\_ \times \_$  : Type  $\rightarrow$  Type  $\rightarrow$  Type
```

Indices intrínsecamente tipados

$$\Gamma := \emptyset \mid \Gamma, x : A$$

data `Context` : `Set` where

`\emptyset` : `Context`

`-, -` : `Context` \rightarrow `Type` \rightarrow `Context`

Indices intrínsecamente tipados

$$\Gamma := \emptyset \mid \Gamma, x : A$$

data Context : Set where

\emptyset : Context

$-, -$: Context \rightarrow Type \rightarrow Context

data \ni : Context \rightarrow Type \rightarrow Set where

Z : $\forall \{ \Gamma \ A \}$

$\rightarrow \Gamma , A \ni A$

S_- : $\forall \{ \Gamma \ A \ B \}$

$\rightarrow \Gamma \ni B$

$\rightarrow \Gamma , A \ni B$

$_ : \emptyset , T \Rightarrow T , T \ni T$
 $_ = Z$

$_ : \emptyset , T \Rightarrow T , T \ni T \Rightarrow T$
 $_ = S Z$

data $\vdash_ :$ Context \rightarrow Type \rightarrow Set where

$\ulcorner_ : \forall \{ \Gamma \ A \}$ -- (ax)

$\rightarrow \Gamma \ni A$

$\rightarrow \Gamma \vdash A$

$\star : \forall \{ \Gamma \} \rightarrow \Gamma \vdash \top$ -- (\top_i)

\vdots

$$\frac{}{\Gamma, x : A \vdash x : A} \text{ (ax)}$$

$$\frac{}{\Gamma \vdash \star : \top} (\top_i)$$

\vdots
 $\lambda_- : \forall \{ \Gamma \ A \ B \} \text{ -- } (\Rightarrow_i)$
 $\rightarrow \Gamma, A \vdash B$

 $\rightarrow \Gamma \vdash A \Rightarrow B$

 $\cdot_- : \forall \{ \Gamma \ A \ B \} \text{ -- } (\Rightarrow_e)$
 $\rightarrow \Gamma \vdash A \Rightarrow B$
 $\rightarrow \Gamma \vdash A$

 $\rightarrow \Gamma \vdash B$
 \vdots

$$\frac{\Gamma, x : A \vdash r : B}{\Gamma \vdash \lambda x. r : A \rightarrow B} (\Rightarrow_i)$$

$$\frac{\Gamma \vdash r : A \rightarrow B \quad \Gamma \vdash s : A}{\Gamma \vdash rs : B} (\Rightarrow_e)$$

⋮

$\langle _, _ \rangle : \forall \{ \Gamma \ A \ B \} \text{ -- } (\times_i)$

$\rightarrow \Gamma \vdash A$

$\rightarrow \Gamma \vdash B$

 $\rightarrow \Gamma \vdash A \times B$

$\pi : \forall \{ \Gamma \ A \ B \} \text{ -- } (\times_e)$

$\rightarrow (C : \text{Type})$

$\rightarrow \{ \text{proof} : (C \cong A) \uplus (C \cong B) \}$

$\rightarrow \Gamma \vdash A \times B$

 $\rightarrow \Gamma \vdash C$

⋮

$$\frac{\Gamma \vdash r : A \quad \Gamma \vdash s : B}{\Gamma \vdash \langle r, s \rangle : A \times B} (\times_i)$$

$$\frac{\Gamma \vdash r : A \times B}{\Gamma \vdash \pi_A(r) : A} (\times_e)$$

⋮

$[_] \equiv_- : \forall \{ \Gamma \ A \ B \} \text{ -- } (\equiv)$

$\rightarrow A \equiv B$

$\rightarrow \Gamma \vdash A$

$\rightarrow \Gamma \vdash B$

$$\frac{A \equiv B \quad \Gamma \vdash r : A}{\Gamma \vdash r : B} (\equiv)$$

Ejemplo

$_ : \emptyset, T \Rightarrow T \vdash T \Rightarrow T$

$_ = \lambda (' S Z . (' S Z . ' Z)) \text{ -- } \lambda x . y (y x)$

Ejemplo

$_ : \emptyset, T \Rightarrow T \vdash T \Rightarrow T$

$_ = \lambda (' S Z . (' S Z . ' Z)) \text{ -- } \lambda x . y (y x)$

$_ : \emptyset \vdash (T \Rightarrow T) \Rightarrow T \Rightarrow T$

$_ = \lambda \lambda (' S Z . (' S Z . ' Z)) \text{ -- } \lambda y . \lambda x . y (y x)$

$$\begin{aligned} & _ : \emptyset, T \Rightarrow T \vdash T \Rightarrow T \\ & _ = \lambda ('S\ Z \cdot ('S\ Z \cdot 'Z)) \text{ -- } \lambda\ x \cdot y\ (y\ x) \end{aligned}$$
$$\begin{aligned} & _ : \emptyset \vdash (T \Rightarrow T) \Rightarrow T \Rightarrow T \\ & _ = \lambda\ \lambda ('S\ Z \cdot ('S\ Z \cdot 'Z)) \text{ -- } \lambda\ y \cdot \lambda\ x \cdot y\ (y\ x) \end{aligned}$$
$$\begin{aligned} \text{swap}^\times & : \emptyset \vdash A \times B \Rightarrow B \times A \\ \text{swap}^\times & = \lambda \langle \pi\ B\ ('Z), \pi\ A\ ('Z) \rangle \end{aligned}$$

Substituciones explícitas

$\text{Subst} : \text{Context} \rightarrow \text{Context} \rightarrow \text{Set}$

$\text{Subst } \Gamma \Delta = \forall \{A\} \rightarrow \Gamma \ni A \rightarrow \Delta \vdash A$

Substituciones explícitas

$\text{Subst} : \text{Context} \rightarrow \text{Context} \rightarrow \text{Set}$

$\text{Subst } \Gamma \Delta = \forall \{A\} \rightarrow \Gamma \ni A \rightarrow \Delta \vdash A$

$\text{ids} : \forall \{\Gamma\} \rightarrow \text{Subst } \Gamma \Gamma$

$\text{ids } x = 'x$

Substituciones explícitas

$\text{Subst} : \text{Context} \rightarrow \text{Context} \rightarrow \text{Set}$
 $\text{Subst } \Gamma \Delta = \forall \{A\} \rightarrow \Gamma \ni A \rightarrow \Delta \vdash A$

$\text{ids} : \forall \{\Gamma\} \rightarrow \text{Subst } \Gamma \Gamma$
 $\text{ids } x = 'x$

$\bullet : \forall \{\Gamma \Delta A\} \rightarrow (\Delta \vdash A) \rightarrow \text{Subst } \Gamma \Delta \rightarrow \text{Subst } (\Gamma , A) \Delta$
 $(M \bullet \sigma) Z = M$
 $(M \bullet \sigma) (S x) = \sigma x$

Substituciones explícitas

$_[-] : \forall \{ \Gamma \ A \ B \}$

$\rightarrow \Gamma , B \vdash A$

$\rightarrow \Gamma \vdash B$

$\rightarrow \Gamma \vdash A$

$N \ [\ M \] = \langle\langle \ M \bullet \text{ids} \ \rangle\rangle \ N$

Substituciones explícitas

$\text{Rename} : \text{Context} \rightarrow \text{Context} \rightarrow \text{Set}$

$\text{Rename } \Gamma \Delta = \forall \{A\} \rightarrow \Gamma \ni A \rightarrow \Delta \ni A$

$\text{rename} : \forall \{\Gamma \Delta\} \rightarrow \text{Rename } \Gamma \Delta \rightarrow \Gamma \vdash A \rightarrow \Delta \vdash A$

$\text{rename } \rho N = \dots$

Substituciones explícitas

$\text{exts} : \forall \{ \Gamma \ \Delta \ A \}$

$\rightarrow \text{Subst } \Gamma \ \Delta$

 $\rightarrow \text{Subst } (\Gamma, A) (\Delta, A)$

$\text{exts } \sigma = ' Z \bullet \uparrow \circ \sigma$

Substituciones explícitas

$\text{exts} : \forall \{ \Gamma \Delta A \}$

$\rightarrow \text{Subst } \Gamma \Delta$

 $\rightarrow \text{Subst } (\Gamma, A) (\Delta, A)$

$\text{exts } \sigma = ' Z \bullet \uparrow \circ \sigma$

$\uparrow : \forall \{ \Gamma A B \} \rightarrow \Gamma \vdash A \rightarrow \Gamma, B \vdash A$

$\uparrow = \lambda N \rightarrow \text{rename } S_ N$

Substituciones explícitas

$\text{exts} : \forall \{ \Gamma \Delta A \}$

$\rightarrow \text{Subst } \Gamma \Delta$

$\rightarrow \text{Subst } (\Gamma, A) (\Delta, A)$

$\text{exts } \sigma = \text{'Z} \bullet \uparrow \circ \sigma$

$\uparrow : \forall \{ \Gamma A B \} \rightarrow \Gamma \vdash A \rightarrow \Gamma, B \vdash A$

$\uparrow = \lambda N \rightarrow \text{rename } S_ N$

$\langle\!\langle\!-\!\rangle\!\rangle_ : \forall \{ \Gamma \Delta A \} \rightarrow \text{Subst } \Gamma \Delta \rightarrow \Gamma \vdash A \rightarrow \Delta \vdash A$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (\text{' } k) = \sigma k$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (\lambda N) = \lambda (\langle\!\langle\! \text{exts } \sigma \!\rangle\!\rangle N)$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (L \cdot M) = (\langle\!\langle\! \sigma \!\rangle\!\rangle L) \cdot (\langle\!\langle\! \sigma \!\rangle\!\rangle M)$

$\langle\!\langle\! \sigma \!\rangle\!\rangle \langle M, N \rangle = \langle \langle\!\langle\! \sigma \!\rangle\!\rangle M, \langle\!\langle\! \sigma \!\rangle\!\rangle N \rangle$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (\pi C \{p\} L) = \pi C \{p\} (\langle\!\langle\! \sigma \!\rangle\!\rangle L)$

$\langle\!\langle\! \sigma \!\rangle\!\rangle \star = \star$

Substituciones explícitas

$\text{exts} : \forall \{ \Gamma \Delta A \}$

$\rightarrow \text{Subst } \Gamma \Delta$

$\rightarrow \text{Subst } (\Gamma, A) (\Delta, A)$

$\text{exts } \sigma = \text{'Z} \bullet \uparrow \circ \sigma$

$\uparrow : \forall \{ \Gamma A B \} \rightarrow \Gamma \vdash A \rightarrow \Gamma, B \vdash A$

$\uparrow = \lambda N \rightarrow \text{rename } S_ N$

$\langle\!\langle\!-\!\rangle\!\rangle_ : \forall \{ \Gamma \Delta A \} \rightarrow \text{Subst } \Gamma \Delta \rightarrow \Gamma \vdash A \rightarrow \Delta \vdash A$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (\text{' } k) = \sigma k$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (\lambda N) = \lambda (\langle\!\langle\! \text{exts } \sigma \!\rangle\!\rangle N)$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (L \cdot M) = (\langle\!\langle\! \sigma \!\rangle\!\rangle L) \cdot (\langle\!\langle\! \sigma \!\rangle\!\rangle M)$

$\langle\!\langle\! \sigma \!\rangle\!\rangle \langle M, N \rangle = \langle \langle\!\langle\! \sigma \!\rangle\!\rangle M, \langle\!\langle\! \sigma \!\rangle\!\rangle N \rangle$

$\langle\!\langle\! \sigma \!\rangle\!\rangle (\pi C \{p\} L) = \pi C \{p\} (\langle\!\langle\! \sigma \!\rangle\!\rangle L)$

$\langle\!\langle\! \sigma \!\rangle\!\rangle \star = \star$

Reducción I

data $_ \hookrightarrow _ : (\Gamma \vdash A) \rightarrow (\Gamma \vdash A) \rightarrow \text{Set}$ where

$\xi_{- \cdot 1} : \forall \{t \ t' : \Gamma \vdash A \Rightarrow B\} \{s : \Gamma \vdash A\}$
 $\rightarrow t \hookrightarrow t'$

 $\rightarrow t \cdot s \hookrightarrow t' \cdot s$

$\xi_{- \cdot 2} : \forall \{t : \Gamma \vdash A \Rightarrow B\} \{s \ s' : \Gamma \vdash A\}$
 $\rightarrow s \hookrightarrow s'$

 $\rightarrow t \cdot s \hookrightarrow t \cdot s'$

$\zeta : \forall \{t \ t' : \Gamma, B \vdash A\}$
 $\rightarrow t \hookrightarrow t'$

 $\rightarrow \lambda t \hookrightarrow \lambda t'$

⋮

$$\frac{t \hookrightarrow t'}{t \ s \hookrightarrow t' \ s}$$

$$\frac{s \hookrightarrow s'}{t \ s \hookrightarrow t \ s'}$$

$$\frac{t \hookrightarrow t'}{\lambda t \hookrightarrow \lambda t'}$$

$$\begin{array}{c} \vdots \\ \beta\text{-}\lambda : \forall \{t : \Gamma, A \vdash B\} \{s : \Gamma \vdash A\} \\ \hline \rightarrow (\lambda t) . s \hookrightarrow t [s] \\ \vdots \end{array}$$

$$(\lambda t) s \hookrightarrow t[s]$$

Reducción III

⋮

$$\xi-\langle, \rangle_1 : \forall \{r \ r' : \Gamma \vdash A\} \{s : \Gamma \vdash B\}$$

$$\rightarrow r \hookrightarrow r'$$

$$\rightarrow \langle r, s \rangle \hookrightarrow \langle r', s \rangle$$

$$\xi-\langle, \rangle_2 : \forall \{r : \Gamma \vdash A\} \{s \ s' : \Gamma \vdash B\}$$

$$\rightarrow s \hookrightarrow s'$$

$$\rightarrow \langle r, s \rangle \hookrightarrow \langle r, s' \rangle$$

$$\xi-\pi : \forall \{t \ t' : \Gamma \vdash A \times B\}$$

$$\rightarrow t \hookrightarrow t'$$

$$\rightarrow \pi \ C \ \{p\} \ t \hookrightarrow \pi \ C \ \{p\} \ t'$$

⋮

$$\frac{r \hookrightarrow r'}{\langle r, s \rangle \hookrightarrow \langle r', s \rangle}$$

$$\frac{s \hookrightarrow s'}{\langle r, s \rangle \hookrightarrow \langle r, s' \rangle}$$

$$\frac{t \hookrightarrow t'}{\pi_C t \hookrightarrow \pi_C t'}$$

$$\begin{array}{l}
 \vdots \\
 \beta\text{-}\pi_1 : \forall \{r : \Gamma \vdash A\} \{s : \Gamma \vdash B\} \\
 \hline
 \rightarrow \pi A \{ \text{inj}_1 \text{ refl} \} \langle r, s \rangle \hookrightarrow r \\
 \\
 \beta\text{-}\pi_2 : \forall \{r : \Gamma \vdash A\} \{s : \Gamma \vdash B\} \\
 \hline
 \rightarrow \pi B \{ \text{inj}_2 \text{ refl} \} \langle r, s \rangle \hookrightarrow s \\
 \vdots
 \end{array}$$

Si $r : A$ $\pi_A \langle r, s \rangle \hookrightarrow r$

⋮

$\xi{-}\equiv : \forall \{t \ t' : \Gamma \vdash A\} \{iso : A \equiv B\}$

$\rightarrow t \hookrightarrow t'$

$\rightarrow [iso] \equiv t \hookrightarrow [iso] \equiv t'$

$$\frac{t \hookrightarrow t'}{[iso] \equiv t \hookrightarrow [iso] \equiv t'}$$

$$(\pi_{A \rightarrow A} \langle \lambda x.x, \star \rangle) y \hookrightarrow_{\pi} (\lambda x.x) y$$

$$(\pi_{A \rightarrow A} \langle \lambda x.x, \star \rangle) y \hookrightarrow_{\pi} (\lambda x.x) y$$

$$T_3 : \emptyset, \top \vdash \top$$

$$T_3 = (\pi (\top \Rightarrow \top) \langle \lambda ' Z, \star \rangle) . ' Z$$

$$_ : T_3 \hookrightarrow (\lambda ' Z) . ' Z$$

$$_ = \xi^{-\cdot 1} \beta^{-\pi_1}$$

Isomorfismos de tipos

data $_ \equiv _$: Type \rightarrow Type \rightarrow Set where

comm : $\forall \{A\ B\} \rightarrow A \times B \equiv B \times A$

asso : $\forall \{A\ B\ C\} \rightarrow A \times (B \times C) \equiv (A \times B) \times C$

dist : $\forall \{A\ B\ C\} \rightarrow (A \Rightarrow B) \times (A \Rightarrow C) \equiv A \Rightarrow B \times C$

curry : $\forall \{A\ B\ C\} \rightarrow A \Rightarrow B \Rightarrow C \equiv (A \times B) \Rightarrow C$

id- \times : $\forall \{A\} \rightarrow A \times \top \equiv A$

id- \Rightarrow : $\forall \{A\} \rightarrow \top \Rightarrow A \equiv A$

abs : $\forall \{A\} \rightarrow A \Rightarrow \top \equiv \top$

sym : $\forall \{A\ B\} \rightarrow A \equiv B \rightarrow B \equiv A$

cong \Rightarrow_1 : $\forall \{A\ B\ C\} \rightarrow A \equiv B \rightarrow A \Rightarrow C \equiv B \Rightarrow C$

cong \Rightarrow_2 : $\forall \{A\ B\ C\} \rightarrow A \equiv B \rightarrow C \Rightarrow A \equiv C \Rightarrow B$

cong \times_1 : $\forall \{A\ B\ C\} \rightarrow A \equiv B \rightarrow A \times C \equiv B \times C$

cong \times_2 : $\forall \{A\ B\ C\} \rightarrow A \equiv B \rightarrow C \times A \equiv C \times B$

$_ : \forall \{A\ B\} \rightarrow A \times B \Rightarrow T \equiv B \times A \Rightarrow T$
 $_ = \text{cong} \Rightarrow_1 \text{comm}$

$T_4 : \forall \{A\ B\} \rightarrow \emptyset \vdash (A \Rightarrow B \Rightarrow A) \times (A \Rightarrow B \Rightarrow B)$
 $T_4 = [\text{sym dist}] \equiv (\lambda [\text{sym dist}] \equiv (\lambda \langle \text{' (S Z) , ' Z} \rangle))$

Equivalencias entre términos

$$\langle r, s \rangle \Leftrightarrow \langle s, r \rangle$$

```
data _ $\Leftrightarrow$ _ : ( $\Gamma \vdash A$ )  $\rightarrow$  ( $\Gamma \vdash A$ )  $\rightarrow$  Set where
  comm :  $\forall \{r : \Gamma \vdash A\} \rightarrow \{s : \Gamma \vdash B\}$ 
     $\rightarrow$  [ comm ]  $\equiv \langle r, s \rangle \Leftrightarrow \langle s, r \rangle$ 
  sym-comm :  $\forall \{r : \Gamma \vdash A\} \rightarrow \{s : \Gamma \vdash B\}$ 
     $\rightarrow$  [ sym comm ]  $\equiv \langle r, s \rangle \Leftrightarrow \langle s, r \rangle$ 
  ...
```

Equivalencias entre términos

$$\langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle$$

⋮

$\text{asso} : \forall \{r : \Gamma \vdash A\} \rightarrow \{s : \Gamma \vdash B\} \rightarrow \{t : \Gamma \vdash C\}$
 $\rightarrow [\text{asso}] \equiv \langle r, \langle s, t \rangle \rangle \Leftrightarrow \langle \langle r, s \rangle, t \rangle$

$\text{sym-asso} : \forall \{r : \Gamma \vdash A\} \rightarrow \{s : \Gamma \vdash B\} \rightarrow \{t : \Gamma \vdash C\}$
 $\rightarrow [\text{sym asso}] \equiv \langle \langle r, s \rangle, t \rangle \Leftrightarrow \langle r, \langle s, t \rangle \rangle$

⋮

$$\langle r, s \rangle \xleftrightarrow{\text{SPLIT}}$$

$$\langle r, s \rangle \xleftrightarrow{\text{SPLIT}} \langle r, \langle \pi_B(s), \pi_C(s) \rangle \rangle$$

Equivalencias entre términos

$$\langle r, s \rangle \xleftrightarrow{\text{SPLIT}} \langle r, \langle \pi_B(s), \pi_C(s) \rangle \rangle \xleftrightarrow{\text{ASSO}} \langle \langle r, \pi_B(s) \rangle, \pi_C(s) \rangle$$

Equivalencias entre términos

$$\langle r, s \rangle \xleftrightarrow{\text{SPLIT}} \langle r, \langle \pi_B(s), \pi_C(s) \rangle \rangle \xleftrightarrow{\text{ASSO}} \langle \langle r, \pi_B(s) \rangle, \pi_C(s) \rangle$$

⋮

asso-split : $\forall \{r : \Gamma \vdash A\} \rightarrow \{s : \Gamma \vdash B \times C\}$
 $\rightarrow [\text{asso}] \equiv \langle r, s \rangle \rightleftharpoons \langle \langle r, \pi_B s \rangle, \pi_C s \rangle$

sym-asso-split : $\forall \{r : \Gamma \vdash A \times B\} \rightarrow \{s : \Gamma \vdash C\}$
 $\rightarrow [\text{sym asso}] \equiv \langle r, s \rangle \rightleftharpoons \langle \pi_A r, \langle \pi_B r, s \rangle \rangle$

⋮

Equivalencias entre términos

$$\lambda \langle r, s \rangle \Leftrightarrow \langle \lambda r, \lambda s \rangle$$

⋮

$$\begin{aligned} \text{dist-}\lambda &: \forall \{r : \Gamma, C \vdash A\} \rightarrow \{s : \Gamma, C \vdash B\} \\ &\rightarrow [\text{dist}] \equiv \langle \lambda r, \lambda s \rangle \Leftrightarrow \lambda \langle r, s \rangle \end{aligned}$$

$$\begin{aligned} \text{sym-dist-}\lambda &: \forall \{r : \Gamma, C \vdash A\} \rightarrow \{s : \Gamma, C \vdash B\} \\ &\rightarrow [\text{sym dist}] \equiv (\lambda \langle r, s \rangle) \Leftrightarrow \langle \lambda r, \lambda s \rangle \end{aligned}$$

⋮

Equivalencias entre términos

$$\lambda x^A. \lambda y^B. r \iff \lambda z^{A \times B}. r[\pi_B(z)/y, \pi_A(z)/x]$$

Equivalencias entre términos

$$\lambda x^A. \lambda y^B. r \iff \lambda z^{A \times B}. r[\pi_B(z)/y, \pi_A(z)/x]$$

$$\lambda r[\pi_B(0), \pi_A(0)]$$

Equivalencias entre términos

$$\lambda x^A. \lambda y^B. r \iff \lambda z^{A \times B}. r[\pi_B(z)/y, \pi_A(z)/x]$$

$$\lambda r[\pi_B(0), \pi_A(0)]$$

$\sigma\text{-curry} : \forall \{ \Gamma \ A \ B \} \rightarrow \text{Subst} \ (\Gamma \ , \ A \ , \ B) \ (\Gamma \ , \ A \times B)$

$\sigma\text{-curry} = \pi \ B \ (' \ Z) \bullet \pi \ A \ (' \ Z) \bullet \text{ids} \circ S_-$

Equivalencias entre términos

$$\lambda x^A. \lambda y^B. r \Leftrightarrow \lambda z^{A \times B}. r[\pi_B(z)/y, \pi_A(z)/x]$$

$$\lambda r[\pi_B(0), \pi_A(0)]$$

$\sigma\text{-curry} : \forall \{ \Gamma \ A \ B \} \rightarrow \text{Subst } (\Gamma, A, B) (\Gamma, A \times B)$

$\sigma\text{-curry} = \pi \ B \ (' \ Z) \bullet \pi \ A \ (' \ Z) \bullet \text{ids} \circ S_$

\vdots

$\text{curry} : \forall \{ r : \Gamma, A, B \vdash C \}$
 $\rightarrow [\text{curry}] \equiv (\lambda \lambda \ r) \Leftrightarrow \lambda \ll \sigma\text{-curry} \gg r$

\vdots

Equivalencias entre términos

$$\lambda x^{A \times B}.r \iff \lambda y^A.\lambda z^B.r[\langle y, z \rangle/x]$$

Equivalencias entre términos

$$\lambda x^{A \times B}.r \Leftrightarrow \lambda y^A.\lambda z^B.r[\langle y, z \rangle / x]$$

$$\lambda \lambda r[\langle 1, 0 \rangle]$$

Equivalencias entre términos

$$\lambda x^{A \times B}.r \Leftrightarrow \lambda y^A.\lambda z^B.r[\langle y, z \rangle / x]$$

$$\lambda \lambda r[\langle 1, 0 \rangle]$$

$\sigma\text{-uncurry} : \forall \{ \Gamma \ A \ B \} \rightarrow \text{Subst } (\Gamma, A \times B) (\Gamma, A, B)$

$\sigma\text{-uncurry} = \langle ' \text{S Z}, ' \text{Z} \rangle \bullet \text{ids} \circ \text{S}_- \circ \text{S}_-$

Equivalencias entre términos

$$\lambda x^{A \times B}.r \Leftrightarrow \lambda y^A.\lambda z^B.r[\langle y, z \rangle / x]$$

$$\lambda \lambda r[\langle 1, 0 \rangle]$$

$\sigma\text{-uncurry} : \forall \{ \Gamma \ A \ B \} \rightarrow \text{Subst } (\Gamma, A \times B) (\Gamma, A, B)$

$\sigma\text{-uncurry} = \langle 'S\ Z, 'Z \rangle \bullet \text{ids} \circ S_ \circ S_$

\vdots

$\text{uncurry} : \forall \{ r : \Gamma, A \times B \vdash C \}$

$\rightarrow [\text{sym curry}] \equiv (\lambda r) \Leftrightarrow \lambda \lambda \ll \sigma\text{-uncurry} \gg r$

\vdots

Equivalencias entre términos

⋮

$$\xi-\langle, \rangle_1 : \forall \{r \ r' : \Gamma \vdash A\} \{s : \Gamma \vdash B\}$$

$$\rightarrow r \rightleftharpoons r'$$

$$\rightarrow \langle r, s \rangle \rightleftharpoons \langle r', s \rangle$$

$$\xi-\langle, \rangle_2 : \forall \{r : \Gamma \vdash A\} \{s \ s' : \Gamma \vdash B\}$$

$$\rightarrow s \rightleftharpoons s'$$

$$\rightarrow \langle r, s \rangle \rightleftharpoons \langle r, s' \rangle$$

$$\xi-\pi : \forall \{t \ t' : \Gamma \vdash A \times B\}$$

$$\rightarrow t \rightleftharpoons t'$$

$$\rightarrow \pi \ C \ \{p\} \ t \rightleftharpoons \pi \ C \ \{p\} \ t'$$

$$\xi-\equiv : \forall \{t \ t' : \Gamma \vdash A\} \{iso : A \equiv B\}$$

$$\rightarrow t \rightleftharpoons t'$$

$$\rightarrow ([iso] \equiv t) \rightleftharpoons ([iso] \equiv t')$$

$$T_4 : \forall \{A\ B\} \rightarrow \emptyset \vdash (A \Rightarrow B \Rightarrow A) \times (A \Rightarrow B \Rightarrow B)$$

$$T_4 = [\text{sym dist}] \equiv (\lambda [\text{sym dist}] \equiv (\lambda \langle ' (S\ Z) , ' Z \rangle))$$

$$_ : T_4 \Leftrightarrow [\text{sym dist}] \equiv (\lambda \langle \lambda ' (S\ Z) , \lambda ' Z \rangle)$$

$$_ = \xi\text{-}\equiv (\zeta\ \text{sym-dist-}\lambda)$$

$$\lambda x. \lambda y. \langle x, y \rangle \Leftrightarrow \lambda x. \langle \lambda y. x, \lambda y. y \rangle$$

- 1 Marco teórico
 - Introducción a LCST
 - Introducción a Sistema I
- 2 Aportes
 - Formalización
 - Normalización fuerte
- 3 Conclusiones

Definición (SN)

Un término s es fuertemente normalizante si todos sus reductos también lo son:

$$\frac{\forall t. s \hookrightarrow t \implies t \in SN}{s \in SN}$$

Definición (SN)

Un término s es fuertemente normalizante si todos sus reductos también lo son:

$$\frac{\forall t. s \hookrightarrow t \implies t \in SN}{s \in SN}$$

```
data SN { Γ A } (t : Γ ⊢ A) : Set where  
  sn : (∀ {t'} → t ↦ t' → SN t') → SN t
```

Ejemplos

$SN_{\star} : SN \star$

$SN_{\star} = sn \ (\lambda \ ())$

Ejemplos

$SN_{\star} : SN \ \star$
 $SN_{\star} = sn \ (\lambda \ ())$

$SN_{\langle, \rangle} : SN \ \langle \star, 'Z' \rangle$
 $SN_{\langle, \rangle} = sn \ \lambda \{ (\xi - \langle, \rangle_1 \ ())$
 $\quad ; (\xi - \langle, \rangle_2 \ ()) \}$

$SN_{\star} : SN \ \star$
 $SN_{\star} = sn \ (\lambda \ ())$

$SN_{\langle, \rangle} : SN \ \langle \star, 'Z' \rangle$
 $SN_{\langle, \rangle} = sn \ \lambda \{ (\xi - \langle, \rangle_1 \ ())$
 $\quad ; (\xi - \langle, \rangle_2 \ ()) \}$

$SN_{\pi} : SN \ (\pi \top \{inj_1 \ refl\} \ \langle \star, 'Z' \rangle)$
 $SN_{\pi} = sn \ \lambda \{ (\xi - \pi \ (\xi - \langle, \rangle_1 \ ()))$
 $\quad ; (\xi - \pi \ (\xi - \langle, \rangle_2 \ ()))$
 $\quad ; \beta - \pi_1 \rightarrow SN_{\star} \}$

Approach 1

Realizamos inducción directa sobre los términos:

`strong-norm` : $\forall \{ \Gamma \ A \} \ (t : \Gamma \vdash A) \rightarrow \text{SN } t$

Realizamos inducción directa sobre los términos:

```
strong-norm :  $\forall \{ \Gamma \ A \} \ (t : \Gamma \vdash A) \rightarrow SN \ t$   
strong-norm  $\star$            = sn ( $\lambda \ ()$ )  
strong-norm ( $' \ v$ )      = sn ( $\lambda \ ()$ )  
strong-norm ( $\lambda \ t$ )      = lemma- $\lambda$  (strong-norm  $t$ )  
strong-norm ( $a \cdot b$ )     = lemma- $\cdot$  (strong-norm  $a$ ) (strong-norm  $b$ )  
strong-norm  $\langle a \ , \ b \ \rangle$  = lemma- $\langle, \rangle$  (strong-norm  $a$ ) (strong-norm  $b$ )  
strong-norm ( $\pi \_ t$ )     = lemma- $\pi$  (strong-norm  $t$ )
```

- Por H.I. sabemos que SN vale para los subterminos.

Realizamos inducción directa sobre los términos:

```
strong-norm :  $\forall \{ \Gamma \ A \} \ (t : \Gamma \vdash A) \rightarrow SN \ t$   
strong-norm  $\star$            = sn ( $\lambda \ ()$ )  
strong-norm ( $\text{' } v$ )       = sn ( $\lambda \ ()$ )  
strong-norm ( $\lambda \ t$ )       = lemma- $\lambda$  (strong-norm  $t$ )  
strong-norm ( $a \cdot b$ )      = lemma- $\cdot$  (strong-norm  $a$ ) (strong-norm  $b$ )  
strong-norm  $\langle a, b \rangle$     = lemma- $\langle, \rangle$  (strong-norm  $a$ ) (strong-norm  $b$ )  
strong-norm ( $\pi \_ t$ )     = lemma- $\pi$  (strong-norm  $t$ )
```

- Por H.I. sabemos que SN vale para los subterminos.
- Definimos algunos lemas para construir SN para todo el término.

Approach 1

Definimos una función auxiliar para realizar *pattern matching* sobre el paso de reducción:

```
lemma-λ : ∀ {Γ A B} → {t : Γ , B ⊢ A} → SN t → SN (λ t)
lemma-λ SNt = sn (λ step → aux SNt step)
  where aux : ∀ {t λt'} → SN t → (λ t) ↦ λt' → SN λt'
        aux (sn f) (ζ step) = lemma-λ (f step)
```

Approach 1

Definimos una función auxiliar para realizar *pattern matching* sobre el paso de reducción:

```
lemma-λ : ∀ {Γ A B} → {t : Γ , B ⊢ A} → SN t → SN (λ t)
lemma-λ SNt = sn (λ step → aux SNt step)
  where aux : ∀ {t λ t'} → SN t → (λ t) ↦ λ t' → SN λ t'
        aux (sn f) ((ζ step)) = lemma-λ (f step)
```

$step : t \mapsto t'$

Approach 1

Definimos una función auxiliar para realizar *pattern matching* sobre el paso de reducción:

```
lemma-λ : ∀ {Γ A B} → {t : Γ , B ⊢ A} → SN t → SN (λ t)
lemma-λ SNt = sn (λ step → aux SNt step)
  where aux : ∀ {t λ t'} → SN t → (λ t) ⇝ λ t' → SN λ t'
        aux (sn f) (ζ step) = lemma-λ (f step)
```

$step : t \hookrightarrow t'$
 $f\ step : SN\ t'$

Approach 1

Definimos una función auxiliar para realizar *pattern matching* sobre el paso de reducción:

```
lemma-λ : ∀ {Γ A B} → {t : Γ , B ⊢ A} → SN t → SN (λ t)
lemma-λ SNt = sn (λ step → aux SNt step)
  where aux : ∀ {t λt'} → SN t → (λ t) ↦ λt' → SN λt'
         aux (sn f) (ζ step) = lemma-λ (f step)
```

$step : t \hookrightarrow t'$

$f\ step : SN\ t'$

$lemma-\lambda\ (f\ step) : SN\ (\lambda\ t')$

Approach 1

```
lemma-· : ∀ {a : Γ ⊢ A ⇒ B} {b : Γ ⊢ A} → SN a → SN b → SN (a · b)
lemma-· SNa SNb = sn (λ step → aux SNa SNb step)
  where aux : ∀ {a b t'} → SN a → SN b → a · b ⇔ t' → SN t'
        aux (sn f) SNb (ξ-·₁ step) = lemma-· (f step) SNb
        aux SNa (sn f) (ξ-·₂ step) = lemma-· SNa (f step)
        aux {a = λ t} SNa SNb β-λ = {! !} -- SN (t [ b ])
```

Approach 1

```
lemma-· : ∀ {a : Γ ⊢ A ⇒ B} {b : Γ ⊢ A} → SN a → SN b → SN (a · b)
lemma-· SNa SNb = sn (λ step → aux SNa SNb step)
  where aux : ∀ {a b t'} → SN a → SN b → a · b ⇔ t' → SN t'
        aux (sn f) SNb (ξ-·₁ step) = lemma-· (f step) SNb
        aux SNa (sn f) (ξ-·₂ step) = lemma-· SNa (f step)
        aux {a = λ t} SNa SNb β-λ = {! !} -- SN (t [ b ])
```

No podemos concluir nada sobre $t[b]$, de hecho, la substitución podría crear nuevos redexes dentro de t .

Approach 1

El lema de introducción del par es simple...

```
lemma-⟨,⟩ : ∀ {Γ A B} → {a : Γ ⊢ A} {b : Γ ⊢ B} → SN a → SN b → SN ⟨ a , b ⟩
lemma-⟨,⟩ SNa SNb = sn (λ step → aux SNa SNb step)
  where aux : ∀ {a b t'} → SN a → SN b → ⟨ a , b ⟩ ⇔ t' → SN t'
        aux (sn f) SNb (ξ-⟨,⟩1 step) = lemma-⟨,⟩ (f step) SNb
        aux SNa (sn f) (ξ-⟨,⟩2 step) = lemma-⟨,⟩ SNa (f step)
```

Approach 1

Pero la eliminación también da problemas:

```
lemma- $\pi$  :  $\forall \{ \Gamma \ A \ B \ C \ p \} \rightarrow \{ t : \Gamma \vdash A \times B \} \rightarrow \text{SN } t \rightarrow \text{SN } (\pi \ C \ \{p\} \ t)$   
lemma- $\pi$   $\text{SN}t$  =  $\text{sn}$  ( $\lambda \text{ step} \rightarrow \text{aux } \text{SN}t \ \text{step}$ )  
  where  $\text{aux} : \forall \{ C \ p \ t \ t' \} \rightarrow \text{SN } t \rightarrow (\pi \ C \ \{p\} \ t) \hookrightarrow t' \rightarrow \text{SN } t'$   
     $\text{aux} (\text{sn } f) (\xi\text{-}\pi \ \text{step}) = \text{lemma-}\pi \ (f \ \text{step})$   
     $\text{aux} \{ t = \langle a , b \rangle \} (\text{sn } f) \beta\text{-}\pi_1 = \{! \ !\} \text{ -- SN } a$   
     $\text{aux} \{ t = \langle a , b \rangle \} (\text{sn } f) \beta\text{-}\pi_2 = \{! \ !\} \text{ -- SN } b$ 
```


Approach 2

Definimos un SN más general que añade un predicado sobre el término:

```
data SN* { $\Gamma$   $A$ } ( $P : \Gamma \vdash A \rightarrow \text{Set}$ ) ( $t : \Gamma \vdash A$ ) : Set where  
  sn* : ( $P$   $t$ )  $\rightarrow$  ( $\forall \{t'\} \rightarrow t \hookrightarrow t' \rightarrow \text{SN* } P$   $t'$ )  $\rightarrow \text{SN* } P$   $t$ 
```

Approach 2

Definimos un SN más general que añade un predicado sobre el término:

```
data SN* {Γ A} (P : Γ ⊢ A → Set) (t : Γ ⊢ A) : Set where
  sn* : (P t) → (∀ {t'} → t ↪ t' → SN* P t') → SN* P t
```

Es fácil ver que SN^* implica SN :

```
SN*-SN : ∀ {Γ A P} {t : Γ ⊢ A} → SN* P t → SN t
SN*-SN (sn* P Snt) = sn (λ step → SN*-SN (Snt step))
```

Approach 2

Definimos un SN más general que añade un predicado sobre el término:

```
data SN* {Γ A} (P : Γ ⊢ A → Set) (t : Γ ⊢ A) : Set where
  sn* : (P t) → (∀ {t'} → t ↪ t' → SN* P t') → SN* P t
```

Es fácil ver que SN^* implica SN :

```
SN*-SN : ∀ {Γ A P} {t : Γ ⊢ A} → SN* P t → SN t
SN*-SN (sn* P Snt) = sn (λ step → SN*-SN (Snt step))
```

Luego definimos la **interpretación del término**:

```
[_] : ∀ {Γ A} → Γ ⊢ A → Set
[ (λ t) ] = ∀ {u} → SN* [ ] u → SN* [ ] (t [ u ])
[ < a , b > ] = SN* [ ] a ⊗ SN* [ ] b
[ t ] = Top
```

Approach 2

La prueba queda dividida en dos pasos:

- Primero se prueba el teorema fundamental:

$$\text{adequacy} : \forall \{ \Gamma \ A \} \rightarrow (t : \Gamma \vdash A) \rightarrow \text{SN}^* \llbracket - \rrbracket t$$

Approach 2

La prueba queda dividida en dos pasos:

- Primero se prueba el teorema fundamental:

$$\text{adequacy} : \forall \{ \Gamma A \} \rightarrow (t : \Gamma \vdash A) \rightarrow \text{SN}^* \llbracket - \rrbracket t$$

- Luego es fácil probar SN :

$$\begin{aligned} \text{strong-norm} &: \forall \{ \Gamma A \} (t : \Gamma \vdash A) \rightarrow \text{SN } t \\ \text{strong-norm } t &= \text{SN}^* \text{-SN } (\text{adequacy } t) \end{aligned}$$

- La interpretación de la abstracción permite probar el caso de la aplicación.

```

lemma-· : ∀ {a : Γ ⊢ A ⇒ B} {b : Γ ⊢ A} → SN* [-] a → SN* [-] b → SN* [-] (a · b)
lemma-· SNa SNb = sn* tt (λ step → aux SNa SNb step)
  where aux : ∀ {a b t'} → SN* [-] a → SN* [-] b → a · b ⇔ t' → SN* [-] t'
        aux (sn* _ f) SNb (ξ-·₁ step) = lemma-· (f step) SNb
        aux SNa (sn* _ f) (ξ-·₂ step) = lemma-· SNa (f step)
        aux (sn* (La _) SNb) β-λ      = (La SNb)
  
```

- La interpretación del par resuelve el caso de la proyección.

```

lemma- $\pi$  :  $\forall \{ \Gamma \ A \ B \ C \ p \} \rightarrow \{ t : \Gamma \vdash A \times B \} \rightarrow \text{SN}^* \llbracket \_ \rrbracket t \rightarrow \text{SN}^* \llbracket \_ \rrbracket (\pi \ C \ \{ p \} \ t)$ 
lemma- $\pi$   $\text{SN}t$  =  $\text{sn}^* \ \text{tt} \ (\lambda \ \text{step} \rightarrow \text{aux} \ \text{SN}t \ \text{step})$ 
  where aux :  $\forall \{ C \ p \ t \ t' \} \rightarrow \text{SN}^* \llbracket \_ \rrbracket t \rightarrow (\pi \ C \ \{ p \} \ t) \hookrightarrow t' \rightarrow \text{SN}^* \llbracket \_ \rrbracket t'$ 
    aux (sn* _ f) ( $\xi$ - $\pi$  step) = lemma- $\pi$  (f step)
    aux { t =  $\langle a, b \rangle$  } (sn* (SNa, -) f)  $\beta$ - $\pi_1$  = SNa
    aux { t =  $\langle a, b \rangle$  } (sn* (-, SNb) f)  $\beta$ - $\pi_2$  = SNb
  
```

Approach 2

Las introducciones ahora deben construir las interpretaciones.

- El caso del par es simple:

```
lemma-⟨,⟩ : ∀ {Γ A B} → {a : Γ ⊢ A} {b : Γ ⊢ B} → SN* [[_]] a → SN* [[_]] b → SN*  
lemma-⟨,⟩ SNa SNb = sn* ( SNa , SNb ) (...)
```


Approach 2

Las introducciones ahora deben construir las interpretaciones.

- El caso del par es simple:

$\text{lemma-}\langle, \rangle : \forall \{ \Gamma \ A \ B \} \rightarrow \{ a : \Gamma \vdash A \} \{ b : \Gamma \vdash B \} \rightarrow \text{SN}^* \llbracket _ \rrbracket a \rightarrow \text{SN}^* \llbracket _ \rrbracket b \rightarrow \text{SN}^*$
 $\text{lemma-}\langle, \rangle \text{ SN}a \text{ SN}b = \text{sn}^* (\text{SN}a , \text{SN}b) (\dots)$

- El caso de la abstracción es complejo:

$\text{lemma-}\lambda : \forall \{ \Gamma \ A \ B \} \rightarrow \{ t : \Gamma , B \vdash A \} \rightarrow \llbracket \lambda \ t \rrbracket \rightarrow \text{SN}^* \llbracket _ \rrbracket t \rightarrow \text{SN}^* \llbracket _ \rrbracket (\lambda \ t)$
 $\text{lemma-}\lambda \ Lt \ (\text{sn}^* _ f) = \text{sn}^* \ Lt \ (\dots)$

Las introducciones ahora deben construir las interpretaciones.

- El caso del par es simple:

$\text{lemma-}\langle, \rangle : \forall \{ \Gamma \ A \ B \} \rightarrow \{ a : \Gamma \vdash A \} \{ b : \Gamma \vdash B \} \rightarrow \text{SN}^* \llbracket _ \rrbracket a \rightarrow \text{SN}^* \llbracket _ \rrbracket b \rightarrow \text{SN}^* \llbracket _ \rrbracket \langle a, b \rangle$
 $\text{lemma-}\langle, \rangle \text{ SN} a \text{ SN} b = \text{sn}^* (\text{SN} a , \text{SN} b) (\dots)$

- El caso de la abstracción es complejo:

$\text{lemma-}\lambda : \forall \{ \Gamma \ A \ B \} \rightarrow \{ t : \Gamma , B \vdash A \} \rightarrow \llbracket \lambda t \rrbracket \rightarrow \text{SN}^* \llbracket _ \rrbracket t \rightarrow \text{SN}^* \llbracket _ \rrbracket (\lambda t)$
 $\text{lemma-}\lambda \text{ Lt } (\text{sn}^* _ f) = \text{sn}^* \text{ Lt } (\dots)$

Approach 2

```
adequacy (λ t)      = lemma-λ
                      (λ SNu → {! !}) -- SN* [-] (t [ u ])
                      (adequacy t)
adequacy (a . b)    = lemma-. (adequacy a) (adequacy b)
adequacy ⟨ a , b ⟩ = lemma-⟨,⟩ (adequacy a) (adequacy b)
adequacy (π _ t)    = lemma-π (adequacy t)
```

Approach 3

Generalizamos el teorema fundamental, vamos a probar que vale para cualquier t con cualquier substitución aplicada:

$$\text{adequacy} : \forall \{ \Gamma \Delta A \} \rightarrow (t : \Gamma \vdash A) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{SN}^* [-] (\ll \sigma \gg t)$$

Approach 3

Generalizamos el teorema fundamental, vamos a probar que vale para cualquier t con cualquier substitución aplicada:

$$\text{adequacy} : \forall \{ \Gamma \Delta A \} \rightarrow (t : \Gamma \vdash A) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{SN}^* [-] (\llbracket \sigma \rrbracket t)$$

Approach 3

Generalizamos el teorema fundamental, vamos a probar que vale para cualquier t con cualquier substitución aplicada:

$$\text{adequacy} : \forall \{ \Gamma \Delta A \} \rightarrow (t : \Gamma \vdash A) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{SN}^* [-] (\ll \sigma \gg t)$$
$$\begin{aligned} \text{strong-norm} &: \forall \{ \Gamma A \} (t : \Gamma \vdash A) \rightarrow \text{SN } t \\ \text{strong-norm } t &= (\text{SN}^* \text{-SN } (\text{adequacy } t \text{ (ids)})) \end{aligned}$$

Approach 3

```
adequacy (' v) σ = {! !} -- SN* [-] (σ v)
adequacy (λ t) σ =
  lemma-λ
    (λ {u = u} SNu → (adequacy t (u • σ)))
    (adequacy t (exts σ))
adequacy (a · b) σ = lemma-. (adequacy a σ) (adequacy b σ)
```

Approach 3

```
adequacy (' v) σ = {! !} -- SN* [] (σ v)
adequacy (λ t) σ =
  lemma-λ
    (λ {u = u} SNu → (adequacy t (u • σ)))
    (adequacy t (exts σ))
adequacy (a · b) σ = lemma-· (adequacy a σ) (adequacy b σ)
```

Ahora el problema está en el caso de las variables, una substitución arbitraria podría “romper” el término.

Approach 4

Definimos las **substituciones adecuadas**:

$$\begin{aligned} \models & : \forall \{\Delta\} \rightarrow (\Gamma : \text{Context}) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{Set} \\ \Gamma \models \sigma & = \forall (v : \Gamma \ni A) \rightarrow \text{SN* } \llbracket _ \rrbracket (\sigma \ v) \end{aligned}$$

Approach 4

Definimos las **substituciones adecuadas**:

$$\begin{aligned} \models_{-} &: \forall \{\Delta\} \rightarrow (\Gamma : \text{Context}) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{Set} \\ \Gamma \models \sigma &= \forall (v : \Gamma \ni A) \rightarrow \text{SN}^* \llbracket _ \rrbracket (\sigma \ v) \end{aligned}$$

En particular *ids* es adecuada:

$$\begin{aligned} \models \text{ids} &: \forall \{\Gamma\} \rightarrow \Gamma \models \text{ids} \\ \models \text{ids } _ &= \text{sn}^* \text{ tt } (\lambda \ ()) \end{aligned}$$

Approach 4

Definimos las **substituciones adecuadas**:

$$\begin{aligned} \models_{-} &: \forall \{\Delta\} \rightarrow (\Gamma : \text{Context}) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{Set} \\ \Gamma \models \sigma &= \forall (v : \Gamma \ni A) \rightarrow \text{SN}^* \llbracket _ \rrbracket (\sigma v) \end{aligned}$$

En particular *ids* es adecuada:

$$\begin{aligned} \models_{\text{ids}} &: \forall \{\Gamma\} \rightarrow \Gamma \models \text{ids} \\ \models_{\text{ids}} _ &= \text{sn}^* \text{ tt } (\lambda ()) \end{aligned}$$

El cons entre un término *SN* y una subst. adecuada, también es adecuada:

$$\begin{aligned} \models_{-} \bullet_{-} &: \forall \{\Gamma \Delta A \sigma\} \{t : \Gamma \vdash A\} \rightarrow \text{SN}^* \llbracket _ \rrbracket t \rightarrow \Delta \models \sigma \rightarrow (\Delta, A) \models (t \bullet \sigma) \\ (\models t \bullet \sigma) \text{ Z} &= t \\ (\models t \bullet \sigma) (\text{S } v) &= \sigma v \end{aligned}$$

Approach 4

Definimos las **substituciones adecuadas**:

$$\begin{aligned} \models_{-} &: \forall \{ \Delta \} \rightarrow (\Gamma : \text{Context}) \rightarrow (\sigma : \text{Subst } \Gamma \Delta) \rightarrow \text{Set} \\ \Gamma \models \sigma &= \forall (v : \Gamma \ni A) \rightarrow \text{SN}^* \llbracket - \rrbracket (\sigma v) \end{aligned}$$

En particular *ids* es adecuada:

$$\begin{aligned} \models_{\text{ids}} &: \forall \{ \Gamma \} \rightarrow \Gamma \models \text{ids} \\ \models_{\text{ids}} _ &= \text{sn}^* \text{ tt } (\lambda ()) \end{aligned}$$

El cons entre un término *SN* y una subst. adecuada, también es adecuada:

$$\begin{aligned} \models_{-} \bullet_{-} &: \forall \{ \Gamma \Delta A \sigma \} \{ t : \Gamma \vdash A \} \rightarrow \text{SN}^* \llbracket - \rrbracket t \rightarrow \Delta \models \sigma \rightarrow (\Delta, A) \models (t \bullet \sigma) \\ (\models t \bullet \sigma) \text{ Z} &= t \\ (\models t \bullet \sigma) (\text{S } v) &= \sigma v \end{aligned}$$

Nuevamente fortalecemos el teorema:

$$\text{adequacy} : \forall \{ \Gamma \Delta A \sigma \} \rightarrow (t : \Gamma \vdash A) \rightarrow \boxed{\Gamma \models \sigma} \rightarrow \text{SN}^* \llbracket - \rrbracket (\llbracket \sigma \rrbracket t)$$

Approach 4

```
adequacy (‘  $v$ )  $\models \sigma = \models \sigma v$   
adequacy ( $\lambda t$ )  $\models \sigma =$   
  lemma- $\lambda$   
    ( $\lambda \{u = u\} SNu \rightarrow (\text{adequacy } t (\models SNu \bullet \models \sigma)))$   
    ( $\text{adequacy } t (\models \text{exts } \models \sigma))$ 
```

$$(\ll \text{exts } \sigma \gg t)[u] = \ll u \bullet \sigma \gg t$$

Approach 4

```
adequacy (' v)  $\models \sigma = \models \sigma$  v  
adequacy ( $\lambda$  t)  $\models \sigma =$   
  lemma- $\lambda$   
    ( $\lambda$  {u = u} SNu  $\rightarrow$  (adequacy t ( $\models$  SNu  $\bullet \models \sigma$ )))  
    (adequacy t ( $\models$  exts  $\models \sigma$ ))
```

$$(\ll \text{exts } \sigma \gg t)[u] = \ll u \bullet \sigma \gg t$$

```
strong-norm :  $\forall \{ \Gamma A \} (t : \Gamma \vdash A) \rightarrow \text{SN } t$   
strong-norm t = (SN*-SN (adequacy t  $\models$  ids))
```

Approach 4

```
adequacy (‘  $v$ )  $\models \sigma = \models \sigma$   $v$   
adequacy ( $\lambda$   $t$ )  $\models \sigma =$   
  lemma- $\lambda$   
    ( $\lambda$  { $u = u$ }  $SNu \rightarrow$  (adequacy  $t$  ( $\models SNu \bullet \models \sigma$ )))  
    (adequacy  $t$  ( $\models_{\text{exts}} \models \sigma$ ))
```


$$(\ll \text{exts } \sigma \gg t)[u] = \ll u \bullet \sigma \gg t$$

```
strong-norm :  $\forall \{ \Gamma A \} (t : \Gamma \vdash A) \rightarrow SN t$   
strong-norm  $t = (SN^* - SN$  (adequacy  $t \models_{\text{ids}}$ ))
```

Approach 4

La extensión de una subst. adecuada es también adecuada:

```
 $\models_{\text{exts}} : \forall \{ \Gamma \ \Delta \ A \} \{ \sigma : \text{Subst } \Gamma \ \Delta \} \rightarrow \Gamma \models \sigma \rightarrow (\Gamma , A) \models (\text{exts } \sigma)$   
 $\models_{\text{exts}} \vdash_{\sigma} Z = \text{sn* tt } (\lambda ()) \quad \text{-- SN* } \llbracket \_ \rrbracket (' Z)$   
 $\models_{\text{exts}} \vdash_{\sigma} (S \ v) = \text{SN*-rename } S\_ (\vdash_{\sigma} v) \quad \text{-- SN* } \llbracket \_ \rrbracket (\text{rename } S\_ (\sigma \ v))$ 
```


Approach 4

La extensión de una subst. adecuada es también adecuada:

$$\begin{aligned} \models_{\text{exts}} &: \forall \{ \Gamma \ \Delta \ A \} \ \{ \sigma : \text{Subst } \Gamma \ \Delta \} \rightarrow \Gamma \models \sigma \rightarrow (\Gamma , A) \models (\text{exts } \sigma) \\ \models_{\text{exts}} \vdash_{\sigma} Z &= \text{sn* tt } (\lambda ()) \quad \text{-- SN* } \llbracket _ \rrbracket (' Z) \\ \models_{\text{exts}} \vdash_{\sigma} (S \ v) &= \text{SN*-rename } S_ (\models_{\sigma} v) \quad \text{-- SN* } \llbracket _ \rrbracket (\text{rename } S_ (\sigma \ v)) \end{aligned}$$

Los renombres preservan la propiedad SN^* :

$$\begin{aligned} \text{SN*-rename} &: \forall \{ t \} \rightarrow (\rho : \text{Rename } \Gamma \ \Delta) \rightarrow \text{SN* } \llbracket _ \rrbracket t \rightarrow \text{SN* } \llbracket _ \rrbracket (\text{rename } \rho \ t) \\ \text{SN*-rename } \rho (\text{sn* } Lt \ f) &= \text{sn* } (\llbracket _ \rrbracket \text{-rename } \rho \ Lt) (\dots) \end{aligned}$$

Approach 4

La extensión de una subst. adecuada es también es adecuada:

```
⊢exts : ∀{Γ Δ A} {σ : Subst Γ Δ} → Γ ⊢ σ → (Γ , A) ⊢ (exts σ)
⊢exts ⊢σ Z      = sn* tt (λ ())          -- SN* [] (' Z)
⊢exts ⊢σ (S v) = SN*-rename S_ (⊢σ v)  -- SN* [] (rename S_ (σ v))
```

Los renombres preservan la propiedad SN^* :

```
SN*-rename : ∀{t} → (ρ : Rename Γ Δ) → SN* [] t → SN* [] (rename ρ t)
SN*-rename ρ (sn* Lt f) = sn* ([[]-rename ρ Lt) (...)
```

Los renombres preservan la interpretación:

```
[]-rename : ∀{t : Γ ⊢ A} → (ρ : Rename Γ Δ) → [] t → [] (rename ρ t)
>[]-rename {t = λ n} ρ Ln = λ SNu → {! !}
⋮                               -- SN* [] ((rename ρ n) [ u ])
```

Approach 5

Generalizamos la interpretación de la abstracción:

$$\begin{aligned} \llbracket - \rrbracket &: \forall \{ \Gamma \vdash A \} \rightarrow \Gamma \vdash A \rightarrow \text{Set} \\ \llbracket (\lambda t) \rrbracket &= \forall \{ \rho \} \{ u \} \rightarrow \text{SN}^* \llbracket - \rrbracket u \rightarrow \text{SN}^* \llbracket - \rrbracket (\llbracket u \bullet (\text{ids} \circ \rho) \rrbracket t) \\ \llbracket \langle a, b \rangle \rrbracket &= \text{SN}^* \llbracket - \rrbracket a \otimes \text{SN}^* \llbracket - \rrbracket b \\ \llbracket t \rrbracket &= \text{Top} \end{aligned}$$

Approach 5

Generalizamos la interpretación de la abstracción:

$$\begin{aligned} \llbracket _ \rrbracket &: \forall \{ \Gamma \ A \} \rightarrow \Gamma \vdash A \rightarrow \text{Set} \\ \llbracket (\lambda t) \rrbracket &= \forall \{ \rho \} \{ u \} \rightarrow \text{SN}^* \llbracket _ \rrbracket u \rightarrow \text{SN}^* \llbracket _ \rrbracket (\llbracket u \bullet (\text{ids} \circ \rho) \rrbracket t) \\ \llbracket \langle a, b \rangle \rrbracket &= \text{SN}^* \llbracket _ \rrbracket a \otimes \text{SN}^* \llbracket _ \rrbracket b \\ \llbracket t \rrbracket &= \text{Top} \end{aligned}$$

$$\begin{aligned} \llbracket _ \rrbracket\text{-rename} &: \forall \{ \Gamma \ \Delta \ A \} \{ t : \Gamma \vdash A \} \rightarrow (\rho : \text{Rename } \Gamma \ \Delta) \rightarrow \llbracket t \rrbracket \rightarrow \llbracket \text{rename } \rho \ t \rrbracket \\ \llbracket _ \rrbracket\text{-rename } \{ t = \lambda n \} \rho \ L n \{ \rho_1 \} &= \lambda \{ \text{SN} u \rightarrow L n \{ \rho = \rho_1 \circ \rho \} \text{SN} u \} \\ &\vdots \end{aligned}$$

Generalizamos la interpretación de la abstracción:

$$\begin{aligned}
 \llbracket _ \rrbracket &: \forall \{ \Gamma \ A \} \rightarrow \Gamma \vdash A \rightarrow \text{Set} \\
 \llbracket (\lambda \ t) \rrbracket &= \forall \{ \rho \} \{ u \} \rightarrow \text{SN}^* \llbracket _ \rrbracket u \rightarrow \text{SN}^* \llbracket _ \rrbracket (\llbracket u \bullet (\text{ids} \circ \rho) \rrbracket t) \\
 \llbracket \langle a, b \rangle \rrbracket &= \text{SN}^* \llbracket _ \rrbracket a \otimes \text{SN}^* \llbracket _ \rrbracket b \\
 \llbracket t \rrbracket &= \text{Top}
 \end{aligned}$$

$$\begin{aligned}
 \llbracket _ \rrbracket \text{-rename} &: \forall \{ \Gamma \ \Delta \ A \} \{ t : \Gamma \vdash A \} \rightarrow (\rho : \text{Rename } \Gamma \ \Delta) \rightarrow \llbracket t \rrbracket \rightarrow \llbracket \text{rename } \rho \ t \rrbracket \\
 \llbracket _ \rrbracket \text{-rename } \{ t = \lambda \ n \} \rho \ L n \{ \rho_1 \} &= \lambda \{ \text{SN} u \rightarrow L n \{ \rho = \rho_1 \circ \rho \} \text{SN} u \} \\
 &\vdots
 \end{aligned}$$

La composición entre un renombre y una subst. adecuada, también es adecuada:

$$\begin{aligned}
 \models \text{rename} &: \forall \{ \Gamma \ \Delta \ \Delta_1 \ \sigma \} \rightarrow \Gamma \models \sigma \rightarrow (\rho : \text{Rename } \Delta \ \Delta_1) \rightarrow \Gamma \models (\llbracket \text{ids} \circ \rho \rrbracket \circ \sigma) \\
 \models \text{rename } \models \sigma \ \rho \ v &= (\text{SN}^* \text{-rename } \rho \ (\models \sigma \ v))
 \end{aligned}$$

Approach 5

```
adequacy ( $\lambda$   $t$ )  $\models \sigma$  =  
  lemma- $\lambda$   
    ( $\lambda$  { { $\rho$ }  $SNu \rightarrow$  (adequacy  $t$  ( $\models SNu \bullet$  ( $\models$ rename  $\models \sigma$   $\rho$ )))})  
      (adequacy  $t$  ( $\models$ exts  $\models \sigma$ )))
```

```
adequacy ( $\lambda$   $t$ )  $\models \sigma$  =  
  lemma- $\lambda$   
    ( $\lambda$  { { $\rho$ }  $SNu$   $\rightarrow$  (adequacy  $t$  ( $\models SNu \bullet (\models \text{rename } \models \sigma \rho)))$  } )  
    (adequacy  $t$  ( $\models \text{exts } \models \sigma$ ))
```

Prueba para Sistema I

Definimos la SN para la unión entre \hookrightarrow y \Leftrightarrow :

$_ \rightsquigarrow _ : \forall \{ \Gamma \ A \} \rightarrow (t \ t' : \Gamma \vdash A) \rightarrow \text{Set}$

$t \rightsquigarrow t' = t \hookrightarrow t' \uplus t \Leftrightarrow t'$

`data SN {Γ A} (t : Γ ⊢ A) : Set where`

`sn : (∀ {t'} → t ~> t' → SN t') → SN t`

Prueba para Sistema I

Definimos la SN para la unión entre \hookrightarrow y \rightleftarrows :

$$_ \rightsquigarrow _ : \forall \{ \Gamma \ A \} \rightarrow (t \ t' : \Gamma \vdash A) \rightarrow \mathbf{Set}$$
$$t \rightsquigarrow t' = t \hookrightarrow t' \uplus t \rightleftarrows t'$$

data $SN \{ \Gamma \ A \} (t : \Gamma \vdash A) : \mathbf{Set}$ where

$sn : (\forall \{ t' \} \rightarrow t \rightsquigarrow t' \rightarrow SN \ t') \rightarrow SN \ t$

La suma \uplus funciona igual al **Either** de Haskell:

Prueba para Sistema I

Definimos la SN para la unión entre \hookrightarrow y \rightleftarrows :

$$_ \rightsquigarrow _ : \forall \{ \Gamma \ A \} \rightarrow (t \ t' : \Gamma \vdash A) \rightarrow \mathbf{Set}$$
$$t \rightsquigarrow t' = t \hookrightarrow t' \uplus t \rightleftarrows t'$$

data $SN \ \{ \Gamma \ A \} \ (t : \Gamma \vdash A) : \mathbf{Set}$ where

$$\mathbf{sn} : (\forall \{ t' \} \rightarrow t \rightsquigarrow t' \rightarrow SN \ t') \rightarrow SN \ t$$

La suma \uplus funciona igual al **Either** de Haskell:

$$\mathbf{data \ Either \ a \ b = \ Left \ a \mid \ Right \ b}$$

Prueba para Sistema I

Definimos la SN para la unión entre \hookrightarrow y \rightleftarrows :

$$\begin{aligned} _ \rightsquigarrow _ &: \forall \{ \Gamma \ A \} \rightarrow (t \ t' : \Gamma \vdash A) \rightarrow \mathbf{Set} \\ t \rightsquigarrow t' &= t \hookrightarrow t' \uplus t \rightleftarrows t' \end{aligned}$$

```
data SN {Γ A} (t : Γ ⊢ A) : Set where
  sn : (∀ {t'} → t ~> t' → SN t') → SN t
```

La suma \uplus funciona igual al **Either** de Haskell:

$$\text{data } \mathbf{Either} \ a \ b = \mathbf{Left} \ a \mid \mathbf{Right} \ b$$

Donde **Left** y **Right** se llaman inj_1 y inj_2 resp.

Prueba para Sistema I

Definimos la SN para la unión entre \hookrightarrow y \rightleftharpoons :

$$_ \rightsquigarrow _ : \forall \{ \Gamma \ A \} \rightarrow (t \ t' : \Gamma \vdash A) \rightarrow \mathbf{Set}$$
$$t \rightsquigarrow t' = t \hookrightarrow t' \uplus t \rightleftharpoons t'$$

```
data SN {Γ A} (t : Γ ⊢ A) : Set where
  sn : (∀ {t'} → t ~> t' → SN t') → SN t
```

La suma \uplus funciona igual al **Either** de Haskell:

$$\text{data } \mathbf{Either} \ a \ b = \mathbf{Left} \ a \mid \mathbf{Right} \ b$$

Donde **Left** y **Right** se llaman inj_1 y inj_2 resp.

```
data SN* {Γ A} (P : Γ ⊢ A → Set) (t : Γ ⊢ A) : Set where
  sn* : P t → (∀ {t'} → t ~> t' → SN* P t') → SN* P t
```

Prueba para Sistema I

Se agrega el caso en adequacy:

```
⋮  
adequacy ([ iso ]  $\equiv$  n)  $\models \sigma$  = lemma- $\equiv$  (adequacy n  $\models \sigma$ )  
⋮
```

Prueba para Sistema I

Se agrega el caso en adequacy:

```
⋮  
adequacy ([ iso ] ≡ n) ⊢ σ = lemma-≡ (adequacy n ⊢ σ)  
⋮
```

El caso de la congruencia es simple:

```
lemma-≡ : ∀ {t : Γ ⊢ A} → SN* [-] t → SN* {A = B} [-] ([ iso ] ≡ t)  
lemma-≡ SN*t = sn* tt (aux SN*t)  
where aux : ∀ {Γ A iso t'} → {t : Γ ⊢ A} →  
      SN* [-] t → ([ iso ] ≡ t) ~> t' → SN* [-] t'  
aux (sn* _ f) (inj1 (ξ-≡ step)) = lemma-≡ (f (inj1 step))  
aux (sn* _ f) (inj2 (ξ-≡ step)) = lemma-≡ (f (inj2 step))  
⋮
```

Para las equivalencias la idea es:

- “Desarmar” las hipótesis del lado izquierdo.
- Reconstruir SN^* del lado derecho usando los lemas anteriores.

```
⋮  
aux (sn* ( SNr , SNs ) _) (inj2 comm)      = lemma-⟨,⟩ SNs SNr  
aux (sn* ( SNr , SNs ) _) (inj2 sym-comm) = lemma-⟨,⟩ SNs SNr  
⋮
```

$$\lambda x^A. \lambda y^B. t \iff \lambda z^{A \times B}. t[\pi_A(z)/x, \pi_B(z)/y]$$

$$\lambda x^A.\lambda y^B.t \iff \lambda z^{A \times B}.t[\pi_A(z)/x, \pi_B(z)/y]$$

Por hipótesis sabemos:

$$\forall u_1, u_2 \in SN^* \implies t[u_1/x, u_2/y] \in SN^*$$

$$\lambda x^A. \lambda y^B. t \Leftrightarrow \lambda z^{A \times B}. t[\pi_A(z)/x, \pi_B(z)/y]$$

Por hipótesis sabemos:

$$\forall u_1, u_2 \in SN^* \implies t[u_1/x, u_2/y] \in SN^*$$

Debemos probar:

$$\lambda z. t[\pi_A(z)/x, \pi_B(z)/y] \in SN^*$$

$$\lambda x^A. \lambda y^B. t \iff \lambda z^{A \times B}. t[\pi_A(z)/x, \pi_B(z)/y]$$

Por hipótesis sabemos:

$$\forall u_1, u_2 \in SN^* \implies t[u_1/x, u_2/y] \in SN^*$$

Debemos probar:

$$\lambda z. t[\pi_A(z)/x, \pi_B(z)/y] \in SN^*$$

$$\forall u \in SN^* \implies (t[\pi_A(z)/x, \pi_B(z)/y])[u/z]$$

$$\lambda x^A.\lambda y^B.t \iff \lambda z^{A \times B}.t[\pi_A(z)/x, \pi_B(z)/y]$$

Por hipótesis sabemos:

$$\forall u_1, u_2 \in SN^* \implies t[u_1/x, u_2/y] \in SN^*$$

Debemos probar:

$$\lambda z.t[\pi_A(z)/x, \pi_B(z)/y] \in SN^*$$

$$\forall u \in SN^* \implies (t[\pi_A(z)/x, \pi_B(z)/y])[u/z] = t[\pi_A(u)/x, \pi_B(u)/y] \in SN^*$$

$$\lambda x^A.\lambda y^B.t \Leftrightarrow \lambda z^{A \times B}.t[\pi_A(z)/x, \pi_B(z)/y]$$

Por hipótesis sabemos:

$$\forall u_1, u_2 \in SN* \implies t[u_1/x, u_2/y] \in SN*$$

Debemos probar:

$$\lambda z.t[\pi_A(z)/x, \pi_B(z)/y] \in SN*$$

$$\forall u \in SN* \implies (t[\pi_A(z)/x, \pi_B(z)/y])[u/z] = t[\pi_A(u)/x, \pi_B(u)/y] \in SN*$$

Basta con instanciar la hipótesis usando:

$$u_1 = \pi_A(u) \quad u_2 = \pi_B(u)$$

$$\lambda x^{A \times B}.t \rightleftharpoons \lambda y^A.\lambda z^B.t[\langle y, z \rangle / x]$$

$$\lambda x^{A \times B}.t \rightleftharpoons \lambda y^A.\lambda z^B.t[\langle y, z \rangle/x]$$

Por hipótesis sabemos:

$$\forall u \in SN^* \implies t[u/x] \in SN^*$$

$$\lambda x^{A \times B}.t \iff \lambda y^A.\lambda z^B.t[\langle y, z \rangle/x]$$

Por hipótesis sabemos:

$$\forall u \in SN* \implies t[u/x] \in SN*$$

Debemos probar:

$$\lambda y.\lambda z.t[\langle y, z \rangle/x] \in SN*$$

$$\lambda x^{A \times B}.t \rightleftharpoons \lambda y^A.\lambda z^B.t[\langle y, z \rangle / x]$$

Por hipótesis sabemos:

$$\forall u \in SN* \implies t[u/x] \in SN*$$

Debemos probar:

$$\lambda y.\lambda z.t[\langle y, z \rangle / x] \in SN*$$

$$\forall u_1, u_2 \in SN* \implies ((t[\langle y, z \rangle / x])[u_1/y])[u_2/z]$$

$$\lambda x^{A \times B}.t \rightleftharpoons \lambda y^A.\lambda z^B.t[\langle y, z \rangle/x]$$

Por hipótesis sabemos:

$$\forall u \in SN^* \implies t[u/x] \in SN^*$$

Debemos probar:

$$\lambda y.\lambda z.t[\langle y, z \rangle/x] \in SN^*$$

$$\forall u_1, u_2 \in SN^* \implies ((t[\langle y, z \rangle/x])[u_1/y])[u_2/z] = t[\langle u_1, u_2 \rangle/x] \in SN^*$$

$$\lambda x^{A \times B}.t \rightleftharpoons \lambda y^A.\lambda z^B.t[\langle y, z \rangle/x]$$

Por hipótesis sabemos:

$$\forall u \in SN^* \implies t[u/x] \in SN^*$$

Debemos probar:

$$\lambda y.\lambda z.t[\langle y, z \rangle/x] \in SN^*$$

$$\forall u_1, u_2 \in SN^* \implies ((t[\langle y, z \rangle/x])[u_1/y])[u_2/z] = t[\langle u_1, u_2 \rangle/x] \in SN^*$$

Basta con instanciar la hipótesis usando:

$$u = \langle u_1, u_2 \rangle$$

- 1 Marco teórico
 - Introducción a LCST
 - Introducción a Sistema I
- 2 Aportes
 - Formalización
 - Normalización fuerte
- 3 Conclusiones

- Introducción teórica.

- Introducción teórica.
- Formalización de Sistema I en Agda.

- Introducción teórica.
- Formalización de Sistema I en Agda.
- Prueba de normalización fuerte.

- Inferencia de tipos.

- Inferencia de tipos.
- Formalizar Sistema I Polimórfico.

- Inferencia de tipos.
- Formalizar Sistema I Polimórfico.
- Formalización con tipos extrínsecos.