CAS 760
Simple Type Theory
Winter 2026

4 Alonzo: Syntax and Semantics

William M. Farmer

Department of Computing and Software
McMaster University

December 29, 2025

Outline

1. Syntax.
2. Semantics.
3. Additional notation.
4. Beta-reduction and substitution.

1. Syntax

Notation

- There are two kinds of notation for Alonzo:
  1. Formal.
     - Machine-oriented "internal" notation.
  2. Compact.
     - Human-oriented "external" notation.
     - Introduced by notational definitions and conventions.
- A notational definition has the form

  A stands for B.

  It defines A to be an alternate notation for B.

Symbols

- $\mathcal{S}_{\mathsf{bt}}$ is a set of base type symbols that contains the symbols $A, B, C \ldots, X, Y, Z$, etc.
- $\mathcal{S}_{\mathsf{var}}$ is a set of variable symbols that contains the symbols $a, b, c \ldots, x, y, z$, etc.
- $\mathcal{S}_{\mathsf{con}}$ is a set of constant symbols that contains the symbols $A, B, C \ldots, X, Y, Z$, etc., numeric symbols, nonalphanumeric symbols, and words in lowercase sans sarif font.
- $\mathcal{S}_{\mathsf{bt}}$, $\mathcal{S}_{\mathsf{var}}$, and usually $\mathcal{S}_{\mathsf{con}}$ are countably infinite.

Metavariables

- $\mathbf{a}, \mathbf{b}$, etc. range over $\mathcal{S}_{\mathsf{bt}}$.
- $\mathbf{f}, \mathbf{g}, \mathbf{h}, \mathbf{i}, \mathbf{j}, \mathbf{k}, \mathbf{m}, \mathbf{n}, \mathbf{u}, \mathbf{v}, \mathbf{w}, \mathbf{x}, \mathbf{y}, \mathbf{z}$, etc. range over $\mathcal{S}_{\mathsf{var}}$.
- $\mathbf{c}, \mathbf{d}$, etc. range over $\mathcal{S}_{\mathsf{con}}$.
- $\alpha, \beta, \gamma, \delta$, etc. range over types.
- $\mathbf{A}_\alpha, \mathbf{B}_\alpha, \mathbf{C}_\alpha, \ldots, \mathbf{X}_\alpha, \mathbf{Y}_\alpha, \mathbf{Z}_\alpha$, etc. range over expressions of type α.

Types [1/2]

- A type of Alonzo is a string of symbols defined inductively by the following formation rules:
  - T1. Type of truth values: BoolTy is a type.
  - T2. Base type: $\mathsf{BaseTy}(\mathbf{a})$ is a type.
  - T3. Function type: $\mathsf{FunTy}(\alpha, \beta)$ is a type.
  - T4. Product type: $\mathsf{ProdTy}(\alpha, \beta)$ is a type.
- Compact notation for types:

o	stands for	BoolTy.
$\mathbf{a}$	stands for	$\mathsf{BaseTy}(\mathbf{a})$.
$(\alpha \rightarrow \beta)$	stands for	$\mathsf{FunTy}(\alpha, \beta)$.
$(\alpha \times \beta)$	stands for	$\mathsf{ProdTy}(\alpha, \beta)$.

- Matching pairs of parentheses may be dropped (NC 1).
- Function type formation associates to the right (NC 2).

Types [2/2]

- A type α denotes a nonempty set D_α of values.
  - o denotes the set $D_o = \mathbb{B}$ of the boolean (truth) values F and T.
  - $(\alpha \rightarrow o)$ denotes a set $D_{\alpha \rightarrow o}$ of some total functions from D_α to D_o where $\beta \neq o$.
  - $(\alpha \rightarrow \beta)$ denotes a set $D_{\alpha \rightarrow \beta}$ of some (partial and total) functions from D_α to D_β.
  - $(\alpha \times \beta)$ denotes the Cartesian product $D_{\alpha \times \beta} = D_\alpha \times D_\beta$.
- We will use base types to denote the base domains of structures.

Expressions [1/3]

- An expression of type α of Alonzo is a string of symbols defined inductively by the following formation rules:
  - E1. Variable: $\mathsf{Var}(\mathbf{x}, \alpha)$ is an expression of type α.
  - E2. Constant: $\mathsf{Con}(\mathbf{c}, \alpha)$ is an expression of type α.
  - E3. Equality: $\mathsf{Eq}(\mathbf{A}_\alpha, \mathbf{B}_\alpha)$ is an expression of type BoolTy.
  - E4. Function application: $\mathsf{FunApp}(\mathbf{F}_{\mathsf{FunTy}(\alpha,\beta)}, \mathbf{A}_\alpha)$ is an expression of type β.
  - E5. Function abstraction: $\mathsf{FunAbs}(\mathsf{Var}(\mathbf{x}, \alpha), \mathbf{B}_\beta)$ is an expression of type $\mathsf{FunTy}(\alpha, \beta)$.
  - E6. Definite description: $\mathsf{DefDes}(\mathsf{Var}(\mathbf{x}, \alpha), \mathbf{A}_{\mathsf{BoolTy}})$ is an expression of type α where $\alpha \neq \mathsf{BoolTy}$.
  - E7. Ordered pair: $\mathsf{OrdPair}(\mathbf{A}_\alpha, \mathbf{B}_\beta)$ is an expression of type $\mathsf{ProdTy}(\alpha, \beta)$.
- A formula is an expression of type BoolTy.

Expressions [2/3]

- Compact notation for expressions:

$(\mathbf{x} : \alpha)$	stands for	$\mathsf{Var}(\mathbf{x}, \alpha)$.
$\mathbf{c}_\alpha$	stands for	$\mathsf{Con}(\mathbf{c}, \alpha)$.
$(\mathbf{A}_\alpha = \mathbf{B}_\alpha)$	stands for	$\mathsf{Eq}(\mathbf{A}_\alpha, \mathbf{B}_\alpha)$.
$(\mathbf{F}_{\alpha \rightarrow \beta}\, \mathbf{A}_\alpha)$	stands for	$\mathsf{FunApp}(\mathbf{F}_{\alpha \rightarrow \beta}, \mathbf{A}_\alpha)$.
$(\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{B}_\beta)$	stands for	$\mathsf{FunAbs}(\mathsf{Var}(\mathbf{x}, \alpha), \mathbf{B}_\beta)$.
$(\mathrm{I}\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o)$	stands for	$\mathsf{DefDes}(\mathsf{Var}(\mathbf{x}, \alpha), \mathbf{A}_o)$.
$(\mathbf{A}_\alpha, \mathbf{B}_\beta)$	stands for	$\mathsf{OrdPair}(\mathbf{A}_\alpha, \mathbf{B}_\beta)$.

- Matching pairs of parentheses may be dropped (NC 3).
- Function application formation associates to the left (NC 4).
- The type of a constant may be dropped if it is known from the context (NC 5).

Expressions [3/3]

- The type of a bound variable in the body of a binder may be dropped (NC 6 and NC 7).
- The type of a variable may be dropped if it is known from the context (NC 8).
- An expression of type α is always defined if $\alpha = o$ and may be either defined or undefined if $\alpha \neq o$.
  - If defined, it denotes a value in D_α.
  - If undefined, it denotes nothing at all.
- We will use constants to denote the distinguished values of structures.

Bound and Free Variables and Substitution

- An occurrence of a variable $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$ is bound [free] if it is [not] within a subexpression of $\mathbf{B}_\beta$ of either the form $\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{C}_\gamma$ or the form $\mathrm{I}\, \mathbf{x} : \alpha \,.\, \mathbf{C}_o$.
- A variable $(\mathbf{x} : \alpha)$ is bound [free] in $\mathbf{B}_\beta$ if there is a bound [free] occurrence of $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$.
- An expression is closed if it contains no free variables.
- A sentence is a closed formula.
- $\mathbf{A}_\alpha$ is free for $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$ if no free occurrence of $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$ is within a subexpression of $\mathbf{B}_\beta$ of either the form $\lambda\, \mathbf{y} : \gamma \,.\, \mathbf{C}_\delta$ or the form $\mathrm{I}\, \mathbf{y} : \gamma \,.\, \mathbf{C}_o$ where $(\mathbf{y} : \gamma)$ is free in $\mathbf{A}_\alpha$.
- The substitution of $\mathbf{A}_\alpha$ for $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$, written $\mathbf{B}_\beta[(\mathbf{x} : \alpha) \mapsto \mathbf{A}_\alpha]$, is the result of replacing each free occurrence of $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$ with $\mathbf{A}_\alpha$.

Languages [1/2]

- A language (or signature) is a pair $L = (\mathcal{B}, \mathcal{C})$ where $\mathcal{B}$ is a finite set of base types and $\mathcal{C}$ is a set of constants $\mathbf{c}_\alpha$ where each base type occurring in α is a member of $\mathcal{B}$.
- A type α is a type of L if all the base types occurring in α are members of $\mathcal{B}$.
- An expression $\mathbf{A}_\alpha$ is an expression of L if all the base types occurring in $\mathbf{A}_\alpha$ are members of $\mathcal{B}$ and all the constants occurring in $\mathbf{A}_\alpha$ are members of $\mathcal{C}$.
- When the set of constants is finite, we may write

  $$L = (\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}, \{\mathbf{c}^1_{\beta_1}, \ldots, \mathbf{c}^n_{\beta_n}\})$$

  as a tuple

  $$(\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{c}^1_{\beta_1}, \ldots, \mathbf{c}^n_{\beta_n})$$

  in the same way a structure can be written as a tuple.

Languages [2/2]

- Let $L_i = (\mathcal{B}_i, \mathcal{C}_i)$ be a language for $i \in \{1, 2\}$.
- L_2 is an extension of L_1 (or L_1 is a sublanguage of L_2), written $L_1 \leq L_2$, if $\mathcal{B}_1 \subseteq \mathcal{B}_2$ and $\mathcal{C}_1 \subseteq \mathcal{C}_2$.
- The minimum language is the language $L_{\mathrm{min}} = (\emptyset, \emptyset)$.
- $L_{\mathrm{min}} \leq L$ for every language L.
- The power of a language $L = (\mathcal{B}, \mathcal{C})$, written $\|L\|$, is $|\mathcal{E}(L)|$.
  - In the usual case, when $\mathcal{C}$ is countable, $\|L\| = \omega$.
  - When $\mathcal{C}$ is uncountable, $\|L\| = |\mathcal{C}|$.

2. Semantics

Frames

- Let $L = (\mathcal{B}, \mathcal{C})$ be a language of Alonzo.
- A frame for L is a collection $\mathcal{D} = \{D_\alpha \mid \alpha \in \mathcal{T}(L)\}$ of nonempty domains (sets) of values such that:
  - F1. Domain of truth values: $D_o = \mathbb{B} = \{\mathrm{F}, \mathrm{T}\}$.
  - F2. Predicate domain: $D_{\alpha \rightarrow o}$ is a set of some total functions from D_α to D_o for $\alpha \in \mathcal{T}(L)$.
  - F3. Function domain: $D_{\alpha \rightarrow \beta}$ is a set of some partial and total functions from D_α to D_β for $\alpha, \beta \in \mathcal{T}(L)$, $\beta \neq o$.
  - F4. Product domain: $D_{\alpha \times \beta} = D_\alpha \times D_\beta$ for $\alpha, \beta \in \mathcal{T}(L)$.
- A predicate domain $D_{\alpha \rightarrow o}$ is full if it is the set of all total functions from D_α to D_o
- A function domain $D_{\alpha \rightarrow \beta}$ with $\beta \neq o$ is full if it is the set of all partial and total functions from D_α to D_β.
- The frame is full if $D_{\alpha \rightarrow \beta}$ is full for all $\alpha, \beta \in \mathcal{T}(L)$.

Interpretations and Assignments

- An interpretation of L is a pair $M = (\mathcal{D}, I)$ where $\mathcal{D} = \{D_\alpha \mid \alpha \in \mathcal{T}(L)\}$ is a frame for L and I is an interpretation function that maps each constant in $\mathcal{C}$ of type α to an element of D_α.
- An assignment into $\mathcal{D}$ is a function φ whose domain is the set of variables of L such that $\varphi((\mathbf{x} : \alpha)) \in D_\alpha$ for each variable $(\mathbf{x} : \alpha)$ of L.
- Given an assignment φ, a variable $(\mathbf{x} : \alpha)$ of L, and $d \in D_\alpha$, let $\varphi[(\mathbf{x} : \alpha) \mapsto d]$ be the assignment ψ in $\mathcal{D}$ such that $\psi((\mathbf{x} : \alpha)) = d$ and $\psi((\mathbf{y} : \beta)) = \varphi((\mathbf{y} : \beta))$ for all variables $(\mathbf{y} : \beta)$ of L distinct from $(\mathbf{x} : \alpha)$.
- Given an interpretation M of L, let $\mathrm{assign}(M)$ be the set of assignments into the frame of M.

General Models [1/3]

- Let $\mathcal{D} = \{D_\alpha \mid \alpha \in \mathcal{T}(L)\}$ be a frame for L and $M = (\mathcal{D}, I)$ be an interpretation of L.
- M is a general model of L if there is a partial binary valuation function V^M such that, for all assignments $\varphi \in \text{assign}(M)$ and expressions $\mathbf{C}_\gamma$ of L, (1) either $V^M_\varphi(\mathbf{C}_\gamma) \in D_\gamma$ or $V^M_\varphi(\mathbf{C}_\gamma)$ is undefined and (2) each of the following conditions is satisfied:

V1. $V^M_\varphi((\mathbf{x} : \alpha)) = \varphi((\mathbf{x} : \alpha))$.

V2. $V^M_\varphi(\mathbf{c}_\alpha) = I(\mathbf{c}_\alpha)$.

V3. $V^M_\varphi(\mathbf{A}_\alpha = \mathbf{B}_\alpha) = \mathrm{T}$ if $V^M_\varphi(\mathbf{A}_\alpha)$ is defined, $V^M_\varphi(\mathbf{B}_\alpha)$ is defined, and $V^M_\varphi(\mathbf{A}_\alpha) = V^M_\varphi(\mathbf{B}_\alpha)$. Otherwise, $V^M_\varphi(\mathbf{A}_\alpha = \mathbf{B}_\alpha) = \mathrm{F}$.

General Models [2/3]

V4. $V_\varphi^M(\mathbf{F}_{\alpha\rightarrow\beta}\,\mathbf{A}_\alpha) = V_\varphi^M(\mathbf{F}_{\alpha\rightarrow\beta})(V_\varphi^M(\mathbf{A}_\alpha))$ if $V_\varphi^M(\mathbf{F}_{\alpha\rightarrow\beta})$ is defined, $V_\varphi^M(\mathbf{A}_\alpha)$ is defined, and $V_\varphi^M(\mathbf{F}_{\alpha\rightarrow\beta})$ is defined at $V_\varphi^M(\mathbf{A}_\alpha)$. Otherwise, $V_\varphi^M(\mathbf{F}_{\alpha\rightarrow\beta}\,\mathbf{A}_\alpha) = \mathrm{F}$ if $\beta = o$ and $V_\varphi^M(\mathbf{F}_{\alpha\rightarrow\beta}\,\mathbf{A}_\alpha)$ is undefined if $\beta \neq o$.

V5. $V_\varphi^M(\lambda\,\mathbf{x} : \alpha\;.\;\mathbf{B}_\beta)$ is the (partial or total) function $f \in D_{\alpha\rightarrow\beta}$ such that, for each $d \in D_\alpha$, $f(d) = V_{\varphi[(\mathbf{x}:\alpha)\mapsto d]}^M(\mathbf{B}_\beta)$ if $V_{\varphi[(\mathbf{x}:\alpha)\mapsto d]}^M(\mathbf{B}_\beta)$ is defined and $f(d)$ is undefined if $V_{\varphi[(\mathbf{x}:\alpha)\mapsto d]}^M(\mathbf{B}_\beta)$ is undefined.

V6. $V_\varphi^M(\mathrm{I}\,\mathbf{x} : \alpha\;.\;\mathbf{A}_o)$ is the $d \in D_\alpha$ such that $V_{\varphi[(\mathbf{x}:\alpha)\mapsto d]}^M(\mathbf{A}_o) = \mathrm{T}$ if there is exactly one such d. Otherwise, $V_\varphi^M(\mathrm{I}\,\mathbf{x} : \alpha\;.\;\mathbf{A}_o)$ is undefined.

V7. $V_\varphi^M((\mathbf{A}_\alpha, \mathbf{B}_\beta)) = (V_\varphi^M(\mathbf{A}_\alpha), V_\varphi^M(\mathbf{B}_\beta))$ if $V_\varphi^M(\mathbf{A}_\alpha)$ and $V_\varphi^M(\mathbf{B}_\beta)$ are defined. Otherwise, $V_\varphi^M((\mathbf{A}_\alpha, \mathbf{B}_\beta))$ is undefined.

General Models [3/3]

- Proposition. General models for L exist.
- Theorem. Alonzo satisfies the three principles of TATU.
- The size of M, written $|M|$, is the cardinality of

  $$\bigcup_{\mathbf{a} \in \mathcal{B}} D^{M}_{\mathbf{a}}.$$

- M is finite if its size is finite and is infinite otherwise.
- The power of M, written $\|M\|$, is the least cardinal κ such that $|D^{M}_{\alpha}| \leq \kappa$ for all $\alpha \in \mathcal{T}(L)$.

Finite and Standard Models

- Proposition. Let M be a general model of L.
  1. M is finite iff D_α^M is finite for all $\alpha \in \mathcal{T}(L)$.
  2. If M is finite, then $\|M\| = \omega$.
- An interpretation $M = (\mathcal{D}, I)$ of L is a standard model of L if $\mathcal{D}$ is full.
- Proposition. A standard model of L is a general model of L.
- A nonstandard model of L is a general model of L that is not a standard model.
- Theorem. Every finite general model is a standard model.

Satisfiability, Validity, Semantic Consequence [1/2]

- φ satisfies $\mathbf{A}_o$ in M, written $M \vDash_\varphi \mathbf{A}_o$, if $V_\varphi^M(\mathbf{A}_o) = \mathrm{T}$.
- $\mathbf{A}_o$ is satisfiable in M if $M \vDash_\varphi \mathbf{A}_o$ for some $\varphi \in \mathsf{assign}(M)$.
- $\mathbf{A}_o$ is satisfiable if $M \vDash_\varphi \mathbf{A}_o$ for some general model M and some $\varphi \in \mathsf{assign}(M)$.
- $\mathbf{A}_o$ is valid in M (or M is a model of $\mathbf{A}_o$), written $M \vDash \mathbf{A}_o$, if $M \vDash_\varphi \mathbf{A}_o$ for all $\varphi \in \mathsf{assign}(M)$.
- If $\mathbf{A}_o$ is a sentence, then $\mathbf{A}_o$ is true [false] in M if $V_\varphi^M(\mathbf{A}_o) = \mathrm{T}$ $[\mathrm{F}]$ (for all $\varphi \in \mathsf{assign}(M)$).
- $\mathbf{A}_o$ is valid (in the general sense), written $\vDash \mathbf{A}_o$, if $M \vDash \mathbf{A}_o$ for all general models M that interpret $\mathbf{A}_o$.
- $\mathbf{A}_o$ is valid in the standard sense, written $\vDash^{\mathsf{s}} \mathbf{A}_o$, if $M \vDash \mathbf{A}_o$ for all standard models M that interpret $\mathbf{A}_o$.
- $\mathbf{A}_o$ is logically equivalent to $\mathbf{B}_o$ if $V_\varphi^M(\mathbf{A}_o) = V_\varphi^M(\mathbf{B}_o)$ for all general models M that interpret $\mathbf{A}_o$ and $\mathbf{B}_o$ and all $\varphi \in \mathsf{assign}(M)$.

Satisfiability, Validity, Semantic Consequence [2/2]

- φ satisfies Γ in M, written $M \vDash_\varphi \Gamma$, if $M \vDash_\varphi \mathbf{A}_o$ for all $\mathbf{A}_o \in \Gamma$.
- Γ is satisfiable in M if $M \vDash_\varphi \Gamma$ for some $\varphi \in \mathrm{assign}(M)$.
- Γ is satisfiable if $M \vDash_\varphi \Gamma$ for some general model M for L and some $\varphi \in \mathrm{assign}(M)$.
- M is a model of Γ, written $M \vDash \Gamma$, if $M \vDash \mathbf{A}_o$ for all $\mathbf{A}_o \in \Gamma$.
- $\mathbf{A}_o$ is a semantic consequence of Γ (in the general sense), written $\Gamma \vDash \mathbf{A}_o$, if $M \vDash_\varphi \Gamma$ implies $M \vDash_\varphi \mathbf{A}_o$ for all general models M that interpret $\mathbf{A}_o$ and Γ and all $\varphi \in \mathrm{assign}(M)$.
- $\mathbf{A}_o$ is a semantic consequence of Γ in the standard sense, written $\Gamma \vDash^{\mathrm{s}} \mathbf{A}_o$, if $M \vDash_\varphi \Gamma$ implies $M \vDash_\varphi \mathbf{A}_o$ for all standard models M that interpret $\mathbf{A}_o$ and Γ and all $\varphi \in \mathrm{assign}(M)$.

Expansion of a General Model

- Let M_i be a general model of L_i for $i \in \{1, 2\}$. Assume $L_1 \leq L_2$.
- M_2 is an expansion of M_1 to L_2 (or M_1 is a reduct of M_2 to L_1), written $M_1 \leq M_2$, if $\mathcal{D}_1 \subseteq \mathcal{D}_2$ and $I_1 \sqsubseteq I_2$.
- The minimum model is the unique standard model $M_{\rm min} = (\mathcal{D}, I)$ of $L_{\rm min}$, where $\mathcal{D}$ is the unique full frame for $L_{\rm min}$ and I is the empty function.
- $M_{\rm min} \leq M$ for every general model M.

Standard vs. General Semantics

- Alonzo has two semantics:
  1. A logic-oriented general semantics based on general models.
  2. A mathematics-oriented standard semantics based on standard models.
- The distinction between standard and nonstandard models is perfectly clear in Alonzo, but not so in first-order logic.

3. Additional Notation

Notational Definitions for Boolean Operators

T_o	stands for	$(\lambda\, x : o \,.\, x) = (\lambda\, x : o \,.\, x)$.
F_o	stands for	$(\lambda\, x : o \,.\, T_o) = (\lambda\, x : o \,.\, x)$.
$\wedge_{o \to o \to o}$	stands for	$\lambda\, x : o \,.\, \lambda\, y : o \,.$ $(\lambda\, g : o \to o \to o \,.\, g\, T_o\, T_o) =$ $(\lambda\, g : o \to o \to o \,.\, g\, x\, y)$.
$(\mathbf{A}_o \wedge \mathbf{B}_o)$	stands for	$\wedge_{o \to o \to o}\, \mathbf{A}_o\, \mathbf{B}_o$.
$\Rightarrow_{o \to o \to o}$	stands for	$\lambda\, x : o \,.\, \lambda\, y : o \,.\, x = (x \wedge y)$.
$(\mathbf{A}_o \Rightarrow \mathbf{B}_o)$	stands for	$\Rightarrow_{o \to o \to o}\, \mathbf{A}_o\, \mathbf{B}_o$.
$\neg_{o \to o}$	stands for	$\lambda\, x : o \,.\, x = F_o$.
$(\neg \mathbf{A}_o)$	stands for	$\neg_{o \to o}\, \mathbf{A}_o$.
$\vee_{o \to o \to o}$	stands for	$\lambda\, x : o \,.\, \lambda\, y : o \,.\, \neg(\neg x \wedge \neg y)$.
$(\mathbf{A}_o \vee \mathbf{B}_o)$	stands for	$\vee_{o \to o \to o}\, \mathbf{A}_o\, \mathbf{B}_o$.

Notational Definitions for Binary Operators

$(\mathbf{A}_\alpha \ \mathbf{c} \ \mathbf{B}_\alpha)$	stands for	$\mathbf{c}_{\alpha \rightarrow \alpha \rightarrow \beta} \, \mathbf{A}_\alpha \, \mathbf{B}_\alpha$ or $\mathbf{c}_{(\alpha \times \alpha) \rightarrow \beta} \, (\mathbf{A}_\alpha, \mathbf{B}_\alpha)$.
$(\mathbf{A}_o \Leftrightarrow \mathbf{B}_o)$	stands for	$\mathbf{A}_o = \mathbf{B}_o$.
$(\mathbf{A}_\alpha \neq \mathbf{B}_\alpha)$	stands for	$\neg(\mathbf{A}_\alpha = \mathbf{B}_\alpha)$.
$(\mathbf{A}_\alpha < \mathbf{B}_\alpha)$	stands for	$(\leq_{\alpha \rightarrow \alpha \rightarrow o} \mathbf{A}_\alpha \, \mathbf{B}_\alpha) \wedge (\mathbf{A}_\alpha \neq \mathbf{B}_\alpha)$.
$(\mathbf{A}_\alpha > \mathbf{B}_\alpha)$	stands for	$\mathbf{B}_\alpha < \mathbf{A}_\alpha$.
$(\mathbf{A}_\alpha \geq \mathbf{B}_\alpha)$	stands for	$\mathbf{B}_\alpha \leq \mathbf{A}_\alpha$.
$(\mathbf{A}_\alpha = \mathbf{B}_\alpha = \mathbf{C}_\alpha)$	stands for	$(\mathbf{A}_\alpha = \mathbf{B}_\alpha) \wedge (\mathbf{B}_\alpha = \mathbf{C}_\alpha)$.
$(\mathbf{A}_\alpha \ \mathbf{c} \ \mathbf{B}_\alpha \ \mathbf{d} \ \mathbf{C}_\alpha)$	stands for	$(\mathbf{A}_\alpha \ \mathbf{c} \ \mathbf{B}_\alpha) \wedge (\mathbf{B}_\alpha \ \mathbf{d} \ \mathbf{C}_\alpha)$.

Notational Definitions for Quantifiers

$(\forall\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o)$	stands for	$(\lambda\, x : \alpha \,.\, T_o) = (\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o)$.
$(\exists\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o)$	stands for	$\neg(\forall\, \mathbf{x} : \alpha \,.\, \neg\mathbf{A}_o)$.
$(\exists!\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o)$	stands for	$\exists\, y : \alpha \,.\, (\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o) = (\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{x} = y)$ where y does not occur in $(\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{A}_o)$.

Notational Definitions for Definedness

$\bot_o$	stands for	F_o.
$\bot_\alpha$	stands for	$\mathrm{I}\, x : \alpha \,.\, x \neq x$ where $\alpha \neq o$.
$\Delta_{\alpha \rightarrow \beta}$	stands for	$\lambda\, x : \alpha \,.\, \bot_\beta$ where $\beta \neq o$.
$(\mathbf{A}_\alpha\downarrow)$	stands for	$\mathbf{A}_\alpha = \mathbf{A}_\alpha$.
$(\mathbf{A}_\alpha\uparrow)$	stands for	$\neg(\mathbf{A}_\alpha\downarrow)$.
$(\mathbf{A}_\alpha \simeq \mathbf{B}_\alpha)$	stands for	$(\mathbf{A}_\alpha\downarrow \vee \mathbf{B}_\alpha\downarrow) \Rightarrow \mathbf{A}_\alpha = \mathbf{B}_\alpha$.
$(\mathbf{A}_\alpha \not\simeq \mathbf{B}_\alpha)$	stands for	$\neg(\mathbf{A}_\alpha \simeq \mathbf{B}_\alpha)$.
$\mathrm{IF}(\mathbf{A}_o, \mathbf{B}_o, \mathbf{C}_o)$	stands for	$(\mathbf{A}_o \Rightarrow \mathbf{B}_o) \wedge (\neg \mathbf{A}_o \Rightarrow \mathbf{C}_o)$.
$\mathrm{IF}(\mathbf{A}_o, \mathbf{B}_\alpha, \mathbf{C}_\alpha)$	stands for	$\mathrm{I}\, x : \alpha \,.$ $(\mathbf{A}_o \Rightarrow x = \mathbf{B}_\alpha) \wedge (\neg \mathbf{A}_o \Rightarrow x = \mathbf{C}_\alpha)$ where $\alpha \neq o$.
$(\mathbf{A}_o \mapsto \mathbf{B}_\alpha \mid \mathbf{C}_\alpha)$	stands for	$\mathrm{IF}(\mathbf{A}_o, \mathbf{B}_\alpha, \mathbf{C}_\alpha)$.

Notational Definitions for Sets

$\{\alpha\}$	stands for	$\alpha \rightarrow o$.
$(\mathbf{A}_\alpha \in \mathbf{B}_{\{\alpha\}})$	stands for	$\mathbf{B}_{\{\alpha\}}\ \mathbf{A}_\alpha$.
$(\mathbf{A}_\alpha \not\in \mathbf{B}_{\{\alpha\}})$	stands for	$\neg(\mathbf{A}_\alpha \in \mathbf{B}_{\{\alpha\}})$.
$\{\mathbf{x} : \alpha \mid \mathbf{A}_o\}$	stands for	$\lambda\, \mathbf{x} : \alpha\ .\ \mathbf{A}_o$.
$\emptyset_{\{\alpha\}}$	stands for	$\lambda\, x : \alpha\ .\ F_o$.
$\{\,\}_{\{\alpha\}}$	stands for	$\emptyset_{\{\alpha\}}$.
$U_{\{\alpha\}}$	stands for	$\lambda\, x : \alpha\ .\ T_o$.
n-α-SET	stands for	$\lambda\, x_1 : \alpha\ . \cdots .\ \lambda\, x_n : \alpha\ .\ \lambda\, x : \alpha\ .$ $x = x_1 \vee \cdots \vee x = x_n$ where $n \geq 1$.
$\{\mathbf{A}^1_\alpha, \ldots, \mathbf{A}^n_\alpha\}$	stands for	n-α-SET $\mathbf{A}^1_\alpha \cdots \mathbf{A}^n_\alpha$ where $n \geq 1$.
$\subseteq_{\{\alpha\} \rightarrow \{\alpha\} \rightarrow o}$	stands for	$\lambda\, a : \{\alpha\}\ .\ \lambda\, b : \{\alpha\}\ .$ $\forall\, x : \alpha\ .\ x \in a \Rightarrow x \in b$.
$\cup_{\{\alpha\} \rightarrow \{\alpha\} \rightarrow \{\alpha\}}$	stands for	$\lambda\, a : \{\alpha\}\ .\ \lambda\, b : \{\alpha\}\ .$ $\{x : \alpha \mid x \in a \vee x \in b\}$.
$\cap_{\{\alpha\} \rightarrow \{\alpha\} \rightarrow \{\alpha\}}$	stands for	$\lambda\, a : \{\alpha\}\ .\ \lambda\, b : \{\alpha\}\ .$ $\{x : \alpha \mid x \in a \wedge x \in b\}$.
$\overline{\cdot}_{\{\alpha\} \rightarrow \{\alpha\}}$	stands for	$\lambda\, a : \{\alpha\}\ .\ \{x : \alpha \mid x \not\in a\}$.
$\overline{\mathbf{A}_{\{\alpha\}}}$	stands for	$\overline{\cdot}_{\{\alpha\} \rightarrow \{\alpha\}}\ \mathbf{A}_{\{\alpha\}}$.
$\setminus_{\{\alpha\} \rightarrow \{\alpha\} \rightarrow \{\alpha\}}$	stands for	$\lambda\, a : \{\alpha\}\ .\ \lambda\, b : \{\alpha\}\ .\ a \cap \overline{b}$.

Notational Definitions for Tuples

(α)	stands for	α.
$(\alpha_1 \times \cdots \times \alpha_n)$	stands for	$(\alpha_1 \times (\alpha_2 \times \cdots \times \alpha_n))$ where $n \geq 2$.
$(\mathbf{A}_\alpha)$	stands for	$\mathbf{A}_\alpha$.
$(\mathbf{A}^1_{\alpha_1}, \ldots, \mathbf{A}^n_{\alpha_n})$	stands for	$(\mathbf{A}^1_{\alpha_1}, (\mathbf{A}^2_{\alpha_1}, \ldots, \mathbf{A}^n_{\alpha_n}))$ where $n \geq 2$.
$\mathsf{fst}_{(\alpha \times \beta) \rightarrow \alpha}$	stands for	$\lambda\, p : \alpha \times \beta \,.\, \mathrm{I}\, x : \alpha \,.\, \exists\, y : \beta \,.\, p = (x, y)$.
$\mathsf{snd}_{(\alpha \times \beta) \rightarrow \beta}$	stands for	$\lambda\, p : \alpha \times \beta \,.\, \mathrm{I}\, y : \beta \,.\, \exists\, x : \alpha \,.\, p = (x, y)$.

Notational Definitions for Functions

$\mathrm{id}_{\alpha\to\alpha}$	stands for	$\lambda\, x : \alpha \,.\, x$.
$\mathrm{dom}_{(\alpha\to\beta)\to\{\alpha\}}$	stands for	$\lambda\, f : \alpha \to \beta \,.\, \{x : \alpha \mid (f\,x){\downarrow}\}$.
$\mathrm{ran}_{(\alpha\to\beta)\to\{\beta\}}$	stands for	$\lambda\, f : \alpha \to \beta \,.\, \{y : \beta \mid \exists\, x : \alpha \,.\, f\,x = y\}$.
$\sqsubseteq_{(\alpha\to\beta)\to(\alpha\to\beta)\to o}$	stands for	$\lambda\, f : \alpha \to \beta \,.\, \lambda\, g : \alpha \to \beta \,.\, \forall\, x : \alpha \,.\, x \in \mathrm{dom}_{(\alpha\to\beta)\to\{\alpha\}}\, f \Rightarrow f\,x = g\,x$.
$\circ_{(\alpha\to\beta)\to(\beta\to\gamma)\to(\alpha\to\gamma)}$	stands for	$\lambda\, f : \alpha \to \beta \,.\, \lambda\, g : \beta \to \gamma \,.\, \lambda\, x : \alpha \,.\, g\,(f\,x)$.
$(\mathbf{F}_{\alpha\to\beta} \circ \mathbf{G}_{\beta\to\gamma})$	stands for	$\circ_{(\alpha\to\beta)\to(\beta\to\gamma)\to(\alpha\to\gamma)}\, \mathbf{F}_{\alpha\to\beta}\, \mathbf{G}_{\beta\to\gamma}$.
$\vert_{(\alpha\to\beta)\to\{\alpha\}\to(\alpha\to\beta)}$	stands for	$\lambda\, f : \alpha \to \beta \,.\, \lambda\, s : \{\alpha\} \,.\, \lambda\, x : \alpha \,.\, x \in s \mapsto f\,x \mid \bot_\beta$.
$(\mathbf{F}_{\alpha\to\beta}\vert_{\mathbf{A}_{\{\alpha\}}})$	stands for	$\vert_{(\alpha\to\beta)\to\{\alpha\}\to(\alpha\to\beta)}\, \mathbf{F}_{\alpha\to\beta}\, \mathbf{A}_{\{\alpha\}}$.

Miscellaneous Notational Definitions

$\mathsf{TOTAL}(\mathbf{F}_{\alpha \to \beta})$	stands for	$\forall\, x : \alpha \,.\, (\mathbf{F}_{\alpha \to \beta}\, x){\downarrow}$.
$\mathsf{TOTAL2}(\mathbf{F}_{\alpha \to \beta \to \gamma})$	stands for	$\forall\, x : \alpha,\ y : \beta \,.\, (\mathbf{F}_{\alpha \to \beta \to \gamma}\, x\, y){\downarrow}$.
$\mathsf{SURJ}(\mathbf{F}_{\alpha \to \beta})$	stands for	$\forall\, y : \beta \,.\, \exists\, x : \alpha \,.\, \mathbf{F}_{\alpha \to \beta}\, x = y$.
$\mathsf{SURJ2}(\mathbf{F}_{\alpha \to \beta \to \gamma})$	stands for	$\forall\, z : \gamma \,.\, \exists\, x : \alpha,\ y : \beta \,.\, \mathbf{F}_{\alpha \to \beta \to \gamma}\, x\, y = z$.
$\mathsf{INJ}(\mathbf{F}_{\alpha \to \beta})$	stands for	$\forall\, x, x' : \alpha \,.\, \mathbf{F}_{\alpha \to \beta}\, x = \mathbf{F}_{\alpha \to \beta}\, x' \Rightarrow x = x'$.
$\mathsf{INJ2}(\mathbf{F}_{\alpha \to \beta \to \gamma})$	stands for	$\forall\, x, x' : \alpha,\ y, y' : \beta \,.$ $\mathbf{F}_{\alpha \to \beta \to \gamma}\, x\, y = \mathbf{F}_{\alpha \to \beta \to \gamma}\, x'\, y' \Rightarrow$ $(x = x' \wedge y = y')$.
$\mathsf{BIJ}(\mathbf{F}_{\alpha \to \beta})$	stands for	$\mathsf{TOTAL}(\mathbf{F}_{\alpha \to \beta}) \wedge \mathsf{SURJ}(\mathbf{F}_{\alpha \to \beta}) \wedge \mathsf{INJ}(\mathbf{F}_{\alpha \to \beta})$.
$\mathsf{DISTINCT}(\mathbf{A}^1_\alpha, \mathbf{A}^2_\alpha)$	stands for	$(\mathbf{A}^1_\alpha \neq \mathbf{A}^2_\alpha)$.
$\mathsf{DISTINCT}(\mathbf{A}^1_\alpha, \ldots, \mathbf{A}^n_\alpha)$	stands for	$(\mathbf{A}^1_\alpha \neq \mathbf{A}^2_\alpha) \wedge \cdots \wedge (\mathbf{A}^1_\alpha \neq \mathbf{A}^n_\alpha) \wedge$ $\mathsf{DISTINCT}(\mathbf{A}^2_\alpha, \ldots, \mathbf{A}^n_\alpha)$ where $n \geq 3$.

Notational Definitions for Quasitypes [1/2]

$(\lambda\, \mathbf{x} : \mathbf{Q}_{\{\alpha\}} \,.\, \mathbf{B}_\beta)$	stands for	$\lambda\, \mathbf{x} : \alpha \,.\, (\mathbf{x} \in \mathbf{Q}_{\{\alpha\}} \mapsto \mathbf{B}_\beta \mid \bot_\beta)$.
$(\forall\, \mathbf{x} : \mathbf{Q}_{\{\alpha\}} \,.\, \mathbf{B}_o)$	stands for	$\forall\, \mathbf{x} : \alpha \,.\, (\mathbf{x} \in \mathbf{Q}_{\{\alpha\}} \Rightarrow \mathbf{B}_o)$.
$(\exists\, \mathbf{x} : \mathbf{Q}_{\{\alpha\}} \,.\, \mathbf{B}_o)$	stands for	$\exists\, \mathbf{x} : \alpha \,.\, (\mathbf{x} \in \mathbf{Q}_{\{\alpha\}} \wedge \mathbf{B}_o)$.
$(\mathrm{I}\, \mathbf{x} : \mathbf{Q}_{\{\alpha\}} \,.\, \mathbf{B}_o)$	stands for	$\mathrm{I}\, \mathbf{x} : \alpha \,.\, (\mathbf{x} \in \mathbf{Q}_{\{\alpha\}} \wedge \mathbf{B}_o)$.
$(\mathbf{A}_\alpha \downarrow \mathbf{Q}_{\{\alpha\}})$	stands for	$\mathbf{A}_\alpha\!\downarrow \wedge\, \mathbf{A}_\alpha \in \mathbf{Q}_{\{\alpha\}}$.
$(\mathbf{A}_\alpha \uparrow \mathbf{Q}_{\{\alpha\}})$	stands for	$\neg(\mathbf{A}_\alpha \downarrow \mathbf{Q}_{\{\alpha\}})$.
$\rightarrow_{\{\alpha\}\rightarrow\{\beta\}\rightarrow\{\alpha\rightarrow\beta\}}$	stands for	$\lambda\, s : \{\alpha\} \,.\, \lambda\, t : \{\beta\} \,.$ $\{f : \alpha \rightarrow \beta \mid \forall\, x : \alpha \,.$ $(f\, x)\!\downarrow \Rightarrow (x \in s \wedge f\, x \in t)\}$ where $\beta \neq o$.
$\times_{\{\alpha\}\rightarrow\{\beta\}\rightarrow\{\alpha\times\beta\}}$	stands for	$\lambda\, s : \{\alpha\} \,.\, \lambda\, t : \{\beta\} \,.$ $\{p : \alpha \times \beta \mid$ $\mathrm{fst}_{(\alpha\times\beta)\rightarrow\alpha}\, p \in s \wedge$ $\mathrm{snd}_{(\alpha\times\beta)\rightarrow\beta}\, p \in t\}$
$(\mathbf{Q}_{\{\alpha\}} \rightarrow o)$	stands for	$\{s : \{\alpha\} \mid s \subseteq \mathbf{Q}_{\{\alpha\}}\}$.
$\mathcal{P}(\mathbf{Q}_{\{\alpha\}})$	stands for	$\mathbf{Q}_{\{\alpha\}} \rightarrow o$.
$(\mathbf{Q}_{\{\alpha\}} \rightarrow \mathbf{R}_{\{\beta\}})$	stands for	$\rightarrow_{\{\alpha\}\rightarrow\{\beta\}\rightarrow\{\alpha\rightarrow\beta\}}\; \mathbf{Q}_{\{\alpha\}}\, \mathbf{R}_{\{\beta\}}$ where $\beta \neq o$.
$(\alpha \rightarrow \mathbf{R}_{\{\beta\}})$	stands for	$U_{\{\alpha\}} \rightarrow \mathbf{R}_{\{\beta\}}$ where $\beta \neq o$.
$(\mathbf{Q}_{\{\alpha\}} \rightarrow \beta)$	stands for	$\mathbf{Q}_{\{\alpha\}} \rightarrow U_{\{\beta\}}$ where $\beta \neq o$.
$(\mathbf{Q}_{\{\alpha\}} \times \mathbf{R}_{\{\beta\}})$	stands for	$\times_{\{\alpha\}\rightarrow\{\beta\}\rightarrow\{\alpha\times\beta\}}\; \mathbf{Q}_{\{\alpha\}}\, \mathbf{R}_{\{\beta\}}$.

Notational Definitions for Quasitypes [2/2]

$(\alpha \times \mathbf{R}_{\{\beta\}})$	stands for	$U_{\{\alpha\}} \times \mathbf{R}_{\{\beta\}}$.
$(\mathbf{Q}_{\{\alpha\}} \times \beta)$	stands for	$\mathbf{Q}_{\{\alpha\}} \times U_{\{\beta\}}$.
$\mathsf{TOTAL\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}})$	stands for	$\forall\, x : \mathbf{Q}_{\{\alpha\}} \;.\; (\mathbf{F}_{\alpha\to\beta}\, x) \downarrow \mathbf{R}_{\{\beta\}}$.
$\mathsf{TOTAL\text{-}ON2}(\mathbf{F}_{\alpha\to\beta\to\gamma}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}}, \mathbf{S}_{\{\gamma\}})$	stands for	$\forall\, x : \mathbf{Q}_{\{\alpha\}}, y : \mathbf{R}_{\{\beta\}} \;.\; (\mathbf{F}_{\alpha\to\beta\to\gamma}\, x\, y) \downarrow \mathbf{S}_{\{\gamma\}}$.
$\mathsf{SURJ\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}})$	stands for	$\forall\, y : \mathbf{R}_{\{\beta\}} \;.\; \exists\, x : \mathbf{Q}_{\{\alpha\}} \;.\; \mathbf{F}_{\alpha\to\beta}\, x = y$.
$\mathsf{SURJ\text{-}ON2}(\mathbf{F}_{\alpha\to\beta\to\gamma}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}}, \mathbf{S}_{\{\gamma\}})$	stands for	$\forall\, z : \mathbf{S}_{\{\gamma\}} \;.\; \exists\, x : \mathbf{Q}_{\{\alpha\}}, y : \mathbf{R}_{\{\beta\}} \;.\; \mathbf{F}_{\alpha\to\beta\to\gamma}\, x\, y = z$.
$\mathsf{INJ\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}})$	stands for	$\forall\, x, x' : \mathbf{Q}_{\{\alpha\}} \;.\; \mathbf{F}_{\alpha\to\beta}\, x = \mathbf{F}_{\alpha\to\beta}\, x' \Rightarrow x = x'$.
$\mathsf{INJ\text{-}ON2}(\mathbf{F}_{\alpha\to\beta\to\gamma}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}})$	stands for	$\forall\, x, x' : \mathbf{Q}_{\{\alpha\}}, y, y' : \mathbf{R}_{\{\beta\}} \;.\; \mathbf{F}_{\alpha\to\beta\to\gamma}\, x\, y = \mathbf{F}_{\alpha\to\beta\to\gamma}\, x'\, y' \Rightarrow (x = x' \wedge y = y')$.
$\mathsf{BIJ\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}})$	stands for	$\mathsf{TOTAL\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}}) \wedge \mathsf{SURJ\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}}, \mathbf{R}_{\{\beta\}}) \wedge \mathsf{INJ\text{-}ON}(\mathbf{F}_{\alpha\to\beta}, \mathbf{Q}_{\{\alpha\}})$.
$\mathsf{INF}(\mathbf{Q}_{\{\alpha\}})$	stands for	$\exists\, s : \{\alpha\} \;.\; s \subset \mathbf{Q}_{\{\alpha\}} \wedge \exists\, f : \alpha \to \alpha \;.\; \mathsf{BIJ\text{-}ON}(f, s, \mathbf{Q}_{\{\alpha\}})$.
$\mathsf{FIN}(\mathbf{Q}_{\{\alpha\}})$	stands for	$\neg \mathsf{INF}(\mathbf{Q}_{\{\alpha\}})$.
$\mathsf{COUNT}(\mathbf{Q}_{\{\alpha\}})$	stands for	$\forall\, s : \{\alpha\} \;.\; (s \subseteq \mathbf{Q}_{\{\alpha\}} \wedge \mathsf{INF}(s)) \Rightarrow \exists\, f : \alpha \to \alpha \;.\; \mathsf{BIJ\text{-}ON}(f, s, \mathbf{Q}_{\{\alpha\}})$.

Notational Definitions for Dependent Quasitypes

Let $\gamma = \{\alpha\} \rightarrow (\alpha \rightarrow \{\beta\}) \rightarrow \{\alpha \rightarrow \beta\}$
and $\delta = \{\alpha\} \rightarrow (\alpha \rightarrow \{\beta\}) \rightarrow \{\alpha \times \beta\}$.

Π_γ	stands for	$\lambda\, s : \{\alpha\} \,.\, \lambda\, t : \alpha \rightarrow \{\beta\} \,.$ $\{f : \alpha \rightarrow \beta \mid \forall\, x : \alpha \,.$ $(f\, x)\!\downarrow \Rightarrow (x \in s \wedge f\, x \in t\, x)\}$ where $\beta \neq o$.
Σ_δ	stands for	$\lambda\, s : \{\alpha\} \,.\, \lambda\, t : \alpha \rightarrow \{\beta\} \,.$ $\{p : \alpha \times \beta \mid$ $\mathsf{fst}_{(\alpha \times \beta) \rightarrow \alpha}\, p \in s \wedge$ $\mathsf{snd}_{(\alpha \times \beta) \rightarrow \beta}\, p \in t\, (\mathsf{fst}_{(\alpha \times \beta) \rightarrow \alpha}\, p)\}$
$(\Pi\, \mathbf{x} : \mathbf{Q}_{\{\alpha\}} \,.\, \mathbf{R}_{\{\beta\}})$	stands for	$\Pi_\gamma\, \mathbf{Q}_{\{\alpha\}} (\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{R}_{\{\beta\}})$ where $\beta \neq o$.
$(\Sigma\, \mathbf{x} : \mathbf{Q}_{\{\alpha\}} \,.\, \mathbf{R}_{\{\beta\}})$	stands for	$\Sigma_\delta\, \mathbf{Q}_{\{\alpha\}} (\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{R}_{\{\beta\}})$.

Additional Notational Conventions

- Constant symbols are used to write pseudoconstants and parametric pseudoconstants (NC 9).
- There are implicit notational definitions for the infix operators associated with a weak order operator (NC 10).
- Like quantifiers over types can be merged (NC 11).
- Like quantifiers over quasitypes can be merged (NC 12).
- Abbreviations are written in uppercase (NC 13).
- Bound variables introduced in the RHS of an abbreviation are chosen so they are not free in the LHS (NC 14).

4. Beta-Reduction and Substitution

Beta-Reduction Theorem

- Theorem. If $\mathbf{A}_\alpha$ is free for $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$, then

$$\mathbf{A}_\alpha{\downarrow} \Rightarrow (\lambda\, \mathbf{x} : \alpha\, .\, \mathbf{B}_\beta)\, \mathbf{A}_\alpha \simeq \mathbf{B}_\beta[(\mathbf{x} : \alpha) \mapsto \mathbf{A}_\alpha]$$

  is valid.

- Lemma. Let M be a general model of L that interprets $\mathbf{A}_\alpha$ and $\mathbf{B}_\beta$, and let $\varphi \in \mathsf{assign}(M)$. If $\mathbf{A}_\alpha$ is free for $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$ and $V^M_\varphi(\mathbf{A}_\alpha)$ is defined, then

$$V^M_{\varphi[(\mathbf{x}:\alpha) \mapsto V^M_\varphi(\mathbf{A}_\alpha)]}(\mathbf{B}_\beta) \simeq V^M_\varphi(\mathbf{B}_\beta[(\mathbf{x} : \alpha) \mapsto \mathbf{A}_\alpha]).$$

- Proposition. The formula

$$\mathbf{A}_\alpha{\uparrow} \Rightarrow (\lambda\, \mathbf{x} : \alpha\, .\, \mathbf{B}_\beta)\, \mathbf{A}_\alpha \simeq \bot_\beta$$

  is valid.

Universal Instantiation Theorem

- Theorem. If $\mathbf{A}_\alpha$ is free for $(\mathbf{x} : \alpha)$ in $\mathbf{B}_o$, then

$$((\forall\, \mathbf{x} : \alpha\ .\ \mathbf{B}_o) \wedge \mathbf{A}_\alpha\!\downarrow) \Rightarrow \mathbf{B}_o[(\mathbf{x} : \alpha) \mapsto \mathbf{A}_\alpha]$$

  is valid.

Invalid Beta-Reduction Example

Let $\textbf{F}_{\alpha \rightarrow \alpha \rightarrow \alpha} \equiv \lambda\, \textbf{x} : \alpha \,.\, \lambda\, \textbf{y} : \alpha \,.\, (\textbf{x} : \alpha)$. Then $\textbf{F}_{\alpha \rightarrow \alpha \rightarrow \alpha}\, \textbf{A}_\alpha$ should denote a constant function if $\textbf{A}_\alpha$ is defined and should be undefined if $\textbf{A}_\alpha$ is undefined.

1. $\textbf{F}_{\alpha \rightarrow \alpha \rightarrow \alpha}\, (\textbf{y} : \alpha)$ beta-reduces to:

   $$(\lambda\, \textbf{y} : \alpha \,.\, (\textbf{x} : \alpha))[(\textbf{x} : \alpha) \mapsto (\textbf{y} : \alpha)] \equiv \lambda\, \textbf{y} : \alpha \,.\, (\textbf{y} : \alpha).$$

   Hence, although $(\textbf{y} : \alpha)$ is defined, $\textbf{F}_{\alpha \rightarrow \alpha \rightarrow \alpha}\, (\textbf{y} : \alpha)$ beta-reduces to an identity function, not a constant function as expected. This happens because $(\textbf{y} : \alpha)$ is not free for $(\textbf{x} : \alpha)$ in $\lambda\, \textbf{y} : \alpha \,.\, (\textbf{x} : \alpha)$.
2. $\textbf{F}_{\alpha \rightarrow \alpha \rightarrow \alpha}\, \bot_\alpha$ beta-reduces to:

   $$(\lambda\, \textbf{y} : \alpha \,.\, (\textbf{x} : \alpha))[(\textbf{x} : \alpha) \mapsto \bot_\alpha] \equiv \lambda\, \textbf{y} : \alpha \,.\, \bot_\alpha.$$

   Hence $\textbf{F}_{\alpha \rightarrow \alpha \rightarrow \alpha}\, \bot_\alpha$ beta-reduces to the empty function and is defined, not undefined as expected. This happens because $\bot_\alpha$ is undefined.

Alpha-Conversion Theorem

- Theorem. If $(\mathbf{y} : \alpha)$ is not free in $\mathbf{B}_\beta$ and $(\mathbf{y} : \alpha)$ is free for $(\mathbf{x} : \alpha)$ in $\mathbf{B}_\beta$, then

  $$(\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{B}_\beta) = (\lambda\, \mathbf{y} : \alpha \,.\, \mathbf{B}_\beta[(\mathbf{x} : \alpha) \mapsto (\mathbf{y} : \alpha)])$$

  is valid.
- Corollary. If $(\mathbf{y} : \alpha)$ does not occur in $\mathbf{B}_\beta$, then

  $$(\lambda\, \mathbf{x} : \alpha \,.\, \mathbf{B}_\beta) = (\lambda\, \mathbf{y} : \alpha \,.\, \mathbf{B}_\beta[(\mathbf{x} : \alpha) \mapsto (\mathbf{y} : \alpha)])$$

  is valid.

The End.