Higher-Order Termination
From Kruskal to Computability

Jean-Pierre Jouannaud
École Polytechnique
91400 Palaiseau, France

Project LogiCal, Pôle Commun de Recherche en Informatique du Plateau de Saclay, CNRS, École Polytechnique, INRIA, Université Paris-Sud.

Joint work with Frédéric Blanqui and Albert Rubio

LPAR 2006, Pnomh Penh, november 17, 2006

Outline

Higher-order algebras [Jouannaud, Rubio, JACM to appear]

Types, signatures and terms

- $\mathcal{S}$: set of sort symbols of a fixed arity, denoted by $s : *^n \Rightarrow *$
- $\mathcal{S}^\forall$: set of sort variables
- Types

$$\mathcal{T}_\mathcal{S} := \alpha \mid s(\mathcal{T}_\mathcal{S}^n) \mid (\mathcal{T}_\mathcal{S} \rightarrow \mathcal{T}_\mathcal{S})$$
$$\text{for } \alpha \in \mathcal{S}^\forall \text{ and } s : *^n \Rightarrow * \ \in \mathcal{S}$$

- Terms

$$\mathcal{T} := \mathcal{X} \mid (\lambda\mathcal{X} : \mathcal{T}_\mathcal{S}.\mathcal{T}) \mid @(\mathcal{T}, \mathcal{T}) \mid \mathcal{F}(\mathcal{T}, \ldots, \mathcal{T}).$$

We will sometimes write $\mathcal{T}(\mathcal{T})$ for $@(\mathcal{T}, \mathcal{T})$.

Types, signatures and terms

- $\mathcal{S}$: set of sort symbols of a fixed arity, denoted by $s : *^n \Rightarrow *$
- $\mathcal{S}^\forall$: set of sort variables
- Types

$$\mathcal{T}_\mathcal{S} := \alpha \mid s(\mathcal{T}_\mathcal{S}^n) \mid (\mathcal{T}_\mathcal{S} \to \mathcal{T}_\mathcal{S})$$
$$\text{for } \alpha \in \mathcal{S}^\forall \text{ and } s : *^n \Rightarrow * \;\in \mathcal{S}$$

- Terms

$$\mathcal{T} := \mathcal{X} \mid (\lambda \mathcal{X} : \mathcal{T}_\mathcal{S}.\mathcal{T}) \mid @(\mathcal{T}, \mathcal{T}) \mid \mathcal{F}(\mathcal{T}, \ldots, \mathcal{T}).$$

We will sometimes write $\mathcal{T}(\mathcal{T})$ for $@(\mathcal{T}, \mathcal{T})$.

Types, signatures and terms

- $\mathcal{S}$: set of sort symbols of a fixed arity, denoted by $\mathsf{s} : *^n \Rightarrow *$
- $\mathcal{S}^\forall$: set of sort variables
- Types

$$\mathcal{T}_\mathcal{S} := \alpha \mid \mathsf{s}(\mathcal{T}_\mathcal{S}^n) \mid (\mathcal{T}_\mathcal{S} \rightarrow \mathcal{T}_\mathcal{S})$$

$$\text{for } \alpha \in \mathcal{S}^\forall \text{ and } \mathsf{s} : *^n \Rightarrow * \ \in \mathcal{S}$$

- Terms

$$\mathcal{T} := \mathcal{X} \mid (\lambda\mathcal{X} : \mathcal{T}_\mathcal{S}.\mathcal{T}) \mid @(\mathcal{T}, \mathcal{T}) \mid \mathcal{F}(\mathcal{T}, \ldots, \mathcal{T}).$$

We will sometimes write $\mathcal{T}(\mathcal{T})$ for $@(\mathcal{T}, \mathcal{T})$.

- $\mathcal{S}$: set of sort symbols of a fixed arity, denoted by $s : *^n \Rightarrow *$
- $\mathcal{S}^\forall$: set of sort variables
- Types

$$\mathcal{T}_\mathcal{S} := \alpha \mid s(\mathcal{T}_\mathcal{S}^n) \mid (\mathcal{T}_\mathcal{S} \to \mathcal{T}_\mathcal{S})$$
$$\text{for } \alpha \in \mathcal{S}^\forall \text{ and } s : *^n \Rightarrow * \ \in \mathcal{S}$$

- Terms

$$\mathcal{T} := \mathcal{X} \mid (\lambda \mathcal{X} : \mathcal{T}_\mathcal{S}.\mathcal{T}) \mid @(\mathcal{T}, \mathcal{T}) \mid \mathcal{F}(\mathcal{T}, \ldots, \mathcal{T}).$$

We will sometimes write $\mathcal{T}(\mathcal{T})$ for $@(\mathcal{T}, \mathcal{T})$.

Typing rules

Functions:

$$\frac{f : \sigma_1 \times \ldots \times \sigma_n \Rightarrow \sigma \quad \Gamma \vdash t_1 : \tau_1 \ \ldots \ \Gamma \vdash t_n : \tau_n \quad \theta = mgu(\sigma_1 = \tau_1 \ \& \ \ldots \ \& \ \sigma_n = \tau_n)}{\Gamma \vdash f(t_1, \ldots, t_n) : \sigma\theta}$$

Variables:

$$\frac{x : \sigma \in \Gamma}{\Gamma \vdash x : \sigma}$$

Abstraction:

$$\frac{\Gamma \cup \{x : \sigma\} \vdash t : \tau}{\Gamma \vdash (\lambda x : \sigma . t) : \sigma \rightarrow \tau}$$

Application:

$$\frac{\Gamma \vdash s : \sigma_1 \rightarrow \sigma \quad \Gamma \vdash t : \tau_1 \quad \theta = mgu(\sigma_1 = \tau_1)}{\Gamma \vdash @(s, t) : \sigma\theta}$$

$$
\begin{array}{lcl}
\mathbb{N}, \alpha & : & * \\
0, x & : & \mathbb{N} \\
s & : & \mathbb{N} \Rightarrow \mathbb{N} \\
rec & : & \mathbb{N} \times \alpha \times (\mathbb{N} \rightarrow \alpha \rightarrow \alpha) \Rightarrow \alpha \\
U & : & \alpha \qquad\qquad X : \mathbb{N} \rightarrow \alpha \rightarrow \alpha
\end{array}
$$

$$
\begin{array}{lcl}
rec(0, U, X) & \rightarrow & U \\
rec(s(x), U, X) & \rightarrow & @(X, x, rec(x, U, X))
\end{array}
$$

Rules use first-order pattern matching

Brouwer's ordinals

$$Ord, \alpha : *$$

$$0 : Ord \qquad s : Ord \to Ord \qquad lim : (\mathbb{N} \to Ord) \Rightarrow Ord$$

$$rec : Ord \times \alpha \times (Ord \to \alpha \to \alpha) \times ((\mathbb{N} \to Ord) \to (\mathbb{N} \to \alpha) \Rightarrow \alpha) \\ \to \alpha$$

$$x : Ord \qquad F : \mathbb{N} \to Ord$$

$$U : \alpha \qquad X : Ord \to \alpha \to \alpha \qquad W : (\mathbb{N} \to Ord) \to (\mathbb{N} \to \alpha) \to \alpha$$

$$\begin{array}{lcl} rec(0, U, X, W) & \to & U \\ rec(s(x), U, X, W) & \to & @(X, x, rec(x, U, X, W)) \\ rec(lim(F), U, X, W) & \to & @(W, F, \lambda n.rec(@(F, n), U, X, W)) \end{array}$$

Goal

Automate
strong normalization
proofs

Tait and Girard's computability predicate method

Language

- Simple type discipline
- One rewrite schema:

$$@(\lambda x.u, v) \rightarrow u\{x \mapsto v\}$$

Tait

$[\![\sigma]\!]$, the computability predicate of type σ s.t.:
(i) computable terms are strongly normalizing;
(ii) reducts of computable terms are computable;
(iii) a neutral term u is computable iff all its reducts are computable;
(iv) $u : \sigma \to \tau$ is computable iff so is $@(u, v)$ for all computable v;
(v) (optionnal) $\lambda x.u$ is computable iff so is $u\{x \mapsto v\}$ for all computable v.

Except (v), no explicit mention of β-reduction.

Examples of computability predicates

- Basic types: there are two possibilities

  $s : \sigma \in [\![\sigma]\!]$ iff s is strongly normalizing
  or
  $s : \sigma \in [\![\sigma]\!]$ iff $\forall t : \tau$ s.t. $s \longrightarrow t$ then $t \in [\![\tau]\!]$
  or ...

- Functional types:
  $s : \theta \rightarrow \tau \in [\![\sigma \rightarrow \tau]\!]$ iff $@(s, u) : \tau \in [\![\tau]\!]$ for every $u : \theta \in [\![\theta]\!]$.

Given term s and computable substitution γ, then $s\gamma$ is computable.
By induction on the structure of terms.

1. $s \in \mathcal{X}$. $s\gamma$ computable by assumption.
2. $s = @(u, v)$. $u\gamma$ and $v\gamma$ are computable by induction hypothesis, hence $s\gamma = @(u\gamma, v\gamma)$ is computable by computability property (iv).
3. $s = \lambda x.u$. By property (v), $s\gamma = \lambda x.u\gamma$ is computable iff $u\gamma\{x \mapsto v\} = u(\gamma \cup \{x \mapsto v\})$ is computable for all computable v. We conclude by induction hypothesis.

Recursive path ordering

Recursive path ordering: $s \succ_{rpo} t$ iff

1. $s = f(\overline{s})$ with $f \in \mathcal{F}$, and $u \underset{rpo}{\succeq} t$ for some $u \in \overline{s}$
2. $s = f(\overline{s})$ with $f \in \mathcal{F}$, and $t = g(\overline{t})$ with $f >_{\mathcal{F}} g$, and A
3. $s = f(\overline{s})$ and $t = g(\overline{t})$ with $f =_{\mathcal{F}} g$, and A and $\overline{s} \; (\underset{rpo}{\succ})_{stat_f} \; \overline{t}$

where $s \underset{rpo}{\succeq} t$ iff $s \underset{rpo}{\succ} t$ or $s = t$

Computability is defined as strong normalization, implying all computability properties trivially. We add a new computability property:

(vi) Let $f \in \mathcal{F}_n$ and $\overline{s}$ be computable terms. Then $f(\overline{s})$ is computable.

First: proof of property (vi)

The restriction of $\succ_{rpo}$ to terms smaller than or equal to the terms in $\overline{s}$ w.r.t. $\succ_{rpo}$ is a well-founded ordering which we use for building an outer induction on the pairs $(f, \overline{s})$ ordered by $(>_{\mathcal{F}}, (\succ_{rpo})_{stat_f})_{lex}$.

We now show that $f(\overline{s})$ is computable by proving that t is computable for all t such that $f(\overline{s}) \succ_{rpo} t$. This property is itself proved by an inner induction on $| t |$, and by case analysis upon the proof that $f(\overline{s}) \succ_{rpo} t$.

Proof of (vi) continued

1. subterm: $\exists u \in \overline{s}$ such that $u \succ_{rpo} t$. By assumption, u is computable. Reduct t too.
2. precedence: $t = g(\overline{t})$, $f >_{\mathcal{F}} g$, and $s \succ_{rpo} \overline{t}$. By inner induction, $\overline{t}$ is computable. By outer induction, $g(\overline{t}) = t$ is computable.
3. status: $t = g(\overline{t})$ with $f =_{\mathcal{F}} g \in Lex$, $\overline{s}(\succ_{rpo})_{lex}\overline{t}$, and $s \succ_{rpo} \overline{t}$. By inner induction, $\overline{t}$ is computable. By outer induction, $g(\overline{t}) = t$ is computable. $\square$

Second (Main Lemma) : every term is computable.

Proof by induction on the structure of terms. If t is a variable, done. Otherwise $t = f(\bar{t})$.

By induction hypothesis, $\bar{t}$ is computable.
By property (vi), t is computable. Done.

The well-foundedness of $\succ_{rpo}$ follows by Property (i).

General Schema

Closure and General Schema

The computability closure $\mathcal{CC}(t = f(\bar{t}))$, with $f \in \mathcal{F}$, is the set $\mathcal{CC}(t, \emptyset)$, s.t. $\mathcal{CC}(t, \mathcal{V})$, with $\mathcal{V} \cap \mathcal{V}ar(t) = \emptyset$, is the smallest set of typable terms containing all variables in $\mathcal{V}$ and terms in $\bar{t}$, closed under:

1. basic type subterm; application; abstraction;
2. precedence: let $f >_{\mathcal{F}} g$, and $\bar{s} \in \mathcal{CC}(t, \mathcal{V})$; then $g(\bar{s}) \in \mathcal{CC}(t, \mathcal{V})$;
3. recursive call: let $f(\bar{s})$ be a term s.t. terms in $\bar{s}$ belong to $\mathcal{CC}(t, \mathcal{V})$ and $\bar{t}(\longrightarrow_{\beta \cup \rhd})_{stat_f} \bar{s}$; then $g(\bar{s}) \in \mathcal{CC}(t, \mathcal{V})$ for every $g =_{\mathcal{F}} f$;
4. reduction: let $u \in \mathcal{CC}(t, \mathcal{V})$, and $u \longrightarrow_{\beta \cup \rhd} v$; then $v \in \mathcal{CC}(t, \mathcal{V})$.

General schema [Blanqui, Jouannaud and Okada, TCS 2001]

We say that a rewrite system R satisfies the *general schema* if

$$R = \{f(\bar{l}) \rightarrow r \mid r \in \mathcal{CC}(f(\bar{l}))\}$$

We now consider computability with respect to the rewrite relation $\longrightarrow_R \cup \longrightarrow_\beta$, and add the computability property (vii) whose proof can be easily adapted from the previous one. We can then add a new case in Tait's Main Lemma, for terms headed by an algebraic function symbol.

Conclusion: $\longrightarrow_\beta \cup \longrightarrow_R$ is SN.

Example : System T

$$rec(s(x), U, X) \rightarrow @(X, x, rec(x, U, X))$$

Higher Order Recursive Path Ordering

Higher-Order Recursive Path Ordering: Ingredients

- A type quasi-ordering $\geq_{\mathcal{T}_\mathcal{S}}$ s.t.
  (i) $>_{\mathcal{T}_\mathcal{S}}$ is well-founded;
  (ii) Arrow preservation: $\tau \to \sigma =_{\mathcal{T}_\mathcal{S}} \alpha$ iff $\alpha = \tau' \to \sigma'$, $\tau' =_{\mathcal{T}_\mathcal{S}} \tau$ and $\sigma =_{\mathcal{T}_\mathcal{S}} \sigma'$;
  (iii) Arrow decreasingness: $\tau \to \sigma >_{\mathcal{T}_\mathcal{S}} \alpha$ implies $\sigma \geq_{\mathcal{T}_\mathcal{S}} \alpha$ or $\alpha = \tau' \to \sigma', \tau' =_{\mathcal{T}_\mathcal{S}} \tau$ and $\sigma >_{\mathcal{T}_\mathcal{S}} \sigma'$;
  (iv) Arrow monotonicity: $\tau \geq_{\mathcal{T}_\mathcal{S}} \sigma$ implies $\alpha \to \tau \geq_{\mathcal{T}_\mathcal{S}} \alpha \to \sigma$ and $\tau \to \alpha \geq_{\mathcal{T}_\mathcal{S}} \sigma \to \alpha$;
- A well-founded precedence $\geq_{\mathcal{F}}$ s.t.
  $@ <_{\mathcal{F}} f \in \mathcal{F}$
- A status $stat_f \in \{Mul, Lex\}$ for every $f \in \mathcal{F}$.

Definition : $s \succ_{horpo}$ iff $\sigma \geq_{\mathcal{T}_S} \tau$ and Case

1: $s = f(\bar{s})$ with $f \in \mathcal{F} \cup \{@\}$

1. $u \succeq_{horpo} t$ for $u \in \bar{s}$
2. $t = g(\bar{t})$ with $f >_{\mathcal{F}} g$ and $s \succ_{horpo} \bar{t}$
3. $t = g(\bar{t})$ with $f =_{\mathcal{F}} g$, $s \succ_{horpo} \bar{t}$ and $\bar{s}\ (\succ_{horpo})_{stat_f}\ \bar{t}$

2: $s = @(v, w)$ $v = \lambda x.u$ and $u\{x \mapsto w\} \succ_{horpo} t$

3: $s = \lambda x : \alpha.u$ and

1. $u\{x \mapsto y\} \succ_{horpo} t$, for some fresh $y : \alpha$
2. $t = \lambda y : \beta.v$, $y \notin \mathcal{V}ar(v)$, $\alpha =_{\mathcal{T}_S} \beta$ and $u \succ_{horpo} v$
3. $u = @(v, x)$, $x \notin \mathcal{V}ar(v)$ and $v \succ_{horpo} t$

where $s \succeq_{horpo} t$ iff $s \succ_{horpo} t$ or $s =_{\alpha} t$

Definition : $s \succ_{horpo}$ iff $\sigma \geq_{\mathcal{T}_S} \tau$ and Case

1: $s = f(\overline{s})$ with $f \in \mathcal{F} \cup \{@\}$

1. $u \succeq_{horpo} t$ for $u \in \overline{s}$
2. $t = g(\overline{t})$ with $f >_{\mathcal{F}} g$ and $s \succ_{horpo} \overline{t}$
3. $t = g(\overline{t})$ with $f =_{\mathcal{F}} g$, $s \succ_{horpo} \overline{t}$ and $\overline{s}\ (\succ_{horpo})_{stat_f}\ \overline{t}$

2: $s = @(v, w)$ $v = \lambda x.u$ and $u\{x \mapsto w\} \succ_{horpo} t$

3: $s = \lambda x : \alpha.u$ and

1. $u\{x \mapsto y\} \succ_{horpo} t$, for some fresh $y : \alpha$
2. $t = \lambda y : \beta.v$, $y \notin \mathcal{V}ar(v)$, $\alpha =_{\mathcal{T}_S} \beta$ and $u \succ_{horpo} v$
3. $u = @(v, x)$, $x \notin \mathcal{V}ar(v)$ and $v \succ_{horpo} t$

where $s \succeq_{horpo} t$ iff $s \succ_{horpo} t$ or $s =_{\alpha} t$

1: $s = f(\overline{s})$ with $f \in \mathcal{F} \cup \{@\}$

1. $u \succeq_{horpo} t$ for $u \in \overline{s}$
2. $t = g(\overline{t})$ with $f >_{\mathcal{F}} g$ and $s \succ_{horpo} \overline{t}$
3. $t = g(\overline{t})$ with $f =_{\mathcal{F}} g$, $s \succ_{horpo} \overline{t}$ and $\overline{s}\ (\succ_{horpo})_{stat_f}\ \overline{t}$

2: $s = @(v, w)$ $v = \lambda x.u$ and $u\{x \mapsto w\} \succ_{horpo} t$

3: $s = \lambda x : \alpha.u$ and

1. $u\{x \mapsto y\} \succ_{horpo} t$, for some fresh $y : \alpha$
2. $t = \lambda y : \beta.v$, $y \notin \mathcal{V}ar(v)$, $\alpha =_{\mathcal{T}_S} \beta$ and $u \succ_{horpo} v$
3. $u = @(v, x)$, $x \notin \mathcal{V}ar(v)$ and $v \succ_{horpo} t$

where $s \succeq_{horpo} t$ iff $s \succ_{horpo} t$ or $s =_{\alpha} t$

Example: simple proof of system T

$$rec(s(x), U, X) \longrightarrow @(X, x, rec(x, U, X))$$

HORPO and Closure

Combining HORPO and closure

We change the subterm case:

1. $s = f(\overline{s})$ with $f \in \mathcal{F}$ and $u \underset{horpo}{\succeq} t$ for $u \in \overline{s}$

in

$s = f(\overline{s})$ with $f \in \mathcal{F}$ and $u \underset{horpo}{\succeq} t$ for $u \in \mathcal{CC}(f(\overline{s}))$

Drawbacks:

1. Decidability of HORPO is lost;
2. There are many repetitions;
3. Type checking is no much help, but a lot of burden;
4. Treatment of abstractions remains weak.

New HORPO with integrated closure mechanism

Ingredients:

1. A set of strictly positive inductive types inducing an accessibility relationship $\overline{s} \trianglerighteq_{acc} v$ such that $v \in \overline{u}$ or v is accessible from $u \in \overline{s}$
2. a precedence on function symbols
3. a congruence on types
4. $s \succ^X t$ for the main ordering
5. $s : \sigma \succ^X_{\mathcal{T}_S} t : \tau$ for $s \succ^X t$ and $\sigma =_{\mathcal{T}_S} \tau$
6. $l \succ^{\emptyset}_{\mathcal{T}_S} r$ as initial call for each $l \rightarrow r \in R$

Case 1: $s = f(\overline{s})$ with $f \in \mathcal{F}$ and $t \in X$ or

1. $u \succeq^X_{\mathcal{T}_S} t$ for some u such that $\overline{s} \trianglerighteq_{acc} u$
2. $t = g(\overline{t})$ with $f >_{\mathcal{F}} g \in \mathcal{F} \cup \{@\}$ and $s \succ^X \overline{t}$
3. $t = g(\overline{t})$ with $f =_{\mathcal{F}} g \in \mathcal{F}$ and $s \succ^X \overline{t}$ and $\overline{s}(\succ^X_{\mathcal{T}_S})_{stat_f}\overline{t}$
4. $t = \lambda x.u$ with $x \notin X$ and $f(\overline{s}) \succ^{X \cup \{x\}} u$

Case 2: $s = @(v, w)$ and

1. $t = @(u, r)$ and $(v, w)(\succ^X_{\mathcal{T}_S})_{mon}(u, r)$
2. $v = \lambda x.u$ and $u\{x \mapsto w\} \succ^X t$

Case 3: $s = \lambda x : \alpha.u$ and

1. $t = \lambda x : \beta.v$, $x \notin X$, $\alpha =_{\mathcal{T}_S} \beta$ and $u \succ^{X \cup \{x\}} v$
2. $u = @(v, x)$, $x \notin \mathcal{V}ar(v)$ and $v \succ^X t$

Definition : $s \succ^X t$ iff

Case 1: $s = f(\overline{s})$ with $f \in \mathcal{F}$ and $t \in X$ or

1. $u \succeq^X_{\mathcal{T}_S} t$ for some u such that $\overline{s} \trianglerighteq_{acc} u$
2. $t = g(\overline{t})$ with $f >_{\mathcal{F}} g \in \mathcal{F} \cup \{@\}$ and $s \succ^X \overline{t}$
3. $t = g(\overline{t})$ with $f =_{\mathcal{F}} g \in \mathcal{F}$ and $s \succ^X \overline{t}$ and $\overline{s}(\succ^X_{\mathcal{T}_S})_{stat_f}\overline{t}$
4. $t = \lambda x.u$ with $x \notin X$ and $f(\overline{s}) \succ^{X \cup \{x\}} u$

Case 2: $s = @(v, w)$ and

1. $t = @(u, r)$ and $(v, w)(\succ^X_{\mathcal{T}_S})_{mon}(u, r)$
2. $v = \lambda x.u$ and $u\{x \mapsto w\} \succ^X t$

Case 3: $s = \lambda x : \alpha.u$ and

1. $t = \lambda x : \beta.v$, $x \notin X$, $\alpha =_{\mathcal{T}_S} \beta$ and $u \succ^{X \cup \{x\}} v$
2. $u = @(v, x)$, $x \notin \mathcal{V}ar(v)$ and $v \succ^X t$

Definition : $s \succ^X t$ iff

Case 1: $s = f(\overline{s})$ with $f \in \mathcal{F}$ and $t \in X$ or

1. $u \succeq^X_{\mathcal{T}_S} t$ for some u such that $\overline{s} \trianglerighteq_{acc} u$
2. $t = g(\overline{t})$ with $f >_{\mathcal{F}} g \in \mathcal{F} \cup \{@\}$ and $s \succ^X \overline{t}$
3. $t = g(\overline{t})$ with $f =_{\mathcal{F}} g \in \mathcal{F}$ and $s \succ^X \overline{t}$ and $\overline{s}(\succ^X_{\mathcal{T}_S})_{stat_f}\overline{t}$
4. $t = \lambda x.u$ with $x \notin X$ and $f(\overline{s}) \succ^{X \cup \{x\}} u$

Case 2: $s = @(v, w)$ and

1. $t = @(u, r)$ and $(v, w)(\succ^X_{\mathcal{T}_S})_{mon}(u, r)$
2. $v = \lambda x.u$ and $u\{x \mapsto w\} \succ^X t$

Case 3: $s = \lambda x : \alpha.u$ and

1. $t = \lambda x : \beta.v$, $x \notin X$, $\alpha =_{\mathcal{T}_S} \beta$ and $u \succ^{X \cup \{x\}} v$
2. $u = @(v, x)$, $x \notin \mathcal{V}ar(v)$ and $v \succ^X t$

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F,n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F,n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F,n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F,n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F,n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F,n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F,n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 2.1

Brouwer's ordinals

$lim : (\mathbb{N} \rightarrow Ord) \Rightarrow Ord \qquad F : \mathbb{N} \rightarrow Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \rightarrow \alpha \rightarrow \alpha) \times ((\mathbb{N} \rightarrow Or) \rightarrow (\mathbb{N} \rightarrow \alpha) \rightarrow \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul} \{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$
$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul} \{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Brouwer's ordinals

$lim : (\mathbb{N} \rightarrow Ord) \Rightarrow Ord \qquad F : \mathbb{N} \rightarrow Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \rightarrow \alpha \rightarrow \alpha) \times ((\mathbb{N} \rightarrow Or) \rightarrow (\mathbb{N} \rightarrow \alpha) \rightarrow \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_\mathcal{S}} @(W, F, \lambda n.rec(@(F,n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_\mathcal{S}} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F,n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F,n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F,n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_\mathcal{S}})_{mul}\{@(F,n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_\mathcal{S}} @(F,n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F,n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 4.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 4.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $$rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$$
   yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 4.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$
$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 4.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul} \{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 4.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Brouwer's ordinals

$lim : (\mathbb{N} \to Ord) \Rightarrow Ord \qquad F : \mathbb{N} \to Ord \qquad n : \mathbb{N}$

$rec : Or \times \alpha \times (Or \to \alpha \to \alpha) \times ((\mathbb{N} \to Or) \to (\mathbb{N} \to \alpha) \to \alpha) \Rightarrow \alpha$

1. $rec(lim(F), U, X, W) \succ^{\emptyset}_{\mathcal{T}_S} @(W, F, \lambda n.rec(@(F, n), U, X, W))$ yields 2 subgoals:
2. $\alpha =_{\mathcal{T}_S} \alpha$ which is trivially satisfied, and
3. $rec(lim(F), U, X, W) \succ^{\emptyset} \{W, F, \lambda n.rec(@(F, n), U, X, W)\}$ which simplifies to:
4. $rec(lim(F), U, X, W) \succ^{\emptyset} W$ which succeeds by Case 1.1,
5. $rec(lim(F), U, X, W) \succ^{\emptyset} F$, which succeeds by Case 1.1,
6. $rec(lim(F), U, X, W) \succ^{\emptyset} \lambda n.rec(@(F, n), U, X, W)$ yields
7. $rec(lim(F), U, X, W) \succ^{\{n\}} rec(@(F, n), U, X, W)$ yields
8. $\{lim(F), U, X, W\}(\succ^{\{n\}}_{\mathcal{T}_S})_{mul}\{@(F, n), U, X, W\}$, hence
9. $lim(F) \succ^{\{n\}}_{\mathcal{T}_S} @(F, n)$ whose type-check succeeds, and yields
10. $lim(F) \succ^{\{n\}} F$ which succeeds by Case 1.2, and
11. $lim(F) \succ^{\{n\}} n$ which succeeds by Case 1.
12. $rec(lim(F), U, X, W) \succ^{\{n\}} \{@(F, n), U, X, W\}$, our remaining goal, succeeds easily by Cases 1.2, 1 and 1.1

Conclusion

Achievements: A quite powerful powerful which adapts easily to higher-order rewriting based on higher-order pattern matching. See [Jouannaud and Rubio, RTA'2006]

Remaining problems:

- Use term interpretations instead of a precedence on function symbols;
- Integrate AC;
- Generalization to the Calculus of Inductive Constructions;
- Develop the tool (see our Web page).

Acknowledgments: to Mitsuhiro Okada for our long standing collaboration on these matters.