# Domains for Guarded Recursion:

## Building with Finite Approximations

Yuriy Dupyn

2024

In this mathematical essay we explore $\omega$-sets which model guarded recursion.
The exposition is inspired by reading last two chapters of Lambda-kalkul by Jiří Zlatuška.
The content is primarily based on the paper

[1] First steps in Synthetic Guarded Domain Theory:
Step-indexing in the topos of trees

but also on

[2] The Guarded Lambda-Calculus:
Programming and reasoning with guarded recursion for coinductive types

[3.1] Recursive Types Reduced to Inductive Types (by Peter Freyd)
[3.2] Remarks on Algebraically Compact Categories

# Recollection on Set

Let us consider sets. Denote the category of sets by Set.

Write $X : Set$ for $X$ is a set,

$f : X \to Y : Set$ for $f$ is a set map from set $X$ to set $Y$.

When Set is clear from context, will just write

$$f : X \longrightarrow Y$$

Set is equipped with various constructions.

We recall some of them.

(*) cartesian product $X, Y \rightsquigarrow X \times Y$

equipped with pairing $x, y \rightsquigarrow (x, y)$

and projections

$$X \times Y \xrightarrow{fst} X \qquad (x,y) \rightsquigarrow x$$

$$X \times Y \longrightarrow Y \qquad (x,y) \rightsquigarrow y$$

Todo: Change this to $Z \to X,\ Z \to Y \rightsquigarrow Z \to X \times Y$

(*) terminal set (singleton set) $1$

which has unique element $() : 1$.

Note $1 : Set$ can be thought of as cartesian product of 0 sets.

(*) coproduct (disjoint sum) $X, Y \rightsquigarrow left(X) + right(Y)$

equipped with injections

$$left : X \longrightarrow left(X) + right(Y) \qquad x \rightsquigarrow left(x)$$

$$right : Y \longrightarrow left(X) + right(Y) \qquad y \rightsquigarrow right(y)$$

and a copairing (case analysis):

for any $Z : Set$,

given $f : X \longrightarrow Z$,

$g : Y \longrightarrow Z$,

we can glue them into

$$case[f|g] : left(X) + right(Y) \longrightarrow Z$$

$$case[f|g](x) := \begin{array}{l} case\ x\ of \\ \quad \mid left(x) \to f(x) \\ \quad \mid right(y) \to g(y) \end{array}$$

[in more classical notation $:= \begin{cases} f(x) & \text{if } x = left(x) \\ g(x) & \text{if } x = right(y) \end{cases}$]

i.e. $case[f|g](left(x)) = f(x)$

$case[f|g](right(y)) = g(x)$

(*) initial set (empty set) $0$,

it doesn't have any elements.

So we can think that $0$ is canonically included in any other set.

Let us denote this inclusion by,

for any $Z : Set$,

$$case[] : 0 \longrightarrow Z$$

The notation comes from thinking of $0$ as a sum of 0 sets.

Another way of thinking about $case[]$:

hypothetically if we had an element $x : 0$, we could do anything since that's contradictory. Hence we could do anything preposterous

we would even construct a canonical element

$case[](x) : Z$ in any other set (which is clearly impossible).

(*) function set $X, Y \rightsquigarrow [X \to Y]$

equipped with application

$$app : [X \longrightarrow Y] \times X \longrightarrow Y \qquad f, x \rightsquigarrow f(x)$$

and currying (partial application):

for any $Z : Set$,

given $g : Z \times X \longrightarrow Y$,

we have

$$curry(g) : Z \longrightarrow [X \to Y] \qquad z \rightsquigarrow \lambda x.\, g(z, x)$$

# Universal Properties

The equipment of all the above constructions determines the construction uniquely up to isomorphism in Set, since the equipment characterizes the construction by either mapping-in or mapping-out property.

mapping-in:

$$\frac{X \longrightarrow A \times B}{X \longrightarrow A,\ X \to B}$$

$$\frac{X \longrightarrow 1}{\text{1 element}}$$

$$\frac{X \longrightarrow [A \to B]}{X \times A \to B}$$

mapping-out:

$$\frac{left(A) + right(B) \longrightarrow X}{A \to X,\ B \to X}$$

$$\frac{0 \longrightarrow X}{\text{1 element}}$$

Such universal properties can be formulated in arbitrary categories.

They guide us towards finding the correct constructions.

We will exploit them when looking at a model of domains for guarded recursion.

# Finite Approximation

## Streams & ω-chains

For fixed $A : \mathrm{Set}$, we can think of

$$\mathrm{Stream}[A] := \mathbb{N} \to A$$

as a set of infinite sequences of elements in $A$.

Note that we have a consing operation

$$\mathrm{cons} : A \times \mathrm{Stream}[A] \longrightarrow \mathrm{Stream}[A]$$
$$a_0, (a_0, a_1, a_2, \dots) \rightsquigarrow (a_0, a_0, a_1, a_2, \dots)$$

In fact cons is invertible; its inverse consists of two components

$$\mathrm{hd} : \mathrm{Stream}[A] \longrightarrow A$$
$$(a_0, a_1, a_2, \dots) \rightsquigarrow a_0$$

$$\mathrm{tl} : \mathrm{Stream}[A] \longrightarrow \mathrm{Stream}[A]$$
$$(a_0, a_1, a_2, \dots) \rightsquigarrow (a_1, a_2, \dots)$$

Write $a :: s$ for $\mathrm{cons}(a, s)$.

Any stream

$$a_0 :: a_1 :: a_2 :: a_3 :: \cdots$$

can be thought of as built in stages from its finite approximations

$$a_0 :: \bullet \text{ ——— } a_0 :: a_1 :: \bullet \text{ ——— } a_0 :: a_1 :: a_2 :: \bullet \text{ ——— } \cdots$$

where we think of $\bullet$ as an element that's to be refined later.

Let us explicitly define the sets that contain these finite approximations

$$\mathrm{Stream}[A](0) := \{ \mathrm{cons}(a_0, \bullet) \mid a_0 : A \}$$
$$:= \mathrm{cons}(A, \bullet)$$

$$\mathrm{Stream}[A](1) := \mathrm{cons}(A, \mathrm{cons}(A, \bullet))$$

$$\mathrm{Stream}[A](2) := \mathrm{cons}(A, \mathrm{cons}(A, \mathrm{cons}(A, \bullet)))$$

$$\vdots$$

$$\forall n : \mathbb{N}$$
$$\mathrm{Stream}[A](n+1) := \mathrm{cons}(A, \mathrm{Stream}[A](n))$$

Note that in some sense

$$a_0 :: \bullet : \mathrm{Stream}[A](0) \text{ refines (or approximates) } a_0 :: a_1 :: \bullet : \mathrm{Stream}[A](1)$$

To formalize this notion of refinement note that we have a sequence of erasure map that erase tails

$$\mathrm{Stream}[A](0) \xleftarrow{\mathrm{erase}_0} \mathrm{Stream}[A](1) \xleftarrow{\mathrm{erase}_1} \mathrm{Stream}[A](2) \xleftarrow{\mathrm{erase}_2} \cdots$$
$$a_0 :: \bullet \quad \leftsquigarrow \quad a_0 :: a_1 :: \bullet \quad \leftsquigarrow \quad a_0 :: a_1 :: a_2 :: \bullet$$

Note erasure is defined recursively

$$\mathrm{erase}_{n+1}(\mathrm{cons}(a_0, u)) := \mathrm{cons}(a_0, \mathrm{erase}_n(u)).$$

Then $u : \mathrm{Stream}[A](n)$ refines $v : \mathrm{Stream}[A]$

$$:\Leftrightarrow \quad \mathrm{erase}_n(v) = u$$

i.e. $v$ is in the inverse image of the erasure.

We can think of $\mathrm{Stream}[A]$ as a limit built from its finite approximations. To make this precise we abstract the sequence of approximating sets together with the refinement/erasure/projection structure.

**define** An ω-chain $A$ of sets consists of a sequence

$$A(0), A(1), A(2), \dots : \mathrm{Set}$$

with a sequence of maps

$$A(0) \xleftarrow{p_0} A(1) \xleftarrow{p_1} A(2) \xleftarrow{p_2} \cdots : \mathrm{Set}.$$

We call the maps projections (but in general we do not require surjectivity).

For $a : A(n)$, $a' : A(n+1)$ we can define refinement relation

$$a \sqsubseteq a' \quad :\Leftrightarrow \quad p_n a' = a \quad \blacksquare$$

Given an ω-chain $A$, we have a notion of a limiting sequence

$$a_0 : A(0), \quad a_1 : A(1), \quad a_2 : A(2), \dots$$

where $\forall n : \mathbb{N}$
$$p_n a_{n+1} = a_n$$

i.e. $a_0 \sqsubseteq a_1 \sqsubseteq a_2 \sqsubseteq \cdots$

**define** The limiting set of $A$ is

$$A(\omega) := \varprojlim_{n:\omega} A(n) := \{ (a_0 \sqsubseteq a_1 \sqsubseteq a_2 \sqsubseteq \cdots) \text{ limiting sequences of } A \}. \quad \blacksquare$$

Now it is easy to see that

$$\mathrm{Stream}[A](-) \text{ is an } \omega\text{-chain of sets}$$

and $\mathrm{Stream}[A] \simeq \mathrm{Stream}[A](\omega)$

Further examples follow.

**example** [Conatural Numbers]

We wish to have $\mathrm{CoNat} : \mathrm{Set}$ where

$$\mathrm{CoNat} \simeq \{ 0, 1, 2, 3, \dots, \infty \}$$

which we can think of as

$$\{ 0, \mathrm{s}0, \mathrm{ss}0, \mathrm{sss}0, \dots, \mathrm{ssssss}\cdots \}$$

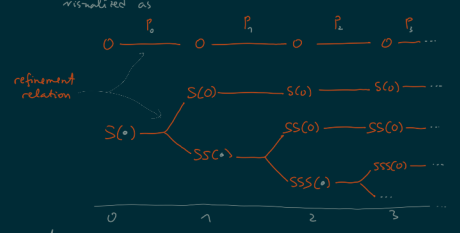
Note $\mathrm{CoNat} \simeq \{0\} + S(\mathrm{CoNat})$

This suggests an ω-chain whose limit is CoNat

$$\mathrm{CoNat}(0) := \{ 0, S(\bullet) \}$$
$$\forall n : \mathbb{N}$$
$$\mathrm{CoNat}(n+1) := \{0\} + S(\mathrm{CoNat}(n))$$

$$\left[\begin{array}{l} p_{n+1} : \mathrm{CoNat}(n+2) \longrightarrow \mathrm{CoNat}(n+1) \\ p_{n+1}(0) := 0 \\ p_{n+1}(S(u)) := S(p_n(u)) \end{array}\right]$$

visualized as

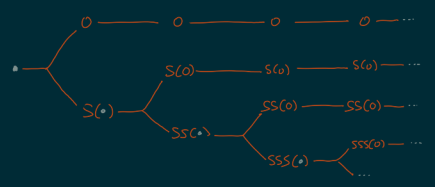



Note that limiting sequences precisely correspond to $0, 1, 2, \dots, \infty$

Also note that we would have shifted the ω-chain starting not with $\mathrm{CoNat}(0) := \{0, S(\bullet)\}$

but with $\mathrm{CoNat}(0) := \{\bullet\}$ a single totally unrefined element.

The picture would change to




but limit would be the same (up to iso in Set) $\blacksquare$

**example** [Cantor Space]

We wish to have $\mathrm{Cantor} : \mathbf{Set}$ s.t.

$$\mathrm{Cantor} \simeq a(\mathrm{Cantor}) + b(\mathrm{Cantor})$$

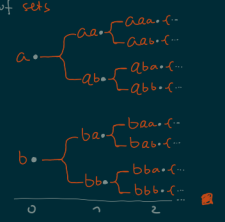
i.e. the space whose elements are infinite choices of a's & b's.

This suggests

$$\mathrm{Cantor}(0) := \{a\bullet,\ b\bullet\}$$

$$\forall n : \mathbb{N}.\quad \mathrm{Cantor}(n+1) := a(\mathrm{Cantor}(n)) + b(\mathrm{Cantor}(n))$$

an $\omega$-chain of sets



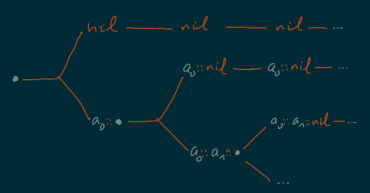

**example** [finite or infinite sequences]

For $A : \mathbf{Set}$, consider $\omega$-chain $S[A]$

$$S[A](0) := \{\bullet\}$$

$$\forall n : \mathbb{N}.\quad S[A](n+1) := \{\mathrm{nil}\} + \mathrm{cons}(A, S[A](n))$$

Then we get both finite sequences & infinite streams as a limit




There are many branches, a new one for a new choice of $a_0, a_1, \ldots$ but it is kinda pointless drawing them all

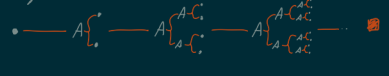
**example** [Infinitely Deep Binary Trees with data in $A$]

For $A : \mathbf{Set}$, $\omega$-chain $BT[A]$

$$BT[A](0) := \{\bullet\}$$

$$\forall n : \mathbb{N}.\quad BT[A](n+1) := \mathrm{branch}(BT[A](n), A, BT[A](n))$$

visualizing




**example** [$\omega$]

Now from the point of view of $\mathbf{Set}$, there is no difference between these singleton sets

$$1 := \{()\} \quad \& \quad \{\infty\}$$

where we think of $\infty$ as

$$\infty = S(\infty)$$

i.e. $\infty = SSSSS\ldots$

Consider the $\omega$-chain of sets, which we will denote $\omega$,

$$\omega(0) := \{\bullet\} \quad (\text{singleton set containing symbol } S\bullet)$$

$$\forall n : \mathbb{N}.\quad \omega(n+1) := S(\omega(n)) \quad (\text{this is still a singleton})$$

Visualizing

$$\bullet \text{ —— } S(\bullet) \text{ —— } SS(\bullet) \text{ —— } SSS(\bullet) \text{ —— } \cdots$$

Then the limit is the set $\{SSSS\ldots\}$ with a single infinite element.

This perspective will be important later.

# $\omega$-Sets

In a domain $D$ given a chain $a_0 \sqsubseteq a_1 \sqsubseteq a_2 \sqsubseteq \cdots : D$ we are able to form $\bigsqcup_n a_n : D$ their limit.

What we did so far is to consider the category $D := \mathbf{Set}$ as a kind of a domain in which analogue of a chain is an $\omega$-chain of sets $A(0) \xleftarrow{p_0} A(1) \xleftarrow{p_1} A(2) \xleftarrow{p_2} \cdots$ & the limit of the chain is $A(\omega) : \mathbf{Set}$

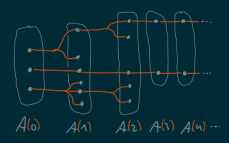
But we saw in the above examples that $\omega$-chain of sets $A(-)$ feels like an algebraic domain — it has just separated its finite approximations into stages that are connected by the projections.

Now we shift perspective and consider $\omega$-chains of sets as the primary objects of study by making a category out of them. This category should be thought of as a category of some sort of domains, or even graded sets. Thus from now on we rename $\omega$-chain of sets $A(-)$ to $\omega$-set $A$. Repeating the definition

**define** An $\omega$-set $A$ is a diagram

$$A(0) \xleftarrow{p_0} A(1) \xleftarrow{p_1} A(2) \xleftarrow{p_2} \cdots : \mathbf{Set}$$

A generic $\omega$-set $A$ can be visualized as a forest




Note that we do not require the projection maps to be surjective, so it may happen that some $a : A(n)$ can't be refined beyond stage $n$. This can be observed in the above picture.

**define** $\omega$-set $A$ is said to be total

$:\Leftrightarrow$ $\forall n : \mathbb{N},\ x : A(n).$
$\exists y : A(n+1).$
$x \sqsubseteq y$
i.e. $py = x.$

This says that

$\forall n : \mathbb{N}$
$A(n) \xleftarrow{p_n} A(n+1)$
is surjective.

Non-total $\omega$-sets will play an important role too.

**define** Given $A, B$ $\omega$-sets, a map $f$ from $A$ to $B$ written $f : A \to B : \omega\text{-}\mathbf{Set}$, or simply $f : A \to B$ (when under is clear) consists of a sequence of maps

$$\begin{array}{ccccc} A(0) & A(1) & A(2) & & \\ \downarrow f_0 & \downarrow f_1 & \downarrow f_2 & & \\ B(0), & B(1), & B(2), & & \ldots \end{array}$$

s.t.

$$\begin{array}{ccccccc} A(0) & \xleftarrow{p_0} & A(1) & \xleftarrow{p_1} & A(2) & \xleftarrow{p_2} & \cdots \\ \downarrow f_0 & = & \downarrow f_1 & = & \downarrow f_2 & = & \\ B(0) & \xleftarrow[p_0]{} & B(1) & \xleftarrow[p_1]{} & B(2) & \xleftarrow[p_2]{} & \cdots \end{array}$$

i.e. $\forall n : \mathbb{N},\ x : A(n+1).$

$$p_n f_{n+1}(x) = f_n(p_n x)$$

We will often drop indices, so above equation becomes

$$p\, f(x) = f(p\, x).$$

Note that for $f : A \to B : \omega\text{-}\mathbf{Set}$,

$$\begin{array}{ccccccc} & A(0) & & A(1) & & A(2) & \\ \text{and} & a_0 & \sqsubseteq & a_1 & \sqsubseteq & a_2 & \sqsubseteq \cdots \end{array}$$

we have

$$\begin{array}{ccccccc} & B(0) & & B(1) & & B(2) & \\ & f(a_0) & \sqsubseteq & f(a_1) & \sqsubseteq & f(a_2) & \sqsubseteq \cdots \end{array}$$

This may look like just monotonicity, but it is much stronger — $f$ is also required to preserve the stages of approximations!

Note that for each $\omega$-set $A$, we have special identity map

$$id_A : A \to A$$

$$\begin{tikzcd} A(0) \arrow[d, "id_{A(0)}"'] & A(1) \arrow[l, "p"'] \arrow[d, "id_{A(1)}"] & \cdots \arrow[l] \\ A(0) & A(1) \arrow[l, "p"] & \cdots \arrow[l] \end{tikzcd}$$

Also given $f : A \to B$, $g : B \to C$ of $\omega$-sets,

we can compose them $gf : A \to C$ by glueing diagrams in the obvious way,

$$\begin{tikzcd} A(0) \arrow[d, "f"'] & A(1) \arrow[l, "p"'] \arrow[d, "f"] & A(2) \arrow[l, "p"'] \arrow[d, "f"] & \cdots \arrow[l, "p"'] \\ B(0) \arrow[d, "g"'] & B(1) \arrow[l, "p"'] \arrow[d, "g"] & B(2) \arrow[l, "p"'] \arrow[d, "g"] & \cdots \arrow[l, "p"'] \\ C(0) & C(1) \arrow[l, "p"'] & C(2) \arrow[l, "p"'] & \cdots \arrow[l, "p"'] \end{tikzcd}$$

This defines a category

**define** $\omega$-Set is a category of $\omega$-sets & their maps ■

We will explore various constructions in $\omega$-Set.

## Terminal

Recall $\omega : \omega$-Set,

$$\omega := \underset{0}{\bullet} \text{———} \underset{1}{S(\bullet)} \text{———} \underset{2}{SS(\bullet)} \text{———} \underset{3}{SSS(\bullet)} \text{———} \cdots$$

whose components are all singleton sets. This is the terminal object in $\omega$-Set,

$\forall A : \omega$-Set.

$\exists$ unique map

$$! : A \to \omega$$

since

$$\begin{tikzcd} A(0) \arrow[d, "!"'] & A(1) \arrow[l, "p"'] \arrow[d, "!"] & A(2) \arrow[l, "p"'] \arrow[d, "!"] & \cdots \arrow[l, "p"'] \\ \underset{\cong 1}{\{\bullet\}} & \underset{\cong 1}{\{S\bullet\}} \arrow[l, "p"'] & \underset{\cong 1}{\{SS\bullet\}} \arrow[l, "p"'] & \cdots \arrow[l, "p"'] \end{tikzcd}$$

Note the a map

$$x : \omega \to A : \omega\text{-Set}$$

corresponds to a limiting sequence

$$\overset{A(0)}{x_0} \sqsubseteq \overset{A(1)}{x_1} \sqsubseteq \overset{A(2)}{x_2} \sqsubseteq \cdots$$

Also note, given $f : A \to B : \omega$-Set,

$$fx : \omega \to B : \omega\text{-Set}$$

corresponds to

$$\overset{B(0)}{f(a_0)} \sqsubseteq \overset{B(1)}{f(a_1)} \sqsubseteq \overset{B(2)}{f(a_2)} \sqsubseteq \cdots$$

This gives us a functor

$$(-)(\omega) : \omega\text{-Set} \to \text{Set}$$

$$A \rightsquigarrow \{\text{limiting sequence of } A\}$$

that's representable by $\omega$ i.e.

$\forall A : \omega$-Set.

$$A(\omega) \simeq \omega \to A \quad \text{as sets}$$

We also have a functor in the other direction,

Given $X : \text{Set}$, we have the constant $\omega$-set

$$\text{const}[X] := X \xleftarrow{p := id_X} X \xleftarrow{p := id_X} X \leftarrow \cdots$$

This defines a functor

$$\text{const} : \text{Set} \to \omega\text{-Set}.$$

It is easy to see that for $X : \text{Set}$, $A : \omega$-Set

a map $f : \text{const}(X) \to A : \omega$-Set

just picks $X$-many limiting sequences in $A$ i.e.

$$f : X \to A(\omega) : \text{Set}.$$

This makes an adjunction

$$\text{Set} \underset{(-)(\omega)}{\overset{\text{const}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\perp}}}} \omega\text{-Set}$$

In fact Set embeds fully-faithfully into $\omega$-Set via const,

i.e. constant $\omega$-sets & $\omega$-maps between them are just sets & set functions.

**Convention:** When writing $x : A$

we mean $x : A(\omega)$.

## Subterminals & Representables

**define** $A$ subset $X$ of $A : \omega$-Set, written $X \subseteq A$,

consists of a sequence of subsets

$$X(0) \subseteq A(0), \quad X(1) \subseteq A(1), \ \ldots$$

where $\forall n : \mathbb{N}$, $x : A(n+1)$.

if $x : X(n+1)$ then $px : X(n)$,

i.e. $X$ is closed under projection,

so $X$ also defines an $\omega$-set ■

**example** For $\omega$, it has subsets

$$\emptyset \subseteq \omega \quad (\text{the initial object in } \omega\text{-Set})$$

$$\& \quad \omega \subseteq \omega,$$

but also for each $n : \mathbb{N}$, we have

$$[n] \subseteq \omega$$

uniquely determined by

$$S^n(\bullet) : [n](n)$$

& for $m > n$,

$$[n](m) = \emptyset$$

These are all the subsets, & they form a chain

$$\emptyset \subseteq [0] \subseteq [1] \subseteq [2] \subseteq \cdots \quad \subseteq \omega$$

They are called sub-terminals.

Note

$$\begin{array}{lccccccccc} & 0 & & 1 & & 2 & & 3 & & 4 \\ \hline [0] \simeq & \bullet & — & \emptyset & — & \emptyset & — & \emptyset & — & \emptyset & — \cdots \\ [1] \simeq & \bullet & — & \bullet & — & \emptyset & — & \emptyset & — & \emptyset & — \cdots \\ [2] \simeq & \bullet & — & \bullet & — & \bullet & — & \emptyset & — & \emptyset & — \cdots \\ \vdots & & & & & & & & & & \end{array}$$

are not total. ■

Note that

$$x : [n] \to A$$

corresponds to a finite chain in $A$ of size $n+1$

$$\overset{A(0)}{x_0} \sqsubseteq \overset{A(1)}{x_1} \sqsubseteq \cdots \sqsubseteq \overset{A(n)}{x_n}$$

but these are determined by the last element via projections,

so

$$A(n) \simeq [n] \to A$$

The $\omega$-sets $[0], [1], [2], \ldots$

are called representables

## Product & Coproduct

Given $A, B : \omega$-Set, then

$$[A \times B](n) := A(n) \times B(n)$$

$$[A + B](n) := A(n) + B(n)$$

with the obvious projection maps &

$$\pi_A : A \times B \to A$$
$$\pi_B : A \times B \to B$$
$$i_A : A \to A + B$$
$$i_B : B \to A + B$$

give cartesian product & coproduct in $\omega$-Set.

# Arrow Object & Finite Approximations of Maps

Is $\omega$-Set Cartesian Closed? i.e. for $A, B : \omega\text{-Set}$, do we have an $\omega$-set

$$[A \to B] \lhd$$

it somehow encodes information about functions?

Whatever $[A \to B]$ may be, we would hope that

$$f : [A \to B] \quad \left(\text{i.e. by our convention } f : [A \to B](\omega)\right)$$

would correspond to usual maps

$$f : A \to B : \omega\text{-Set}.$$

In which case

$$[A \to B](0), [A \to B](1), \ldots$$

should contain some sort of finite approximations to $f$...

**define** Let $X, Y : \omega\text{-Set}$, an $n$-map from $X$ to $Y$ consists of a finite sequence of $n+1$ functions $(f_0, f_1, \ldots, f_n)$ where

$$\begin{tikzcd} X_0 \arrow[d, "f_0"'] & X_1 \arrow[l, "p"] \arrow[d, "f_1"'] \arrow[r, phantom, "="] & \cdots \arrow[l] & X_n \arrow[l] \arrow[d, "f_n"] \\ Y_0 & Y_1 \arrow[l, "p"] \arrow[r, phantom, "="] & \cdots \arrow[l] & Y_n \arrow[l] \end{tikzcd}$$

Write $X \xrightarrow[n]{} Y$ for the set of $n$-maps. ■

Note that we have a canonical

$$\mathrm{id}_{X,n} : X \xrightarrow[n]{} X$$

& given $f : X \xrightarrow[n]{} Y$, $g : Y \xrightarrow[n]{} Z$, we can make the obvious composite

$$gf : X \xrightarrow[n]{} Y$$

In fact for each $n$, we have a category whose objects are $\omega$-sets, but whose morphisms are $n$-maps.

Given $f : X \xrightarrow[n+1]{} Y$, i.e. $f = (f_0, f_1, \ldots, f_n, f_{n+1})$, we can forget the last component and define

$$Pf : X \xrightarrow[n]{} Y$$

$$Pf := (f_0, f_1, \ldots, f_n)$$

This defines our $\omega$-Set

$$[X \to Y] \quad (\text{often also written } Y^X)$$

$$[X \to Y](n) := X \xrightarrow[n]{} Y$$

Given $f := (f_0, \ldots, f_n) : X \xrightarrow[n]{} Y$, $x : X(n)$, we get $f(x) := f_n(x) : Y_n$.

In fact consider the chain determined by $x$

$$x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \ldots \sqsubseteq x_n := x$$

then applying components of $f$ we get

$$f_0(x_0) \sqsubseteq f_1(x_1) \sqsubseteq f_2(x_2) \sqsubseteq \cdots \sqsubseteq f_n(x_n) = f(x),$$

i.e. the application $f(x)$ respects projection

$$P(f(x)) = P(f)(P(x))$$

This means that the application gives rise to a map

$$\mathrm{app} : [X \to Y] \times X \to Y : \omega\text{-Set}$$

$$f : X \xrightarrow[n]{} Y, \quad x : X(n) \rightsquigarrow f(x) : Y(n)$$

We also have the obvious currying map:

given $f : Z \times X \to Y : \omega\text{-Set}$,

then $\mathrm{curr}(f) : Z \to [X \to Y] : \omega\text{-Set}$

$$z : Z(n) \rightsquigarrow \begin{tikzcd} X(0) \arrow[d, "f(z,-)"'] & X(1) \arrow[l] \arrow[d, "f(z_1,-)"'] & \cdots \arrow[l] & X(n) \arrow[l] \arrow[d, "f(z_n,-)"'] \\ Y(0) & Y(1) \arrow[l] & \cdots \arrow[l] & Y(n) \arrow[l] \end{tikzcd}$$

where $z_0 \sqsubseteq z_1 \sqsubseteq \cdots \sqsubseteq z_n = z$

Hence we get an iso

$$Z \to [X \to Y] \simeq Z \times X \to Y$$

making $[X \to Y]$ into the arrow object in $\omega$-Set.

Note that $[n] \times X : \omega\text{-Set}$ is $\lhd$.

$$[n] \times X(\ell) \simeq X(\ell) \quad \text{for } \ell < n,$$
$$[n] \times X(\ell) \simeq 0 \quad \text{for } \ell \geq n.$$

We call $[n] \times X$ the $n$-truncation of $X$.

It is easy to see that it classifies $n$-maps out of $X$

**prop** $$\frac{X \xrightarrow[n]{} Y}{[n] \times X \to Y}$$

With $0, 1, +, \times, [- \to -]$ we can now interpret simply typed $\lambda$-calculus.

A closed term $\vdash M : A$ is interpreted as a limiting sequence $[\![M]\!] : [\![A]\!]$,

& an open term $x_1 : A_1, \ldots, x_n : A_n \vdash M : A$ is interpreted as a map

$$[\![M]\!] : [\![A_1]\!] \times \ldots \times [\![A_n]\!] \to [\![A]\!]$$

We already saw that composition works incrementally on approximations of maps, but now we can define it more compactly

$$\lambda(g, f).\, \lambda x.\, g(f(x)) : [B \to C] \times [A \to B] \to [A \to C] : \omega\text{-Set}$$

# Later

Recall that for fixed $A : \mathrm{Set}$, we defined

$$\mathrm{Stream}[A] : \omega\text{-Set}$$

by recursion

$$\mathrm{Stream}[A](0) := \mathrm{cons}(A, \bullet)$$

$$\forall n : \mathbb{N}.$$

$$\mathrm{Stream}[A](n+1) := \mathrm{cons}(A, \mathrm{Stream}[A](n))$$

On the level of $\omega$-Sets this almost looks like we are trying to solve

$$\mathrm{Stream}[A] \simeq \mathrm{cons}(A, \mathrm{Stream}[A]) : \omega\text{-Set}$$

(i.e. find $X : \omega\text{-Set}$ s.t. $X \simeq A \times X : \omega\text{-Set}$ where $A$ is a constant $\omega$-Set)

but such an equation would imply

$$\forall n : \mathbb{N}$$

$$\mathrm{Stream}[A](n) \simeq \mathrm{cons}(A, \mathrm{Stream}[A](n)),$$

which is not quite what we want.

The correct stream equation suggests a new construction, that will allow us the shift in indices.

**define** For $X : \omega\text{-Set}$, let

$$\blacktriangleright X : \omega\text{-Set}$$

$$\blacktriangleright X(0) := \{\bullet\} \quad (\text{terminal set})$$

$$\forall n : \mathbb{N}$$

$$\blacktriangleright X(n+1) := X(n)$$

with the obvious shifted projections of original $X$

$$\blacktriangleright X \simeq \underset{0}{\bullet} \xleftarrow{!} \underset{1}{X(0)} \xleftarrow{p} \underset{2}{X(1)} \xleftarrow{p} \underset{\cdots}{\cdots} \quad ■$$

Then the correct recursive $\omega$-set equation becomes

$$\mathrm{Stream}[A] \simeq \mathrm{cons}(A, \blacktriangleright \mathrm{Stream}[A])$$

& now it makes sense even for $A : \omega\text{-Set}$ and not just a set:

$$\mathrm{Stream}[A](0) := \mathrm{cons}(A(0), \bullet)$$

$$\forall n : \mathbb{N}.$$

$$\mathrm{Stream}[A](n+1) := \mathrm{cons}(A(n+1), \mathrm{Stream}[A](n))$$

Note how we could have used

$$\mathrm{Stream}[A] \simeq \blacktriangleright \mathrm{cons}(A, \mathrm{Stream}[A])$$

instead. It generates the same limiting sequences, but gives slightly different $\omega$-set.

$$\mathrm{Stream}[A](0) := \mathrm{cons}(A(0), \bullet)$$

$$\forall n : \mathbb{N}.$$

$$\mathrm{Stream}[A](n+1) := \mathrm{cons}(A(n), \mathrm{Stream}[A](n))$$

— note how both head & the tail are delayed.

When defining e.g.

$$f(x : X,\ y : \blacktriangleright Y) :=$$

$x$ is available now,

$y$ is not yet fully available,

but will be available in the next time step

i.e. $y$ is available later.

That's why $\blacktriangleright$ is pronounced "later"

It is clear that $\blacktriangleright : \omega\text{-}Set \to \omega\text{-}Set$ is a functor.

It is equipped with a map

$$p : X \longrightarrow \blacktriangleright X$$

whose components are the projections of $X$

$$\begin{tikzcd} X \arrow[d, "p"'] \\ \blacktriangleright X \end{tikzcd} := \begin{tikzcd} X(0) \arrow[d, "!"] & X(1) \arrow[l, "p_0"'] \arrow[d, "!"] & X(2) \arrow[l, "p_1"'] \arrow[d, "!"] & \cdots \arrow[l, "p_2"'] \\ \bullet_0 & X(0) \arrow[l, "!"'] & X(1) \arrow[l, "p_0"'] & \cdots \arrow[l, "p_1"'] \end{tikzcd}$$

## Fixpoints of Maps

Let us unpack what it means to have a map

$$f : \blacktriangleright A \longrightarrow B : \omega\text{-}Set.$$

Then

$$f := \begin{tikzcd} \bullet \arrow[d, "f_0"'] & A(0) \arrow[l, "!"'] \arrow[d, "f_1"] & A(1) \arrow[l, "p"'] \arrow[d, "f_2"] & \cdots \arrow[l, "p"'] \\ B(0) & B(1) \arrow[l, "p"] & B(2) \arrow[l, "p"] & \cdots \arrow[l, "p"] \end{tikzcd}$$

Note that the 0-th component just picks out an element of $B(0)$,

otherwise given $x : A(n)$, then $f(x) : B(n+1)$, i.e.

$f$ is able to compute output that's more refined than its input.

It strictly increases the information content.

For $B := A$, we have $f : \blacktriangleright A \longrightarrow A$

which gives us

$$a_0 : A(0) \xrightarrow{f_1} A(1) \xrightarrow{f_2} A(2) \longrightarrow \cdots$$

And because $\forall a$ $\quad f(p\,a) = p\,f(a)$

we may form a limiting sequence in $A$

$$a_0 \sqsubseteq f_1(a_0) \sqsubseteq f_2 f_1(a_0) \sqsubseteq f_3 f_2 f_1(a_0) \sqsubseteq \cdots$$

call it $fix[f] : A$.

Note $\quad p\ fix[f] : \blacktriangleright A$, & from

$$\begin{tikzcd} \bullet \arrow[d, "a_0"'] & A(0) \arrow[l, "!"'] \arrow[d, "f_1"] & A(1) \arrow[l, "p"'] \arrow[d, "f_2"] & \cdots \arrow[l, "p"'] \\ A(0) & A(1) \arrow[l, "p"] & A(2) \arrow[l, "p"] & \cdots \arrow[l, "p"] \end{tikzcd}$$

it is clear that

$$f(p\ fix[f]) = fix[f].$$

Also given $x : A$ s.t. $f(p\,x) = x$,

we immediately get that

$$x_0 = a_0, \ \& \ \forall n : \mathbb{N} \quad \text{if } x_n = fix[f]_n \quad \text{then } x_{n+1} = f(x_n) = f(fix[f]_n) = fix[f]_n,$$

so $\quad x = fix[f] : A.$

So in some sense every $f : \blacktriangleright A \to A$

has a unique delayed fixpoint $fix[f] : A$.

Note that this fixpoint is computed incrementally,

hence $fix$ lifts to a map

$$fix : [\blacktriangleright A \to A] \longrightarrow A : \omega\text{-}Set$$

**example** Consider streams over $\mathbb{N} : Set$ (thought of as constant $\omega$-set)

$$St[\mathbb{N}] := \mathbb{N} :: \blacktriangleright St[\mathbb{N}]$$

This gives us maps

$$_ :: _ : \mathbb{N} \times \blacktriangleright St[\mathbb{N}] \longrightarrow St[\mathbb{N}]$$

$$hd : St[\mathbb{N}] \longrightarrow \mathbb{N}$$

$$tl : St[\mathbb{N}] \longrightarrow \blacktriangleright St[\mathbb{N}].$$

(*) Consider the constant 0 stream

$$zeroes := 0 :: 0 :: 0 :: \cdots : St[\mathbb{N}],$$

it is uniquely characterized as

$$zeroes = 0 :: p(zeroes),$$

i.e. $\quad zeroes = fix[\lambda\ \underline{zeroes} : \blacktriangleright St[\mathbb{N}].\ 0 :: \underline{zeroes}]$

(*) Consider

$$nats := 0 :: 1 :: 2 :: 3 :: \cdots St[\mathbb{N}]$$

It can be defined as

$$nats := from(0)$$

where $\quad from : \mathbb{N} \longrightarrow St[\mathbb{N}]$

$$from(n) := n :: (n+1) :: (n+2) :: \cdots$$

To define from note,

$$from(n) = n :: p(from(n+1))$$

so

$$from = fix\Big[\overbrace{\lambda\ \underline{from} : \blacktriangleright[\mathbb{N} \to St[\mathbb{N}]].\ \lambda n : \mathbb{N}.\ \underline{from}(\ p(n)\ )}^{\text{This has type } \blacktriangleright[\mathbb{N} \to St[\mathbb{N}]] \to [\mathbb{N} \to St[\mathbb{N}]]}\Big]$$

where we need a new kind of delayed function application

$$-(\!|-|\!) : \blacktriangleright[A \to B] \times \blacktriangleright A \longrightarrow \blacktriangleright B$$

which is defined on finite approximations in the obvious way.

Note that the function inside the fix has type

$\blacksquare$

# Enriched Functors

Now we try to make sense out of recursive type equations on $\omega$-Set.

For that we need to talk about functors. We have the classical notion of functor

$$F : \omega\text{-Set} \to \omega\text{-Set}$$

that maps any $X : \omega\text{-Set}$ to $F[X] : \omega\text{-Set}$, and transforms

$$f : X \to Y : \omega\text{-Set} \rightsquigarrow F[f] : F[X] \to F[Y] : \omega\text{-Set}$$

while respecting identity & composition.

But such a notion of a functor is not sufficient for our purpose.

In the spirit of the idea of finite approximations,

we need a functor that can transform

finite n-approximations of maps

into finite n-approximations of maps

in such a way that it respects refinements of approximations.

More precisely

**define** Enriched Functor $F : \omega\text{-Set} \to \omega\text{-Set}$

consists of

(*) a map $X : \omega\text{-Set} \rightsquigarrow F[X] : \omega\text{-Set}$

(**) For all $n : \mathbb{N}$, $X, Y : \omega\text{-Set}$, a map

$$F_n : (X \xrightarrow[n]{} Y) \longrightarrow (F[X] \xrightarrow[n]{} F[Y]) : \mathrm{Set}$$

where

(id) $\forall n : \mathbb{N}$, $X : \omega\text{-Set}$,

$$F_n(\mathrm{id}_{X_n}) = \mathrm{id}_{F[X]_n} : X \xrightarrow[n]{} Y$$

(compose) $\forall n : \mathbb{N}$, $f : X \xrightarrow[n]{} Y$, $g : Y \xrightarrow[n]{} Z$,

$$F_n(g \circ f) = F_n(g) \circ F_n(f) : X \xrightarrow[n]{} Z$$

(refine) $\forall n : \mathbb{N}$, $f : X \xrightarrow[n+1]{} Y$,

$$p\, F_{n+1}(f) = F_n(p f) : X \xrightarrow[n]{} Y \quad \blacksquare$$

Note that for enriched $F : \omega\text{-Set} \to \omega\text{-Set}$, the data above amounts to a map

$$F : [X \to Y] \longrightarrow [F[X] \to F[Y]] : \omega\text{-Set}$$

So we also get a transform on global maps

$$f : X \to Y : \omega\text{-Set} \rightsquigarrow F[f] : \omega\text{-Set}$$

that satisfies classical functoriality.

We can analogously define enriched

$F : \omega\text{-Set}^{op} \to \omega\text{-Set}$ (contravariant functors)

$B : \omega\text{-Set} \times \omega\text{-Set} \to \omega\text{-Set}$ (bifunctors)

$T : \omega\text{-Set}^{op} \times \omega\text{-Set} \to \omega\text{-Set}$ (profunctors)

equipped with

$$F_n : (X \xrightarrow[n]{} Y) \longrightarrow (F[Y] \xrightarrow[n]{} F[X]) : \mathrm{Set}$$

$$B_n : (X \xrightarrow[n]{} Y) \times (X' \xrightarrow[n]{} Y') \longrightarrow (B[X, X'] \xrightarrow[n]{} B[Y, Y']) : \mathrm{Set}$$

$$T_n : (X \xrightarrow[n]{} Y) \times (X' \xrightarrow[n]{} Y') \longrightarrow (T[Y, X'] \xrightarrow[n]{} T[X, Y']) : \mathrm{Set}$$

satisfying the obvious axioms.

**example** Fix $A : \omega\text{-Set}$, then functors

$$A + _ : \omega\text{-Set} \longrightarrow \omega\text{-Set}$$

$$A \times _ : \omega\text{-Set} \longrightarrow \omega\text{-Set}$$

$$[A \to _] : \omega\text{-Set} \longrightarrow \omega\text{-Set}$$

$$[_ \to A] : \omega\text{-Set}^{op} \longrightarrow \omega\text{-Set}$$

can be lifted to enriched functors by equipping them with obvious maps

$$[X \xrightarrow[n]{} Y] \longrightarrow [A + X \xrightarrow[n]{} A + Y]$$

$$[X \xrightarrow[n]{} Y] \longrightarrow [A \times X \xrightarrow[n]{} A \times Y]$$

$$[X \xrightarrow[n]{} Y] \longrightarrow [[A \to X] \xrightarrow[n]{} [A \to Y]]$$

$$[X \xrightarrow[n]{} Y] \longrightarrow [[Y \to A] \xrightarrow[n]{} [X \to A]]$$

For example for $[_ \to A]$ (which might at first thought be seen as a huge headache)

let $f = (f_0, \dots, f_n) : X \xrightarrow[n]{} Y$, i.e.

$$\begin{tikzcd}
X(0) \arrow[d, "f_0"'] & X(1) \arrow[l, "p"'] \arrow[d, "f_1"'] & \cdots \arrow[l, "p"'] & X(n) \arrow[l, "p"'] \arrow[d, "f_n"'] \\
Y(0) & Y(1) \arrow[l, "p"'] & \cdots \arrow[l, "p"'] & Y(n) \arrow[l, "p"']
\end{tikzcd}$$

let $g : [Y \to A](n)$ which is just $g = (g_0, \dots, g_n) : Y \xrightarrow[n]{} A$ i.e.

$$\begin{tikzcd}
Y(0) \arrow[d, "g_0"'] & Y(1) \arrow[l, "p"'] \arrow[d, "g_1"'] & \cdots \arrow[l, "p"'] & Y(n) \arrow[l, "p"'] \arrow[d, "g_n"'] \\
A(0) & A(1) \arrow[l, "p"'] & \cdots \arrow[l, "p"'] & A(n) \arrow[l, "p"']
\end{tikzcd}$$

we simply compose the above diagrams to get an element

$$gf : [X \to A](n) \quad \blacksquare$$

**example** Fix $A, B : \omega\text{-Set}$, then we have

enriched bifunctors

$$- \times -, - + - : \omega\text{-Set} \times \omega\text{-Set} \to \omega\text{-Set}$$

enriched profunctor

$$[- \to -] : \omega\text{-Set}^{op} \times \omega\text{-Set} \to \omega\text{-Set} \quad \blacksquare$$

Recall that maps $f : \blacktriangleright X \to X$ have a unique fix-point $\mathrm{fix}[f] : X$

Employing wishful thinking, we would like to have an analogues notion for a functor on $\omega\text{-}Set$

$$G : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$$

that would give us a construction of $\mathrm{fix}[G] : \omega\text{-}Set$ s.t.

$$G[\mathrm{fix}[G]] \simeq \mathrm{fix}[G].$$

Since enriched $F : \omega\text{-}Set \to \omega\text{-}Set$ gave us

$$F : [X \to Y] \to [F[X] \to F[Y]] : \omega\text{-}Set$$

we might try to define $G : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$ in such a way that we get

$$G : \blacktriangleright [X \to Y] \to [F[X] \to F[Y]] : \omega\text{-}Set$$

**define** Enriched Functor $G : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$

consists of

(*) a map $X : \omega\text{-}Set \rightsquigarrow G[X] : \omega\text{-}Set$

(*) For all $n : \mathbb{N}$, $X, Y : \omega\text{-}Set$ a map

$$G^{X,Y}_{n+1} : (X \xrightarrow[n]{} Y) \to (G[X] \xrightarrow[n+1]{} G[Y]) : Set$$

This is new

(*) For all $X, Y : \omega\text{-}Set$,

$$G^{X,Y}_0 : G[X] \xrightarrow[0]{} G[Y] : Set$$

where

(id) $\forall n : \mathbb{N}$, $X : \omega\text{-}Set$.

$$G_{n+1}(\mathrm{id}_{X,n}) = \mathrm{id}_{G[X]_{n+1}} : G[X] \xrightarrow[n+1]{} G[X]$$

$$G^{X,X}_0 = \mathrm{id}_{G[X]_0} : G[X] \xrightarrow[0]{} G[X]$$

(compose) $\forall n : \mathbb{N}$, $f : X \xrightarrow[n]{} Y$, $g : Y \xrightarrow[n]{} Z$.

$$G_{n+1}(g \circ f) = G_{n+1}(g) \circ G_{n+1}(f) : G[X] \xrightarrow[n+1]{} G[Z]$$

$$G^{X,Z}_0 = G^{Y,Z}_0 \circ G^{X,Y}_0 : G[X] \xrightarrow[0]{} G[Z]$$ ← This implies that all $G^{X,Y}_0$ are invertible

(refine) $\forall n : \mathbb{N}$, $f : X \xrightarrow[n+1]{} Y$.

$$p\, G_{n+1}(f) = G_{n+1}(p f) : X \xrightarrow[n+1]{} Y$$

in particular $p\, G_1(f : X \xrightarrow[1]{} Y) = G_0 : X \xrightarrow[0]{} Y$ ■

So for any $X, Y : \omega\text{-}Set$,

$$G[X](0) \simeq G[Y](0)$$

hence as a special case, for any $X : \omega\text{-}Set$,

$$G[X](0) \simeq G[\omega](0)$$

& for any $f : X \to Y : \omega\text{-}Set$ the 0-th component of $G(p(f))$ must be $G^{X,Y}_0 : X \xrightarrow[0]{} Y$

We can generalize this to

$F : \blacktriangleright \omega\text{-}Set^{op} \to \omega\text{-}Set$,

$B : \blacktriangleright \omega\text{-}Set \times \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$

$T : \blacktriangleright \omega\text{-}Set^{op} \times \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$

in the obvious way.

**example** From

$$[X \xrightarrow[n]{} Y] \xrightarrow{F} [F[X] \xrightarrow[n]{} F[Y]] \xrightarrow{G} [GF[X] \xrightarrow[n+1]{} GF[Y]]$$

&

$$[X \xrightarrow[n]{} Y] \xrightarrow{G} [G[X] \xrightarrow[n+1]{} G[Y]] \xrightarrow{F} [FG[X] \xrightarrow[n+1]{} FG[Y]]$$

we can see that composition of enriched $F : \omega\text{-}Set \to \omega\text{-}Set$ with enriched $G : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$ gives rise to enriched

$$GF,\ FG : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set.$$ ■

**example** $\blacktriangleright : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$ is enriched.

This amounts to giving a map

$$\blacktriangleright [X \to Y] \to [\blacktriangleright X \to \blacktriangleright Y] : \omega\text{-}Set$$

i.e. for each $n : \mathbb{N}$,

$$\blacktriangleright_n : [X \xrightarrow[n]{} Y] \to [\blacktriangleright X \xrightarrow[n+1]{} \blacktriangleright Y]$$

$$\begin{tikzcd}
X(0) \arrow[d] & X(1) \arrow[l, "p"'] \arrow[d] & \cdots \arrow[l, "p"'] & X(n) \arrow[l, "p"'] \arrow[d] \\
Y(0) & Y(1) \arrow[l, "p"] & \cdots \arrow[l, "p"] & Y(n) \arrow[l, "p"]
\end{tikzcd}
\rightsquigarrow
\begin{tikzcd}
\blacktriangleright X(0) \arrow[d] & \blacktriangleright X(1) \arrow[l] \arrow[d] & \blacktriangleright X(2) \arrow[l] \arrow[d] & \cdots \arrow[l] & \blacktriangleright X(n+1) \arrow[l] \arrow[d] \\
\blacktriangleright Y(0) & \blacktriangleright Y(1) \arrow[l] & \blacktriangleright Y(2) \arrow[l] & \cdots \arrow[l] & \blacktriangleright Y(n+1) \arrow[l]
\end{tikzcd}$$

by definition this is just

$$\begin{tikzcd}
1 \arrow[d, "!"] & X(0) \arrow[l, "!"'] \arrow[d] & X(1) \arrow[l, "p"'] \arrow[d] & \cdots \arrow[l, "p"'] & X(n) \arrow[l, "p"'] \arrow[d] \\
1 & Y(0) \arrow[l, "!"] & Y(1) \arrow[l, "p"] & \cdots \arrow[l, "p"] & Y(n) \arrow[l, "p"]
\end{tikzcd}$$ ■

**example** Combining the two previous examples, we get enriched covariant functors $\blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$

$$\lambda X.\ \blacktriangleright X \quad (\text{Thinking about } \omega,\ \lambda X.\ \mathrm{succ}(\blacktriangleright X))$$

$$\lambda X.\ \blacktriangleright X + \blacktriangleright X \quad (\text{Thinking about Cantor Space})$$

$$\lambda X.\ A \times \blacktriangleright X \quad (\text{Thinking about streams},\ \lambda X.\ \mathrm{cons}(\overset{\text{head}}{A}, \overset{\text{tail}}{\blacktriangleright X}))$$

$$\lambda X.\ 1 + \blacktriangleright X \quad (\text{Thinking about conaturals},\ \lambda X.\ \{0\} + \mathrm{succ}(\blacktriangleright X))$$

$0, 1, 2, \ldots, \infty$

$$\lambda X.\ \blacktriangleright X \times A \times \blacktriangleright X \quad (\text{Thinking about infinitely deep binary trees},\ \lambda X.\ \mathrm{branch}(\blacktriangleright X, A, \blacktriangleright X))$$

$$\simeq \lambda X.\ A \times \blacktriangleright [X \times X]$$

$$\lambda X.\ 1 + A \times \blacktriangleright X \quad (\text{Thinking about finite or infinite streams}\ \lambda X.\ \{\mathrm{nil}\} + \mathrm{cons}(A, \blacktriangleright X))$$

enriched contravariant functors $\blacktriangleright \omega\text{-}Set^{op} \to \omega\text{-}Set$

$$\lambda X.\ [\blacktriangleright X \to A]$$

$$\lambda X.\ [\blacktriangleright X \to [A \to A] \to A] \quad (\text{Thinking about Turing fix-point combinator})$$

Note how in each example the input variable is guarded by $\blacktriangleright$ ■

## Functor Fixpoints

Let $G : \blacktriangleright \omega\text{-}Set \to \omega\text{-}Set$ be enriched.

Note that we have the unique map

$$! : \omega \leftarrow G[\omega] \qquad (\text{since } \omega \text{ is terminal})$$

hence we have

$$p(!) : \blacktriangleright [\omega \leftarrow G[\omega]]$$

So by applying $G$, we get

$$G(p(!)) : G[\omega] \leftarrow G^2[\omega].$$

We can iterate this to get an $\omega$-chain of $\omega$-sets

$$G[\omega] \xleftarrow{G(p(!))} G^2[\omega] \xleftarrow{G(p(G(p(!))))} G^3[\omega] \leftarrow \cdots : \omega\text{-}Set$$

More generally

**define** Given an $\omega$-chain of $\omega$-sets

$$X_0 \xleftarrow{p_0} X_1 \xleftarrow{p_1} X_2 \xleftarrow{p_2} \cdots : \omega\text{-}Set,$$

we can define its limiting $\omega$-set as

$$\varprojlim_i X_i : \omega\text{-}Set$$

$$\left[\varprojlim_i X_i\right](n) := \left\{\text{limiting segments of the } \omega\text{-set } X_0(n) \xleftarrow{p_0} X_1(n) \xleftarrow{p_1} X_2(n) \xleftarrow{p_2} \cdots\right\}$$

& for $n : \mathbb{N}$, we have a map of $\omega$-sets

$$\begin{tikzcd}
X_0(n) & X_1(n) \arrow[l, "p_0"'] & X_2(n) \arrow[l, "p_1"'] & \cdots \arrow[l, "p_2"'] \\
\cdot \arrow[u] & \cdot \arrow[u] & &
\end{tikzcd}$$

This defines an $\omega\text{-}Set$. ■

Now we hope that the limit of $G[\omega] \xleftarrow{G(p(!))} G^2[\omega] \xleftarrow{G(p(G(p(!))))} G^3[\omega] \leftarrow \cdots : \omega\text{-}Set$ will be the fixpoint of $G$ i.e.

$$G\left[\varprojlim_i G^i[\omega]\right] \simeq \varprojlim_i G^i[\omega]$$

But using the generic construction, that was defined above for $\varprojlim_i G^i[\omega]$ violates the spirit of finiteness of approximation, since at each stage it seems we are doing infinite amount of work in constructing the limiting set of $X_0(n) \xleftarrow{p_0} X_1(n) \xleftarrow{p_1} X_2(n) \xleftarrow{p_2} \cdots$

We shall see that because of properties of $G$, the $\omega$-chain

$$G[\omega] \xleftarrow{G(p(!))} G^2[\omega] \xleftarrow{G(p(G(p(!))))} G^3[\omega] \leftarrow \cdots : \omega\text{-}Set$$

is especially simple.

**example** Again we illustrate on streams

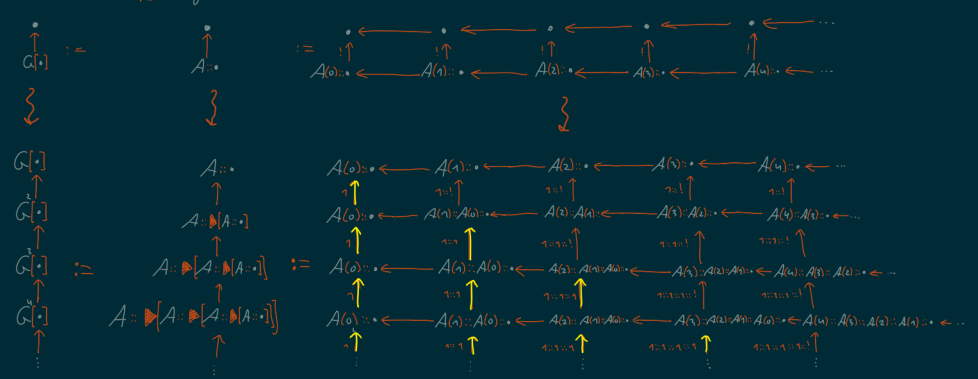
Consider
$$G[X] := A :: \blacktriangleright X,$$
then $G : \blacktriangleright\omega\text{-Set} \to \omega\text{-Set}$ is enriched:
$$G : \blacktriangleright[X \to Y] \longrightarrow [A :: \blacktriangleright X \to A :: \blacktriangleright Y]$$

i.e. $G_0 : A(0) :: \bullet \longrightarrow A(0) :: \bullet$ is the identity,

$\forall n : \mathbb{N}$:
$$G_n : [X \xrightarrow[n]{} Y] \longrightarrow [A :: \blacktriangleright X \xrightarrow[n+1]{} A :: \blacktriangleright Y]$$

$$\begin{tikzcd} X(0) \arrow[d, "f_0"'] & X(1) \arrow[l] \arrow[d, "f_1"'] & X(2) \arrow[l] \arrow[d, "f_2"'] \\ Y(0) & Y(1) \arrow[l] & Y(2) \arrow[l] \end{tikzcd} \rightsquigarrow \begin{tikzcd} A(0) :: \bullet \arrow[d, "1"'] & A(1) :: X(0) \arrow[l] \arrow[d, "1 :: f_0"'] & A(2) :: X(1) \arrow[l] \arrow[d, "1 :: f_1"'] & A(3) :: X(2) \arrow[l] \arrow[d, "1 :: f_2"'] \\ A(0) :: \bullet & A(1) :: Y(0) \arrow[l] & A(2) :: Y(1) \arrow[l] & A(3) :: Y(2) \arrow[l] \end{tikzcd}$$
etc

So the $\omega$-chain generated by $G$ is



Note that the yellow arrows are isomorphisms, so to compute the limit of individual columns, we only need to reach the diagonal, after which everything stays the same, so there is no point in going beyond.

The appearance of isomorphisms is not a coincidence. ∎

**define** Say $f : X \to Y : \omega\text{-Set}$ is $n$-isomorphism

iff $\forall k \le n : \mathbb{N}$:
$$f : X(k) \longrightarrow Y(k) \text{ is iso.}$$ ∎

**prop** For enriched $G : \blacktriangleright\omega\text{-Set} \to \omega\text{-Set}$,

given $f : X \to Y : \omega\text{-Set}$ $n$-iso,

then $G(p(f)) : G[X] \to G[Y] : \omega\text{-Set}$ is $n+1$-iso.

**proof** Note that $p(f) : \blacktriangleright[X \to Y] : \omega\text{-Set}$ is $n+1$-iso,

since $p$ just shifts everything & adds a trivially invertible component at the start.

Let $\bar{f} := (\bar{f}_0, \bar{f}_1, \dots, \bar{f}_n) : X \xrightarrow[n]{} Y$ have invertible components,

then we have an inverse
$$\bar{g} := (\bar{g}_0, \bar{g}_1, \dots, \bar{g}_n) : Y \xrightarrow[n]{} X$$

Then
$$G_n(\bar{g} \circ \bar{f}) = G_n(\bar{g}) \circ G(\bar{f})$$
$$G_n(\underset{\substack{\| \\ id_{G(X),n}}}{id_{X,n}})$$

Similarly for $\bar{f} \circ \bar{g} \dots$ we get that $G(\bar{f}) : X \xrightarrow[n+1]{} Y$ has invertible components.

Going back to $f : X \to Y$, note that

first $n+1$ components of $n$
$$f_n : X \xrightarrow[n]{} Y \text{ is invertible,}$$

so $G(f_n) : X \xrightarrow[n]{} Y$ is invertible,

but $G$ preserves projections, so
$$G(f)_n = G(f_n) : X \xrightarrow[n+1]{} Y \text{ is invertible,}$$

hence $G(f)$ is $n+1$ iso. ∎

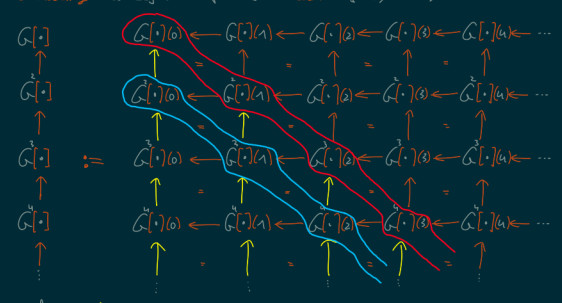
**prop** $G(p(!)) : G(\omega) \longrightarrow G^2(\omega)$ is $0$-iso

**proof** Recall that for any $X : \omega\text{-Set}$,

we have a canonical iso
$$G(\omega)(0) \simeq G(X)(0)$$
& for any $f : X \to Y : \omega\text{-Set}$ (in particular $f := !$),

the $0$-th component of
$$G(p(f)) \text{ must be the canonical iso.}$$ ∎

So unrolling the definition of the $\omega$-chain of $G$, we get



where the yellow maps are iso.

Denote the diagonal $\omega$-chain by $fix[G]$,

then applying $G$ to $fix[G]$, we get the blue chain $G[fix[G]]$ below the diagonal,

which is equipped with an iso $G[fix[G]] \to fix[G] : \omega\text{-Set}$,

whose components are the yellow maps from blue chain to red chain.

Hence any enriched $G : \blacktriangleright\omega\text{-Set} \to \omega\text{-Set}$

has a canonical fixpoint (in fact unique up to canonical iso).

Also given $H : \blacktriangleright\omega\text{-Set}^{op} \to \omega\text{-Set}$,

we can compose it with itself, to get covariant
$$H^2 : \blacktriangleright\omega\text{-Set} \to \omega\text{-Set},$$
whose fixpoint is also a fixpoint of original $H$.

More generally, given $T : \blacktriangleright\omega\text{-Set}^{op} \times \blacktriangleright\omega\text{-Set} \longrightarrow \omega\text{-Set}$,

for each fixed $A : \omega\text{-Set}$, we can construct an enriched function
$$\lambda X.\, T[A, X] : \blacktriangleright\omega\text{-Set} \longrightarrow \omega\text{-Set},$$
that has a fixpoint $fix[\lambda X.\, T[A,X]] : \omega\text{-Set}$.

This can be made into a contravariant enriched function
$$\lambda A.\, fix[\lambda X.\, T[A,X]] : \blacktriangleright\omega\text{-Set}^{op} \longrightarrow \omega\text{-Set}$$
whose fixpoint $\mathcal{U} : \omega\text{-Set}$ is also a fixpoint of original $T$
$$T[\mathcal{U}, \mathcal{U}] \simeq \mathcal{U}$$

**example** The above justifies the following definition:

For each $A : \omega\text{-Set}$, let
$$Rec[A] := [\blacktriangleright Rec[A] \times [\blacktriangleright A \to A] \to A].$$

Then from the isos we get
$$fold : [\blacktriangleright Rec[A] \times [\blacktriangleright A \to A] \to A] \longrightarrow Rec[A]$$
& $$unfold : Rec[A] \times \blacktriangleright Rec[A] \times [\blacktriangleright A \to A] \longrightarrow A$$

This is a bit like self-application

Let
$$\theta(r : \blacktriangleright Rec_A,\ f : \blacktriangleright A \to A) := f(p(unfold)([r, p(r), p(f)])) : A$$
i.e. $\theta : \blacktriangleright Rec_A \times [\blacktriangleright A \to A] \longrightarrow A$.

So we can define
$$fix := \lambda f : \blacktriangleright A \to A.\ \theta(fold(\theta), f)$$
$$fix : [\blacktriangleright A \to A] \longrightarrow A$$
which is the unique fixpoint we have seen before ∎