Inductive Inductive Types

Thomas Posthuma, Pieter-Jan Lavaerts

Radboud University

12 December 2024

Motivation

- Has been implemented in Agda
- Has been used to study type theory within itself
- This paper verifies consistency

What is an inductive-inductive type?

An inductive type $A : Set$ together with **type indexed family** $B : A \to Set$

```
Inductive A : Set :=
| ..
mutual B : A -> Set :=
| ..
```

Inductive-Inductive buildings

$$
\begin{aligned}
\text{ground} &: \text{Platform}, \\
\text{extension} &: ((p : \text{Platform}) \times \text{Building}(p)) \to \text{Platform}, \\
\text{onTop} &: (p : \text{Platform}) \to \text{Building}(p), \\
\text{hangingUnder} &: ((p : \text{Platform}) \times (b : \text{Building}(p))) \to \text{Building}(\text{extension}(\langle p, b \rangle))
\end{aligned}
$$

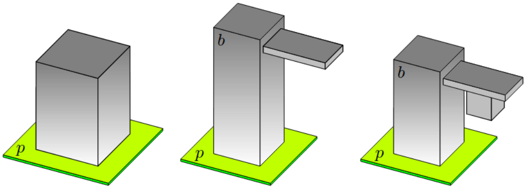

Fig. 1. onTop(p), extension($\langle p, b \rangle$) and hangingUnder($\langle p, b \rangle$).

Simultaneous inductive to Inductive-Inductive

Simultaneous inductive

$$\text{intro}_A : \Phi_A(A, B) \to A \qquad \text{intro}_B : \Phi_B(A, B) \to B$$

Inductive-inductive

$$\text{intro}_A : \Phi_A(A, B) \to A \qquad \text{intro}_B : (a : \Phi_B(A, B)) \to B(i_{A,B}(a))$$

Strictly Positive

Remember from ~~yesterday~~ last week that we had

$$\text{intro} : \Phi(A) \to A$$

where the functor Φ was constructed as follows:

- No premises: $\Phi(A) = \mathbf{1}$
- Non-inductive premise: $\Phi(A) = (x : K) \times \Psi_x(A)$
- Inductive premise: $\Phi(A) = (K \to A) \times \Psi(A)$

Strictly Positive Operators

If we then move to defining two sets, we get

$$\text{intro}_A : \Phi_A(A, B) \to A \qquad \text{intro}_B : \Phi_B(A, B) \to B$$

- No premises: $\Phi(A, B) = \mathbf{1}$
- Non-inductive premise: $\Phi(A, B) = (x : K) \times \Psi_x(A, B)$
- Premise inductive in A: $\Phi(A, B) = (K \to A) \times \Psi(A, B)$
- Premise inductive in B: $\Phi(A, B) = (K \to B) \times \Psi(A, B)$

Strictly Positive Operators

Moving from a simultaneous inductive to an inductive-inductive defition

$$\text{intro}_A : \Phi_A(A, B) \to A \qquad \text{intro}_B : (a : \Phi_B(A, B)) \to B(i_{A,B})$$

- No premises: $\Phi(A, B) = \mathbf{1}$
- Non-inductive premise: $\Phi(A, B) = (x\colon K) \times \Psi_x(A, B)$
- Premise inductive in A: $\Phi(A, B) = (K \to A) \times \Psi(A, B)$ Premise inductive in A: ~~$\Phi(A, B) = (K \to A) \times \Psi(A, B)$~~ Premise inductive in A: $\Phi(A, B) = (f : K \to A) \times \Psi_f(A, B)$*
- Premise inductive in B: $\Phi(A, B) = (K \to B) \times \Psi(A, B)$ Premise inductive in B: ~~$\Phi(A, B) = (K \to B) \times \Psi(A, B)$~~ Premise inductive in B: $\Phi(A, B) = (f : ((x : K) \to B(i_{A,B}(x)))) \times \Psi_F(A, B)*$

* : This Ψ_f is only allowed to depend on $f : K \to A$ for indices of B

Axiomatisation using coding

$$SP_A : \text{Type} \qquad SP_B : \text{Type}$$

together with

$$\text{Arg}_A \qquad \text{Arg}_B$$

SP_A: Formation Rule

$$\frac{A_{ref} : \mathrm{Set} \qquad B_{ref} : \mathrm{Set}}{\mathrm{SP}_\mathrm{A}(A_{ref}, B_{ref}) : \mathrm{Type}}$$

Eventually, we only want to look at codes that don't already have any elements:
$\mathrm{SP'}_\mathrm{A} := \mathrm{SP}_\mathrm{A}(\mathbf{0}, \mathbf{0})$

SP_A: Introduction Rules

$$\frac{}{\mathrm{nil_A} : \mathrm{SP_A}(A_{ref}, B_{ref})}$$

Representing a trivial constructor

$$\frac{K : \mathrm{Set} \qquad \gamma : K \to \mathrm{SP_A}(A_{ref}, B_{ref})}{\mathrm{nonind}(K, \gamma) : \mathrm{SP_A}(A_{ref}, B_{ref})}$$

Representing a constructor with a non-inductive argument

$$\frac{K : \mathrm{Set} \qquad \gamma : \mathrm{SP_A}(A_{ref} + K, B_{ref})}{\text{A-ind}(K, \gamma) : \mathrm{SP_A}(A_{ref}, B_{ref})}$$

Representing a constructor with an A-inductive argument

SP_A: Introduction Rules (cont)

$$\frac{K : \mathrm{Set} \qquad h_{index} : K \to A_{ref} \qquad \gamma : \mathrm{SP}_{\mathrm{A}}(A_{ref}, B_{ref} + K)}{\mathrm{B\text{-}ind}(K, h_{index}, \gamma) : \mathrm{SP}_{\mathrm{A}}(A_{ref}, B_{ref})}$$

Representing a constructor with a B-inductive argument

Example

If we look at the following constructor:

$$\text{extension} : ((p : \text{Platform}) \times \text{Building}(p)) \to \text{Platform}$$

Let's rewrite it so that the rule will fit on the slide:

$$\text{ext} : ((p : \text{A})) \times \text{B}(p) \to \text{A}$$

Then this rule would have following code:

$$\gamma_{ext} = \text{A-ind}(\mathbf{1}, \text{B-ind}(\mathbf{1}, \lambda * .\hat{p}, \text{nil}_\text{A}))$$

Where then γ_{ext} : $\text{SP'}_\text{A} = \text{SP}_\text{A}(\mathbf{0}, \mathbf{0})$, and $\hat{p} = inr(*)$ is the element representing the "induction hypothesis"

Arg_A: Formation Rule

$$
\frac{\begin{array}{c} A_{ref}, B_{ref} : \mathrm{Set} \\ \gamma : \mathrm{SP}_\mathrm{A}(A_{ref}, B_{ref}) \end{array} \qquad \begin{array}{c} A : \mathrm{Set} \\ B : A \to \mathrm{Set} \end{array} \qquad \begin{array}{c} \mathrm{rep}_\mathrm{A} : A_{ref} \to A \\ \mathrm{rep}_\mathrm{index} : B_{ref} \to A \\ \mathrm{rep}_\mathrm{B} : (x : B_{ref}) \to B(\mathrm{rep}_\mathrm{index}(x)) \end{array}}{\mathrm{Arg}_\mathrm{A}(A_{ref}, B_{ref}, \gamma, A, B, \mathrm{rep}_\mathrm{A}, \mathrm{rep}_\mathrm{index}, \mathrm{rep}_\mathrm{B}) : \mathit{Set}}
$$

$\mathrm{Arg_A}$: Formation Rule

$$\frac{\begin{array}{c} A_{ref}, B_{ref} : \mathrm{Set} \\ \gamma : \mathrm{SP_A}(A_{ref}, B_{ref}) \end{array} \quad \begin{array}{c} A : \mathrm{Set} \\ B : A \to \mathrm{Set} \end{array} \quad \begin{array}{c} \mathrm{rep_A} : A_{ref} \to A \\ \mathrm{rep_{index}} : B_{ref} \to A \\ \mathrm{rep_B} : (x : B_{ref}) \to B(\mathrm{rep_{index}}(x)) \end{array}}{\mathrm{Arg_A}(A_{ref}, B_{ref}, \gamma, A, B, \mathrm{rep_A}, \mathrm{rep_{index}}, \mathrm{rep_B}) : Set}$$

γ represents a constructor, which can make use of the elements represented by codes in A_{ref} and B_{ref}

Arg_A: Formation Rule

$$\frac{\begin{matrix} A_{ref}, B_{ref} : \mathrm{Set} \\ \gamma : \mathrm{SP_A}(A_{ref}, B_{ref}) \end{matrix} \qquad \begin{matrix} A : \mathrm{Set} \\ B : A \to \mathrm{Set} \end{matrix} \qquad \begin{matrix} \mathrm{rep_A} : A_{ref} \to A \\ \mathrm{rep_{index}} : B_{ref} \to A \\ \mathrm{rep_B} : (x : B_{ref}) \to B(\mathrm{rep_{index}}(x)) \end{matrix}}{\mathrm{Arg_A}(A_{ref}, B_{ref}, \gamma, A, B, \mathrm{rep_A}, \mathrm{rep_{index}}, \mathrm{rep_B}) : Set}$$

Since A and B are yet to be defined, these input sets are allowed to be arbitrary for now

Arg_A: Formation Rule

$$\frac{A_{ref}, B_{ref} : \mathrm{Set} \quad \gamma : \mathrm{SP_A}(A_{ref}, B_{ref}) \qquad A : \mathrm{Set} \quad B : A \to \mathrm{Set} \qquad \mathrm{rep_A} : A_{ref} \to A \quad \mathrm{rep_{index}} : B_{ref} \to A \quad \mathrm{rep_B} : (x : B_{ref}) \to B(\mathrm{rep_{index}}(x))}{\mathrm{Arg_A}(A_{ref}, B_{ref}, \gamma, A, B, \mathrm{rep_A}, \mathrm{rep_{index}}, \mathrm{rep_B}) : Set}$$

The various rep functions map elements to their real counterparts

Arg_A: Formation Rule

$$
\frac{\begin{array}{c} A_{ref}, B_{ref} : \mathrm{Set} \\ \gamma : \mathrm{SP}_\mathrm{A}(A_{ref}, B_{ref}) \end{array} \qquad \begin{array}{c} A : \mathrm{Set} \\ B : A \to \mathrm{Set} \end{array} \qquad \begin{array}{c} \mathrm{rep}_\mathrm{A} : A_{ref} \to A \\ \mathrm{rep}_\mathrm{index} : B_{ref} \to A \\ \mathrm{rep}_\mathrm{B} : (x : B_{ref}) \to B(\mathrm{rep}_\mathrm{index}(x)) \end{array}}{\mathrm{Arg}_\mathrm{A}(A_{ref}, B_{ref}, \gamma, A, B, \mathrm{rep}_\mathrm{A}, \mathrm{rep}_\mathrm{index}, \mathrm{rep}_\mathrm{B}) : Set}
$$

The code γ represents a constructor. Arg_A gives the domain of that constructor.

Another definition: Arg'_A

We are mostly interested in the case where $A_{ref} = B_{ref} = \mathbf{0}$, in that case:

- $\gamma : \text{SP'}_\text{A}$
- $\text{rep}_\text{A} : \mathbf{0} \to A$
- $\text{rep}_\text{index} : \mathbf{0} \to A$
- $\text{rep}_\text{B} : (x : \mathbf{0}) \to B(\text{rep}_\text{index}(x))$

Since their types already determines our choices for these functions, we define:

$$\text{Arg'}_\text{A}(\gamma, A, B) := \text{Arg}_\text{A}(\mathbf{0}, \mathbf{0}, \gamma, A, B, !_A, !_A, !_{B \circ !_A})$$

Arg_A

The code nil_A represents a constructor with no argument, and as we saw earlier, the domain for that constructor is $\mathbf{1}$

$$Arg_A(A_{ref}, B_{ref}, nil_A, A, B, rep_A, rep_{index}, rep_B) = \mathbf{1}$$

The code $nonind(K, \gamma)$ represents a constructor with a non-inductive argument

$$Arg_A(A_{ref}, B_{ref}, nonind(K, \gamma), A, B, rep_A, rep_{index}, rep_B) = (k : K) \times Arg_A(\ldots, \gamma(k), \ldots)$$

$\mathrm{Arg}_{\mathrm{A}}$

The code $\text{A-ind}(K, \gamma)$ represents a constructor with an A-inductive argument

$$\mathrm{Arg}_{\mathrm{A}}(A_{ref}, B_{ref}, \text{A-ind}(K, \gamma), A, B, \mathrm{rep}_{\mathrm{A}}, \mathrm{rep}_{\mathrm{index}}, \mathrm{rep}_{\mathrm{B}}) = (j : K \to A) \times \mathrm{Arg}_{\mathrm{A}}(\ldots, \gamma(k), \ldots)$$

Arg_A

And $\text{B-ind}(K, h_{\text{index}}, \gamma)$ one with a B-inductive argument

$$
\begin{aligned}
ArgA(A_{ref}, B_{ref}, \text{B-ind}(K, h_{\text{index}}, \gamma), A, B, \text{rep}_\text{A}, \text{rep}_{\text{index}}, \text{rep}_\text{B}) = \\
(j : (k : K) \to B((\text{rep}_\text{A} \circ h_{\text{index}})(k))) \\
\times \mathrm{Arg}_\mathrm{A}(\ldots, B_{ref} + K, \gamma(k), \ldots, \text{rep}_{\text{index}} \sqcup (\text{rep}_\text{A} \circ h_{\text{index}}), \text{rep}_\text{B} \sqcup j)
\end{aligned}
$$

Example

If we go back to our example from earlier with extension, it had the following code:

$$\gamma_{ext} = \text{A-ind}(\mathbf{1}, \text{B-ind}(\mathbf{1}, \lambda * .\hat{p}, \text{nil}_{\text{A}}))$$

It would the following Arg'$_{\text{A}}$:
$\text{Arg'}_{\text{A}}(\gamma_{ext}, \text{Platform}, \text{Building}) = (p : \mathbf{1} \to \text{Platform}) \times \mathbf{1} \to \text{Building}(p(*)) \times \mathbf{1}$
$\text{Arg'}_{\text{A}}(\gamma_{ext}, \text{Platform}, \text{Building}) = (p : \text{Platform}) \times \text{Building}(p)$

Motivation

- We now have representations for (eventual) elements of A and B, and we can reference those representations
- We might want to reference a *constructor* of A as an index for B, but such a constructor will need arguments
- We need to represent an element of $\text{Arg'}_{\text{A}}(\gamma, A, B)$

Intuitively, we might want to construct $\text{Arg'}_{\text{A}}(\gamma, A_{ref}, B_{ref})$ and then use elements from there as representations.

But: A_{ref} and B_{ref} are not quite of the right form yet

The Idea

We will construct:

- $\overline{A_{ref}} : Set$
- $\overline{B_{ref}} : \overline{A_{ref}} \to Set$
- $\overline{\mathrm{rep_A}} : \overline{A_{ref}} \to A$
- $\overline{\mathrm{rep_B}} : (x : \overline{A_{ref}}) \to \overline{B_{ref}}(x) \to B(\overline{\mathrm{rep_A}}(x))$

From these we will then get a function

$$\mathrm{lift}'(\overline{\mathrm{rep_A}}, \overline{\mathrm{rep_A}}) : \mathrm{Arg'}_\mathrm{A}(\gamma, \overline{A_{ref}}, \overline{B_{ref}}) \to \mathrm{Arg'}_\mathrm{A}(\gamma, A, B)$$

A_{ref}

- A_{ref} : Everything we need to represent A
- B_{ref} : Everything we need to represent B
  - So including elements a from A to serve as incices
- $\overline{A_{ref}}$: Everything that *actually* represents an a in A
  - So including those elements from B_{ref}
- $\overline{A_{ref}} := A_{ref} + B_{ref}$.

B_{ref}

- If $\bar{a}$ from $\overline{A_{ref}}$ represents a from A, then elements from $\overline{B_{ref}}(\bar{a})$ should represent elements from $B(a)$
- If $\bar{a}$ is from $\overline{A_{ref}}$ then it is either from A_{ref} or from B_{ref}
- If it is from A_{ref} then we don't know any elements from $B(a)$
- If it is from B_{ref} then we know one element: $\mathrm{rep}_{\mathrm{B}}(\bar{a})$
- $\overline{B_{ref}} := (\lambda x.\mathbf{0}) \sqcup (\lambda x.\mathbf{1})$

$\overline{\mathrm{rep}_\mathrm{A}}$

We define:

- $\overline{\mathrm{rep}_\mathrm{A}} : \overline{A_{ref}} \rightarrow A = (A_{ref} + B_{ref}) \rightarrow A$
- How to map those to the elements of A they represent we already know:
- $\overline{\mathrm{rep}_\mathrm{A}} := \mathrm{rep}_\mathrm{A} \sqcup \mathrm{rep}_\mathrm{index}$

$\overline{\mathrm{rep}_\mathrm{B}}$

- $\overline{\mathrm{rep}_\mathrm{b}} : (x : \overline{A_{ref}}) \to \overline{B_{ref}}(x) \to B(\overline{\mathrm{rep}_\mathrm{A}}(x))$
- If x comes from A_{ref} then $\overline{B_{ref}}(x) = \mathbf{0}$ we have nothing to map, and we use $!_A$ to construct a function of the right type
- If x comes from B_{ref} then $\overline{B_{ref}}(x) = \mathbf{1}$ and we need to map that element to the one element we know exists
- $\overline{\mathrm{rep}_\mathrm{b}} := (\lambda x.!_{B \circ !_A}) \sqcup (\lambda x : *.\mathrm{rep}_\mathrm{B}(x))$

lift

If we have $g : A \to A^*$ and $g' : (x : A) \to B(x) \to B^*(g(x))$ then we can also construct:

$$\text{lift}'(g, g') : \text{Arg'}_{\text{A}}(\gamma, A, B) \to \text{Arg'}_{\text{A}}(\gamma, A^*, B^*)$$

We skip the proof for time reasons

Using the lift function

We now give the following two definitions

- $\overline{\text{arg}_A}(\gamma, A_{ref}, B_{ref}) := \text{Arg'}_A(\gamma, \overline{A_{ref}}, \overline{B_{ref}})$
- $\overline{\text{lift}}(\text{rep}_A, \text{rep}_{\text{index}}, \text{rep}_B) := \text{lift}'(\overline{\text{rep}_A}, \overline{\text{rep}_B})$
  - $\overline{\text{lift}}(\text{rep}_A, \text{rep}_{\text{index}}, \text{rep}_B) : \overline{\text{arg}_A}(\gamma, A_{ref}, B_{ref}) \to \text{Arg'}_A(\gamma, A, B)$
  - $\overline{\text{lift}}(\text{rep}_A, \text{rep}_{\text{index}}, \text{rep}_B) : \overline{\text{Arg'}_A}(\gamma, \overline{A_{ref}}, \overline{B_{ref}}) \to \text{Arg'}_A(\gamma, A, B)$

Representation for arguments

- $\mathrm{rep}_{\mathrm{A},1} := \overline{\mathrm{lift}}(\mathrm{rep}_{\mathrm{A}}, \mathrm{rep}_{\mathrm{index}}, \mathrm{rep}_{\mathrm{B}})$
- $\mathrm{rep}_{\mathrm{A},1} : \overline{arg_A}(\gamma, A_{ref}, B_{ref}) \to \mathrm{Arg'}_{\mathrm{A}}(\gamma, A, B)$
- We now have represenations for *arguments* to constructors

Example

Let's look at γ_{ext} again:

$$\text{extension} : ((p : \text{Platform}) \times \text{Building}(p)) \to \text{Platform}$$

$$\gamma_{ext} = \text{A-ind}(\mathbf{1}, \text{B-ind}(\mathbf{1}, \lambda * .\hat{p}, \text{nil}_\text{A}))$$

and

$$\text{Arg'}_\text{A}(\gamma_{ext}, \text{Platform}, \text{Building}) = (p : \mathbf{1} \to \text{Platform}) \times \mathbf{1} \to \text{Building}(p(*)) \times \mathbf{1}$$

$$\text{Arg'}_\text{A}(\gamma_{ext}, \text{Platform}, \text{Building}) = (p : \text{Platform}) \times \text{Building}(p) \times \mathbf{1}$$

Also assume we have $A_{ref} = B_{ref} = \mathbf{0} + \mathbf{1}$
Then $\overline{A_{ref}} = A_{ref} + B_{ref}$ has two elements: $\hat{p} = \text{inl}(\text{inr}(*))$ and $\widehat{pb} = \text{inr}(\text{inr}(*))$

Example

- $\overline{B_{ref}}(\hat{p}) = \mathbf{0}$
- $\overline{B_{ref}}(\widehat{pb}) = \mathbf{1}$
- $\widehat{\langle pb \rangle} = \langle \widehat{pb}, *, * \rangle$ is the only element in $\overline{\text{arg}_A}(\gamma_{ext}, A_{ref}, B_{ref})$
- $\text{rep}_{A,1}(\widehat{\langle pb \rangle}) = \langle \text{rep}_{\text{index}}(\widehat{pb}), \text{rep}_{\text{B}}(\widehat{pb}), * \rangle = \langle p, b, * \rangle$

Nested Constructors

Our arg fuction has given us the tools to go from a representation for A and B to represenations of arguments of constructors

Now, we want to be able to nest those constructors as well

Nested Constructors

Let's say we have a sequence $\vec{B}_{ref(n)} = B_{ref,0}, B_{ref,1}, \ldots, B_{ref,n-1}$. (Note that $\vec{B}_{ref(0)}$ is just an empty sequence.)
We now define:

$$\mathrm{arg}_A^0(\gamma, A_{ref}, \vec{B}_{ref(0)}) = A_{ref}$$

$$\mathrm{arg}^{n+1}0_A(\gamma, A_{ref}, \vec{B}_{ref(n+1)}) = \overline{\mathrm{arg}_A}(\gamma, \bigplus_{i=0}^{n} \mathrm{arg}_{\mathrm{A}}^i(\gamma, A_{ref}, \vec{B}_{ref(i)}), B_{ref,n})$$

$\mathrm{arg}_{\mathrm{A}}^k$ represents k nested constructors

Looking at arg^1_{A}

$$\text{arg}^1_{\text{A}}(\gamma, A_{ref}, \vec{B}_{ref(1)}) = \overline{\text{arg}_{\text{A}}}(\gamma, \text{arg}^0_{\text{A}}(\gamma, A_{ref}, \vec{B}_{ref(0)}), B_{ref,0})$$
$$= \overline{\text{arg}^0_{\text{A}}}(\gamma, A_{ref}, B_{ref,0})$$

In the "real" world

$$\mathrm{Arg}_{\mathrm{A}}^{0}(\gamma, A, \vec{B}_{(0)}) = A$$

$$\mathrm{Arg}_{\mathrm{A}}^{n+1}(\gamma, A_{ref}, \vec{B}_{n+1}) = \mathrm{Arg'}_{\mathrm{A}}(\gamma, \bigplus_{i=0}^{n} \mathrm{Arg}_{\mathrm{A}}^{i}(\gamma, A, \vec{B}_{(i)}), \bigsqcup_{i=0}^{n} B_i)$$

Where $\vec{B}_{(n)} = B_0, B_1, ..., B_{n-1}$, with $B_i : \mathrm{Arg}_{\mathrm{A}}^{i}(\gamma_A, A, \vec{B}_{(i-1)}) \to \mathrm{Set}$

$\mathrm{rep}_{\mathrm{index,\ i}}$

If we now have the following:

- $\mathrm{rep}_{\mathrm{A}} : A_{ref} \to A$
- $\mathrm{rep}_{\mathrm{index,\ i}} : B_{ref,i} \to \mathrm{Arg}^{i}_{\mathrm{A}}(\gamma, A, \vec{B})$
- $\mathrm{rep}_{\mathrm{B,i}} : (x : B_{ref,i}) \to B_i(\mathrm{rep}_{\mathrm{index,\ i}}(x))$

Then we can construct:

- $\mathrm{rep}_{\mathrm{A,\ n}} : \mathrm{arg}^{n}_{\mathrm{A}}(\gamma, A_{ref}, \vec{B}_{ref}) \to \mathrm{Arg}^{n}_{\mathrm{A}}(\gamma, A, \vec{B})$
  - $\mathrm{rep}_{\mathrm{A,\ 0}} = \mathrm{rep}_{\mathrm{A}}$
  - $\mathrm{rep}_{\mathrm{A,\ n\,+\,1}} = \overline{\mathrm{lift}}(\|^{n}_{i=0}\mathrm{rep}_{\mathrm{A,\ i}}, \mathrm{in}_n \circ \mathrm{rep}_{\mathrm{index,\ n}}, \mathrm{rep}_{\mathrm{B,\ n}})$

SP_B

- SP_B Codes for constructors
- Arg_B Maps codes on types
- $Index_B$ assigns elements $b : B(a)$ to their index a

Formation rule for SP_B

SP_B is like SP_A but two differences

- We can refer to constructors of A ($\gamma_A : SP'_A$ and $B_{\text{ref}}, 0, \ldots B_{\text{ref}}, i$)
- We need an index for codomain of constructor

Formation rule for SP_B

$$\frac{\gamma_A : \mathrm{SP}'_{\mathrm{A}} \quad A_{\mathrm{ref}} : \mathrm{Set} \quad B_{\mathrm{ref},\ 0}, B_{\mathrm{ref},\ 1}, \ldots, B_{\mathrm{ref},\ k} : \mathrm{Set}}{\mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\mathrm{ref}}, B_{\mathrm{ref},\ 0}, B_{\mathrm{ref},\ 1}, \ldots, B_{\mathrm{ref},\ k}) : \mathrm{Type}}$$

Formation rule for SP_B

$$\frac{A_{\text{ref}} : \text{Set} \qquad B_{\text{ref}} : \text{Set}}{\text{SP}_\text{A}(A_{\text{ref}}, B_{\text{ref}}) : \text{Type}}$$

$$\frac{\boxed{\gamma_A : \text{SP}'_\text{A}} \qquad A_{\text{ref}} : \text{Set} \qquad B_{\text{ref, 0}}, \boxed{B_{\text{ref, 1}}, \ldots, B_{\text{ref, }k}} : \text{Set}}{\text{SP}_\text{B}(\gamma_A, A_{\text{ref}}, B_{\text{ref, 0}}, B_{\text{ref, 1}}, \ldots, B_{\text{ref, }k}) : \text{Type}}$$

Formation rule for SP_B

$$\text{hangingUnder} : ((p : \text{Platform}) \times (b : \text{Building}(p))) \to \text{Building}(\underline{\text{extension}}(\langle p, b \rangle)).$$

$$\frac{\gamma_A : \mathrm{SP}'_{\mathrm{A}} \quad A_{\mathrm{ref}} : \mathrm{Set} \quad B_{\mathrm{ref},\ 0}, \boxed{B_{\mathrm{ref},\ 1}, \ldots, B_{\mathrm{ref},\ k}} : \mathrm{Set}}{\mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\mathrm{ref}}, B_{\mathrm{ref},\ 0}, B_{\mathrm{ref},\ 1}, \ldots, B_{\mathrm{ref},\ k}) : \mathrm{Type}}$$

Introduction rules for SP_B

$\text{nil}_{\text{B}}(a_{\text{index}})$

$\text{nonind}(K, \gamma)$

$\text{A-ind}(K, \gamma) :$

$\text{B}_\ell\text{-ind}(K, h_{\text{index}}, \gamma)$

Introduction rules for SP_B

$$\frac{a_{\text{index}} : +_{i=0}^{k} \operatorname{arg}_{\mathrm{A}}^{i}(\gamma_A, A_{\text{ref}}, \vec{B}_{\text{ref}})}{\mathrm{nil}_{\mathrm{B}}(a_{\text{index}}) : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}$$

$$\frac{K : \mathrm{Set} \quad \gamma : K \to \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}{\mathrm{nonind}(K, \gamma) : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}$$

$$\frac{K : \mathrm{Set} \quad \gamma : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}} + K, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}{\text{A-ind}(K, \gamma) : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}$$

$$\frac{\begin{array}{c} h_{\text{index}} : K \to \operatorname{arg}_{\mathrm{A}}^{\ell}(\gamma_A, A_{\text{ref}}, \vec{B}_{\text{ref}}) \\ K : \mathrm{Set} \quad \gamma : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,\ell} + K, \ldots, B_{\text{ref},\,k}) \end{array}}{\mathrm{B}_{\ell}\text{-ind}(K, h_{\text{index}}, \gamma) : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}$$

Introduction rules for SP_B

$$\frac{a_{\text{index}} : +_{i=0}^{k} \text{arg}_{\text{A}}^{i}(\gamma_A, A_{\text{ref}}, \vec{B}_{\text{ref}})}{\text{nil}_{\text{B}}(a_{\text{index}}) : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ k})}$$

$$\frac{}{\text{nil}_{\text{A}} : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}})}$$

$$\frac{K : \text{Set} \quad \gamma : K \to \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ k})}{\text{nonind}(K, \gamma) : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ k})}$$

$$\frac{K : \text{Set} \quad \gamma : K \to \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}})}{\text{nonind}(K, \gamma) : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}})}$$

$$\frac{K : \text{Set} \quad \gamma : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}} + K, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ k})}{\text{A-ind}(K, \gamma) : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ k})}$$

$$\frac{K : \text{Set} \quad \gamma : \text{SP}_{\text{A}}(A_{\text{ref}} + K, B_{\text{ref}})}{\text{A-ind}(K, \gamma) : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}})}$$

$$\frac{\begin{array}{c} h_{\text{index}} : K \to \text{arg}_{\text{A}}^{\ell}(\gamma_A, A_{\text{ref}}, \vec{B}_{\text{ref}}) \\ K : \text{Set} \quad \gamma : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ \ell} + K, \ldots, B_{\text{ref},\ k}) \end{array}}{\text{B}_{\ell}\text{-ind}(K, h_{\text{index}}, \gamma) : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\ 0}, \ldots, B_{\text{ref},\ k})}$$

$$\frac{K : \text{Set} \quad h_{\text{index}} : K \to A_{\text{ref}} \quad \gamma : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}} + K)}{\text{B-ind}(K, h_{\text{index}}, \gamma) : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}})}$$

Introduction rules for SP_B

$$\frac{a_{\text{index}} : +_{i=0}^{k} \operatorname{arg}^{i}_{\mathrm{A}}(\gamma_A, A_{\text{ref}}, \vec{B}_{\text{ref}})}{\operatorname{nil}_{\mathrm{B}}(a_{\text{index}}) : \mathrm{SP}_{\mathrm{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref. }0}, \ldots, B_{\text{ref. }k})}$$

$$\frac{}{\operatorname{nil}_{\mathrm{A}} : \mathrm{SP}_{\mathrm{A}}(A_{\text{ref}}, B_{\text{ref}})}$$

Introduction rules for SP_B

$$\frac{K : \text{Set} \qquad \begin{array}{c} h_{\text{index}} : K \to \text{arg}^{\ell}_{\text{A}}(\gamma_A, A_{\text{ref}}, \vec{B}_{\text{ref}}) \\ \gamma : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,\ell} + K, \ldots, B_{\text{ref},\,k}) \end{array}}{\text{B}_{\ell}\text{-ind}(K, h_{\text{index}}, \gamma) : \text{SP}_{\text{B}}(\gamma_A, A_{\text{ref}}, B_{\text{ref},\,0}, \ldots, B_{\text{ref},\,k})}$$

$$\frac{K : \text{Set} \qquad \begin{array}{c} h_{\text{index}} : K \to A_{\text{ref}} \\ \gamma : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}} + K) \end{array}}{\text{B-ind}(K, h_{\text{index}}, \gamma) : \text{SP}_{\text{A}}(A_{\text{ref}}, B_{\text{ref}})}$$

Introduction rules for SP_B

$$\frac{K : \mathrm{Set} \quad \dfrac{h_{\mathrm{index}} : K \to \arg^{\ell}_{\mathrm{A}}(\gamma_A, A_{\mathrm{ref}}, \vec{B}_{\mathrm{ref}})}{\gamma : \mathrm{SP_B}(\gamma_A, A_{\mathrm{ref}}, B_{\mathrm{ref},\,0}, \ldots, B_{\mathrm{ref},\,\ell} + K, \ldots, B_{\mathrm{ref},\,k})}}{\mathrm{B}_{\ell}\text{-ind}(K, h_{\mathrm{index}}, \gamma) : \mathrm{SP_B}(\gamma_A, A_{\mathrm{ref}}, B_{\mathrm{ref},\,0}, \ldots, B_{\mathrm{ref},\,k})}$$

$$\frac{K : \mathrm{Set} \quad \dfrac{h_{\mathrm{index}} : K \to A_{\mathrm{ref}}}{\gamma : \mathrm{SP_A}(A_{\mathrm{ref}}, B_{\mathrm{ref}} + K)}}{\mathrm{B}\text{-ind}(K, h_{\mathrm{index}}, \gamma) : \mathrm{SP_A}(A_{\mathrm{ref}}, B_{\mathrm{ref}})}$$

Arg_B

nil_b, nonind, A-ind are analogous to Arg_A

$$\begin{aligned} \mathrm{nil}_B(a_{\mathrm{index}}) &\to \mathbf{1} \\ \mathrm{nonind}(K, \gamma) &\to (k : K) \times \text{recursive call} \\ \text{A-ind}(K, \gamma) &\to (j : K \to A) \times \text{recursive call} \end{aligned}$$

Arg_B

$$
\begin{aligned}
&\mathrm{B}_I\text{-ind}(K, h_{index}, \gamma) \to \\
&\qquad (j : (k : K) \to B_I((Rep_{A,I} \circ h_{index}(k))) \times \text{ recursive call}
\end{aligned}
$$

The last missing piece is now Index_B
Again we do case distinction on the codes

$$\text{Index}_\text{B}(\gamma_A, A_\text{ref}, \vec{B}_\text{ref}, \underline{\text{nil}_\text{B}(a_\text{index})}, A, \vec{B}, \text{rep}_\text{A}, \vec{\text{rep}}_\text{index}, \vec{\text{rep}}_\text{B}, \star) = (\coprod_{i=0}^{k} \text{rep}_{\text{A},i})(a_\text{index})$$

$$\mathrm{Index_B}(\gamma_A, A_{\mathrm{ref}}, \vec{B}_{\mathrm{ref}}, \underline{\mathrm{nonind}(K, \gamma)}, A, \vec{B}, \mathrm{rep_A}, \vec{\mathrm{rep}}_{\mathrm{index}}, \vec{\mathrm{rep}}_{\mathrm{B}}, \langle \underline{k,} y \rangle) = \mathrm{Index_B}(\_, \_, \_\_, \gamma(k), \_, \_\_, \_, \_\_, \_\_, y)$$

$$\mathrm{Index_B}(\gamma_A, A_{\mathrm{ref}}, \vec{B}_{\mathrm{ref}}, \underline{\text{A-ind}(K,\gamma)}, A, \vec{B}, \mathrm{rep_A}, \vec{\mathrm{rep}}_{\mathrm{index}}, \vec{\mathrm{rep}}_{\mathrm{B}}, \langle j, y\rangle) = \mathrm{Index_B}(\_, A_{\mathrm{ref}} + K, \_\_, \gamma, \_, \_\_, \mathrm{rep_A} \sqcup j, \_\_, \_\_, y)$$

$$\mathrm{Index}_\mathrm{B}(\gamma_A, A_\mathrm{ref}, \vec{B}_\mathrm{ref}, \underline{\mathrm{B}_n\text{-ind}(K, h, \gamma)}, A, \vec{B}, \mathrm{rep}_\mathrm{A}, \vec{\mathrm{rep}}_\mathrm{index}, \vec{\mathrm{rep}}_\mathrm{B}, \langle j, y \rangle) =$$
$$\mathrm{Index}_\mathrm{B}(\_, \_, \_\_, B_{\mathrm{ref},\ n+K}, \_\_, \gamma, \_, \_\_, \_, \_\_, \mathrm{rep}_{\mathrm{index},n} \sqcup (\mathrm{rep}_{\mathrm{A},n} \circ h), \_\_, \_\_, \mathrm{rep}_{\mathrm{B},n} \sqcup j, \_\_, y).$$

Formation rules

$$\frac{\gamma_A : \text{SP'}_\text{A} \qquad \gamma_B : \text{SP'}_\text{B}(\gamma_A)}{A_{\gamma_A,\gamma_B} : \text{Set}}$$

$$\frac{\gamma_A : \text{SP'}_\text{A} \qquad \gamma_B : \text{SP'}_\text{B}(\gamma_A)}{B_{\gamma_A,\gamma_B} : A_{\gamma_A,\gamma_B} \to \text{Set}}$$

All rules will have the premises $\gamma_A : \text{SP'}_\text{A}$ and $\gamma_B : \text{SP'}_\text{B}(\gamma_A)$, so from now on we'll leave them out

Introduction rule for A

$$\frac{a : \text{Arg'}_{\text{A}}(\gamma_A, A_{\gamma_A,\gamma_B}, B_{\gamma_A,\gamma_B})}{\text{intro}_A(a) : A_{\gamma_A,\gamma_B}}$$

Introduction Rule for B

$$\frac{b : \text{Arg'}_{\text{B}}(\gamma_A, A_{\gamma_A,\gamma_B}, B_{\gamma_A,\gamma_B}, B_1, \ldots, B_k)}{\text{intro}_B(b) : B_{\gamma_A,\gamma_B}(\overline{\text{index}}(b))}$$

We don't have these yet!

B_i's

We still need the various functions $B_i : \mathrm{Arg}^i_{\mathrm{B}}(\gamma_A, A_{\gamma_A,\gamma_B}, B_{\gamma_A,\gamma_B}) \to \mathrm{Set}$
We will need to define:

$$\mathrm{intro}_n : \mathrm{Arg}^n_{\mathrm{A}}(\gamma_A, A_{\gamma_A,\gamma_B}, B_0, \ldots, B_{n-1}) \to A_{\gamma_A,\gamma_B}$$
$$B_n : \mathrm{Arg}^n_{\mathrm{A}}(\gamma_A, A_{\gamma_A,\gamma_B}, B_0, \ldots, B_{n-1}) \to \mathrm{Set}$$

B_i's

$$\mathrm{intro}_0 = \mathrm{id}$$

$$\mathrm{intro}_{n+1} = \mathrm{intro}_A \circ \mathrm{lift}'(\bigsqcup_{i=0}^{n} \mathrm{intro}_i, \bigsqcup_{i=0}^{n} (\lambda a.id))$$

$$B_i(x) = B_{\gamma_A,\gamma_B}(\mathrm{intro}_i(x))$$

One more definition

$$\overline{index} = \left(\bigsqcup_{i=0}^{k} \mathrm{intro}_i\right) \circ \mathrm{Index}'_B(\gamma_A, \gamma_B, A_{\gamma_A,\gamma_B}, B_0, ..., B_k)$$

Introduction Rule for B

$$\frac{b : \text{Arg'}_{\text{B}}(\gamma_A, A_{\gamma_A,\gamma_B}, B_{\gamma_A,\gamma_B}, B_1, \ldots, B_k)}{\text{intro}_B(b) : B_{\gamma_A,\gamma_B}(\overline{\text{index}}(b))}$$

Questions?