A cartesian bicategory of polynomial functors in homotopy type theory

Samuel Mimram (joint with Eric Finster, Maxime Lucas and Thomas Seiller)
Journées SCALP – November 4, 2021
In a nutshell

The situation:

- the category of polynomial functors is cartesian closed

Our contributions:

- we have formalized polynomials in groupoids (or spaces) in HoTT/Agda
- we have shown that the resulting bicategory is cartesian closed
- we have provided a small axiomatization of the type $B$ of natural numbers and bijections
The situation:

- the category of polynomial functors is cartesian closed
- the 2-category of polynomial functors is not cartesian closed [Girard]

Our contributions:
In a nutshell

The situation:

- the category of polynomial functors is cartesian closed
- the 2-category of polynomial functors is not cartesian closed [Girard]
- we should consider polynomial functors in groupoids [Kock]

Our contributions:
In a nutshell

The situation:

- the category of polynomial functors is cartesian closed
- the 2-category of polynomial functors is not cartesian closed [Girard]
- we should consider polynomial functors in groupoids [Kock]

Our contributions:

- we have formalized polynomials in groupoids (or spaces) in HoTT/Agda
In a nutshell

The situation:

- The category of polynomial functors is cartesian closed
- The 2-category of polynomial functors is not cartesian closed [Girard]
- We should consider polynomial functors in groupoids [Kock]

Our contributions:

- We have formalized polynomials in groupoids (or spaces) in HoTT/Agda
- We have shown that the resulting bicategory is cartesian closed
In a nutshell

The situation:

- the category of polynomial functors is cartesian closed
- the 2-category of polynomial functors is not cartesian closed [Girard]
- we should consider polynomial functors in groupoids [Kock]

Our contributions:

- we have formalized polynomials in groupoids (or spaces) in HoTT/Agda
- we have shown that the resulting bicategory is cartesian closed
- we have provided a small axiomatization of the type $\mathbb{B}$ of natural numbers and bijections
Part I

Polynomial functors
A **polynomial** is a sum of monomials

\[ P(X) = \sum_{0 \leq i < k} X^{n_i} \]

(no coefficients, but repetitions allowed)
Categorifying polynomials

A **polynomial** is a sum of monomials

$$P(X) = \sum_{0 \leq i < k} X^{n_i}$$

(no coefficients, but repetitions allowed)

We can **categorify** this notion: replace natural numbers by elements of a set.

$$P(X) = \sum_{b \in B} X^{E_b}$$
This data can be encoded as a polynomial $P$, which is a diagram in $\textbf{Set}$:

$$E \xrightarrow{p} B$$

where

- $b \in B$ is a monomial
- $E_b = p^{-1}(b)$ is the set of instances of $X$ in the monomial $b$. 
Polynomial functors

This data can be encoded as a polynomial $P$, which is a diagram in $\text{Set}$:

$$E \xrightarrow{p} B$$

where

- $b \in B$ is a monomial
- $E_b = p^{-1}(b)$ is the set of instances of $X$ in the monomial $b$.

It induces a polynomial functor

$$[P] : \text{Set} \to \text{Set}$$

$$X \mapsto \sum_{b \in B} X^{E_b}$$
For instance, consider the polynomial corresponding to the function

$$E \xrightarrow{p} B$$

The associated polynomial functor is

$$\mathcal{P}(X) : \text{Set} \to \text{Set}$$

$$X \mapsto X \times X \sqcup X \times X \times X$$
Polynomial functors

For instance, consider the polynomial corresponding to the function

\[ \mathbb{N} \xrightarrow{p} 1 \]

\[ \vdots \]

\[ \bullet \]

The associated polynomial functor is

\[ [[P]](X) : \textbf{Set} \to \textbf{Set} \]

\[ X \mapsto X \times X \times X \times \ldots \]
For instance, consider the polynomial corresponding to the function

\[ \mathbb{N} \xrightarrow{p} 1 \]

The associated polynomial functor is

\[ \lbrack P \rbrack(X) : \textbf{Set} \to \textbf{Set} \]

\[ X \mapsto X \times X \times X \times \ldots \]

A polynomial is \textit{finitary} when each monomial is a finite product.
Polynomial functors: typed variant

We will more generally consider a “colored variant” of polynomials $P$

\[
I \xleftarrow{s} E \xrightarrow{p} B \xrightarrow{t} J
\]

this means that

- each monomial $b$ has a color $t(b) \in J$,
- each occurrence of a variable $e \in E$ has a color $s(e) \in I$.

\[
\begin{array}{c}
\vdots \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
b \\
\vdots \\
j
\end{array}
\]

\[i_1 \quad i_2 \quad i_{n-1}i_n\]
Polynomial functors: typed variant

We will more generally consider a “colored variant” of polynomials $P$

\[
I \xleftarrow{s} E \xrightarrow{p} B \xrightarrow{t} J
\]

this means that

- each monomial $b$ has a color $t(b) \in J$,
- each occurrence of a variable $e \in E$ has a color $s(e) \in I$.

It induces a polynomial functor

\[
\llbracket P \rrbracket (X) : \text{Set}^I \to \text{Set}^J
\]

\[
(X_i)_{i \in I} \mapsto \left( \sum_{b \in t^{-1}(j)} \prod_{e \in p^{-1}(b)} X_{s(e)} \right)_{j \in J}
\]
The category of polynomial functors

**Proposition**

*The composite of two polynomial functors is again polynomial:*

\[
\begin{align*}
\text{Set}^I &\xrightarrow{[P]} \text{Set}^J \xrightarrow{[Q]} \text{Set}^K \\
[Q] \circ [P] &= [Q \circ P]
\end{align*}
\]

We can thus build a category **PolyFun** of sets and polynomial functors:

- an object is a set \( I \),
- a morphism

\[
F : I \to J
\]

is a polynomial functor

\[
[P] : \text{Set}^I \to \text{Set}^J
\]
A polynomial $P$

$$I \leftarrow^s E \overset{p}{\rightarrow} B \overset{t}{\rightarrow} J$$

induces a polynomial functor

$$\llbracket P \rrbracket : \text{Set}^I \rightarrow \text{Set}^J$$

We have mentioned that composition is defined for polynomials. However, on polynomials, it is not strictly associative: we can build a bicategory $\textbf{Poly}$ of sets and polynomial functors.

This suggests that 2-cells are an important part of the story!
A morphism between two polynomials is

\[
\begin{array}{cccccc}
I & \xleftarrow{s} & E & \xrightarrow{p} & B & \xrightarrow{t} & J \\
| & \downarrow{\varepsilon} & \downarrow{} & \downarrow{\beta} & \downarrow{} & \downarrow{} & | \\
I & \xleftarrow{s'} & E' & \xrightarrow{p'} & B' & \xrightarrow{t'} & J
\end{array}
\]

We send monomials to monomials, preserving typing and arities:

\[
\begin{array}{ccc}
i_1 & i_2 & i_{n-1}i_n \\
\downarrow & \downarrow & \downarrow \\
b & \beta(b) & j
\end{array}
\]

We can build a bicategory \(\text{Poly}\) of sets, polynomials and morphisms of polynomials. In the following, we will restrict to the case where 2-cells are equivalences.
Morphisms between polynomials

A morphism between two polynomials is

\[
I \xleftarrow{s} E \xrightarrow{p} B \xrightarrow{t} J
\]

\[
I \xleftarrow{s'} E' \xrightarrow{p'} B' \xrightarrow{t'} J
\]

We send monomials to monomials, preserving typing and arities:

We can build a bicategory \textbf{Poly} of sets, polynomials and morphisms of polynomials.
Morphisms between polynomials

A morphism between two polynomials is

\[
\begin{array}{ccc}
I & \xleftarrow{s} & E & \xrightarrow{p} & B & \xrightarrow{t} & J \\
& & \downarrow{\varepsilon} & \downarrow{\imath} & \downarrow{\beta} & \\
I & \xleftarrow{s'} & E' & \xrightarrow{p'} & B' & \xrightarrow{t'} & J
\end{array}
\]

We send monomials to monomials, preserving typing and arities:

\[
\begin{array}{ccc}
i_1 & i_2 & \cdots & i_{n-1} & i_n \\
\downarrow & \downarrow & \vdots & \downarrow & \downarrow \\
b & \beta(b) & \mapsto & \beta(b) & j
\end{array}
\]

We can build a bicategory \textbf{Poly} of sets, polynomials and morphisms of polynomials. In the following, we will restrict to the case where 2-cells are \textit{equivalences}. 
A morphism between polynomial functors

\([P], [Q] : \text{Set}^I \rightarrow \text{Set}^J\)

is a “suitable” natural transformation, and we can build a 2-category \textbf{PolyFun}. 
The category $\text{PolyFun}$ is cartesian. Namely, given two polynomial functors in $\text{Poly}$

$$P : I \to J \quad Q : I \to K$$

i.e., in $\text{Cat}$,

$$[P] : \text{Set}^I \to \text{Set}^J \quad [Q] : \text{Set}^I \to \text{Set}^K$$

we have, in $\text{Cat}$,

$$\langle P, Q \rangle : \text{Set}^I \to \text{Set}^J \times \text{Set}^K \cong \text{Set}^{J \sqcup K}$$

and the constructions preserve polynomiality: in $\text{PolyFun}$,

$$\langle P, Q \rangle : I \to (J \sqcup K)$$
For the closed structure, we can hope for the same: given, in PolyFun,

\[ P : I \sqcup J \rightarrow K \]

i.e., in \textbf{Cat},

\[ \left[ P \right] : \text{Set}^{I \sqcup J} \rightarrow \text{Set}^K \]

we have

\[ \text{Set}^{I \sqcup J} \rightarrow \text{Set}^K \]
For the closed structure, we can hope for the same: given, in \textbf{PolyFun},

\[ P : I \sqcup J \to K \]

i.e., in \textbf{Cat},

\[ \lbrack P \rbrack : \text{Set}^{I \sqcup J} \to \text{Set}^K \]

we have

\[
\frac{\text{Set}^{I \sqcup J} \to \text{Set}^K}{\text{Set}^I \times \text{Set}^J \to \text{Set}^K}
\]
For the closed structure, we can hope for the same: given, in **PolyFun**, 

\[ P : I \sqcup J \to K \]

i.e., in **Cat**, 

\[ \left[ P \right] : \text{Set}^{I \sqcup J} \to \text{Set}^{K} \]

we have

\[
\begin{align*}
\text{Set}^{I \sqcup J} & \to \text{Set}^{K} \\
\text{Set}^{I} \times \text{Set}^{J} & \to \text{Set}^{K} \\
\text{Set}^{I} & \to (\text{Set}^{K})^{\text{Set}^{J}}
\end{align*}
\]
Closed structure

For the closed structure, we can hope for the same: given, in PolyFun,

\[ P : I \sqcup J \to K \]

i.e., in Cat,

\[ [P] : \text{Set}^{I \sqcup J} \to \text{Set}^K \]

we have

\[
\begin{align*}
\text{Set}^{I \sqcup J} & \to \text{Set}^K \\
\text{Set}^I \times \text{Set}^J & \to \text{Set}^K \\
\text{Set}^I & \to (\text{Set}^K)^{\text{Set}^J} \\
\text{Set}^I & \to \text{Set}^{\text{Set}^J \times K}
\end{align*}
\]
For the closed structure, we can hope for the same: given, in PolyFun,

\[ P : I \sqcup J \to K \]

i.e., in Cat,

\[
[P] : \text{Set}^{I \sqcup J} \to \text{Set}^K
\]

we have

\[
\text{Set}^{I \sqcup J} \to \text{Set}^K
\]

\[
\text{Set}^I \times \text{Set}^J \to \text{Set}^K
\]

\[
\text{Set}^I \to (\text{Set}^K)^{\text{Set}^J}
\]

\[
\text{Set}^I \to \text{Set}^{\text{Set}^J \times K}
\]

which suggests defining the closure as

\[
[J, K] = \text{Set}^J \times K
\]
For the closed structure, we can hope for the same: given, in \textbf{PolyFun},

\[ P : I \sqcup J \to K \]

i.e., in \textbf{Cat},

\[
\left[ P \right] : \text{Set}^{I \sqcup J} \to \text{Set}^K
\]

we have

\[
\begin{align*}
\text{Set}^{I \sqcup J} & \to \text{Set}^K \\
\text{Set}^I \times \text{Set}^J & \to \text{Set}^K \\
\text{Set}^I & \to (\text{Set}^K)^{\text{Set}^J} \\
\text{Set}^I & \to \text{Set}^{\text{Set}^J \times K}
\end{align*}
\]

which suggests defining the closure as

\[
[J, K] = \text{Set}^J \times K
\]

for LL-people: this looks like \( !J \Rightarrow K \).
In terms of operations, the intuition behind the bijection

\[ \text{PolyFun}(I \sqcup J, K) \cong \text{PolyFun}(I, \text{Set}^J \times K) \]

is that we can formally transform operations as follows.

\[ \begin{array}{c}
\text{I} \\
\vdots \\
\hspace{1cm} \\
K \\
\text{J} \\
\vdots \\
\vdots \\
\text{K} \\
\end{array} \quad \sim \quad \begin{array}{c}
\text{I} \\
\vdots \\
\vdots \\
\text{J} \\
\vdots \\
\hspace{1cm} \\
\text{K} \\
\end{array} \]
In terms of operations, the intuition behind the bijection

\[ \text{PolyFun}(I \sqcup J, K) \cong \text{PolyFun}(I, \text{Set}/J \times K) \]

is that we can formally transform operations as follows

via

\[ \text{Set}^I \cong \text{Set}/J \]
There are two problems with our closure. The first one is that

\[ [I, J] = \text{Set}/I \times J \]

is too large to be an object of our category.
There are two problems with our closure. The first one is that

\[ [I, J] = \text{Set}/I \times J \]

is too large to be an object of our category.

One can restrict to polynomial functors which are finitary: we can then take

\[ [I, J] = \text{Set}_{\text{fin}}/I \times J \]
There are two problems with our closure. The first one is that

$$[l, J] = \text{Set} / I \times J$$

is too large to be an object of our category.

One can restrict to polynomial functors which are finitary: we can then take

$$[l, J] = \text{Set}_{\text{fin}} / I \times J$$

or rather

$$[l, J] = \mathbb{N} / I \times J$$
There are two problems with our closure. The first one is that 

$$[I, J] = \textbf{Set}/I \times J$$

is too large to be an object of our category.

One can restrict to polynomial functors which are \textit{finitary}: we can then take

$$[I, J] = \textbf{Set}_{\text{fin}}/I \times J$$

or rather

$$[I, J] = \mathbb{N}/I \times J$$

Finitary polynomial functors are also known as \textbf{normal functors} [Girard].
Theorem

The category $\text{PolyFun}$ is cartesian closed.
Theorem
The category $\text{PolyFun}$ is cartesian closed.

Remark (Girard)
The 2-category $\text{PolyFun}$ is not cartesian closed.
Failure of the cartesian closed structure

We would like to have an equivalence of categories

\[ \text{PolyFun}(I \sqcup J, K) \cong \text{PolyFun}(I, \mathbb{N}/J \times K) \]
Failure of the cartesian closed structure

We would like to have an equivalence of categories

\[ \text{PolyFun}(I \sqcup J, K) \simeq \text{PolyFun}(I, \mathbb{N}/J \times K) \]

but consider the polynomial functor

\[ \llbracket P \rrbracket(X) = X^2 : \text{Set}^{0 \sqcup 1} \to \text{Set}^1 \]
Failure of the cartesian closed structure

We would like to have an equivalence of categories

$$\text{PolyFun}(I \sqcup J, K) \simeq \text{PolyFun}(I, \mathbb{N}/J \times K)$$

but consider the polynomial functor

$$[P](X) = X^2 : \text{Set}^{0 \sqcup 1} \to \text{Set}^1$$

which is induced by the polynomial

$$1 \leftarrow 2 \rightarrow 1 \rightarrow 1$$
We would like to have an equivalence of categories

\[
\text{PolyFun}(I \sqcup J, K) \simeq \text{PolyFun}(I, \mathbb{N}/J \times K)
\]

but consider the polynomial functor

\[
\lceil P \rceil (X) = X^2 : \text{Set}^{0\sqcup 1} \to \text{Set}^1
\]

which has two automorphisms

\[
\begin{array}{cccccc}
1 & \leftarrow & 2 & \rightarrow & 1 & \rightarrow & 1 \\
\| & & \tau & \downarrow & \text{id} & \downarrow & \| \\
1 & \leftarrow & 2 & \rightarrow & 1 & \rightarrow & 1
\end{array}
\]
Failure of the cartesian closed structure

We would like to have an equivalence of categories

\[ \text{PolyFun}(I \sqcup J, K) \simeq \text{PolyFun}(I, \mathbb{N}/J \times K) \]

but consider the polynomial functor

\[ [P](X) = X^2 : \text{Set}^{0 \sqcup 1} \to \text{Set}^1 \]

whose exponential transpose is

\[ 0 \leftarrow 0 \rightarrow 1 \xrightarrow{\star \mapsto 2} \mathbb{N} \]

and has only one automorphism.
We would like to have an equivalence of categories

\[ \text{PolyFun}(I \sqcup J, K) \simeq \text{PolyFun}(I, \mathbb{N}/J \times K) \]

but consider the polynomial functor

\[ [P](X) = X^2 : \text{Set}^{0 \sqcup 1} \to \text{Set}^1 \]

whose exponential transpose is

\[
\begin{array}{c}
0 & \leftrightarrow & 0 & \rightarrow & 1 & \rightarrow & \mathbb{N}
\end{array}
\]

and has only one automorphism.

The equivalence fails:

\[ \text{PolyFun}(0 \sqcup 1, 1) \not\cong \text{PolyFun}(0, \mathbb{N}/1 \times 1) \]

(two elements on the left, one on the right)
Fixing the cartesian closed structure

The failure of the equivalence

\[ \text{PolyFun}(0 \sqcup 1, 1) \not\cong \text{PolyFun}(0, \mathbb{N}/1 \times 1) \]

can be interpreted as being due to the fact that \(2 \in \mathbb{N}/1\) has no non-trivial isomorphism.

This suggests moving to \textbf{groupoids}!
Fixing the cartesian closed structure

The failure of the equivalence

\[ \text{PolyFun}(0 ⊔ 1, 1) \not\cong \text{PolyFun}(0, \mathbb{N}/1 × 1) \]

can be interpreted as being due to the fact that \(2 \in \mathbb{N}/1\) has no non-trivial isomorphism.

This suggests moving to **groupoids**!

More precisely, we should replace \(\mathbb{N}\) by the groupoid \(\mathbb{B}\) of all symmetric groups.
The notion of polynomial functor generalizes in any locally cartesian closed category.
The notion of polynomial functor generalizes in any locally cartesian closed category.

...but the category $\text{Gpd}$ is not cartesian closed!
The notion of polynomial functor generalizes in any locally cartesian closed category.

...but the category $\mathbf{Gpd}$ is not cartesian closed!

Kock has identified that if we perform all the usual constructions up to homotopy (slice, pullbacks, etc.), we recover a suitable setting to define polynomial functors.
Polynomial functors in groupoids

The notion of polynomial functor generalizes in any locally cartesian closed category.

...but the category $\text{Gpd}$ is not cartesian closed!

Kock has identified that if we perform all the usual constructions up to homotopy (slice, pullbacks, etc.), we recover a suitable setting to define polynomial functors.

This requires properly defining and using all the usual constructions in a suitable 2-categorical sense.
Given a polynomial $P$

\[
E \xrightarrow{p} B
\]

the induced polynomial functor

\[
[P] : \text{Gpd} \rightarrow \text{Gpd}
\]

\[
X \mapsto \int_{b \in B} E_b
\]

where $E_b$ is the homotopy fiber of $p$ at $b$ and

\[
\int_{b \in E} E_b = \sum_{b \in \pi_0(B)} X_b / \text{Aut}(b)
\]

where the quotient is to be taken 2-categorically / homotopically...
Part II

Formalization in Agda
There is a framework in which everything is constructed \textit{up to homotopy} for free: \textbf{homotopy type theory}.

In particular, there is a well-known notion of groupoid in this setting: a type with no non-trivial equalities between equalities.

Let’s formally develop the theory of polynomials in this setting.
A polynomial

\[
I \leftarrow^s E \rightarrow^p B \rightarrow^t J
\]

is a **container**:

record Poly (I J : Type) : Type₁ where
  field
    Op : J → Type
    Pm : (i : I) → \{j : J\} → Op j → Type

We sometimes write

\[I \rightsquigarrow J = \text{Poly } I \text{ J}\]
Composing polynomials

The polynomial functor induced by a polynomial $P$ is

\[
[-] : I \leadsto J \rightarrow (I \rightarrow \text{Type}) \rightarrow (J \rightarrow \text{Type})
\]

\[
[-] P X j = \Sigma (\text{Op } P j) (\lambda c \rightarrow (i : I) \rightarrow (p : Pm P i c) \rightarrow (X i))
\]
Composing polynomials

The polynomial functor induced by a polynomial $P$ is

\[
[_] : I \rightsquigarrow J \rightarrow (I \rightarrow \text{Type}) \rightarrow (J \rightarrow \text{Type})
\]

\[
[_] \ P \ X \ j = \Sigma \ (\text{Op} \ P \ j) \ (\lambda \ c \rightarrow (i : I) \rightarrow (p : \text{Pm} \ P \ i \ c) \rightarrow (X \ i))
\]

The composite of two polynomials is

\[
\_ \cdot \_ : I \rightsquigarrow J \rightarrow J \rightsquigarrow K \rightarrow I \rightsquigarrow K
\]

\[
\text{Op} \ (P \cdot Q) = [ \ Q \ ] \ (\text{Op} \ P)
\]

\[
\text{Pm} \ (_\cdot_ \ P \ Q) \ i \ (c , a) = \Sigma \ J \ (\lambda \ j \rightarrow \Sigma \ (\text{Pm} \ Q \ j \ c) \ (\lambda \ p \rightarrow \text{Pm} \ P \ i \ (a \ j \ p)))
\]
Theorem

We can build a pre-bicategory of types, polynomials and their morphisms.

Note: by univalence, we can use propositional equality for 2-cells, which simplifies the definition.
Theorem
We can build a pre-bicategory of types, polynomials and their morphisms.

Theorem
We can build a bicategory of groupoids, polynomials in groupoids and their morphisms.
Theorem
We can build a pre-bicategory of types, polynomials and their morphisms.

Theorem
We can build a bicategory of groupoids, polynomials in groupoids and their morphisms.

Theorem
This bicategory is cartesian with \( \square \) as coproduct.
Defining the exponential

In order to define the 1-categorical closure, the plan was:

\[
\begin{array}{c}
\text{Set} & \rightsquigarrow & \text{Set}_{\text{fin}} & \rightsquigarrow & \mathbb{N}
\end{array}
\]
In order to define the 1-categorical closure, the plan was:

\[
\text{Set} \, \rightsquigarrow \, \text{Set}_{\text{fin}} \, \rightsquigarrow \, \mathbb{N}
\]

For the 2-categorical closure the plan is

\[
\text{Gpd} \, \rightsquigarrow \, \text{Gpd}_{\text{fin}} \, \rightsquigarrow \, \mathcal{B}
\]

Here, \(\mathcal{B}\) is the groupoid with \(n \in \mathbb{N}\) as objects and \(\Sigma_n\) as automorphisms on \(n\).
Finite types

We write $\text{Fin } n$ for the canonical finite type with $n$ elements: its constructors are 0 to $n-1$. 
Finite types

We write $\text{Fin } n$ for the canonical finite type with $n$ elements: its constructors are 0 to $n-1$.

data Fin : $\mathbb{N} \to \text{Set}$ where
  zero : \{n : $\mathbb{N}$\} $\to$ Fin (suc n)
  suc : \{n : $\mathbb{N}$\} (i : Fin n) $\to$ Fin (suc n)
The predicate of being finite is

\[
is\text{-finite} : \text{Type} \to \text{Type}
\]
\[
is\text{-finite } A = \Sigma \mathbb{N} (\lambda n \to \parallel A \cong \text{Fin } n \parallel)
\]
Finite types

The predicate of being finite is

\[
is\text{-finite} : \text{Type} \to \text{Type}
\]
\[
is\text{-finite } A = \Sigma \mathbb{N} (\lambda n \to \parallel A \simeq \text{Fin } n \parallel)
\]

The type of finite types is

\[
\text{FinType} : \text{Type}_1
\]
\[
\text{FinType} = \Sigma \text{Type} \text{ is-finite}
\]
Finite types

The predicate of being \textit{finite} is

\[
\text{is-finite} : \text{Type} \to \text{Type} \\
\text{is-finite } A = \Sigma N (\lambda n \to \| A \simeq \text{Fin } n \|)
\]

The type of finite types is

\[
\text{FinType} : \text{Type}_1 \\
\text{FinType} = \Sigma \text{Type} \text{ is-finite}
\]

(note that this is a \textit{large} type)
A polynomial is **finitary** when, for each operation, the total space of its parameters is finite:

\[
\text{is-finitary} : (P : I \rightarrow J) \rightarrow \text{Type} \\
\text{is-finitary} \ P = \{j : J\} (c : \text{Op \ P} \ j) \rightarrow \text{is-finite} (\Sigma I (\lambda i \rightarrow \text{Pm \ P} i c))
\]
The type of \texttt{integers} is

\begin{verbatim}
data \texttt{N} : Type where
  \texttt{zero} : \texttt{N}
  \texttt{suc} : \texttt{N} \rightarrow \texttt{N}
\end{verbatim}
A small model for finite types

The type $\mathcal{B}$ is

```
data $\mathcal{B}$ : Type where
  obj : $\mathbb{N} \rightarrow \mathcal{B}$
  hom : \{n : $\mathbb{N}$\} (α : Fin n ≃ Fin n) → obj n ≡ obj n
  id-coh : (n : $\mathbb{N}$) → hom \{n = n\} ≃-refl ≡ refl
  comp-coh : \{m n o : $\mathbb{N}$\} (α : Fin m ≃ Fin n) (β : Fin n ≃ Fin o) →
            hom (≃-trans α β) ≡ hom α · hom β
```

(this is a small higher inductive type!)
A small model for finite types

The type $\mathbb{B}$ is

data $\mathbb{B}$ : Type where
  obj : $\mathbb{N} \to \mathbb{B}$
  hom : $\{n : \mathbb{N}\} \to \text{obj } n \equiv \text{obj } n$
  id-coh : (n : $\mathbb{N}$) \to \text{hom } \{n = n\} \equiv \text{refl}
  comp-coh : $\{m \ n \ o : \mathbb{N}\} \to \text{hom } \{\text{Fin } m \equiv \text{Fin } n\} \ \ (\beta : \text{Fin } n \equiv \text{Fin } o) \to$
          \text{hom } \{\equiv -\text{trans } \alpha \ \beta\} \equiv \text{hom } \alpha \cdot \text{hom } \beta

(this is a small higher inductive type!)

**Theorem**

$\text{FinType } \simeq \mathbb{B}$. 

The closure

We define

\[ \text{Exp} : \text{Type} \rightarrow \text{Type}_1 \]
\[ \text{Exp } I = I \rightarrow \text{Type} \]

**Theorem**

*Ignoring size issues, for polynomials we have*

\[ (I \uplus J) \leadsto K \simeq I \leadsto (\text{Exp } J \times K) \]
We define

\[ \text{Exp} : \text{Type} \rightarrow \text{Type}_1 \]

\[ \text{Exp} \ I = \Sigma (I \rightarrow \text{Type}) \ (\lambda \ F \rightarrow \text{is-finite} \ (\Sigma \ I \ F)) \]

**Theorem**

*Ignoring size issues, for finitary polynomials we have*

\[(I \sqcup J) \leadsto K \simeq I \leadsto (\text{Exp} \ J \times K)\]
We define

\[ \text{Exp} : \text{Type} \to \text{Type}_1 \]

\[ \text{Exp } I = \Sigma \text{FinType} (\lambda N \to \text{fst } N \to I) \]

**Theorem**

*Ignoring size issues, for finitary polynomials we have*

\[ (I \sqcap J) \rightsquigarrow K \simeq I \rightsquigarrow (\text{Exp } J \times K) \]
The closure

We define

\[ \text{Exp} : \text{Type} \to \text{Type} \]

\[ \text{Exp } I = \Sigma B (\lambda b \to B\text{-to-Fin } b \to I) \]

**Theorem**

For finitary polynomials we have

\[ (I \sqcap J) \leadsto K \simeq I \leadsto (\text{Exp } J \times K) \]
Note that

\[ \text{Exp} : \text{Type} \to \text{Type} \]
\[ \text{Exp} \ I = \Sigma \ B \ (\lambda \ b \to \ B\text{-to-Fin} \ b \to \ I) \]

is the free pseudo-commutative monoid!
Questions?