A cartesian (2,1)-category of homotopy polynomial functors in groupoids

***Abstract*—Polynomial functors are a categorical generalization of the usual notion of polynomial, which has found many applications in higher categories and type theory: those are generated by polynomials consisting a set of monomials built from sets of variables. We study here a further generalization of this notion, based on groupoids instead of sets, for which we introduce a new "fibered" presentation. This allows us to conveniently study the structure of the resulting bicategory: we show that it is cartesian closed and thus forms a model of simply typed λ-calculus, close to the one of generalized species. Our computations are based on a formalization in univalent type theory (in Agda) of polynomials in groupoids. Most notably, this required us introducing an axiomatization in a small universe of the type of finite types, as an appropriate higher inductive type of natural numbers and bijections.**

I. Introduction

Polynomial functors have been introduced as a categorical generalization of traditional polynomials and have been intensively studied by Kock and collaborators [1], [2], [3]. They have become an important categorical tool, allowing the definition and manipulation of various structures such as opetopes [4] or type theories [5]. This motivates the study of the categorical structures they bear in order to facilitate constructions on those, and the situation turns out to be quite subtle. For instance, one would expect that they should be cartesian closed (after all, most usual categories are), but it is not the case: the cartesian product functor does not have a satisfactory right adjoint. There are essentially two reasons for that.

The first one is that there are size issues – the naive definition of the exponential is too large to be a proper object in the category of polynomials – which is easily overcome by restricting to polynomials which are finitary, i.e., only involve monomials consisting of products of variables which are finite. The resulting category is isomorphic to the category of Girard's normal functors [6], which is a model of simply typed λ-calculus, and linear logic [7], which can be thought of as a quantitative variant of the relational model (historically, this model is in fact one of the starting points motivating the introduction of linear logic).

The second one is more problematic: polynomial functors carry a very natural 2-categorical structure, but it was observed early on that the closure mentioned above fails to extend to a 2-categorical one [6], [8], [7]. Here, we advocate that a satisfactory answer to this problem is provided by switching from traditional polynomial functors to ones over groupoids, as first considered by Kock [2], see also [9], [10]. We show that the resulting bicategory is cartesian closed; it is more generally a model of intuitionistic linear logic, which we expect to extend as a model of differential linear logic. The resulting category is close to the "equivariant variant" of polynomials, provided by generalized species or analytic functors [11], [12], [13], [14].

In order to work more easily with polynomial functors over groupoids, we introduce here a fibered variant of the definition introduced in [2], which is easier to manipulate in practice. This definition allows us to carry on explicit computations, and construct a structure of cartesian closed category on those. It is also adapted to mechanized proofs: we have formalized the main constructions required to have a formalization of the constructions performed here, in Agda. In order to do so, we work in the setting of univalent type theory, where 1-truncated types are equipped with a structure of groupoid, whose morphisms are identities, and all morphisms are functorial. A salient contribution is the construction, as a higher inductive type, of the type of finite sets and bijections in a small universe, which is required in order to define the exponential in the category of polynomials.

We recall the definition of polynomials, polynomial functors and associated constructions in section II, we then present a formalization in type theory of the cartesian closed bicategory of polynomial functors in groupoids in section III, we also perform this construction in the setting of set-theoretic categories in section IV and show that it is cartesian closed in section V.

II. Polynomial functors

We begin by recalling the traditional definition of polynomial functors, as well as related constructions. All the material in this section is already known, but required in the following.

A. The category of polynomial functors

Polynomial functors are a categorical generalization of the notion of polynomial, see [1] for a detailed presentation. Polynomials as traditionally defined as finite sums of the form $P(X) = \sum_{0 \leq i < k} X^{n_i}$. This notion can be "categorified" by taking a set B of monomials (instead of specifying their number k) and having a set E_b of instances of X in each monomial b (instead of specifying their number n_i). This data can thus collected as a function representing the *polynomial*

$$E \xrightarrow{p} B \tag{1}$$

where B is the set of monomials and for each monomial b, $E_b = p^{-1}(b)$ is the set of instances of X involved in the

monomial. Writing P for the above diagram (1) in sets, it induces a endofunctor $[\![P]\!]$ of $\mathbf{Set}$ defined as

$$[\![P]\!](X) = \sum_{b \in B} X^{E_b}$$

and we call *polynomial functor* such a functor. An interesting point of view on the above data consists in considering the elements of B as abstract *operations*, whose *parameters* are the elements of E_b, so that $[\![P]\!](X)$ corresponds to the set obtained by formally applying the operations in B to the required number of elements of the set X.

In the following, we more generally consider a "typed variant" of polynomials and polynomial functors, where the parameters of an operation are decorated by a "type" in a set I, as well as their output in a set J. This data can be encoded by a diagram P in $\mathbf{Set}$ of the form

$$I \xleftarrow{s} E \xrightarrow{p} B \xrightarrow{t} J \tag{2}$$

which we call a *polynomial* and consists of an untyped polynomial (1) together with functions s and t respectively indicating the types of the parameters and outputs of operations. Note that the previous untyped setting is recovered when I and J are both the terminal set. Writing $\mathbf{Set}/I$ for the slice category of sets over a set I, such data again induces a functor $[\![P]\!]$, called a *polynomial functor*, obtained as the composite

$$\mathbf{Set}/I \xrightarrow{\Delta_s} \mathbf{Set}/E \xrightarrow{\Pi_p} \mathbf{Set}/B \xrightarrow{\Sigma_t} \mathbf{Set}/J \tag{3}$$

where Δ_s is the pullback map along s, and Σ_t (resp. Π_p) is the left (resp. right) adjoint to Δ_t (resp. Δ_p) given by post-composition by t (resp. local cartesian closure).

Given a set I, it is well known that the slice category $\mathbf{Set}/I$ of sets over I is equivalent to the category of families indexed by I,

$$\mathbf{Set}/I \quad \simeq \quad \mathbf{Set}^I \tag{4}$$

Through this equivalence, the above functor can be seen as a functor $\mathbf{Set}^I \to \mathbf{Set}^J$, which is often convenient, obtained as a composite of the following functors on families:

$$\begin{array}{lrcl}
\Delta_s : & \mathbf{Set}^I & \to & \mathbf{Set}^E \\
 & (X_i \mid i \in I) & \mapsto & (X_{s(e)} \mid e \in E) \\
\Pi_p : & \mathbf{Set}^E & \to & \mathbf{Set}^B \\
 & (X_e \mid e \in E) & \mapsto & (\prod_{e \in p^{-1}(b)} X_e \mid b \in B) \\
\Sigma_t : & \mathbf{Set}^B & \to & \mathbf{Set}^J \\
 & (X_b \mid b \in B) & \mapsto & (\sum_{b \in t^{-1}(j)} X_b \mid j \in J)
\end{array}$$

giving rise to the functor

$$\begin{aligned}
&[\![P]\!] : \mathbf{Set}^I \to \mathbf{Set}^J \\
&(X_i \mid i \in I) \mapsto (\sum_{b \in t^{-1}(j)} \prod_{e \in p^{-1}(b)} X_{s(e)} \mid j \in J)
\end{aligned}$$

A functor obtained as in (3) is called a *polynomial functor* and we say that the diagram (2) is a presentation of the polynomial. For instance, the identity functor on $\mathbf{Set}/I$ is polynomial as being presented by

$$I \xleftarrow{\mathrm{id}} I \xrightarrow{\mathrm{id}} I \xrightarrow{\mathrm{id}} I$$

As a slightly richer example, the functor

$$\begin{aligned}
\exp : \mathbf{Set}/1 &\to \mathbf{Set}/1 \\
X &\mapsto \sum_{n \in \mathbb{N}} X^n
\end{aligned}$$

is also polynomial, as being presented by the diagram

$$1 \longleftarrow \mathbb{N} \xrightarrow{\mathrm{id}} \mathbb{N} \longrightarrow 1$$

It can be shown (and this is non-trivial) that the composite of two polynomial functors is again polynomial: this means that given two polynomial functors generated by two diagrams of the form (2), one can find a third diagram of the form (2) which is a presentation for the composite of the functors. We can thus build a category $\mathbf{PolyFun}$ where an object is a set and a morphism $I \to J$ is a polynomial functor $\mathbf{Set}/I \to \mathbf{Set}/J$. Note that even though an operation of composition is defined on polynomials (2), we cannot build a category of those: their composition being defined by using universal constructions, it will not be strictly associative and the best we can hope for is a structure of a bicategory. This motivates investigating the 2-categorical structure of polynomials.

These constructions can easily be generalized to any local cartesian closed category instead of $\mathbf{Set}$: the local cartesian closure ensures the existence of the right adjoints Π_f to change of base functors Δ_f, which are necessary to define the polynomial functor (3) associated to a polynomial, as well as to define the composition of polynomials. We should also note here that the definition of composition crucially depends on the Beck-Chevalley isomorphism.

B. 2-categorical structure

Given two polynomials P and P' of the form (2), both from I to J, a morphism between them consists of two functions $\beta : B \to B'$ and $\varepsilon : E \to E'$ between the operations (resp. parameters) of P and those of P' making the diagram

$$\begin{tikzcd}
I \arrow[d, equal] & E \arrow[l, "s"'] \arrow[r, "p"] \arrow[d, "\varepsilon"'] \arrow[dr, phantom, "\lrcorner"] & B \arrow[r, "t"] \arrow[d, "\beta"] & J \arrow[d, equal] \\
I & E' \arrow[l, "s'"] \arrow[r, "p'"'] & B' \arrow[r, "t'"'] & J
\end{tikzcd} \tag{5}$$

commute and such that the middle square is a pullback (such a morphism is sometimes said to be *cartesian* to insist on this requirement). This last condition can be understood as requiring that β preserves the arity of operations (it is also technically important because there is no sensible way of defining horizontal composition of morphisms without this condition). The composition of polynomials defined above turns out to be associative up to isomorphism, so that one can define a bicategory $\mathbf{Poly}$ whose 0-cells are sets, 1-cells are polynomials and 2-cells are morphisms of polynomials.

The resulting bicategory can be shown to be biequivalent to the 2-category $\mathbf{PolyFun}$, obtained by adding suitable natural transformations to the above category [1, Theorem 2.17]. There is a subtlety concerning the 2-cells: the "natural" notion of morphism between polynomial functors, strong natural

transformations, is more liberal than the notion of morphism defined above on corresponding polynomials, and one can either restrict those transformations (to cartesian natural transformations) or generalize the notion of morphism between polynomials.

C. The cartesian closed structure

The 1-category $\mathbf{PolyFun}$ of polynomial functors is cartesian. Considering polynomial functors as operating on families through (4), as explained above, the paring of two polynomial functors $P : I \to J$ and $Q : I \to K$ is induced by the one in $\mathbf{Cat}$:

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

which motivates defining the product on objects of $\mathbf{PolyFun}$ as the coproduct of sets.

We can hope that the category also has a closure for products induced by the one in $\mathbf{Cat}$. The sequence of bijections of hom-sets

$$\frac{\dfrac{\mathbf{Set}^I \times \mathbf{Set}^J \to \mathbf{Set}^K}{\mathbf{Set}^I \to (\mathbf{Set}^K)^{\mathbf{Set}^J}}}{\mathbf{Set}^I \to \mathbf{Set}^{\mathbf{Set}^J \times K}}$$

suggests that we define the closure as

$$[J, K] = \mathbf{Set}^J \times K \simeq \mathbf{Set}/J \times K \tag{6}$$

However, this does not make sense because the objects of $\mathbf{PolyFun}$ are sets and this lives in a larger universe. This motivates considering polynomials which are "reasonably small". We say that a polynomial P of the form (2) is *finitary* when every operation has a finite set of parameters, i.e., the set $E_b = p^{-1}(b)$ is finite for every operation b. Polynomial functors corresponding to finitary functors can be shown to be those preserving filtered colimits and are sometimes called *normal functors* [6], [7]. From now on, we restrict our category to such polynomials, and consider them up to isomorphism. This change allows us to replace $\mathbf{Set}/J$ by $\mathbf{Set}_{\mathrm{fin}}/J$ in the above definition of the exponential, where $\mathbf{Set}_{\mathrm{fin}}$ is the class of finite sets (i.e., we consider families of finite sets instead of families of sets). While this is certainly "smaller", this is still not a proper set; however, it is now equivalent to a proper set, namely the set $\mathbb{N}/J$ of functions $[n] \to J$ for some $n \in \mathbb{N}$, where $[n] = \{0, \ldots, n-1\}$ is a canonical choice of a finite set with n elements. To sum up, it can be shown that the resulting category is still cartesian and we can define the exponential as $[J, K] = \mathbb{N}/J \times K$.

We have noted above that the category of polynomial functors really is a 2-category (or a bicategory if we consider polynomials instead) and it is natural to expect that the cartesian closure would extend to the 2-categorical setting, by which we mean that the isomorphism

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

between sets of isomorphism classes of polynomial functors should extend to an equivalence of categories. It has however been observed that it is not the case, see [6, Remark 2.19], [8, Example 1.4.2] and [7, Theorem 1.24]. As an illustration, consider the polynomial functor $[\![P]\!](X) = X^2$, which is induced by the following polynomial P:

$$1 \longleftarrow 2 \longrightarrow 1 \longrightarrow 1$$

(we write n for a set with n elements). Writing $\tau : 2 \to 2$ for the transposition, there are two automorphisms on P: the identity and

$$\begin{tikzcd}
1 & 2 \arrow[l] \arrow[r] \arrow[d, "\tau"'] \arrow[dr, phantom, "\lrcorner", very near start] & 1 \arrow[r] & 1 \\
1 \arrow[u, equal] & 2 \arrow[l] \arrow[r] & 1 \arrow[u, equal] \arrow[r] & 1 \arrow[u, equal]
\end{tikzcd}$$

Since there are two distinct morphisms

$$P \Rightarrow P : 0 \sqcup 1 \to 1$$

if we write $P^\sharp$ for the exponential transpose of P, the closure should induce two distinct morphisms

$$P^\sharp \Rightarrow P^\sharp : 0 \to \mathbb{N}^1 \times 1$$

but this is not the case because 0 is initial in the 2-category.

D. Toward polynomial functors in groupoids

This tension is solved in [7] by quotienting the 2-cells under an ad-hoc equivalence relation. In this paper, we advocate that a more satisfactory approach consists in switching from polynomial functors in sets to polynomial functors in groupoids: intuitively, the problem comes from the fact that, $\mathbb{N}$ being a set, there is no non-trivial endomorphism on a natural number, which we should have if we were to have a closure for the cartesian product.

We will see in section IV that a proper definition of polynomials in groupoids requires more than simply considering diagrams of the form (2) in the category of groupoids. For instance, the traditional definition of composition does not immediately extend to those because the category of groupoids is not locally cartesian closed, and thus the right adjoint to the change of base functor Δ_f is not defined for every morphism f. Following [2], this can be explained as the fact that switching from sets to groupoids can be thought of as switching from 0-truncated spaces to 1-truncated spaces, where the strict universal limits and colimits involved in the constructions on polynomials are not the right one: we need to take limits and colimits *up to homotopy*. In practice, it is convenient to avoid resorting to such constructions up to homotopy, by imposing additional properties on the maps involved in polynomials, typically being fibrations.

III. Formalization in homotopy type theory

We begin by presenting a formalization in homotopy type theory, in the Agda language, of our constructions on polynomials. The reason for presenting it first is mainly didactic: we believe that it is simpler to understand because all constructions performed in the internal language of type theory are up to homotopy (and all types are "fibrant"), whereas in the "set

theoretic version" of section IV we have to ensure that they are so by imposing suitable conditions. While the two exhibit similar constructions and proofs, the link between them is not entirely clear yet to us, making it necessary to present both here, as discussed in remark 31. All the code is available in the repository [15], and is based on our own formalization of homotopy type theory (or HoTT), following the reference book [16], to which we suppose the reader already acquainted.

We unfortunately do not have enough space here to expose in details the basic definitions of Agda and homotopy type theory, and only recall notations. We write `Type` for the universe of small types and $\texttt{Type}_1$ for the universe of large types (in particular `Type` is an element of $\texttt{Type}_1$). We write $x \equiv y$ for the type of *identities* (or *equalities* or *paths*) between two terms x and y of the same type. We write $A \simeq B$ for the type of *equivalences* between two types A and B (possibly in different universes): it consists of functions $f : A \to B$ admitting an inverse up to homotopy, in a suitably coherent sense. We recall that a *proposition* is a type in which any two identity proofs between the same terms are equal, and a *set* (resp. a *groupoid*) is a type in which the type $x \equiv y$ of identities between two elements x and y is a proposition (resp. a set). We postulate here the *univalence* axiom which states that the canonical map from identities $x \equiv y$ to equivalences $x \simeq y$ is itself an equivalence.

A. Formalizing polynomials

Our formalization of polynomials can be found in [15, `Polynomial.agda`]. A direct translation of the definition of polynomials (2) would consist in encoding them as

```
record Poly (I J : Type) : Type₁ where
  field
    B : Type        t : B → J
    E : Type        p : E → B
                    s : E → I
```

However, the constructions performed with this definition are much involved because they require manipulating equalities which quickly becomes quite intricate. As an illustration of this, the type of operations of the composite of two polynomials `P` and `Q` is

```
Σ (B Q) (λ b → (x : hfib (p Q) b) →
                hfib (t P) (s Q (fst x)))
```

where `hfib f x` is the *homotopy fiber* of a function `f` at `x`:

```
hfib : {A B : Type} → (f : A → B) → B → Type
hfib {A} f b = Σ A (λ a → f a ≡ b)
```

which involves an identity type (we leave to the reader as an exercise to complete the definition of composition in this style and prove basic properties of polynomials).

A definition which is much easier to manipulate can be obtained as follows. The function `t : B → J` above associates to each operation of the polynomial in `B` a typing in `J`. This data can equivalently be encoded as a family of types `Op : J → Type` which to every element `j` of `J` associates the type `Op j` of operations of type `j` of the polynomial. Formally, we are exploiting the equivalence between slices and families in type theory, which is recalled in theorem 9 below. By performing similar transformations on the rest of the data, we reach the following definition, which is easier to work with because it uses more heavily dependent types: the explicit computations we had to perform with equality above are now implicitly handled by the dependent pattern matching of Agda.

Definition 1. The type of *polynomials* between two types `I` and `J` is

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

Given types `I` and `J`, a polynomial from `I` to `J` consists of:

- a family of types `Op j` indexed by the elements `j` of `J`: the *operations* of the polynomial;
- a family of type `Pm i b` indexed by the element `i` of `I` and the operations `b` in `Pm j` for some `j` of `J`: the *parameters* of the polynomial.

The careful reader will note that the above actually defines "polynomials in types", where there is a type (as opposed to a set or a groupoid) of operations and parameters. The notion of polynomial in groupoids can be obtained by further restricting to the case where all the involved types (`I`, `J`, `Op j` and `Pm i b`) are groupoids, in the sense of HoTT recalled above.

B. First constructions on polynomials

The identity polynomial on a type `I` is defined as

```
Id : Poly I I
Op Id i = ⊤
Pm Id i {j = j} tt = i ≡ j
```

and has, for each element `i` of `I`, exactly one operation of type `i`, whose only parameter is also of type `i`. The polynomial functor induced by a polynomial `P` from `I` to `J` is

```
⟦_⟧ : Poly I J → (I → Type) → (J → Type)
⟦_⟧ P X j = Σ (Op P j) (λ c → (i : I) →
                          (p : Pm P i c) → (X i))
```

Finally, the composite `P · Q` of two polynomials `P` from `I` to `J`, and `Q` from `J` to `K` is

```
_·_ : Poly I J → Poly J K → Poly I K
Op (P · Q) = ⟦ Q ⟧ (Op P)
Pm (_·_ P Q) i (c , a) =
  Σ J (λ j → Σ (Pm Q j c) (λ p → Pm P i (a j p)))
```

Morphisms between polynomials can be encoded as follows:

Definition 2. The type of *morphisms* between two polynomials `P` and `Q` is

```
record Poly→ (P Q : Poly I J) : Type where
  field
```

```
Op→ : {j : J} → Op P j → Op Q j
Pm≃ : {i : I} {j : J} {c : Op P j} →
        Pm P i c ≃ Pm Q i (Op→ c)
```

A morphism thus consists of:

- a morphism between the operations of P and those of Q, which respects the typing;
- a morphism between the parameters of P and those of Q, which respects typing and operations, and is moreover an equivalence (requiring this morphism to be an equivalence corresponds precisely to requiring that the square in the middle of (3) is a pullback).

Definition 3. A morphism φ as above is an *equivalence of polynomials* when the morphism on operations is an equivalence at each element of J:

```
Poly-equiv : {P Q : I ⇝ J} (φ : P ⇝₂ Q) → Type
Poly-equiv φ = {j : J} → is-equiv (Op→ φ {j = j})
```

As it can be observed above, we write I ⇝ J (resp. P ⇝₂ Q) for the type of polynomials from I to J (resp. morphisms of polynomials from P to Q). We also write P ≃₂ Q for the type of equivalences of polynomials between P and Q.

C. A bicategory of polynomials

Starting from there we can build all the structure one expects to find in a bicategory of polynomials:

- we can define the identity polynomial and the composition of polynomials (see above);
- we can show that composition of polynomials is associative and unital up to an equivalence of polynomials;
- we can define the horizontal and vertical composition of morphism of polynomials;
- we can show that those compositions are associative and unital up to a suitable notion of equivalence of morphisms of polynomials.

Moreover, by using univalence, one can show the following.

Proposition 4. *The type of equivalences between two polynomials* P *and* Q *is equivalent to the type* P ≡ Q *of equalities between the two polynomials:*

```
(P ≡ Q) ≃ (P ≃₂ Q)
```

We can therefore build a bicategory (it might be more accurate to call it a $(2,1)$-category) of groupoids, polynomials in groupoids and equivalences, in the following sense.

Definition 5. A *prebicategory* consists of:

- a type of ob *objects*;
- for each objects I and J, we have a groupoid hom I J of *morphisms*;
- for each object I there is a distinguished morphism id in hom I I called *identity*;
- there is a composition operation

  ```
  _⊗_ : hom I J → hom J K → hom I K
  ```

  for every objects I, J and K;
- composition is associative and unital:

  ```
  assoc :
    (P : hom I J) (Q : hom J K) (R : hom K L) →
    (P ⊗ Q) ⊗ R ≡ P ⊗ (Q ⊗ R)
  unit-l : (P : hom I J) → id ⊗ P ≡ P
  unit-r : (P : hom I J) → P ⊗ id ≡ P
  ```
- the traditional pentagon law

  ```
  ap (λ P → P ⊗ S) (assoc P Q R) ·
  assoc P (Q ⊗ R) S ·
  ap (λ Q → P ⊗ Q) (assoc Q R S)
  ≡
  assoc (P ⊗ Q) R S ·
  assoc P Q (R ⊗ S)
  ```

  and triangle law

  ```
  assoc P id Q · ap (λ Q → P ⊗ Q) (unit-l Q)
  ≡
  ap (λ P → P ⊗ Q) (unit-r P)
  ```

  of bicategories are satisfied for composable morphisms P, Q, R and S.

Above, "·" denotes the concatenation of paths (or transitivity of equality) and ap is a proof that every function is a congruence for equality.

Theorem 6. *There is a prebicategory whose objects are groupoids and morphisms are polynomials in groupoids.*

The notion of prebicategory generalizes the notion of precategory in HoTT [16, Section 9.1]. As the name suggests, it lacks a property in order to be called a bicategory, similarly to the situation with categories. A morphism P in hom I J in a prebicategory is an *internal equivalence* when there exists morphisms Q and Q' both in hom J I such that P ⊗ Q ≡ id and Q ⊗ P ≡ id, and we write I ≃' J for the type of internal equivalences from I to J. There is a canonical map associating to any equality of type I ≡ J between two objects I and J an internal equivalence from I to J. A *bicategory* is a prebicategory in which this canonical map is an equivalence, i.e., we have

```
(I ≡ J) ≃ (I ≃' J).
```

Theorem 7. *The prebicategory of theorem 6 is a bicategory.*

Note that proposition 4, in addition to proving the above theorem, allows us to use equalities as 2-cells instead of morphisms in the sense of definition 3. Because of this, the usual structure of bicategory which is "missing" from definition 5 (e.g. horizontal and vertical composition of 2-cells, the exchange law, etc.) is automatically present thanks to the general properties of equality. If this was not the case (for instance, if we wanted to consider morphisms of polynomials instead of equivalences as 2-cells), we would have had to use a much more involved notion of bicategory [17].

The notion of being cartesian for such a bicategory can be formalized in the expected way and one can show:

Theorem 8. *The bicategory of theorem 7 is cartesian, with the product being defined by coproduct ⊔ on objects (the groupoids), first projection polynomial being*

```
projl : (I ⊔ J) ⇝ I
Op projl i = ⊤
Pm projl (inl i) {i'} tt = i ≡ i'
Pm projl (inr j) {i'} tt = ⊥
```

(and second projection is similar), and the pairing operation on polynomials being

```
pair : (I ⇝ J) → (I ⇝ K) → I ⇝ (J ⊔ K)
Op (pair P Q) (inl j) = Op P j
Op (pair P Q) (inr k) = Op Q k
Pm (pair P Q) i {inl j} c = Pm P i c
Pm (pair P Q) i {inr k} c = Pm Q i c
```

D. Naive closed structure

A first step toward constructing a right adjoint to the product is the formalization of the naive closure described in section II-C given in [15, `LargePolynomial.agda`]. Namely, the formula (6) indicates that the hom space from `I` to `J` should be

```
(I → Type) × J.
```

Of course, we encounter the same size issues as mentioned in the introduction, and we need to suppose that `Type` is the same as $\texttt{Type}_1$ (i.e., we disable the checking of universe levels), which makes the logic inconsistent, for this proof to go through. The formalization is still useful because it is a simple version of the actual one for the closure, which is more involved but does not require the extra assumption.

We define the *exponential* of a type as

```
Exp : Type → Type₁
Exp I = I → Type
```

so that the internal hom between two types `I` and `J` should be `Exp I × J`. We can indeed define a curryfication map

```
curry : (I ⊔ J) ⇝ K → I ⇝ (Exp J × K)
Op (curry P) (jj , k) =
  Σ (Op P k)
    (λ c → ((λ j → Pm P (inr j) c) ≡ jj))
Pm (curry P) i c = Pm P (inl i) (fst c)
```

and an uncurryfication map

```
uncurry : I ⇝ (Exp J × K) → (I ⊔ J) ⇝ K
Op (uncurry P) k = Σ (Exp J)
                     (λ jj → Op P (jj , k))
Pm (uncurry P) (inl i) (jj , c) = Pm P i c
Pm (uncurry P) (inr j) (jj , c) = jj j
```

and show that they induce an adjunction in the sense that we have

```
((I ⊔ J) ⇝ K) ≡ (I ⇝ (Exp J × K)).
```

An alternative, and sometimes more convenient, definition for exponential can be given as follows. The equivalence (4) between slices and families, see also (7), generalizes in type theory. Given a type `I`, we have an equivalence (and thus an equality by univalence) between types over `I` and families of types indexed by `I`, see [15, `Fam.agda`]:

Theorem 9. *Given a type* `I`*, we have the equivalence*

```
(Σ Type (λ A → A → I)) ≃ (I → Type).
```

This implies that, if we had defined exponential as

```
Exp : Type → Type₁
Exp I = Σ Type (λ A → A → I)
```

we would also be able to show the adjunction result described above.

E. Finite types

Following the plan of section II-C for the proof, we now restrict to finitary polynomials in order to have a smaller exponential in the closure, for which we can handle the size issues. In this section, we first formalize the notion of finiteness for type which will be used to define finitary functors. The proofs associated to this section can be found in [15, `FinType.agda`].

Definition 10. A type is *finite* when it is merely equivalent to the type with n elements for some natural number n:

```
is-finite : Type → Type
is-finite A = Σ ℕ (λ n → ‖ A ≃ Fin n ‖)
```

Above, `Fin n` is the canonical type with n elements (its constructors are 0 up to $n-1$) and `‖ A ‖` is the *propositional truncation* of `A` (an element of this type can be thought of as a witness that there exists a proof of `A`, without explicitly providing such a proof [16, Section 3.7]). The following lemma shows that, for a finite type `A`, there is a well-defined notion of *cardinality* which is the natural number `n` such that `‖ A ≃ Fin n ‖` holds:

Lemma 11. *Given a type* `A`*, if* `‖ A ≃ Fin m ‖` *and* `‖ A ≃ Fin n ‖` *then* `m ≡ n`.

Proof. By transitivity, we should have `Fin m ≃ Fin n` and thus `Fin m ≡ Fin n` by univalence and thus `m ≡ n` by injectivity of the type constructor `Fin`. This last fact is not obvious (not every type constructor is injective) and can be shown by induction on `m` and `n`. □

Using this, one can deduce that being finite is a proper predicate:

Lemma 12. *Being finite for a type is a proposition.*

We write `FinType` for the type of all finite types:

```
FinType : Type₁
FinType = Σ Type is-finite
```

Note that this type is a large one (it lives in $\texttt{Type}_1$), which will give rise to major difficulties in the following. The above lemma ensures that equality of finite types is equivalent to equality of the underlying types and moreover

Lemma 13. `FinType` *is a groupoid.*

Proof. The types `Fin n` being decidable, they are sets by Hedberg's theorem [16, Theorem 7.2.5]. Since a finite type `A`

is equivalent to such a type `Fin n` by definition and being a set is a proposition [16, Lemma 3.3.5], its underlying type is also a set by transport, and therefore `A` is a set by the above remark. □

F. Finitary polynomials

Following definition 34, a functor is *finitary* when, for each operation `c`, the total space of its parameters is finite [15, `FinPolynomial.agda`]:

```
is-finitary : (P : I ⇝ J) → Type
is-finitary P = {j : J} (c : Op P j) →
                is-finite (Σ I (λ i → Pm P i c))
```

Remark 14. Another natural definition of being finitary could be to require that each space of parameters is finite:

```
is-finitary : (P : I ⇝ J) → Type
is-finitary P = (i : I) {j : J} (c : Op P j) →
                is-finite (Pm P i c)
```

but one quickly convinces himself that this notion is not suitable: identity polynomials are not generally finitary in this sense, and being finitary is not stable under composition.

If we restrict to finitary polynomials in groupoids, we expect (this point is not yet fully formalized due to lack of time) that the proofs of section III-D can be refined following section V-C in order to show that the cartesian product admits `Exp I × J` as internal hom, where `Exp` is now defined as

```
Exp : Type → Type₁
Exp I = Σ (I → Type) (λ F → is-finite (Σ I F))
```

i.e., we restrict to families whose total space is finite. Through the equivalence between families and slices (see theorem 9), it can be shown:

Lemma 15. *The above definition of the exponential is equivalent to the following one:*

```
Exp : Type → Type₁
Exp I = Σ FinType (λ N → fst N → I)
```

For the same reasons as in section III-D, the above reasoning requires us to ignore size issues, which forces us working in an inconsistent logic. Again the cause of the problem is the fact that, given a small type `A`, `Exp A` is a large type. If we consider the definition of lemma 15 for the exponential, this is due to the fact that we sum over finite types, and `FinType` is itself a large type. We see however in next section that it is in fact equivalent to a small type, so that the above proof can be modified in order to properly handle size issues.

G. A small axiomatization of finite types

In this section, we show the last missing bit of our construction: the type `FinType` of finite types is equivalent to a small type [15, `Bij.agda`]. This is a generalization of the following simple observation: every finite set is isomorphic to a set of the form $[n] = \{0, \ldots, n-1\}$ for some natural number n, so that the class of finite sets is equivalent to the set of natural numbers. However, `FinType` is a groupoid and not a set, in the sense of HoTT: there are non-trivial equalities between finite types which, by univalence, correspond to isomorphisms of finite types. For this reason, we do not expect that the type `FinType` is equivalent to the traditional type $\mathbb{N}$ of natural numbers, which has no non-trivial path between its element (it has decidable equality and thus is a set by Hedberg's theorem), but rather to a type that we call here $\mathbb{B}$, which has natural numbers as objects, but is moreover such that the group of path endomorphisms on an object `n` is the symmetric group on `n` elements. A more accurate picture of the situation than the equivalence between finite sets and natural numbers is thus the equivalence between the category **Bij** of finite sets an bijections and its skeleton, which has natural numbers as objects.

The type $\mathbb{B}$ being constructed from constructors (the natural numbers), but also paths, it is natural to describe it as a higher inductive type [16, Section 6]. Those are not readily available in current plain version of Agda, but we can simulate those by working axiomatically with them, as done usually. We thus axiomatize $\mathbb{B}$ as the small type such that

- it has natural numbers as objects:
  `obj : ℕ → 𝔹`
- for every equivalence between finite sets there is a path in $\mathbb{B}$ between the corresponding natural numbers:
  ```
  hom : {m n : ℕ} (α : Fin m ≃ Fin n) →
        obj m ≡ obj n
  ```
- the path associated to identity is the identity:
  `id-coh : (n : ℕ) → hom {n = n} ≃-refl ≡ refl`
- paths are compatible with composition:
  ```
  comp-coh : {m n o : ℕ}
    (α : Fin m ≃ Fin n) (β : Fin n ≃ Fin o) →
    hom (≃-trans α β) ≡ hom α · hom β
  ```
- there are no higher paths, i.e., $\mathbb{B}$ is a groupoid:
  `𝔹-is-groupoid : is-groupoid 𝔹`

We also need to postulate an appropriate elimination principle, which can be found in the formalization: it roughly states that, in order to define function of type `f : 𝔹 → A`, for some groupoid `A`, it is enough to define:

- an element `f (obj n)` of `A` for every natural number `n`,
- a path `apd f (hom e) : f (obj m) ≡ f (obj n)` for every equivalence `e : Fin m ≃ Fin n`,

in suitably coherent way. This definition is close to the one performed in [18] in order to define Eilenberg-MacLane spaces in homotopy type theory. In the definition of `hom`, the reader might be surprised that we allow two different cardinalities of finite sets: given an equivalence `Fin m ≃ Fin n`, we necessarily have `m ≡ n`. But this "more general" definition simplifies the proofs in practice.

One of the main contribution of this paper is the following theorem:

Theorem 16. *The large type of finite types and the above type are equivalent:* `FinType ≃ 𝔹`.

Proof. The proof uses the "encode-decode method" introduced to compute the fundamental group of the circle [16, Section 8.1]. Given a natural number `n` and an element `b` of $\mathbb{B}$, we can encode the type of paths of type `obj n ≡ b` in $\mathbb{B}$ as the type `Code n b` defined by induction on `b` (using the elimination rule for $\mathbb{B}$): in particular, for a natural number `m`, we define `Code n (obj m)` to be the type `Fin n ≃ Fin m` of equivalences between `Fin m` and `Fin n`. Namely, we can define two functions

$$f \quad : \quad \texttt{obj n} \equiv \texttt{b} \quad \rightleftarrows \quad \texttt{Code n b} \quad : \quad g$$

such that $g \circ f$ is the identity and $f \circ g$ is then identity on paths of the form `hom e` for `e : Fin n ≃ Fin m`. From there, we can deduce that the canonical function `𝔹-to-Fin` from $\mathbb{B}$ to `FinType` (defined using the elimination principle for $\mathbb{B}$) is an embedding. This map is easily shown to be surjective, and we deduce that it is and equivalence by [16, Theorem 4.6.3]. □

As a byproduct of the above theorem, we have a map `𝔹-to-Fin` from $\mathbb{B}$ to `FinType`. This finally allows us to give the following definition of the exponential

```
Exp : Type → Type
Exp A = Σ 𝔹 (λ b → 𝔹-to-Fin b → A)
```

which remains in small types. The proof of section III-F can be adapted to this definition (because it is equivalent to the one of lemma 15 by theorem 16), resulting in a proof which does not require inconsistent assumptions on universes.

IV. A BICATEGORY OF POLYNOMIAL FUNCTORS IN GROUPOIDS

In this section and the following, we study polynomial functors in groupoids, from a more traditional category-theoretic perspective. We introduce a "fibered variant" of polynomial functors in groupoids and show that it forms a cartesian closed bicategory. We begin by reviewing alternative preexisting approaches in the literature.

A. Homotopy polynomial functors in groupoids

Our aim is to define a notion of polynomial functor in the category of groupoids, which is complete and cocomplete. For a category with pullbacks, any change of base functor Δ_f admits a left adjoint Σ_f. Moreover, being locally cartesian closed is equivalent to requiring the existence of a right adjoint functor Π_f to every change of base functors Δ_f, which we need in order to define composition of polynomials, and this is not the case for the category of groupoids [19] (the morphisms f for which Π_f exists are called *exponentiable*). It is however *weakly* cartesian closed, if we properly take the structure of 2-category of groupoids in account [19].

This allows to define a notion of "weak" or "homotopy" polynomial functors in groupoids: this is the path followed by Kock in [2], which contains a detailed account of what follows. The idea, that we only briefly sketch here, consists in replacing all the constructions of limits and colimits by ones "up to homotopy". For instance, given a diagram

$$\begin{tikzcd} A \times_C B \arrow[r, dotted] \arrow[d, dotted] \arrow[dr, phantom, "\lrcorner", very near start] & B \arrow[d, "g"] \\ A \arrow[r, "f"'] & C \end{tikzcd}$$

in $\mathbf{Set}$, it is well-known that the pullback of f and g is the set

$$A \times_C B = \{(a, b) \in A \times B \mid f(a) = g(b)\}$$

equipped with the obvious projections. In the 2-category $\mathbf{Gpd}$, one can consider the *homotopy pullback* of f and g, whose formula is essentially the same excepting that equality is now replaced by an explicit isomorphism: it is a category whose objects are triples (a, e, b) with $a \in A$, $b \in B$ and $e : fa \to gb$ is a morphism in C. In this way, given a functor $f : I \to J$, one can define a 2-functor $\Delta_f : \mathbf{Gpd}/J \to \mathbf{Gpd}/I$ which is defined on objects by homotopy pullback, and admits a (homotopy) right adjoint Π_f. In this way, we can define a *polynomial in groupoids* as a diagram (2) in $\mathbf{Gpd}$ and adapt the constructions of section II, by suitably replacing all constructions by ones up to homotopy.

As an another approach, instead of generalizing the constructions up to homotopy, we can keep the strict versions of section II, but restrict to polynomials for which those work: this is the approach taken by Weber in [9], see also [10]. The morphisms in $\mathbf{Gpd}$ which are exponentiable are precisely the isofibrations: a functor $F : I \to J$ between groupoids is an *isofibration* when, for every object $i \in I$ and isomorphism $g : Fi \to j$ in J, there is an isomorphism $f : i \to i'$ such that $Ff = g$. In the category of groupoids, isofibrations coincide with Conduché functors (which are exponentiable morphisms in $\mathbf{Cat}$) and with Grothendieck fibrations, and we will simply say *fibrations* for those. Following this observation, we can define a polynomial in $\mathbf{Gpd}$ as a diagram (2) such that both p and t are fibrations: the first one allows the use of Π_p in the definition of polynomials (3) as well as when defining composition of polynomials, and the second one justifies the use of the Beck-Chevalley isomorphism when defining composition. It can be checked that such polynomials are closed under composition. We expect that a comparison between this definition and the previous one can be made by using general arguments of model structures, based the existence of the *canonical model structure* on $\mathbf{Gpd}$ [20], for which weak equivalences are equivalence of categories and fibrations are fibrations in the above sense (in particular, this model category is right proper, from which follows that strict pullbacks along fibrations coincide with homotopy pullbacks). We leave this for future works though.

In this paper, we introduce a third possible definition of polynomials, where the data is not organized in the same way as usual. It is well-known that the equivalence (4) generalizes as an equivalence of 2-categories

$$\mathrm{Fib}(I) \quad \simeq \quad [I, \mathbf{Gpd}]_{\mathrm{pseudo}} \tag{7}$$

between fibrations over a groupoid I and pseudofunctors from I to groupoids (as a variant there is also an equivalence between split fibrations over I and functors from I to groupoids), see for instance [21, Chapter 12], where the left-to-right functor is given by the Grothendieck construction (see below). This means that when defining a polynomial (2), we can equivalently encode the data of the morphism $t : B \to J$ as a functor $t : J \to \mathbf{Gpd}$, and similarly for other morphisms. Our definition is detailed in section IV-C and, again, we leave the detailed comparison with previous ones for later. There are many possible small variants on this definition (using pseudofunctors or functors, fibrations which are split or not, etc.), and this one is the only one for which we were able to construct the closure to the cartesian product, for interesting technical reasons. It is however not customary to detail failed attempts in papers.

B. Preliminaries on groupoids and 2-categories

Let us fix some notations first. We denote isomorphisms of categories by $\cong$ and equivalences by $\simeq$. We recall that a category is a groupoid when every morphism is invertible and we write $\mathbf{Gpd}$ for their (2-)category. We write $\top$ for the terminal groupoid and $\star$ for its only object, and $\bot$ for the initial groupoid.

Definition 17. Given a functor $F : I \to \mathbf{Gpd}$ for some groupoid I, the *dependent sum* $\sum_{i:I} F_i$ is the groupoid whose objects are pairs (i, x) with i an object of I and x an object of F_i, and such that a morphism $(i, x) \to (j, y)$ is the data of an arrow $f : i \to j$ in I, and an arrow $g : F_f(x) \to y$ in F_j. The category $\sum_{i:I} F_i$ is also known as the *Grothendieck construction* of F.

Given a 2-category $\mathcal{C}$, we write:

- $\mathcal{C}^{\simeq}$ for the sub-2-category of $\mathcal{C}$ whose 1-cells are restricted to equivalences and 2-cells to isomorphisms;
- $|\mathcal{C}|_1$ for the *1-truncation* of $\mathcal{C}$ which is the category with the same objects as $\mathcal{C}$ and morphisms are obtained from 1-cells of $\mathcal{C}$ by quotienting by isomorphism 2-cells.

Lemma 18. *We have the following properties of 1-truncation:*

- *the operation $|-|_1$ extends as a functor from 2-categories to categories;*
- *given a 2-category $\mathcal{C}$, the category $|\mathcal{C}^{\simeq}|_1$ is a groupoid;*
- *the 1-truncations of two 2-equivalent 2-categories are equivalent;*
- *1-truncation preserves products.*

The traditional notion of slice category generalizes as follow in the 2-categorical setting.

Definition 19. Given a 2-category $\mathcal{C}$, and X an object of $\mathcal{C}$, the *slice 2-category* $\mathcal{C}/\!\!/X$ is such that

- a 0-cells is a pair of an object A of $\mathcal{C}$ and a morphism $f_A : A \to X$,
- a 1-cell from (A, f_A) to (B, f_B) is a morphism $u : A \to B$ and a 2-isomorphism $\alpha : f_B \circ u \Rightarrow f_A$,
- a 2-cell from (u, α) to (v, β) is a 2-cell from u to v such that the appropriate diagram commutes.

Lemma 20. *Supposing that $\mathcal{C}$ has coproducts, given two objects X and Y, we have $\mathcal{C}/\!\!/(X \sqcup Y) \cong \mathcal{C}/\!\!/X \times \mathcal{C}/\!\!/Y$.*

Lemma 21. *Given a functor $F : \mathcal{C} \to \mathcal{D}$ which is a 2-equivalence of 2-categories and an object X of $\mathcal{C}$, we have an equivalence of slice categories: $\mathcal{C}/\!\!/X \simeq \mathcal{D}/\!\!/FX$.*

Given a functor $F : \mathcal{C} \to \mathcal{D}$ between 2-categories and an object X of $\mathcal{D}$, we can similarly define a notion of *comma category* $F/\!\!/X$ which generalizes the previous definition (a slice category is a comma category over the identity functor), and the above lemmas generalize to those.

C. A fibered view of polynomial functors in groupoids

We can now introduce our definition of polynomial functor in groupoids, which appropriately reformulates in the language of category theory the type theoretic definition 1. Given a groupoid I, we define its *exponential* as the groupoid

$$!I = |\mathbf{Gpd}^{\simeq}/\!\!/I|_1$$

this can be thought of as acting as the type of groupoids over I, except that we need to truncate the resulting 2-category in order to get back to groupoids, see also remark 25. It can be shown to satisfy the following "Seely isomorphism", using the properties of section IV-B:

Lemma 22. *Given groupoids I and J, we have*

$$!(I \sqcup J) \cong !I \times !J.$$

Definition 23. A *fibered polynomial* in groupoids from I to J is the data of:

- an *operations* functor $\mathrm{Op}[P] : J \to \mathbf{Gpd}$, and
- a *parameters* functor $\mathrm{Pm}[P] : \sum_{j:J} \mathrm{Op}[P]_j \to !I$.

Unpacking the definition, the data of $\mathrm{Pm}[P]$ decomposes into three components:

- a functor $\mathrm{Pm}[P] : \sum_{j:J} \mathrm{Op}[P]_j \to \mathbf{Gpd}$,
- for any object $b \in \mathrm{Op}[P]_j$, a functor $\mathrm{s}^P_b : \mathrm{Pm}[P]_b \to I$
- for any morphism $u : (j, b) \to (j', b')$ in $\mathrm{Op}[P]$, a natural transformation $\alpha^P_u : \mathrm{s}^P_{b'} \circ \mathrm{Pm}[P]_u \Rightarrow \mathrm{s}^P_b$:

$$\begin{tikzcd}
\mathrm{Pm}[P]_b \arrow[rr, "\mathrm{Pm}[P]_u"] \arrow[dr, "\mathrm{s}^P_b"'] & \arrow[d, phantom, "\overset{\alpha^P_u}{\Longleftarrow}"] & \mathrm{Pm}[P]_{b'} \arrow[dl, "\mathrm{s}^P_{b'}"] \\
& I &
\end{tikzcd}$$

We note here that every polynomial in groupoids in our sense induces one in the sense of [10], and leave detailed comparison for future work.

Lemma 24. *Any fibered polynomial induces a polynomial in groupoids*

$$I \xleftarrow{\sum_{j:J} \sum_{b:\mathrm{Op}[P]_j} \mathrm{s}^P_b} \sum_{j:J} \sum_{b:\mathrm{Op}[P]_j} \mathrm{Pm}[P]_b \xrightarrow{\pi_1} \sum_{j:J} \mathrm{Op}[P]_j \xrightarrow{\pi_1} J$$

where both projections π_1 are fibrations.

Remark 25. Let us briefly discuss the definition of $!I$. In view of the above lemma, if we had chosen to work with the 1-categorical slice construction $\mathbf{Gpd}/I$ instead of the 2-slice $\mathbf{Gpd}/\!\!/I$, it would have resulted in a much stricter notion of polynomials than traditional ones [10]. Namely, it would impose that, in a polynomial (2), the arrow $E \to I$ sends lifts of isomorphisms of B to identities in I.

The reason why we have to truncate can be seen as an instance of the phenomenon explained in the introduction, but one dimension higher: our polynomials canonically form a tricategory, but the closure is only correct up to dimension 2.

The counterpart of definition 2 for morphisms is the following.

Definition 26. A *morphism* between fibered polynomials $P \nrightarrow Q : I \rightsquigarrow J$ consists of two natural transformations

$$\beta : \mathrm{Op}[P] \Rightarrow \mathrm{Op}[Q] \qquad \varepsilon : \mathrm{Pm}[Q] \circ (\sum_{j:J} \beta_j) \Rightarrow \mathrm{Pm}[P]$$

$$\begin{tikzcd}
 & \sum_{j:J} \mathrm{Op}[P]_j \arrow[dl, "\mathrm{Pm}[P]"'] \arrow[dr, "\pi_1"] \arrow[dd, "\sum_{j:J} \beta_j"] & \\
|\mathbf{Gpd}/\!\!/I|_1 \arrow[rr, phantom, "\xleftarrow{\varepsilon}"] & & J \\
 & \sum_{j:J} \mathrm{Op}[Q]_j \arrow[ul, "\mathrm{Pm}[Q]"] \arrow[ur, "\pi_1"'] &
\end{tikzcd}$$

D. Bicategorical structure

In this section, we focus on fibered polynomials, which we simply call polynomials and detail their structure of bicategory.

1) Horizontal composition:

Definition 27. For any groupoid I, the *unit polynomial* $\mathrm{id}_I : I \rightsquigarrow I$ is given by $\mathrm{Op}[\mathrm{id}]_i = \top$, and $\mathrm{Pm}[\mathrm{id}]$ is given by the following data:

- the functor $\mathrm{Pm}[\mathrm{id}] : \sum_{i:I} \mathrm{Op}[\mathrm{id}] \to \mathbf{Gpd}$ is the constant functor equal to $\top$;
- for any object $i \in I$, $\mathrm{s}^{\mathrm{id}} : \top \to I$ sends $\star$ to i;
- for any morphism $u : i \to i'$ in I, $\alpha^{\mathrm{id}}_u = u^{-1}$.

$$\begin{tikzcd}
\top \arrow[rr] \arrow[dr, "i"'] & \arrow[d, phantom, "\xleftarrow{u^{-1}}"] & \top \arrow[dl, "i'"] \\
 & I &
\end{tikzcd}$$

Definition 28. Given two polynomials $P : I \rightsquigarrow J$ and $Q : J \rightsquigarrow K$ their *composition* $Q \circ P : I \rightsquigarrow K$ is defined as follows:

- Operations are given by
$$\mathrm{Op}[Q \circ P]_k = \sum_{b:\mathrm{Op}[Q]_k} [\mathrm{Pm}[Q]_b, \sum_{j:J} \mathrm{Op}[P]_j],$$
where $[\_,\_]$ denotes the category of morphisms in $!I$;
- An object of $\mathrm{Op}[Q \circ P]_k$ is therefore a triple $(b, f\alpha)$ where $b \in \sum_{k:K} \mathrm{Op}[Q]_k$, $f : \mathrm{Pm}[Q]_b \to \sum_{j:J} \mathrm{Op}[P]_j$, and $\alpha : \pi_1 \circ f \Rightarrow \mathrm{s}^Q$; the groupoid $\mathrm{Pm}[Q \circ P]_{b,f,\alpha}$ is given by:
$$\mathrm{Pm}[Q \circ P]_{b,f,\alpha} = \sum_{x:\mathrm{Pm}[Q]_b} \mathrm{Pm}[P]_{f(x)}.$$
- The arrow $\mathrm{s}^{Q\circ P}_{b,f,\alpha} : \mathrm{Pm}[Q \circ P] \to I$ sends a pair (x, y) to $\mathrm{s}^P_{f(x)}(y)$. A morphism $(x, y) \to (x', y')$ in $\mathrm{Pm}[Q\circ P]_{b,f,\alpha}$ is a pair (u, v), with $u : x \to x'$ in $\mathrm{Pm}[Q]_b$, and $v : \mathrm{Pm}[P]_{f(u)}(y) \to y'$ in $\mathrm{Pm}[P]_{f(x')}$. The image of (u, v) under $\mathrm{s}^{Q\circ P}_{b,f,\alpha}$ is the composite:
$$\mathrm{s}^P_{f(x)}(y) \xrightarrow{\alpha^{P\ -1}_{u,y}} \mathrm{s}^P_{f(x')}(\mathrm{Pm}[P]_{f(u)}(y)) \xrightarrow{\mathrm{s}^P_{f(x')}(v)} \mathrm{s}^P_{f(x')}(y)$$
- We now describe the action of $\mathrm{Pm}[Q \circ P]$ on morphisms. A morphism $(b, f, \alpha) \to (b', f', \alpha')$ in $\mathrm{Op}[Q \circ P]$ is the data of a pair (u, β) , where $u : b \to b'$ is a morphism in $\mathrm{Op}[Q]$, and $\beta : f' \circ \mathrm{Pm}[Q]_u \Rightarrow f$ is a natural transformation such that $(\alpha' \circ \mathrm{Pm}[Q]_u) * \alpha^Q_u = (\mathrm{t}^P \circ \beta) * \alpha$ (those natural transformations go from $\mathrm{t}^P \circ f' \circ \mathrm{Pm}[Q]_u$ to s^Q_b).
We define $\mathrm{Pm}[Q \circ P]_{u,\beta} : \mathrm{Pm}[P]_{b,f,\alpha} \to \mathrm{Pm}[P]_{b',f',\alpha'}$ as the functor sending a pair (x, y) to $(\mathrm{Pm}[Q]_u(x), \mathrm{Pm}[P]_{\beta_x^{-1}}(y))$.
- It remains to define $\alpha^{Q\circ P}_{u,\beta} : \mathrm{s}^{Q\circ P}_{b',f',\alpha'} \circ \mathrm{Pm}[Q\circ P]_{u,\beta} \Rightarrow \mathrm{s}^{Q\circ P}_{b,f,\alpha}$. Computing these functors for any (x, y), we see $\alpha^{Q\circ P}_{u,\beta,(x,y)}$ should have source $\mathrm{s}^P_{f'(\mathrm{Pm}[Q]_u(x))}(\mathrm{Pm}[P]_{\beta_x^{-1}}(y))$, and target $\mathrm{s}^P_{f(x)}(y)$. We define $\alpha^{Q\circ P}_{u,\beta,(x,y)} = \alpha^P_{\beta_x^{-1}}$.

$$\begin{tikzcd}
\sum_{x:\mathrm{Pm}[Q]_b} \mathrm{Pm}[P]_{f(x)} \arrow[rr, "\mathrm{Pm}[Q\circ P]_{u,\beta}"] \arrow[dr, "\mathrm{s}^{Q\circ P}_{b,f,\alpha}"'] & \arrow[d, phantom, "\xleftarrow{\alpha^P_{\beta_x^{-1}}}"] & \sum_{x:\mathrm{Pm}[Q]_{b'}} \mathrm{Pm}[P]_{f'(x)} \arrow[dl, "\mathrm{s}^{Q\circ P}_{b',f',\alpha'}"] \\
 & I &
\end{tikzcd}$$

Definition 29. Let $(\varepsilon, \beta), (\varepsilon', \beta') : P \nrightarrow Q$ be two morphisms of fibered polynomials, and let $\mu : \beta \to \beta'$ be a modification (that is, a family of natural transformations $\mu_b : \beta_b \Rightarrow \beta'_b$ for $b \in \mathrm{Op}[P]$). Then μ induces a natural transformation $\sum_{j:J} \mu_b : \sum_{j:J} \beta_j \Rightarrow \sum_{j:J} \beta'_j$.

A 2-equivalence between fibered polynomial is the data of such a modification μ such that the composite of μ and ε equals ε'.

Proposition 30. *The composition defined above defines a bicategory* **fPoly** *whose:*

- *objects are groupoids;*
- *morphisms are fibered polynomials;*
- *2-cells are morphisms of fibered polynomials, quotiented by 2-equivalences.*

Proof. See appendix B. □

Remark 31. Although the computations performed here and below are quite similar to the ones performed in Agda in section III, we would like to point out that the comparison between the two is not obvious, because they do not construct bicategories in the same sense. For instance, in our type-theoretical notion of bicategory, we have a *type* of objects, whereas traditional bicategories have a set of objects, so that it is not clear whether we can make our formalization apply to usual bicategories.

V. Cartesian closed structure structure

A. Cartesian structure

The cartesian product in $\mathbf{fPoly}$ is written $\oplus$ in order to distinguish it from the cartesian product $\times$ of groupoids. We define it on objects I and J as the coproduct $I \oplus J = I \sqcup J$ of groupoids. We write $\iota_1 : I \to I \sqcup J$ and $\iota_2 : J \to I \sqcup J$ for the canonical injections. The first projection polynomial $\Pi_1^{I,J} : I \sqcup J \rightsquigarrow I$ is defined as follows:

- the operations functor $\mathrm{Op}[\Pi_1^{I,J}] : I \to \mathbf{Gpd}$ is the constant functor equal to $\top$;
- the parameters functor $\mathrm{Pm}[\Pi_1^{I,J}] : I \times \top \to \mathbf{Gpd}$ is the constant functor equal to $\top$, and the functor $s_i : \mathrm{Pm}[\Pi_1^{I,J}]_i \to I \sqcup J$ sends $\star$ to i;
- for any morphism $u : i \to i'$ in I, $\alpha_u^{\mathrm{id}} = u^{-1}$.

$$\begin{tikzcd} \top \arrow[rr] \arrow[dr, "i"'] & \arrow[d, phantom, "\overset{u^{-1}}{\Longleftarrow}"] & \top \arrow[dl, "i'"] \\ & I & \end{tikzcd}$$

The second projection $\Pi_2^{I,J} : I \sqcup J \rightsquigarrow I$ is defined similarly.

Given two polynomials $P : I \to J$ and $Q : I \to K$, their pairing $\langle P, Q\rangle : I \to (J \sqcup K)$ is defined as follows.

- $\mathrm{Op}[\langle P, Q\rangle] : J \sqcup K \to \mathbf{Gpd}$ is $\mathrm{Op}[\langle P, Q\rangle]_{\iota_1 j} = \mathrm{Op}[P]_j$, and $\mathrm{Op}[\langle P, Q\rangle]_{\iota_2 k} = \mathrm{Op}[Q]_k$.
- Notice that there is an isomorphism:
$$\sum_{x:J\sqcup K} \mathrm{Op}[\langle P, Q\rangle] \cong \sum_{j:J} \mathrm{Op}[P]_j \sqcup \sum_{k:K} \mathrm{Op}[Q]_k,$$
and therefore we can define $\mathrm{Pm}[\langle P, Q\rangle]$ as the copairing $\{\mathrm{Pm}[P], \mathrm{Pm}[Q]\}$.

Proposition 32. *The operations defined above induce a categorical product.*

Proof. We prove that using the equational definition of products. I.e. we show that for polynomials $P_1 : I \rightsquigarrow J_1$ and $P_2 : I \rightsquigarrow J_2$, the pairing $\langle P_1, P_2\rangle : I \rightsquigarrow J_1 \sqcup J_2$ satisfies:

- $\Pi_i \circ \langle P_1, P_2\rangle \cong P_i$;
- $\langle \Pi_1 \circ P, \Pi_2 \circ P\rangle \cong P$ for $P : I \rightsquigarrow J_1 \sqcup J_2$.

Details of the computations are given in appendix C. □

B. Naive closed structure

Leaving size issues aside, we study here a "naive" closure for the cartesian product, which is akin to the one defined at the end of section III-D: given two groupoids I and J, we define a large groupoid $[I, J]$ which satisfies the usual equivalence required for a right adjoint to the cartesian product. This proof will be refined in next section in order to handle size issues formally.

We claim that the closure to the product can be obtained as

$$[I, J] = !I \times J.$$

Given a polynomial $P : I \sqcup J \rightsquigarrow K$, its parameters can be seen as a functor

$$\mathrm{Pm}[P] : \sum_{j:J} \mathrm{Op}[P]_j \to !I \times !J,$$

by post-composition with the isomorphism lemma 22. We define its transpose $P^\sharp : I \rightsquigarrow !J \times K$ as follows, where $\mathbf{Iso}_{\mathcal{C}}$ denotes the maximal subgroupoid of a category $\mathcal{C}$:

- operations $\mathrm{Op}[P^\sharp] : !J \times K \to \mathbf{Gpd}$ are defined by
$$\mathrm{Op}[P^\sharp]_{f,k} = \sum_{b:\mathrm{Op}[P]_k} \mathbf{Iso}_{!J}(\pi_2(\mathrm{Pm}[P]_{k,b}), f);$$
- the parameters of $P^\sharp$ are defined as the composite of
  - the canonical inclusion
  $$\textstyle\sum_{(f,k):!J\times K} \mathrm{Op}[P^\sharp]_{f,k} \hookrightarrow \sum_{j:J} \mathrm{Op}[P]_j;$$
  - the parameters $\mathrm{Pm}[P] : \sum_{j:J} \mathrm{Op}[P]_j \to !I \times !J$;
  - the projection $\pi_1 : !I \times !J \to !I$.

Conversely, to a polynomial $Q : I \rightsquigarrow !J \times K$, we associate a polynomial $Q^\flat : I \sqcup J \rightsquigarrow K$ as follows:

- operations $\mathrm{Op}[Q^\flat] : K \to \mathbf{Gpd}$ are given by
$$\mathrm{Op}[Q^\flat]_k = \sum_{f:!J} \mathrm{Op}[Q]_{f,k};$$
- parameters $\mathrm{Pm}[Q^\flat] : \sum_{k:K} \mathrm{Op}[Q^\flat]_k \to !(I \sqcup J)$ are defined, up to the isomorphism of lemma 20, by
$$\mathrm{Pm}[Q^\flat]_{k,(f,b)} = (\mathrm{Pm}[Q]_{k,b}, f).$$

Proposition 33. *Given polynomials $P : I \sqcup J \rightsquigarrow K$ and $Q : I \rightsquigarrow !J \times K$, we have $(P^\sharp)^\flat \simeq P$ and $(Q^\flat)^\sharp \simeq Q$.*

C. A cartesian closed bicategory

In order to avoid the size issues of above approach, we further restrict to finitary polynomial functors in the following. Given a natural number n, recall that we write $[n]$ for the set $n = \{0, \ldots, n-1\}$. A groupoid is said to be *finite* when it is merely equivalent to the set $[n]$, seen as a discrete groupoid, for some $n \in \mathbb{N}$. We write $\mathbf{Gpd}_{\mathrm{fin}}$ for the full subcategory of $\mathbf{Gpd}$ whose objects are finite groupoids.

Definition 34. A polynomial functor $P : I \rightsquigarrow J$ is *finitary* if for any operation $b \in \mathrm{Op}[P]_j$, the groupoid which is the source of $\mathrm{Pm}[P]_{j,b} \in \mathbf{Gpd}/\!\!/I$ is finite.

Proposition 35. *Finitary polynomials form a subcartesian bicategory of fibered polynomials.*

We write $\mathbb{B}$ the groupoid whose objects are natural numbers, and such that a morphism $m \to n$ is a bijective function $[m] \to [n]$. The groupoid $\mathbb{B}$ satisfies that $\mathbb{B}(m, n) = \emptyset$ for $m \neq n$ and the group of endomorphisms on an object n is the symmetric group on n elements. This groupoid provides a small model of the category of finite groupoids in the sense that it is small and equivalent to it:

Proposition 36. *There is a 2-equivalence of 2-categories $\mathbf{Gpd}_{\mathrm{fin}}^{\simeq} \simeq \mathbb{B}$.*

Proof. There is an inclusion functor $F : \mathbb{B} \to \mathbf{Gpd}_{\mathrm{fin}}^{\simeq}$ sending an object n to the discrete groupoid $[n]$ and a bijection to the corresponding functor. This functor is essentially surjective (every finite groupoid is equivalent to the discrete groupoid $[n]$ where n is the number of its

isomorphism classes of objects) and fully faithful (the functor $F_{m,n} : \mathbb{B}(m,n) \to \mathbf{Gpd}^{\simeq}_{\mathrm{fin}}(Fm, Fn)$ is an equivalence of groupoids). Therefore, assuming the axiom of choice, it is an equivalence of (2,1)-categories. $\square$

By lemmas 18 and 21, we immediately deduce:

Corollary 37. *We have an equivalence* $!I \simeq \mathbb{B}/I$.

This finally allows to show that our bicategory is cartesian closed, properly taking size issues in account.

Theorem 38. *The cartesian bicategory of finitary polynomials is closed with exponentials being given by* $[I, J] = \mathbb{B}/\!\!/ I \times J$.

VI. Future work

In this work, we have constructed a bicategory of polynomial functors in groupoids, both in the setting of type theory, and the more traditional one of (set-theoretic) category theory. As mentioned in remark 31, we would like to explore further the links between the two: we expect that the second can be deduced from the first, although the way of formally doing this so eludes us at the moment.

A second aspect in which we wish to push investigations on this model is its relation with linear logic. Namely, the bicategory of spans in groupoids can be understood as the full subbicategory of polynomials as in Eq. (2) where the morphism p is the identity. This subbicategory is monoidal with the tensor induced on objects by cartesian product of groupoids. The inclusion functor admits a left adjoint which is a strong lax monoidal functor. This allows to model the exponential of linear logic [22] and will be detailed elsewhere. We also expect that the constructions of differential linear logic can be interpreted in the model.

Finally, we already mentioned that our model is close to the one of generalized species, which is also a model of (differential) linear logic [11], [13] inspired by combinatorial species [23]. We would like to understand the relationship between those two models, which is hinted at in [2, Section 3.9]: one of the main difference is that whereas generalized species are be composed using traditional composition of profunctors, which involves a quotient, homotopy polynomial functors perform a homotopy quotient, and we should be able to obtain the first from the second by suitably discarding homotopical information (i.e., "taking π_0").

On the long term, we finally want to investigate the vast generalization to polynomial in ∞-groupoids, as first studied in [3]. This would make more transparent the comparison with the type theoretic formalization, but we expect the study of this model to be much more involved on a technical level.

References

[1] P. functors and polynomial monads, "Polynomial functors and polynomial monads," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 154, no. 1. Cambridge University Press, 2013, pp. 153–192.

[2] J. Kock, "Data types with symmetries and polynomial functors over groupoids," *Electronic Notes in Theoretical Computer Science*, vol. 286, pp. 351–365, 2012.

[3] D. Gepner, R. Haugseng, and J. Kock, "∞-Operads as Analytic Monads," *arXiv preprint arXiv:1712.06469*, 2017.

[4] J. Kock, A. Joyal, M. Batanin, and J.-F. Mascari, "Polynomial functors and opetopes," *Advances in Mathematics*, vol. 224, no. 6, pp. 2690–2737, 2010.

[5] N. Arkor and M. Fiore, "Algebraic models of simple type theories: A polynomial approach," in *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, 2020, pp. 88–101.

[6] J.-Y. Girard, "Normal functors, power series and λ-calculus," *Annals of pure and applied logic*, vol. 37, no. 2, pp. 129–177, 1988.

[7] R. Hasegawa, "Two applications of analytic functors," *Theoretical Computer Science*, vol. 272, no. 1-2, pp. 113–175, 2002.

[8] P. Taylor, "Quantitative domains, groupoids and linear logic," in *Category Theory and Computer Science*. Springer, 1989, pp. 155–181.

[9] M. Weber, "Polynomials in categories with pullbacks," *Theory and applications of categories*, vol. 30, no. 16, pp. 533–598, 2015.

[10] J. Vidmar, "Polynomial functors and W-types for groupoids," Ph.D. dissertation, University of Leeds, 2018.

[11] M. Fiore, "Generalised species of structures: Cartesian closed and differential structure," 2004.

[12] M. P. Fiore, "Differential structure in models of multiplicative biadditive intuitionistic linear logic," in *International Conference on Typed Lambda Calculi and Applications*. Springer, 2007, pp. 163–177.

[13] M. Fiore, N. Gambino, M. Hyland, and G. Winskel, "The cartesian closed bicategory of generalised species of structures," *Journal of the London Mathematical Society*, vol. 77, no. 1, pp. 203–220, 2008.

[14] M. Fiore, "Analytic functors between presheaf categories over groupoids," *Theoretical Computer Science*, vol. 546, pp. 120–131, 2014.

[15] 2021, `https://anonymous.4open.science/r/d08fc83a-f154-42c4-a976-c41f`

[16] T. Univalent Foundations Program, *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study: `https://homotopytypetheory.org/book`, 2013.

[17] B. Ahrens, D. Frumin, M. Maggesi, and N. van der Weide, "Bicategories in Univalent Foundations," in *4th International Conference on Formal Structures for Computation and Deduction*, 2019.

[18] D. R. Licata and E. Finster, "Eilenberg-MacLane spaces in homotopy type theory," in *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2014, pp. 1–9.

[19] E. Palmgren, "Groupoids and local cartesian closure," *Department of Mathematics Technical Report*, vol. 21, 2003.

[20] R. Brown, "Fibrations of groupoids," *Journal of Algebra*, vol. 15, no. 1, pp. 103–132, 1970.

[21] M. Barr and C. Wells, *Category theory for computing science*. Prentice Hall New York, 1990, vol. 49.

[22] P.-A. Mellies, "Categorical semantics of linear logic," *Panoramas et syntheses*, vol. 27, pp. 15–215, 2009.

[23] A. Joyal, "Foncteurs analytiques et especes de structures," in *Combinatoire énumérative*. Springer, 1986, pp. 126–159.

Appendix

A. Dependent sum and products of groupoids

In this section we define and study the dependent product $\prod$ of groupoids, which will be useful in the next sections, in particular in order to give an equivalent definition of the composition of polynomials which is more amenable to computations.

Definition 39. The *dependent product* of F is the following groupoid.

- Objects of $\prod_{i:I} F_i$, denoted c are families of objects c_i in F_i indexed by $i \in I$, and arrows $c_\alpha : F_\alpha(c_i) \to c_j$ for any arrow $\alpha : i \to j$ in I, such that $c_{\mathrm{id}_i} = \mathrm{id}_{c_i}$ and $c_{\alpha\circ\beta} = F_\beta(c_\alpha) \circ c_\beta$.
- A morphism $f : c \to d$ is the data, for any object $i \in I$, of an arrow $f_i : c_i \to d_i$, such that for any $\alpha : i \to j$ we have $f_j \circ c_\alpha = d_\alpha \circ F_\alpha f_i$.

Lemma 40. *The constructions $\sum$ and $\prod$ are functorial, both in I and in $F : I \to \mathbf{Gpd}$:*

- *Any functor $f : J \to I$ induces a functor $\sum_{j:J} F_{f(j)} \to \sum_{i:I} F_i$, which is an isormorphism (resp. an equivalence) whenever f is, and similarly for $\prod$.*
- *Any natural transformation $\alpha : F \Rightarrow G$ induces a functor $\sum_{i:I} F_i \to \sum_{i:I} G_i$, which is an isormorphism (resp. an equivalence) whenever α is, and similarly for $\prod$.*

Lemma 41. *The following isomorphisms hold:*

- *For any groupoids I and J, $[I, J] \cong \prod_{i:I} J$.*
- *For any two functors $f, g : I \to J$ between groupoids, $[f, g] \cong \prod_{i:I} J(fi, gi)$, where $[f, g]$ denotes the set of natural transformations from f to g.*
- *For any two functors $f : \mathcal{C} \to I$ and $f : \mathcal{D} \to I$,*
$$[f, g] \cong \sum_{h:\mathcal{C}\to\mathcal{D}} \prod_{x:\mathcal{C}} I(g(h(x)), f(x)),$$
where $[f, g]$ denotes the homomorphisms from f to g in $\mathbf{Gpd}/\!\!/I$.
- Associativity. *Let $F : I \to \mathbf{Gpd}$ be a functor, and $G : \sum_{i:I} F_i \to \mathbf{Gpd}$ be a functor. Then*
$$\sum_{i,x:\sum_{i:I} F_i} G_{i,x} \cong \sum_{i:I} \sum_{x:F_i} G_{i,x}$$
$$\prod_{i,x:\sum_{i:I} F_i} G_{i,x} \cong \prod_{i:I} \prod_{x:F_i} G_{i,x}$$
- Distributivity. *Let $F : I \to \mathbf{Gpd}$ be a functor, and $G : \sum_{i:I} F_i \to \mathbf{Gpd}$ be a functor. Then*
$$\prod_{i:I} \sum_{x:F_i} G_{i,x} \cong \sum_{\phi:\prod_{i:I} F_i} \prod_{i:I} G_{i,\phi i}$$
- *For any functor $F : I_1 \sqcup I_2 \to \mathbf{Gpd}$,*
$$\sum_{i:I_1\sqcup I_2} F_i \cong \sum_{i_1:I_1} F_{i_1} \sqcup \sum_{i_2:I_2} F_{i_2}$$
$$\prod_{i:I_1\sqcup I_2} F_i \cong \prod_{i_1:I_1} F_{i_1} \times \prod_{i_2:I_2} F_{i_2}$$

Lemma 42. *The following equivalences hold:*

- *For any object i of a groupoid I, $\Sigma_{j:I} I(i, j) \simeq \top$*
- *For any object i of a groupoid I and functor $F : \sum_{j:I} I(i, j) \to \mathbf{Gpd}$,*
$$\prod_{j:I} \prod_{x:I(i,j)} F_{j,x} \simeq F_{i,x}.$$
In particular, for any functor $F : I \to \mathbf{Gpd}$, $\prod_{j:I}[I(i, j), F_j] \simeq F_i$.
- *For any functors $f : \mathcal{C} \to I$ and $B : I \to \mathbf{Gpd}$,*
$$[f, \sum_{i:I} B_i] \simeq \prod_{x:\mathcal{C}} B_{f(x)}$$

B. Proof of Proposition 30

a) Left unit: Let $P : I \rightsquigarrow J$ be a polynomial. Let us define an iso $\mathrm{id}_J \circ P \Rightarrow P$. For that we start by the equivalence $\beta : \mathrm{Op}[\mathrm{id} \circ P] \Rightarrow \mathrm{Op}[P]$. We have, for any object $j : J$:

$$\begin{aligned}\mathrm{Op}[\mathrm{id} \circ P]_j &= \sum_{\star:\top} [\top \to_j J, \sum_{j:J} \mathrm{Op}[P]_j] \\ &\cong [\top \to_j J, \sum_{j:J} \mathrm{Op}[P]_j]\end{aligned}$$

An object of $\mathrm{Op}[\mathrm{id} \circ P]_j$ is therefore the data of an element $(j', b) \in \sum_{j:J} \mathrm{Op}[P]_j$, and an isomorphism $u : j' \to j$. We define $\beta_j(j', b, u) = \mathrm{Op}[P]_u(b)$. Its inverse is given by $\beta_j^{-1} b = (j, b, \mathrm{id}_j)$. This is indeed an equivalence because u induces an isomorphism $(j', b, u) \cong (j, \mathrm{Op}[P]_u b, \mathrm{id})$.

The components of the equivalence ε are functors $\varepsilon_{j,(j',b,u)} : \mathrm{Pm}[\mathrm{id} \circ P]_{j,(j',b,u)} \to \mathrm{Pm}[P]_{j,\mathrm{Op}[P]_u(b)}$ over I. Note that we have an isomrphism:

$$\mathrm{Pm}[\mathrm{id} \circ P]_{j,(j',b,u)} = \sum_{\star:\top} \mathrm{Pm}[P]_{j',b} \simeq \mathrm{Pm}[P]_{j',b},$$

and so we can choose

$$\varepsilon_{j,(j',b,u)} = \mathrm{Pm}[P]_{u,\mathrm{id}} : \mathrm{Pm}[P]_{j',b} \to \mathrm{Pm}[P]_{j,\mathrm{Op}[P]_u b}.$$

The fact that these isomorphisms live over I is given by $\alpha^P_{u,\mathrm{id}}$.

b) Right unit: Let $P : I \rightsquigarrow J$ be a polynomial. Let us define an iso $P \circ \mathrm{id}_I \Rightarrow P$. For that we start by a family of functors $\beta_j : \mathrm{Op}[P \circ \mathrm{id}]_j \to \mathrm{Op}[P]_j$. We first notice that:

$$\begin{aligned}\mathrm{Op}[P \circ \mathrm{id}]_j &= \sum_{b:\mathrm{Op}[P]_j} [\mathrm{Pm}[P]_b, \sum_{j:J} \top] \\ &\cong \sum_{b:\mathrm{Op}[P]_j} [\mathrm{Pm}[P]_b, J]\end{aligned}$$

It remains to show that there is an equivalence $[\mathrm{Pm}[P]_b, J] \simeq \top$: let $f : \mathrm{Pm}[P]_b \to J$ and $\alpha : f \Rightarrow \mathrm{s}^P_b$ be a natural isomorhism, then morphisms between (f, α) and $(\mathrm{s}^P_b, \mathrm{id})$ in $[\mathrm{Pm}[P]_b, J]$ are in bijections with the inverses of α.

So the first projection induces an equivalence $\mathrm{Op}[P \circ \mathrm{id}]_j \to \mathrm{Op}[P]_j$, which we take as β_j.

The equivalences $\varepsilon_{b,f,a} : \mathrm{Pm}[P]_b \to \mathrm{Pm}[P \circ \mathrm{id}]_{b,f,a}$ are given by the isomorphism:

$$\begin{aligned}\mathrm{Pm}[P \circ \mathrm{id}]_{b,f,a} &= \sum_{x:\mathrm{Pm}[P]_b} \top \\ &\simeq \mathrm{Pm}[P]_b\end{aligned}$$

c) Associativity.: Let $P_1 : I \rightsquigarrow J$, $P_2 : J \rightsquigarrow K$ and $P_3 : K \rightsquigarrow L$ be three polynomials. To prove the associativity, notice first that Lemma 42 induces an equivalence:

$$\mathrm{Op}[Q \circ P]_k \simeq \sum_{b:\mathrm{Op}[Q]_k} \prod_{x:\mathrm{Pm}[Q]_b} \mathrm{Op}[P]_{\mathrm{s}_b^Q(x)}$$

The computation shown in Figure 1 defines the associativity equivalences $\beta_l : \mathrm{Op}[(P_3 \circ P_2) \circ P_1] \to \mathrm{Op}[P_3 \circ (P_2 \circ P_1)]$, which send a triple $(b_3, \lambda x_3.b_2, \lambda x_3 x_2.b_1)$ to the pair $(b_3, \lambda x_3.(b_2, \lambda x_2.b_1))$.

Finally, ε consists in defining equivalences $\mathrm{Pm}[(P_3 \circ P_2) \circ P_1]_{b_3,b_2,b_1} \to \mathrm{Pm}[P_3 \circ (P_2 \circ P_1)]_{(b_3,\lambda x_3.(b_2 x_3, b_1 x_3))}$. Those are given by the following composite:

$$\begin{aligned}&\mathrm{Pm}[(P_3 \circ P_2) \circ P_1]_{b_3,b_2,b_1} \\ &= \sum_{x_3,x_2:\sum_{x_3:\mathrm{Pm}[P_3]_{b_3}} \mathrm{Pm}[P_2]_{b_2(x_3)}} \mathrm{Pm}[P_1]_{b_1(x_3,x_2))} \\ &\cong \sum_{x_3:\mathrm{Pm}[P_3]_{b_3}} \sum_{x_2:\mathrm{Pm}[P_2]_{b_2(x_3)}} \mathrm{Pm}[P_1]_{b_1(x_3,x_2))} \\ &= \sum_{x_3:\mathrm{Pm}[P_3]_{b_3}} \mathrm{Pm}[(P_2 \circ P_1)]_{b_2 x_3, b_1 x_3} \\ &= \mathrm{Pm}[P_3 \circ (P_2 \circ P_1)]_{b_1,\lambda x_3.(b_2 x_3, b_1 x_3)}\end{aligned}$$

C. Proof of Proposition 32

We prove the result using the equational definition of products. I.e. we show that for polynomials $P_1 : I \rightsquigarrow J_1$ and $P_2 : I \rightsquigarrow J_2$, the pairing $\langle P_1, P_2 \rangle : I \rightsquigarrow J_1 \sqcup J_2$ satisfies:

- $\Pi_i \circ \langle P_1, P_2 \rangle = P_i$;
- $\langle \Pi_1 \circ P, \Pi_2 \circ P \rangle = P$ for $P : I \rightsquigarrow J_1 \sqcup J_2$.

For the first equality :

$$\begin{aligned}&\mathrm{Op}[\Pi_1 \circ \langle P_1, P_2 \rangle]_j \\ &= \sum_{b:\mathrm{Op}[\Pi_1]_j} [\mathrm{Pm}[\Pi_1]_j, \sum_{j:J_1} \mathrm{Op}[\langle P_1, P_2 \rangle]_{\iota_1 j}] \\ &\cong [\top \to_j J, \mathrm{Pm}[P_1]_{\iota_1 j}] \\ &\cong \prod_{j':J_1} [J_1(j', j), \mathrm{Op}[P_1]_{j'}] \\ &\cong \mathrm{Op}[\mathrm{id} \circ P]_j \cong \mathrm{Op}[P]_j.\end{aligned}$$

The map β_j therefore sends a pair $(\star, b, \alpha)$ to b. For ε, we have the following equivalence:

$$\begin{aligned}\mathrm{Pm}[\Pi_1 \circ \langle P_1, P_2 \rangle]_{\star,b,\alpha} &= \sum_{\star:\mathrm{Pm}[\Pi_1]} \mathrm{Pm}[\langle P_1, P_2 \rangle]_b \\ &\cong \mathrm{Pm}[P_1]_b\end{aligned}$$

The equivalence $\Pi_2 \circ \langle P_1, P_2 \rangle = P_2$ is similar.

We now prove the last equivalence, by distinguishing two cases depending on whether $j \in J_1$ or $j \in J_2$. In the first case:

$$\begin{aligned}&\mathrm{Op}[\langle \Pi_1 \circ P, \Pi_2 \circ P \rangle]_{\iota_1 j} \\ &= \mathrm{Op}[\Pi_1 \circ P]_j \\ &= \sum_{\star:\top} [\top \to_{\iota_1 j} J_1 \sqcup J_2, \sum_{j':J_1 \sqcup J_2} \mathrm{Op}[P]_{j'}] \\ &\cong [\top \to_{\iota_1 j} J_1 \sqcup J_2, \sum_{j_1:J_1} \mathrm{Op}[P]_{\iota_1 j_1} \sqcup \sum_{j_2:J_2} \mathrm{Op}[P]_{\iota_2 j_2}] \\ &\cong [\top \to_{\iota_1 j} J_1, \sum_{j_1:J_1} \mathrm{Op}[P]_{\iota_1 j_1}] \\ &\cong \mathrm{Op}[P]_{\iota_1 j_1}\end{aligned}$$

And $\beta_{\iota_1 j}$ therefore sends a pair $(\star, \iota_1(b, \alpha))$ to b. The equivalence ε is then given by:

$$\begin{aligned}&\mathrm{Pm}[\langle \Pi_1 \circ P, \Pi_2 \circ P \rangle]_{(\star,\iota_1(b,\alpha))} \\ &= \mathrm{Pm}[\Pi_1 \circ P]_{\star,b,\alpha)} \\ &= \sum_{\star:\top} \mathrm{Pm}[P]_b \\ &\cong \mathrm{Pm}[P]_b\end{aligned}$$

The case where $j \in J_2$ is symmetric.

D. Proof of Proposition 35

Proof. We need to show that the identity polynomial and the two projections are finitary, and that finitary polynomials are closed under composition and pairing.

The fact that the identity and the projections are finitary comes from the fact that $\top$ is finite.

Let now $P : I \rightsquigarrow J$ and $Q : J \rightsquigarrow K$ be two finitary polynomials, and take $(b, f, \alpha) \in \mathrm{Op}[Q \circ P]_k$. By hypothesis there exists $n \in \mathbb{N}$ such that there is an equivalence $\phi : [n] \to \mathrm{Pm}[Q]_b$, and for any $b' \in \mathrm{Op}[P]_j$, there exists $n_{b'}$ such that $\mathrm{Pm}[P]_{b'} \equiv [n_{b'}]$. We then have:

$$\begin{aligned}\mathrm{Pm}[Q \circ P]_{b,f,\alpha} &= \sum_{y:\mathrm{Pm}[Q]_b} \mathrm{Pm}[P]_{f(y)} \\ &\cong \sum_{y:[n]} \mathrm{Pm}[P]_{f(\phi(y))} \\ &\equiv \sum_{y:[n]} [n_{f(\phi(y))}] \cong \left[\sum_{k=1}^{n} n_{f(\phi(k))}\right]\end{aligned}$$

This proves that finitary polynomial are stable under composition.

Take now $P : I \rightsquigarrow J_1$ and $P_2 : I \rightsquigarrow J_2$ two finitary polynomials, and let us show that $\langle P_1, P_2 \rangle$ is finitary. This is immediate since $\mathrm{Pm}[\langle P_1, P_2 \rangle]_{\iota_1 b} = \mathrm{Pm}[P_1]_b$ and $\mathrm{Pm}[\langle P_1, P_2 \rangle]_{\iota_2 b} = \mathrm{Pm}[P_2]_b$. $\square$

$$
\begin{aligned}
&\mathrm{Op}[(P_3 \circ P_2) \circ P_1]_l \\
&\simeq \sum_{b,f:\mathrm{Op}[P_3 \circ P_2]_l} \prod_{x,y:\mathrm{Pm}[P_2 \circ P_2]_{b,f}} \mathrm{Op}[P_1]_{\mathrm{s}^{P_3 \circ P_2}_{b,f}(x,y)} \\
&\simeq \sum_{b_3:\mathrm{Op}[P_3]_l} \sum_{f:\prod_{x:\mathrm{Pm}[P_3]_b} \mathrm{Op}[P_2]_{\mathrm{s}^{P_3}_b(x)}} \prod_{x,y:\sum_{x:\mathrm{Pm}[P_3]_{b_3}} \mathrm{Pm}[P_2]_{f(x)}} \mathrm{Op}[P_1]_{\mathrm{s}^{P_2}_{f(x)}(y)} \\
&\cong \sum_{b_3:\mathrm{Op}[P_3]_l} \sum_{f:\prod_{x:\mathrm{Pm}[P_3]_{b_3}} \mathrm{Op}[P_2]_{\mathrm{s}^{P_3}_b(x)}} \prod_{x\mathrm{Pm}[P_3]_{b_3}} \prod_{y:\mathrm{Pm}[P_2]_{f(x)}} \mathrm{Op}[P_1]_{\mathrm{s}^{P_2}_{f(x)}(y)} \\
&\cong \sum_{b_3:\mathrm{Op}[P_3]_l} \prod_{x:\mathrm{Pm}[P_3]_{b_3}} \sum_{b_2:\mathrm{Op}[P_2]_{\mathrm{s}^{P_3}_{b_3}(x)}} \prod_{y:\mathrm{Pm}[P_2]_{b_2}} \mathrm{Op}[P_1]_{\mathrm{s}^{P_2}_{b_2}(y)} \\
&\simeq \sum_{b_3:\mathrm{Op}[P_3]_l} \prod_{x:\mathrm{Pm}[P_3]_{b_3}} \mathrm{Op}[P_2 \circ P_1]_{s[P_3]_{b_3}(x)} \\
&\simeq \mathrm{Op}[P_3 \circ (P_2 \circ P_1)]_l
\end{aligned}
$$

Figure 1. Computation for the associativity.