Cent. Eur. J. Math. • 11(3) • 2013 • 401-422
DOI: 10.2478/s11533-012-0152-1

VERSITA

Central European Journal of **Mathematics**

Categorification of Hopf algebras of rooted trees

Research Article

Joachim Kock[1]*

1 *Departament de Matemàtiques, Universitat Autònoma de Barcelona, 08193 Bellaterra (Barcelona), Spain*

Received 8 October 2011; accepted 26 April 2012

Abstract: We exhibit a monoidal structure on the category of finite sets indexed by P-trees for a finitary polynomial endofunctor P. This structure categorifies the monoid scheme (over Spec $\mathbb{N}$) whose semiring of functions is (a P-version of) the Connes–Kreimer bialgebra H of rooted trees (a Hopf algebra after base change to $\mathbb{Z}$ and collapsing H_0). The monoidal structure is itself given by a polynomial functor, represented by three easily described set maps; we show that these maps are the same as those occurring in the polynomial representation of the free monad on P.

MSC: 05C05, 16T05, 18A99

Keywords: Rooted trees • Hopf algebras • Categorification • Monoidal categories • Polynomial functors • Finite sets

1. Introduction

1.1. The Connes–Kreimer Hopf algebra and the Butcher group

The Hopf algebra $\mathcal{H}$ of rooted trees is now a well-established object in mathematics, thanks in particular to the seminal works of Connes and Kreimer. Kreimer [19] discovered that $\mathcal{H}$ controls the combinatorics of renormalisation in perturbative quantum field theory, and his collaboration with Connes, e.g. [8, 9], to cite a few, uncovered deep connections with noncommutative geometry, number theory, Lie theory, and algebraic combinatorics, stimulating a lot of further activity by many mathematicians and physicists. The Connes–Kreimer Hopf algebra has now been characterised by several different universal properties [7, 8, 23]. The group of characters of $\mathcal{H}$, now called the Butcher group, was in fact studied by Butcher [5] some 30 years earlier, in relation with order conditions for Runge–Kutta methods in numerical integration. The link back to this work from the Connes–Kreimer Hopf algebra was provided by Brouder [4].

* *E-mail: kock@mat.uab.cat*

1.2. Categorification of the Connes–Kreimer Hopf algebra

This article investigates a categorification of the bialgebra $\mathcal{H}$, i.e. a lift from the level of algebra to the level of sets. To be precise, the antipode is *not* categorified. Indeed, the antipode is a feature depending on the additive inverses of the ground ring. But in fact, the most natural 'ring' of definition of $\mathcal{H}$ is the semiring $\mathbb{N}$ of natural numbers, and lacking additive inverses there is no antipode to be had here. The semiring $\mathbb{N}$ appears as the Burnside semiring of the distributive category $\mathbb{F}$ of finite sets, i.e. is the set of isomorphism classes of finite sets, with addition and multiplication inherited from the categorical sum and product. Exhibiting a distributive category whose Burnside semiring is $\mathcal{H}$, and describing at this level the comultiplication, is what we mean by categorification, following a popular terminology which goes back to Crane and Yetter [10]; the specific process employed can more precisely be called objectification, cf. Lawvere, Schanuel, and their collaborators. A recommended introduction to categorification is the beautiful paper [2] of Baez and Dolan.

While the combinatorial nature of $\mathcal{H}$ makes it clear that this categorification should be possible, a considerable amount of categorical algebra is needed in order to make all the algebraic structure explicit at the objective level. The categorification of $\mathcal{H}$ will be the distributive category $\mathbb{F}[\mathbf{T}]$ of polynomial functors on the category $\mathbb{A}^{\mathbf{T}}$ of $\mathbf{T}$-indexed finite sets, where $\mathbf{T}$ is the set of trees. For technical reasons we mostly work with P-trees for a finitary polynomial endofunctor P, and we start out by explaining the differences. The comultiplication will be a comonoidal structure on $\mathbb{F}[\mathbf{T}]$, relative to a certain tensor product of distributive categories. It is conceptually much easier to take the dual viewpoint. The category $\mathbb{A}^{\mathbf{T}}$ will be the categorification of the Butcher group (or rather the Butcher monoid), and the task is to describe the monoidal structure on $\mathbb{A}^{\mathbf{T}}$ dual to the comultiplication. This monoidal structure, a functor $M\colon \mathbb{A}^{\mathbf{T}}\times\mathbb{A}^{\mathbf{T}} \to \mathbb{A}^{\mathbf{T}}$, is itself a polynomial functor. The comultiplication is now given by precomposition with M.

1.3. Polynomial functors

The notion of polynomial functor is central to this work. The theory of polynomial functors has roots in topology, representation theory, combinatorics, logic and computer science, but the task of unifying these developments has only recently begun [13]. An important feature of polynomial functors is that they can be manipulated in terms of a few representing sets, just as polynomial functions can be manipulated in terms of their coefficients and exponents.

In the present work, polynomial functors enter at two levels: firstly, and most importantly, the notion of polynomial functor categorifies the notion of polynomial function; second, the trees that index the involved variables are operadic trees, and they are themselves defined in terms of polynomial endofunctors. For the first aspect we need to develop some basic theory, which constitutes Section 4; for the second, the theory needed is already available from [13, 15].

1.4. Free monads, P-trees, and beyond

Historically, one starting point for the general project of categorification is the quest in combinatorics for bijective proofs: it is well appreciated that a bijection between sets represents better understanding than a mere equation between numbers. One insight into the Hopf algebra of rooted trees which results from its categorification and the polynomial viewpoint is the relation with free monads: it is shown that the set maps occurring in the polynomial representation of the new monoidal structure are the same maps as occur in the polynomial representation of the free monad construction. Unfortunately, this is not completely true for abstract trees. It is true for P-trees, and we develop the theory in this setting — the free monad in question is then the free monad on P. The notion of P-tree, cf. 5.4 below, covers many notions of structured and decorated trees, such as binary and planar trees, but abstract trees themselves are not an example: abstract trees should be P-trees for the terminal polynomial endofunctor, but the category of polynomial endofunctors over sets, see 5.2, does not have a terminal object!

Nevertheless, with a little care, the constructions made for P-trees work also for abstract trees, only the relation with free monads is then less direct. This is explained in Section 7, where it is also explained how these issues disappear when upgrading the whole theory from sets to groupoids: in this fancier setting, the terminal polynomial endofunctor does exist, and abstract trees become a particular case of the notion of P-tree.

1.5. Outline of the paper

In Section 2 we introduce the bialgebra of operadic trees, and explain its relation with the usual Connes–Kreimer Hopf algebra. Section 3 collects the notions and results needed from the theory of polynomial functors, notably the explicit formula for substitution of polynomials. In Section 4 we set up a framework for dealing with categories of polynomial functors and categories of indexed finite sets as 'polynomial rings' and 'affine space', respectively. This section contains many observations that seem not to have been made before, regarding polynomial functors as categorification of polynomial functions. In Section 5 we introduce trees and P-trees and review in detail the construction of the free monad on a polynomial endofunctor P. Finally, in Section 6 we establish the monoidal structure on $\mathbb{A}^{\mathsf{T}}$, and relate it to the bialgebra of trees. Section 7 contains some remarks about the difference between P-trees and abstract trees, and hints at a groupoid version of all the constructions as a more comprehensive framework.

1.6. Related work

This work was presented at the 2010 International Category Theory Conference in Genova. At the same conference, Matías Menni spoke about his work with Lawvere [21] on categorification of incidence algebras of Möbius categories. While it is known that the Connes–Kreimer Hopf algebra can be constructed as the incidence algebra of a suitable family of posets, it seems unlikely that it can be given as the incidence algebra of a single Möbius category, so at present the Lawvere–Menni approach does not apply to categorify the Connes–Kreimer Hopf algebra. Work is under way to develop a higher-categorical notion of Möbius categories in order to unify the two approaches.

2. Hopf algebras of rooted trees

The standard Connes–Kreimer Hopf algebra of rooted trees concerns combinatorial trees, whereas in this paper we prefer to work with operadic trees. This section explains the differences.

2.1. Combinatorial trees

The trees usually employed, which here we call combinatorial trees, are often defined as finite connected graphs without loops or cycles, and with a designated root vertex. If a connected subgraph of a rooted tree T does not contain the root of T, then instead it has a vertex nearest the root, which is then defined to be its root.

2.2. The Connes–Kreimer bialgebra of rooted trees

The bialgebra of rooted trees of Connes and Kreimer [19] is the polynomial $\Bbbk$-algebra $\mathcal{H}$ on the set of isomorphism classes of combinatorial trees. Here $\Bbbk$ can be any commutative $\mathbb{N}$-algebra. The comultiplication is given on generators by

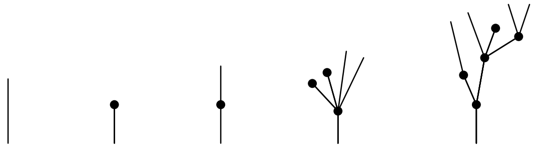
$$\Delta\colon \mathcal{H} \longrightarrow \mathcal{H}\otimes_{\Bbbk}\mathcal{H}, \qquad T \longmapsto \sum_c P_c \otimes R_c,$$

where the sum is over all admissible cuts of T; the left-hand factor P_c is the forest (interpreted as a monomial) found above the cut, and R_c is the subtree found below the cut (or the empty forest, in case the cut is below the root). 'Admissible cut' means upper-set in the poset underlying the tree (oriented from leaves (inputs) to root (output)), i.e. either a subtree containing the root, or the empty set.

$\mathcal{H}$ is a connected bialgebra: the grading is by the number of nodes, and $\mathcal{H}_0$ is spanned by the unit. Therefore, by general principles, see for example [11], it acquires an antipode and becomes a Hopf algebra — provided $\Bbbk$ has additive inverses, i.e. is a $\mathbb{Z}$-algebra. (In any case, the antipode exists after base change to $\mathbb{Z}$.)

2.3. Operadic trees

We shall need trees with slightly more expressive power, by allowing loose ends (leaves): these are *operadic trees*, also called finite rooted trees with boundary — a formal definition is given in 5.1. For the moment, the following drawings should suffice to exemplify operadic trees — as usual the planar aspect inherent in a drawing should be disregarded:



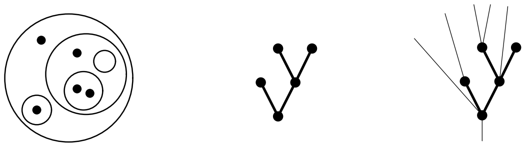
Note that certain edges (the *leaves*) do not start in a node and that one edge (the obligatory *root edge*) does not end in a node. A node without incoming edges is not the same thing as a leaf; it is a nullary operation (i.e. a constant) in the sense of operads. In operad theory, the nodes represent operations, and trees are formal combinations of operations. The small incoming edges drawn at every node serve to keep track of the arities of the operations. Furthermore, for coloured operads, the operations have type constraints on their inputs, encoded as attributes of the edges.

2.4. Operadic trees in QFT

The use in quantum field theory of more refined notions of trees, and operadic trees in particular, has been hinted at by Kreimer in several papers, most concretely with Bergbauer [3], where trees with loose edges are used to analyse combinatorial Dyson–Schwinger equations.

In fact, the role of trees in the Connes–Kreimer bialgebra is to encode nestings of 1-particle irreducible Feynman graphs [19], and one can argue [17] that they are naturally operadic: each tree naturally comes equipped with decorations by primitive 1PI graphs on nodes and interaction labels on edges. To fully encode the compatibility conditions involved in these decorations, and to allow to recover the graph from the decorated trees, it is necessary to keep track of the arities of the nodes, so that even vacant input slots are represented; this leads naturally to operadic trees. A more thorough analysis of the relationship between graphs and trees is given elsewhere [17].

Kreimer [20] stresses the general utility of trees as expression of nestings of structures, not only of Feynman graphs. In the following picture we see first a nesting of subsets, then a combinatorial-tree expression of the nesting, and finally an operadic-tree expression of the same nesting, in which the leaves correspond to the elements of the nested sets:



One feature of operadic trees is that they admit colimit descriptions of basic operations: most importantly, grafting can be expressed as a pushout, and every tree is the colimit of its elementary subtrees, i.e. trees without inner edges [15]. This makes them well suited for constructions (rather than just decomposition). Another advantage is that symmetries of the original nested structure (for example a nesting of Feynman graphs) are better captured by operadic trees than by combinatorial trees, as the above picture also illustrates: the combinatorial tree has a symmetry which does not reflect a symmetry in the nesting, and fails to detect the inner symmetries of the nesting. (The symmetry issues play a crucial role in the treatment of Green functions, where the operadic viewpoint seems important [12].)

2.5. The bialgebra of operadic trees

A *cut* of an operadic tree is defined to be a subtree containing the root — note that the arrows in the category of operadic trees, Section 5, are arity preserving (5.3), meaning that if a node is in the subtree, then so are all the incident edges of that node.

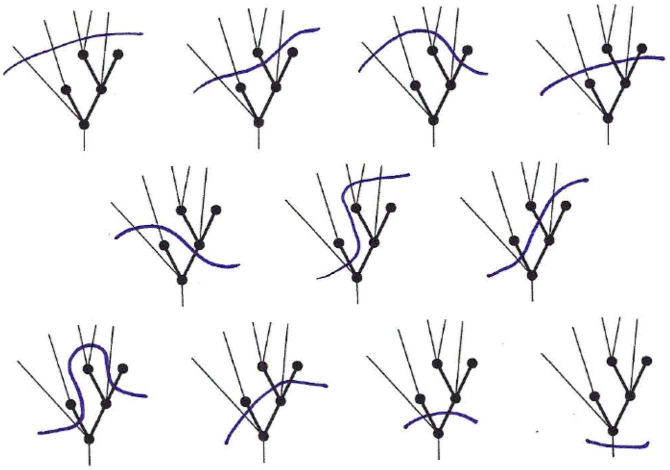
If $c\colon R \subset T$ is a subtree containing the root, then each leaf e of R determines an *ideal subtree* of T (5.3), namely consisting of e (which becomes the new root) and all the descendant edges and nodes. This is still true when e is also a leaf of T: in this case, the ideal tree is the trivial tree consisting solely of e. Figuratively, this means that cuts can go through the leaves but are not allowed to go *above* the leaves. Note also that the root edge is a subtree; the ideal tree of the root edge is of course the tree itself. This is the analogue of the cut-below-the-root in the combinatorial case. For a cut $c\colon R \subset T$, define P_c to be the forest consisting of all the ideal trees generated by the leaves of R.

Let $\mathcal{B}$ be the polynomial $\Bbbk$-algebra on the set of isomorphism classes of operadic trees. With comultiplication defined on the generators by

$$\Delta\colon \mathcal{B} \longrightarrow \mathcal{B}\otimes_{\Bbbk}\mathcal{B}, \qquad T \longmapsto \sum_{c:R\subset T} P_c \otimes R,$$

as for combinatorial trees, $\mathcal{B}$ becomes a graded bialgebra. It is not connected: $\mathcal{B}_0$ is spanned by powers of the trivial tree $|$ (including the empty power, which is the algebra unit 1). These are all group-like, so one could obtain a connected bialgebra by imposing the equation $1 = |$. Note that the comultiplication for operadic trees is linear in the right-hand factor, unlike the comultiplication for combinatorial trees, where $\Delta(T)$ always contains a factor $T \otimes 1$ corresponding to the empty-cut-below-the-root.

As an example, here are the eleven possible cuts of the tree in the previous picture:



Note that the first and the last term of the comultiplication of this tree T are

$$|||||\otimes T \; + \; T \otimes |,$$

not $1 \otimes T + T \otimes 1$.

2.6. From operadic trees to combinatorial trees

The *core* of a non-trivial operadic tree T is the combinatorial tree $T^\bullet$ obtained by pruning off all leaves as well as the root edge. Taking core is functorial in root-preserving inclusions. For any non-trivial operadic tree T, there is a natural bijection between the set of non-trivial subtrees of T and the set of combinatorial subtrees of $T^\bullet$. This bijection sends $R \subset T$ to $R^\bullet \subset T^\bullet$. By extending the assignment by defining the core of the trivial tree to be the empty (combinatorial) forest, it is clear that the operation is compatible with comultiplication, as illustrated in the previous picture, where the core is highlighted with fat lines.

In conclusion we have

2.7. Proposition

Taking core defines a surjective homomorphism of graded bialgebras $\mathcal{B} \longrightarrow \mathcal{H}$, $T \longmapsto T^\bullet$.

2.8. The bialgebra of P-trees

The definition makes sense equally well for P-trees (5.4). If there are more than one edge colour involved, there is a trivial tree for each edge colour, and the monomials in these trees (including the trivial monomial 1) span $\mathcal{B}_0$. Clearly they are all group-like.

3. Polynomial functors

In this section, we recall the basic notions of polynomial functors, referring to Gambino–Kock [13] for all details.

3.1. Slices, pullback, and adjoints

We write $+$ and $\sum$ for the disjoint union of sets (i.e. the categorical coproduct). The equality sign denotes canonical isomorphism of sets. Let B be a set. Recall, e.g. from [1, § 9.7], the *slice category* $\mathbf{Set}/B$: its objects are maps $X \to B$, and its arrows are commutative triangles

$$\begin{tikzcd} X \arrow[rr] \arrow[dr] & & X' \arrow[dl] \\ & B. & \end{tikzcd}$$

For a set map $p\colon X \to B$, we denote the fibre over b by $X_b = p^{-1}(b)$. We can then write X as the union of the fibres, $X = \sum_{b\in B} X_b$, and interpret p as a B-indexed family of sets. In fact, there is a canonical equivalence of categories

$$\mathbf{Set}/B \longrightarrow \mathbf{Set}^B, \qquad [X \to B] \longmapsto [b \mapsto X_b].$$

Here $\mathbf{Set}^B$ denotes the category of functors $B \to \mathbf{Set}$ (considering the set B as a discrete category). For either interpretation of B indexed family of sets, we shall use the notation $(X_b \,|\, b \in B)$.

Given a map $f\colon A \to B$, we have the *lowershriek* functor

$$f_!\colon \mathbf{Set}/A \longrightarrow \mathbf{Set}/B, \qquad [X \to A] \longmapsto [X \to A \to B].$$

Using decomposition into fibres, the functor has the following description:

$$(X_a \,|\, a \in A) \longmapsto \left(\sum_{a\in A_b} X_a \,\middle|\, b \in B \right),$$

i.e. sum along the fibres. The functor $f_!$ has a right adjoint, denoted by *upperstar*, given by pullback:

$$f^*\colon \mathbf{Set}/B \longrightarrow \mathbf{Set}/A, \qquad [X \to B] \longmapsto [A \times_B X \to A].$$

In fibre notation,

$$(X_b \,|\, b \in B) \longmapsto (X_{f(a)} \,|\, a \in A).$$

Finally, also f^* in turn has a right adjoint, denoted *lowerstar*, which is more involved to describe synthetically, but whose description in terms of fibres is "multiply along the fibres":

$$f_*\colon \mathbf{Set}/A \longrightarrow \mathbf{Set}/B, \qquad (X_a \,|\, a \in A) \longmapsto \left(\prod_{a\in A_b} X_a \,\middle|\, b \in B \right).$$

If the categories $\mathbf{Set}/A$ and $\mathbf{Set}/B$ are replaced by the equivalent categories $\mathbf{Set}^A$ and $\mathbf{Set}^B$, the three functors $f_!$, f^*, and f_* can still be interpreted: f^* is just precomposition with f. Its adjoints are left and right Kan extension respectively, cf. [22, Chapter X] for this notion. In this paper we shall not need the explicit form of these functors. Although we actually formulate many results in terms of the functor categories $\mathbf{Set}^B$, when it comes to functors we prefer to work with slices.

3.2. Polynomial functors

A diagram of sets

$$\begin{tikzcd}
& E \arrow[r, "p"] \arrow[dl, "s"'] & B \arrow[dr, "t"] & \\
I & & & J
\end{tikzcd} \tag{1}$$

defines a *polynomial functor*

$$\mathbf{Set}/I \xrightarrow{s^*} \mathbf{Set}/E \xrightarrow{p_*} \mathbf{Set}/B \xrightarrow{t_!} \mathbf{Set}/J.$$

In terms of indexed families and fibres, the formula for this polynomial functor is

$$\mathbf{Set}/I \longrightarrow \mathbf{Set}/J, \qquad (X_i \,|\, i \in I) \longmapsto \left(\sum_{b \in B_j} \prod_{e \in E_b} X_{s(e)} \,\middle|\, j \in J \right).$$

It is hence a J-indexed family of sums of products of I-indexed families. Particularly suggestive is the case $I = J = 1$, so that we are talking about a 'single polynomial in one variable'. In this case the formula boils down to

$$\mathbf{Set} \longrightarrow \mathbf{Set}, \qquad X \longmapsto \sum_{b \in B} X^{E_b}.$$

3.3. Morphisms

Morphisms of polynomial functors are just natural transformations. One can show [13] that a natural transformation $P' \Rightarrow P$ between polynomial functors is uniquely represented by diagrams of the form

$$\begin{tikzcd}
P': & I \arrow[d, equal] & E' \arrow[l] \arrow[r] & B' \arrow[r] \arrow[d, equal] & J \arrow[d, equal] \\
 & & \bullet \arrow[u] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & B' \arrow[d] & \\
P: & I & E \arrow[l] \arrow[r] & B \arrow[r] & J.
\end{tikzcd} \tag{2}$$

By $\mathbf{Poly}(I, J)$ we denote the category of polynomial functors $\mathbf{Set}/I \to \mathbf{Set}/J$, and their natural transformations. For the manipulation of polynomial functors in terms of the representing sets, the following two facts are basic.

3.4. Beck–Chevalley

Given a pullback square

$$\begin{tikzcd}
\overline{A} \arrow[r, "\overline{\varphi}"] \arrow[d, "\alpha"'] \arrow[dr, phantom, "\lrcorner", very near start] & \overline{B} \arrow[d, "\beta"] \\
A \arrow[r, "\varphi"'] & B
\end{tikzcd}$$

there are natural isomorphisms of functors $\alpha_! \circ \overline{\varphi}^* \xrightarrow{\sim} \varphi^* \circ \beta_!$ and $\beta^* \circ \varphi_* \xrightarrow{\sim} \overline{\varphi}_* \circ \alpha^*$, usually called the *Beck–Chevalley isomorphisms*.

3.5. Distributivity

Starting with maps $A \xrightarrow{\varphi} B \xrightarrow{\psi} C$, we can construct the following diagram by applying ψ_* to the map $\varphi \colon A \to B$:

$$\begin{tikzcd}
\psi^*\psi_*A \arrow[r, "\overline{\psi}"] \arrow[d, "\epsilon"'] \arrow[dr, phantom, very near start, "\lrcorner"] & \psi_*A \arrow[dd, "\widetilde{\varphi}"] \\
A \arrow[d, "\varphi"'] & \\
B \arrow[r, "\psi"'] & C.
\end{tikzcd}$$

Here ϵ is the A-component of the counit for the adjunction $\psi^* \dashv \psi_*$. A diagram of this form is called a *distributivity pentagon*; it can be characterised by a universal property [24]. For such a diagrams the *distributive law* holds:

$$\psi_* \circ \varphi_! \ \simeq \ \widetilde{\varphi}_! \circ \overline{\psi}_* \circ \epsilon^* .$$

This is the categorical expression of the distributive law of elementary arithmetic, as it amounts to distributing a product (lowerstar) over a sum (lowershriek); see [13] for more discussion, and [24] for a deeper treatment.

3.6. Composition

The composition of two polynomial functors is again polynomial [13]. This is a consequence of the Beck–Chevalley isomorphisms and distributivity. The important fact is that the bridge diagram representing the composite can be constructed explicitly in terms of a few operations on sets. Namely, given polynomial functors P and Q as in the bottom of the diagram (in which the labels Δ, Π, and Σ merely indicate which sort of polynomial operation is performed along each map: Δ indicates pullback, Σ indicates lowershriek, and Π indicates lowerstar)

$$\begin{tikzcd}
 & & & & & & \cdot \arrow[rrr, "\Pi"] \arrow[dll, "\Delta"'] \arrow[dr, phantom, "\mathrm{pb}"] & & & \cdot \arrow[rr, "\Pi"] \arrow[dll, "\Delta"] \arrow[ddrr, phantom, "\mathrm{distr}"] & & \cdot \arrow[dd, "\Sigma"] & & \\
 & & & & \cdot \arrow[rrr, "\Pi"'] \arrow[dll, "\Delta"'] \arrow[dr, phantom, "\mathrm{pb}"] & & & \cdot \arrow[dll, "\Delta"] \arrow[drr, "\Sigma"] \arrow[dd, phantom, "\mathrm{pb}"] & & & & & & \\
 & & \cdot \arrow[rrr, "\Pi"'] \arrow[dll, "\Delta"'] & & & \cdot \arrow[drr, "\Sigma"'] & & & & \cdot \arrow[dll, "\Delta"] \arrow[rr, "\Pi"'] & & \cdot \arrow[drr, "\Sigma"] & & \\
\cdot & & & Q & & & & \cdot & & & P & & & \cdot
\end{tikzcd}$$

the composite $P \circ Q$ is constructed as the top outline: start by taking the pullback at the common middle set, then lowerstar the result to arrive at what will be the top right-hand corner of the composite, and pull back to complete the distributivity pentagon. The diagram is completed by taking two more pullbacks as indicated. The bridge diagram constructed is naturally isomorphic to the composite of the original functors by the Beck–Chevalley isomorphisms and distributivity. The construction reflects closely how one composes two polynomial functions in elementary algebra: the key point is of course distributing the products involved in the outer polynomial over the sums of the inner.

3.7. The 2-category of polynomial functors

Polynomial functors form a 2-category[1] **Poly** in which the objects are the slices of **Set** (or equivalently, the categories $\mathbf{Set}^I$ for I a set), the 1-cells are the polynomial functors (i.e. those isomorphic to one given by a diagram (1)), and 2-cells are

[1] *The notion of 2-category, see for example [22, Chapter XII], is not essential to understand this work, but it is an efficient framework to set up some of the involved notions correctly.*

arbitrary natural transformations. **Poly** is a strict 2-category. By [13, Theorem 2.17] it is biequivalent to a bicategory whose objects are sets I, whose 1-cells are diagrams like (1) and whose 2-cells are diagrams like (2) (and where composition is described as in 3.6). The theorem allows us to blur the distinction between bridge diagrams and the polynomial functors they represent, allowing for the conceptual benefit of the strict 2-category **Poly** and the computational benefit of the representing diagrams. This is a characteristic aspect of the theory of polynomial functors.

4. Elementary algebra of polynomial functors

We are here concerned with the aspect of polynomial functors as a categorification of the elementary algebra of polynomial functions. In this section we introduce a set-up to deal with 'rings' of polynomial functors and their associated 'affine spaces', in the sense of algebraic geometry [14]. The results in this section appear for the first time, but they are not difficult. One key issue is to impose the correct finiteness conditions. Certainly, our 'ground category' should be the category of finite sets: all coefficients and exponents will now be required to be finite. However, the polynomial rings we are interested in have infinitely many variables, so we cannot just take the theory of polynomial functors internally to the category of finite sets.

4.1. Finite sets

The category of finite sets, which we denote

$$\mathbb{F} = \mathbf{FinSet},$$

will be our coefficient 'ring'. More precisely, the monoidal operations of finite sums and products make it a distributive category [6]. Clearly the set of isomorphism classes of $\mathbb{F}$ is the set of natural numbers, and sum and products then yield the usual addition and multiplication of numbers. In short,

$\mathbb{N}$ is the Burnside semiring of $\mathbb{F}$,

which is categorification in its purest form.

The notation $\mathbb{F}$ stresses the algebraic aspect of **FinSet**, and we use this notation when we think of that category as the ground ring. The same category also plays a geometric role, and as such we denote it

$$\mathbb{A} = \mathbf{FinSet},$$

leading to a natural notation for what plays the role of affine space. Namely, if I is a set, then the functor category $\mathbb{A}^I = \mathbf{FinSet}^I$ is the domain for the finite polynomial functors in I-many variables. (Note that $\mathbf{FinSet}^I$ is not equivalent to $\mathbf{FinSet}/I$ when I is an infinite set: the latter is the category of maps $E \to I$ with E finite. The former is equivalent to the category of maps $E \to I$ with finite fibres.)

4.2. Finite polynomials

A polynomial $I \leftarrow E \rightarrow B \rightarrow 1$ is called *finite* when B and E are finite sets. These are the polynomial functors on $\mathbb{A}^I$ with values in finite sets. We denote the category of these polynomial functors

$$\mathbb{F}[I] = \mathbf{FinPoly}(I, 1).$$

Clearly $\mathbb{F}[I]$ is a distributive category and

the Burnside semiring of $\mathbb{F}[I]$ is $\mathbb{N}[I]$, the polynomial semiring in I-many variables,

as the notation also suggests. Moreover,

$\mathbb{F}[I]$ is the free distributive category on the set I,

just as $\mathbb{N}[I]$ is the free commutative semiring on I. Indeed, the category $\mathbb{F}[I]$ is generated freely under finite sums by the monomials, i.e. those $I \leftarrow E \rightarrow B \rightarrow 1$ for which $B = 1$. From the characterisation of natural transformations in 3.3, we see that the category of monomial functors in I-many variables has arrows given by commutative triangles

$$\begin{tikzcd} & E' \arrow[dl] \\ I & \\ & E, \arrow[ul] \arrow[uu] \end{tikzcd}$$

and so is equivalent to $(\mathbf{FinSet}/I)^{\mathrm{op}}$. But it is well known that $(\mathbf{FinSet}/I)^{\mathrm{op}}$ is the finite-product completion of I.

4.3. 'Polynomial rings' and 'affine space'

The distributive category $\mathbb{F}[I]$ is the category of finite polynomial functors on $\mathbb{A}^I$. Conversely, the category $\mathbb{A}^I$ can be reconstructed from $\mathbb{F}[I]$: define a *character* on a distributive category $\mathcal{D}$ to be a functor $\mathcal{D} \rightarrow \mathbb{F}$ preserving finite sums and finite products. These form a category in which the morphisms are the finite-sum-and products-compatible natural transformations.

$\mathbb{A}^I$ is equivalent to the category of characters of $\mathbb{F}[I]$.

Indeed, the category of sum-and-product preserving functors $\mathbb{F}[I] \rightarrow \mathbb{F}$ is equivalent to the product-preserving functors $(\mathbf{FinSet}/I)^{\mathrm{op}} \rightarrow \mathbb{F}$, and since the domain is the product-completion of I, we are finally left with the category of functors $I \rightarrow \mathbb{F}$, which is what we denote $\mathbb{A}^I$.

4.4. Appropriate maps: locally finite polynomial functors

For a given set I we now have the category $\mathbb{F}[I]$ playing the role of a polynomial ring, and the category $\mathbb{A}^I$ playing the role of affine space. We proceed to assemble these objects into 2-categories **PolyAlg** and **Aff**, respectively, which will be the ambient setting for defining the algebraic structures promised in the introduction.

The 2-category **PolyAlg** is defined as follows. Its objects are the categories of the form $\mathbb{F}[I]$ (i.e. free distributive categories on a set). The 1-cells are the finite-sum-and-product-preserving functors, and the 2-cells are the compatible natural transformations (i.e. those whose component on a finite sum is the sum of the components, and similarly with finite products). The universal property of $\mathbb{F}[J]$ implies that the hom categories in **PolyAlg** are

$$\mathbf{PolyAlg}(\mathbb{F}[J], \mathbb{F}[I]) \simeq \mathbb{F}[I]^J.$$

On the other hand, the 2-category **Aff** is defined as having the categories $\mathbb{A}^I$ as objects, and as hom categories the categories of locally finite polynomial functors and their natural transformations, denoted $\mathbf{LocFinPoly}(I, J)$: A polynomial functor

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

is called *locally finite* when the maps p and t have finite fibres. This means that only finite sums and finite products are involved. It is not difficult to see that we have an equivalence of categories

$$\mathbf{Aff}(\mathbb{A}^I, \mathbb{A}^J) = \mathbf{LocFinPoly}(I, J) \simeq \mathbb{F}[I]^J.$$

One may observe that the locally finite polynomial functors can also be charaterised as those $F\colon \mathbb{A}^I \rightarrow \mathbb{A}^J$ such that for any P belonging to $\mathbb{F}[J]$, the composition $P \circ F$ belongs to $\mathbb{F}[I]$, in analogy with the definition of regular maps in algebraic geometry.

Almost tautologically we have an equivalence of 2-categories $\mathbf{Aff} \simeq \mathbf{PolyAlg}^{\mathrm{op}}$, justifying the symbols and the terminology.

4.5. The tensor product and comonoidal structure

The 2-category **Aff** has finite products: it is given by the equivalence of categories

$$\mathbb{A}^{I_1}\times\mathbb{A}^{I_2}\simeq\mathbb{A}^{I_1+I_2}.$$

Accordingly, the category **PolyAlg** has categorical sums, which we denote by $\otimes_{\mathbb{F}}$. All we need to know about it is the equivalence

$$\mathbb{F}[I_1]\otimes_{\mathbb{F}}\mathbb{F}[I_2]\simeq\mathbb{F}[I_1+I_2],$$

which is just dual to the previous display. The neutral element for this tensor product is the category $\mathbb{F}=\mathbb{F}[\emptyset]$ itself, which is an initial object in **PolyAlg**: the unique sum-and-product-preserving functor $\mathbb{F}\to\mathbb{F}[I]$ sends a finite set S to the polynomial functor

$$I\longleftarrow 0\longrightarrow S\longrightarrow 1.$$

The objects in $(\mathbf{PolyAlg},\otimes,\mathbb{F})$ are algebras. We would like to define bialgebras as comonoids in $(\mathbf{PolyAlg},\otimes,\mathbb{F})$, or rather pseudo-comonoids (i.e. coassociative only up to given coherent isomorphisms). A bialgebra is then a category $\mathbb{F}[I]$ equipped with a comonoidal structure, i.e. functors $\mathbb{F}\leftarrow\mathbb{F}[I]\to\mathbb{F}[I]\otimes\mathbb{F}[I]$. Every $\mathbb{F}[I]$ has two canonical such comonoidal structures, as we shall see in the next paragraph. The main result of this paper establishes another comonoidal structure for the special case $I=\mathsf{T}$. However, it is much more convenient to describe these structures on the other side of the duality, namely in $(\mathbf{Aff},\times,1)$, where the background structure is just the product, and we are talking about the familiar notion of monoidal structure instead of comonoidal structure.

4.6. Canonical monoidal structures on $\mathbb{A}^I$

Fix a set I. The category $\mathbb{A}^I$ has two canonical monoidal structures: pointwise sum and pointwise product. These are given by locally finite polynomial functors.

The *sum* of $x\colon I\to\mathbb{A}$ and $y\colon I\to\mathbb{A}$ is the function $x+y\colon I\to\mathbb{A}$ which sends i to $x(i)+y(i)$. Under the equivalence $\mathbb{A}^I\times\mathbb{A}^I\simeq\mathbb{A}^{I+I}$ the sum map is polynomial: it is represented by

$$I+I\longleftarrow I+I\longrightarrow I+I\longrightarrow I.$$

The corresponding comonoidal structure on $\mathbb{F}[I]$ is given by

$$\Delta\colon\ \mathbb{F}[I]\longrightarrow\mathbb{F}[I]\otimes_{\mathbb{F}}\mathbb{F}[I],\qquad x_i\longmapsto x_i\otimes 1+1\otimes x_i.$$

Here $x_i\in\mathbb{F}[I]$ denotes the polynomial functor $I\xleftarrow{\ulcorner i\urcorner}1\xrightarrow{=}1\xrightarrow{=}1$, where $\ulcorner i\urcorner\colon 1\to I$ picks out the element i. The neutral object for the sum is of course the constant empty set 0. Under the equivalence $1\simeq\mathbb{A}^0$, the corresponding map is represented by $0\leftarrow 0\to 0\to I$. The corresponding counit for the comultiplication is given by $\epsilon(x_i)=0$.

The *product* of $x\colon I\to\mathbb{A}$ and $y\colon I\to\mathbb{A}$ is the function $xy\colon I\to\mathbb{A}$ which sends i to $x(i)y(i)$. It is represented by

$$I+I\longleftarrow I+I\longrightarrow I\longrightarrow I.$$

The unit is the constant function 1. The corresponding comonoidal structure on $\mathbb{F}[I]$ is given by $\Delta(x_i)=x_i\otimes x_i$ and $\epsilon(x_i)=1$. This categorifies the construction of the monoid-algebra on the free abelian monoid on I. Before coming to the promised new monoidal structure on $\mathbb{A}^{\mathsf{T}}$ in Section 6, we need some background on trees.

5. P-trees and free monads

We briefly summarise the polynomial formalism of trees from [15].

5.1. Trees

It was observed in [15] that operadic trees can be conveniently encoded by diagrams of the same shape as polynomial functors. We take this as the definition of tree: An *operadic tree* is a diagram of finite sets

$$A \xleftarrow{s} M \xrightarrow{p} N \xrightarrow{t} A$$

satisfying the following three conditions:

(i) t is injective.

(ii) s is injective with singleton complement (called the *root* and denoted 1).

With $A = 1 + M$, define the walk-to-the-root function $\sigma\colon A \to A$ by $1 \mapsto 1$ and $e \mapsto t(p(e))$ for $e \in M$.

(iii) For all $x \in A$ there exists $k \in \mathbb{N}$ such that $\sigma^k(x) = 1$.

The elements of A are called *edges*. The elements of N are called *nodes*. For $b \in N$, the edge $t(b)$ is called the *output edge* of the node. That t is injective is just to say that each edge is the output edge of at most one node. For $b \in N$, the elements of the fibre $M_b = p^{-1}(b)$ are called *input edges* of b. Hence the whole set $M = \sum_{b \in N} M_b$ can be thought of as the set of nodes-with-a-marked-input-edge, i.e. pairs (b, e) where b is a node and e is an input edge of b. The map s returns the marked edge. Condition (ii) says that every edge is the input edge of a unique node, except the root edge. Condition (iii) says that if you walk towards the root, in a finite number of steps you arrive there. The edges not in the image of t are called *leaves*. From now on we just say *tree* for 'operadic tree'.

The tree $1 \leftarrow 0 \to 0 \to 1$ is the *trivial tree*, which we denote by $|$.

5.2. Cartesian morphisms (cf. [13])

A *cartesian morphism* of polynomial endofunctors is by definition a diagram

$$\begin{tikzcd} I' \arrow[d, "\alpha"'] & E' \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & B' \arrow[r] \arrow[d] & I' \arrow[d, "\alpha"] \\ I & E \arrow[l] \arrow[r] & B \arrow[r] & I. \end{tikzcd} \tag{3}$$

If α is an identity map, then this corresponds precisely to cartesian natural transformations between polynomial endofunctors (i.e. whose naturality squares are cartesian). In the general case, the cartesian morphisms like (3) are outside the scope of the 2-category **Poly**, but they can be reduced to it by some base change [13].

5.3. Morphisms of trees (cf. [15, § 1.1])

A *morphism of trees* is by definition a cartesian morphism of the associated polynomial endofunctors. Hence edges go to edges, and the cartesian condition means that a node of arity n is mapped to a node of arity n. The morphisms of trees are also called *tree embeddings* since in fact each of the components of such a map is automatically injective: indeed, the tree axioms above imply that a morphism necessarily commutes with the walk-to-the-root function σ, and together with arity perservation this forces the maps to be injective. Hence the category of trees and tree embeddings, denoted **TEmb**, is mostly concerned with subtrees, but note that it also contains automorphisms of trees.

A tree embedding is *root-preserving* when it sends the root to the root. In formal terms, these are diagrams (3) such that also the left-hand square is cartesian [15, 1.1.13].

An *ideal embedding* (or an *ideal subtree*) is a subtree for which every descendent edge and node is also in the tree [15, 1.1.9]. There is one ideal subtree generated by each edge in the tree. The ideal embeddings are characterised as having also the right-hand square of (3) cartesian. Ideal embeddings and root-preserving embeddings admit pushouts along each other in the category **TEmb** [15, 1.1.20]. The most interesting case is pushout over a trivial tree: this is then the root of one tree and a leaf of another tree, and the pushout is the grafting onto that leaf.

5.4. Decorated trees: P-trees

Let P denote a finitary polynomial endofunctor on $\mathbf{Set}/I$, represented by $I \leftarrow E \to B \to I$. By definition [15, § 1.2], a *P-tree* is a diagram

$$\begin{tikzcd}
A \arrow[d] & M \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & N \arrow[r] \arrow[d] & A \arrow[d] \\
I & E \arrow[l] \arrow[r] & B \arrow[r] & I,
\end{tikzcd}$$

where the top row is a tree. Unfolding the definition, we see that a P-tree is a tree whose edges are decorated in I, whose nodes are decorated in B, and with the additional structure of a bijection for each node $n \in N$ (with decoration $b \in B$) between the set of input edges of n and the fibre E_b, subject to the compatibility condition that such an edge $x \in M_n$ corresponding to $e \in E_b$ has decoration $s(e)$, and the output edge of n has decoration $t(b)$. Note that the I-decoration of the edges is completely specified by the node decoration together with the compatibility requirement, except for the case of a trivial tree.

5.5. The free monad on a polynomial endofunctor

(For the notion of monad, see [22, Chapter VI] or [1, Chapter 10].) By a *polynomial monad* we mean a monad in the 2-category of polynomial functors and cartesian natural transformations, cf. [13]. The forgetful functor from polynomial monads to polynomial endofunctors has a left adjoint, the free-monad functor. It has the following pleasing description in terms of trees, cf. [15, Proposition 1.2.8]: the free monad on a polynomial endofunctor P is the polynomial monad represented by

$$I \longleftarrow \mathsf{T}'_P \xrightarrow{q} \mathsf{T}_P \longrightarrow I,$$

where T_P is the set of (isomorphism classes of) P-trees, and T'_P is the set of (isomorphism classes of) P-trees with a marked leaf. The left-most map returns the decoration of the marked leaf; the right-most map returns the decoration of the root, and the middle map q just forgets the mark. The monad structure is given by grafting of trees, an operation that allows an easy and completely formal description in the category of trees in terms of pushouts, cf. [15, Proposition 1.1.19]. Examples of the notion of P-trees, for suitable choices of P, are planar and binary trees, or more specialised examples like trees with nodes decorated by primitive 1PI graphs of a quantum field theory [17], or the opetopes of higher category theory [18]. The polynomial endofunctor P will be held fixed throughout, and from now on we put $\mathsf{T} = \mathsf{T}_P$. We let F denote the free monad on P.

We shall need an explicit description of the monad structure of F. As in 3.6, the composition $F \circ F$, is built by the following diagram, whose constituents we now make explicit:

$$\begin{tikzcd}[column sep=tiny]
 & & & & & & \overline{\mathsf{T}}' \arrow[rrr] \arrow[dll] \arrow[dr, phantom, "\mathrm{pb}"] & & & \overset{+}{\overline{\mathsf{T}}} \arrow[rr, "p"] \arrow[dll, "\epsilon"] & & \overline{\mathsf{T}} \arrow[dd, "r"] & = q_*(\mathsf{T} \times_I \mathsf{T}') & \\
 & & & & \cdot \arrow[rrr] \arrow[dll] \arrow[dr, phantom, "\mathrm{pb}"] & & & \mathsf{T} \times_I \mathsf{T}' \arrow[dll] \arrow[drr] & & \mathrm{distr} & & & & \\
 & & \mathsf{T}' \arrow[rrr, "q"'] \arrow[dll] & & & \mathsf{T} \arrow[drr] & & \mathrm{pb} & & \mathsf{T}' \arrow[rr, "q"'] \arrow[dll] & & \mathsf{T} \arrow[drr] & & \\
I & & & F & & & & I & & & F & & & I
\end{tikzcd}$$

The construction starts from the bottom by forming the central pullback square: $\mathsf{T} \times_I \mathsf{T}'$ is the set of pairs (S, R) such that the root of S is of the same type as the marked leaf of R, in other words a simple grafting of one tree onto another. This can also be interpreted as the set of trees with one marked edge: the two projections are then: return the ideal subtree generated by that edge (that's S), and return the root-preserving tree obtained by pruning that ideal subtree (that's R).

Next we compute $\overline{\mathsf{T}} = q_*(\mathsf{T} \times_I \mathsf{T}')$. (Note that this set can also be described as $F(\mathsf{T})$.) By definition of the lowerstar operation, the result is a set over T whose fibre over a fixed tree $R \in \mathsf{T}$ is the set of maps from the set of leaves of R to the set of all trees with matching root. This data is equivalent to the grafting of all those leaf-indexed trees onto the leaves of R. Alternatively this data amounts to the inclusion of R into the big tree resulting from the graftings. In other words, $\overline{\mathsf{T}}$ is the set of trees with a cut, and the map $r \colon \overline{\mathsf{T}} \to \mathsf{T}$ returns the tree found below the cut.

The pullback along q is the set $\overset{+}{\mathsf{T}}$ of pairs consisting of a marked tree and graftings onto all leaves, which amounts to trees with a pointed cut; the map $p \colon \overset{+}{\mathsf{T}} \to \overline{\mathsf{T}}$ just forgets the mark in the cut. The evaluation map $\epsilon \colon \overset{+}{\mathsf{T}} \to \mathsf{T} \times_I \mathsf{T}'$ is described like this: if we think of the elements of $\overset{+}{\mathsf{T}}$ as given by a tree together with a marked leaf and a map from each leaf to T (with matching root), then the map ϵ simply applies that map to the marked leaf, hence obtaining a single tree with correct root, and hence an element in $\mathsf{T} \times_I \mathsf{T}'$. (We don't really need the remaining left-hand part of the diagram, but here it is, for completeness: the upper left-hand corner is the set of trees with a pointed cut and a marked leaf in the subtree corresponding to the point. (Since that marked leaf together with the cut automatically provides a point on the cut it is enough to say: tree with a cut and a marked leaf.) The set just below it is the set of trees with a marked edge and a marked descendant leaf.)

Finally, we describe the structure maps of the free monad F: the multiplication $F \circ F \Rightarrow F$ is the cartesian natural transformation

$$\begin{tikzcd}
F \circ F: \arrow[d, Rightarrow] & I \arrow[d, equal] & \overline{\mathsf{T}}' \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, very near start, "\lrcorner"] & \overline{\mathsf{T}} \arrow[r] \arrow[d, "m"] & I \arrow[d, equal] \\
F: & I & \mathsf{T}' \arrow[l] \arrow[r] & \mathsf{T} \arrow[r] & I,
\end{tikzcd}$$

where the maps in the middle simply forget the cut. The unit for the monad, $\mathrm{Id} \Rightarrow F$ is given essentially by the map $e \colon I \to \mathsf{T}$ assigning to each 'colour' $i \in I$ the trivial tree with edge of colour i.

We summarise the maps r, p, f, m, for later use:

$$\begin{tikzcd}
 & & \overset{+}{\mathsf{T}} \arrow[r, "p"] \arrow[dl] \arrow[ddll, bend right, "f"'] & \overline{\mathsf{T}} \arrow[dd, "r"] \arrow[dddd, bend left, "m"] & = F(\mathsf{T}) \\
 & \mathsf{T} \times_I \mathsf{T}' \arrow[dl] \arrow[dr] & & & \\
\mathsf{T} \arrow[dr] & & \mathsf{T}' \arrow[r] \arrow[dl] & \mathsf{T} \arrow[dr] & \\
 & I & F & & I \\
 & & & \mathsf{T} \arrow[ur] &
\end{tikzcd}$$

6. New monoidal structure on $\mathbb{A}^{\mathsf{T}}$

In this section the symbol T stands for the set of isomorphism classes of P-trees, for any fixed finitary polynomial endofunctor P (and similarly for the decorated symbols, $\overline{\mathsf{T}}$, etc.). The construction of the monoidal structure works also for abstract trees, but then the various sets and maps no longer come from the free-monad construction, and need to be defined in a more ad hoc manner. We discuss this in the next section.

6.1. Theorem

The polynomial functor $M\colon \mathbb{A}^{\mathsf{T}} \times \mathbb{A}^{\mathsf{T}} \to \mathbb{A}^{\mathsf{T}}$ defined by

$$\begin{tikzcd} & \overset{+}{\overline{\mathsf{T}}} + \overline{\mathsf{T}} \arrow[ld, "f+r"'] \arrow[r, "\langle p{,}\overline{\mathsf{T}}\rangle"] & \overline{\mathsf{T}} \arrow[rd, "m"] & \\ \mathsf{T}+\mathsf{T} & & M & \mathsf{T} \end{tikzcd}$$

is a monoidal structure on $\mathbb{A}^{\mathsf{T}}$. Its unit is the functor $U\colon \mathbb{A}^0 \to \mathbb{A}^{\mathsf{T}}$ given by

$$\begin{tikzcd} & 0 \arrow[ld] \arrow[r] & I \arrow[rd, "e"] & \\ 0 & & U & \mathsf{T}. \end{tikzcd}$$

6.2. Description of the maps

An interesting feature of this monoidal structure is that all the involved maps are recognised as those occurring in the polynomial description of the free-monad construction. We review the definitions of the sets and maps:

- T is the set of iso-classes of P-trees. If F denotes the free monad on P, then $\mathsf{T} = F(1)$.
- The set $\overline{\mathsf{T}}$ is the set of iso-classes of P-trees with a cut. It appears as $F(\mathsf{T}) = FF(1)$. More formally, the set of iso-classes of trees with a subtree containing the root.
- The map $m\colon \overline{\mathsf{T}} \to \mathsf{T}$ is the multiplication of the monad, $m\colon FF(1) \to F(1)$, i.e. forget the cut.
- The set $\overset{+}{\overline{\mathsf{T}}}$ is the set of iso-classes of P-trees with a pointed cut, and p is the map that forgets the point. More formally,
$$\overset{+}{\overline{\mathsf{T}}} = \mathsf{T}' \times_{\mathsf{T}} \overline{\mathsf{T}} = q^* \overline{\mathsf{T}},$$
where T' is the set of iso-classes of P-trees with a marked leaf (it appears as the total space of $q\colon \mathsf{T}' \to \mathsf{T}$ in the diagram representing F), and $p\colon \overset{+}{\overline{\mathsf{T}}} \to \overline{\mathsf{T}}$ is the projection.
- The map $r\colon \overline{\mathsf{T}} \to \mathsf{T}$ returns the P-tree below the cut.
- $f\colon \overset{+}{\overline{\mathsf{T}}} \to \mathsf{T}$ returns the ideal subtree generated by the edge marked by the pointed cut. This is the most complicated map to explain formally, but it occurs already in the diagram for $F \circ F$, and is composed of the T-component of the counit of the $q^* \dashv q_*$ adjunction (that's the evaluation map $\epsilon\colon q^*\overline{\mathsf{T}} \to \mathsf{T} \times_I \mathsf{T}'$) followed by the projection to T.
- Finally, $e\colon I \to \mathsf{T}$ assigns to each 'colour' $i \in I$ the trivial tree with edge of colour i.

Note that here and throughout, the name of a set is also used to indicate its identity map. So, for example, $\langle p, \overline{\mathsf{T}}\rangle$ denotes the map whose first component is p and whose second component is the identity map on $\overline{\mathsf{T}}$.

Note that the polynomial is a 'sum-wise tensor product' of a non-linear functor (in the first set of variables):

$$\begin{tikzcd} & \overset{+}{\overline{\mathsf{T}}} \arrow[ld, "f"'] \arrow[r, "p"] & \overline{\mathsf{T}} \arrow[rd, "m"] & \\ \mathsf{T} & & & \mathsf{T} \end{tikzcd}$$

(which concerns taking all cuts and retaining the forest of cut-off branches), and a linear functor (in the second set of variables):

$$\begin{tikzcd}
 & \overline{T} \arrow[r, "="] \arrow[dl, "r"'] & \overline{T} \arrow[dr, "m"] & \\
T & & & T
\end{tikzcd}$$

(which is about taking all cuts and retaining the bottom tree, which of course is a single tree, hence the linearity).

6.3. Proof of associativity

In the following diagram, which commutes strictly, the left-hand vertical polynomial is $T + M$, and the bottom is M. Similarly, the top polynomial is $M + T$ and the right-hand vertical polynomial is M. We show that both composites are naturally isomorphic to the polynomial indicated by the diagonal (to be detailed in the proof). In both cases this amounts to performing the constructions as in 3.6. This natural isomorphism provides the associator, which is part of the monoidal structure. (The pentagon equation for the associator is not established explicitly. We invoke instead a general coherence principle: since the associator is constructed by canonical isomorphisms (Beck–Chevalley and distributivity), it is coherent.)

$$\begin{tikzcd}
T+T+T & & (\overset{+}{T}+\overline{T})+T \arrow[ll, "\Delta"'] \arrow[r, "\Pi"] & \overline{T}+T \arrow[r, "\Sigma"] & T+T \\
 & & \cdot \arrow[u, "\Delta"] \arrow[r, "\Pi"] \arrow[ur, phantom, "\text{pb}"] & \cdot \arrow[u, "\Delta"] \arrow[r, "\Sigma"] \arrow[ur, phantom, "\text{pb}"] & \overset{+}{T}+\overline{T} \arrow[u, "\Delta"'] \arrow[dd, "\Pi"] \\
T+(\overset{+}{T}+\overline{T}) \arrow[uu, "\Delta"] \arrow[d, "\Pi"'] \arrow[dr, phantom, "\text{pb}"] & \cdot \arrow[l, "\Delta"'] \arrow[d, "\Pi"'] \arrow[dr, phantom, "\text{pb}"] & \cdot \arrow[l, "\Delta"'] \arrow[u, "\Delta"] \arrow[r, "\Pi"] \arrow[d, "\Pi"'] \arrow[uull, dashed, "\Delta"] \arrow[dr, dashed, "\Pi"] \arrow[ur, phantom, "\text{pb}"] & \cdot \arrow[u, "\Delta"'] \arrow[d, "\Pi"] \arrow[r, phantom, "\text{distr}"] & \\
T+\overline{T} \arrow[d, "\Sigma"'] \arrow[dr, phantom, "\text{pb}"] & \cdot \arrow[l, "\Delta"] \arrow[d, "\Sigma"'] & \cdot \arrow[l, "\Delta"] \arrow[r, "\Pi"'] \arrow[d, phantom, "\text{distr}"] & \overline{\overline{T}} \arrow[r, "\Sigma"] \arrow[d, "\Sigma"'] \arrow[dr, dashed, "\Sigma"] & \overline{T} \arrow[d, "\Sigma"] \\
T+T & \overset{+}{T}+\overline{T} \arrow[l, "\Delta"] \arrow[rr, "\Pi"'] & & \overline{T} \arrow[r, "\Sigma"'] & T
\end{tikzcd}$$

We start with the lower left-hand composite. The first step is to take a pullback in the lower left-hand corner:

$$\begin{tikzcd}
T+\overline{T} \arrow[d, "T+m"'] \arrow[dr, phantom, "\text{pb}"] & \overset{+}{T}+\overline{\overline{T}} \arrow[l, "f+r"'] \arrow[d, "\overset{+}{T}+\widetilde{m}"] \\
T+T & \overset{+}{T}+\overline{T} \arrow[l, "f+r"]
\end{tikzcd}$$

Here $\overline{\overline{T}}$ is the set of trees with two non-crossing cuts, or more formally, a sequence of two root-preserving inclusions $R \subset S \subset T$. The upper cut is the one coming from the lower right-hand corner of the diagram, and the lower cut comes from the upper left-hand corner of the diagram. Hence the map denoted as $\widetilde{m}$ forgets the lower cut.[2]

[2] *Typographical note: if colour output is available, as an extra visual aid, the upper cut and the maps related to it are printed in red, while the lower cut and its maps in blue. The wording is, however, intended to be sufficient for also a black-and-white printing to make sense.*

The right-hand map $\overset{+}{\mathsf{T}} + \widetilde{m}$ we shall now lowerstar along $\langle p, \overline{\mathsf{T}} \rangle$ to complete the distributivity pentagon:

$$\begin{tikzcd}
\overset{+}{\mathsf{T}} + \overline{\overline{\mathsf{T}}} \arrow[d, "\overset{+}{\mathsf{T}}+\widetilde{m}"'] & \cdot \arrow[l] \arrow[r] & \overline{\overline{\mathsf{T}}} \arrow[d, "\widetilde{m}"] \\
\overset{+}{\mathsf{T}} + \overline{\mathsf{T}} \arrow[rr, "\langle p,\overline{\mathsf{T}}\rangle"'] & & \overline{\mathsf{T}}.
\end{tikzcd}$$

The resulting right-hand map is just $\widetilde{m}$. Indeed, the lowerstar operation amounts to multiplying along the fibres. But since in the left-hand summand the map we lowerstar is just the identity of $\overset{+}{\mathsf{T}}$, no contribution comes from this summand, and in the right-hand summand we are lowerstarring along the identity map, hence the result is just $\widetilde{m}$, the map that forgets the lower cut.

We have now shown that the set $\overline{\overline{\mathsf{T}}}$ appears at the crucial point of the diagonal polynomial when constructed from the lower left-hand side. We proceed to construct it also from the upper right-hand side of the big diagram. The argument is different because of the non-symmetry of the tensor product.

The pullback square in the upper right-hand corner is, typographically transposed,

$$\begin{tikzcd}
\overline{\mathsf{T}} + \mathsf{T} \arrow[d, "m+\mathsf{T}"'] \arrow[dr, phantom, "\text{pb}"] & \overset{\mp}{\mathsf{T}} + \overline{\mathsf{T}} \arrow[l, "f+r"'] \arrow[d, "\bar{m}+\overline{\mathsf{T}}"] \\
\mathsf{T} + \mathsf{T} & \overset{+}{\mathsf{T}} + \overline{\mathsf{T}}. \arrow[l, "f+r"]
\end{tikzcd}$$

Here there are two new symbols: $\overset{\mp}{\mathsf{T}}$ denotes the set of trees with a marked cut and a further cut in the tree above the mark. The map $\bar{m}$ forgets the 'short' cut.

We shall now lowerstar the map $\bar{m} + \overline{\mathsf{T}}$ along $f + r$, in order to complete the distributivity pentagon

$$\begin{tikzcd}
\overset{\mp}{\mathsf{T}} + \overline{\mathsf{T}} \arrow[d, "\bar{m}+\overline{\mathsf{T}}"'] & \cdot \arrow[l] \arrow[r] & \overline{\overline{\mathsf{T}}} \arrow[d, "\overline{m}"] \\
\overset{+}{\mathsf{T}} + \overline{\mathsf{T}} \arrow[rr, "\langle p,\overline{\mathsf{T}}\rangle"'] & & \overline{\mathsf{T}}.
\end{tikzcd}$$

The claim is that the right-hand map is just $\overline{m}$, the map that forgets the upper cut. We compute this fibre-wise over an element $R \subset T$ in $\overline{\mathsf{T}}$. The $\langle p, \overline{\mathsf{T}} \rangle$-fibre has one element for each leaf e of R. For each leaf e, the $\bar{m}$-fibre consists of the possible cuts in the ideal subtree $D_e \subset T$ generated by e. Lowerstarring means multiplying these fibres, so it amounts to giving a cut in D_e for each leaf e of R. Altogether, this amounts to giving a total cut in T above the original cut $R \subset T$. This is once again the set $\overline{\overline{\mathsf{T}}}$, but the projection $\overline{m}\colon \overline{\overline{\mathsf{T}}} \to \overline{\mathsf{T}}$ this time forgets the upper cut. (Note that there is no contribution from the right-hand summand, as it has trivial fibres.)

Finally, note that the square appearing where the two constructions meet,

$$\begin{tikzcd}
\overline{\overline{\mathsf{T}}} \arrow[r, "\overline{m}"] \arrow[d, "\widetilde{m}"'] & \overline{\mathsf{T}} \arrow[d, "m"] \\
\overline{\mathsf{T}} \arrow[r, "m"'] & \mathsf{T}
\end{tikzcd}$$

commutes, since it amounts to forgetting first the upper cut and then the lower, or the other way around.

The remaining part of the big diagram is only about taking pullbacks, and presents no difficulties.

6.4. Proof of the unit axiom

The unit is the functor $U\colon \mathbb{A}^0 \to \mathbb{A}^{\mathsf{T}}$ represented by $0 \leftarrow 0 \to I \to \mathsf{T}$, where the last map associates to an edge colour the corresponding trivial tree. That this is the unit object means that when left-added to the identity functor on T,

$$0 + \mathsf{T} \longleftarrow 0 + \mathsf{T} \longrightarrow I + \mathsf{T} \longrightarrow \mathsf{T} + \mathsf{T},$$

and composing with M yields the identity functor. And similarly of course with right-adding the identity functor. Checking this amounts to filling the following big diagram:

$$\begin{tikzcd}
\mathsf{T} & & \mathsf{T} \arrow[ll, "\Delta"'] \arrow[r, "\Pi"] \arrow[dr, phantom, "\mathrm{pb}"] & \mathsf{T}+I \arrow[r, "\Sigma"] \arrow[dr, phantom, "\mathrm{pb}"] & \mathsf{T}+\mathsf{T} \\
 & & \cdot \arrow[u, "\Delta"] \arrow[r, "\Pi"] \arrow[dr, phantom, "\mathrm{pb}"] & \cdot \arrow[u, "\Delta"] \arrow[r, "\Sigma"] \arrow[ddr, phantom, "\mathrm{distr}"] & \overset{+}{\mathsf{T}}+\overline{\mathsf{T}} \arrow[u, "\Delta"'] \arrow[dd, "\Pi"] \\
\mathsf{T} \arrow[uu, "\Delta"] \arrow[d, "\Pi"'] \arrow[dr, phantom, "\mathrm{pb}"] & \cdot \arrow[l, "\Delta"'] \arrow[d, "\Pi"] \arrow[dr, phantom, "\mathrm{pb}"] & \mathsf{T} \arrow[uull, equal] \arrow[l, "\Delta"'] \arrow[u, "\Delta"] \arrow[r, "\Pi"] \arrow[d, "\Pi"] \arrow[dr, equal] & \cdot \arrow[u, "\Delta"] \arrow[d, "\Pi"] & \\
I+\mathsf{T} \arrow[d, "\Sigma"'] \arrow[dr, phantom, "\mathrm{pb}"] & \cdot \arrow[l, "\Delta"] \arrow[d, "\Sigma"] \arrow[drr, phantom, "\mathrm{distr}"] & \cdot \arrow[l, "\Delta"] \arrow[r, "\Pi"'] & \mathsf{T} \arrow[r, "\Sigma"] \arrow[d, "\Sigma"'] \arrow[dr, equal] & \overline{\mathsf{T}} \arrow[d, "\Sigma"] \\
\mathsf{T}+\mathsf{T} & \overset{+}{\mathsf{T}}+\overline{\mathsf{T}} \arrow[l, "\Delta"] \arrow[rr, "\Pi"'] & & \overline{\mathsf{T}} \arrow[r, "\Sigma"'] & \mathsf{T}.
\end{tikzcd}$$

As with associativity, we content ourselves with checking that the two constructions meet at the distributivity squares. We start at the lower left-hand corner with a pullback:

$$\begin{tikzcd}
I+\mathsf{T} \arrow[d, "e+\mathsf{T}"'] \arrow[dr, phantom, "\mathrm{pb}"] & V+\overline{\mathsf{T}} \arrow[l, "f+r"'] \arrow[d, "v+\overline{\mathsf{T}}"] \\
\mathsf{T}+\mathsf{T} & \overset{+}{\mathsf{T}}+\overline{\mathsf{T}}. \arrow[l, "f+r"]
\end{tikzcd}$$

The set V appearing is the set of trees with a marked cut such that the mark itself is a leaf (in other words, the upper tree is a trivial tree).[3] The map $v\colon V \to \overset{+}{\mathsf{T}}$ is the inclusion of these special pointed cuts into the set of all pointed cuts. Now lowerstar along $\langle p, \overline{\mathsf{T}} \rangle$ to complete the distributivity pentagon:

$$\begin{tikzcd}
V+\overline{\mathsf{T}} \arrow[d, "v+\overline{\mathsf{T}}"'] & \cdot \arrow[l] \arrow[r] & W \; = \mathsf{T} \arrow[d, "w"] \\
\overset{+}{\mathsf{T}}+\overline{\mathsf{T}} \arrow[rr, "\langle p, \overline{\mathsf{T}} \rangle"'] & & \overline{\mathsf{T}}.
\end{tikzcd}$$

[3] *The author ran out of onomatopoetic notation at this point, and just chose the letter V at random.*

This amounts to requiring for each mark of the cut that the tree above it is trivial; in other words it is the set W of cuts that only take leaves, i.e. the set of trivial root-preserving inclusions, i.e. the set T itself. (The map $w\colon W \to \overline{\mathsf{T}}$ is of course the inclusion of this particular kind of cuts into the set of all cuts.)

The check is similar for the right-hand unit axiom, except that the set in the corner is the set of trees with the root cut, which again is naturally identified with T itself.

6.5. Bialgebra of polynomial functors

By the duality of 4.4, the monoidal structure on $\mathbb{A}^{\mathsf{T}}$ induces a comonoidal structure on $\mathbb{F}[\mathsf{T}]$,

$$\mathbb{F} \xleftarrow{\epsilon} \mathbb{F}[\mathsf{T}] \xrightarrow{\Delta} \mathbb{F}[\mathsf{T}] \otimes_{\mathbb{F}} \mathbb{F}[\mathsf{T}].$$

The comultiplication is simply defined by precomposition with M and the counit is defined by precomposition with U. In detail, if $F \in \mathbb{F}[\mathsf{T}]$ is a polynomial functor in T-many variables,

$$\mathsf{T} \longleftarrow E \longrightarrow B \longrightarrow 1.$$

we compute $\Delta(F)$ as the composite

$$\begin{tikzcd}
 & \overset{+}{\overline{\mathsf{T}}} + \overline{\mathsf{T}} \arrow[r, "\langle p{,}\overline{\mathsf{T}}\rangle"] \arrow[dl, "f+r"'] & \overline{\mathsf{T}} \arrow[dr, "m"] & & E \arrow[r] \arrow[dl] & B \arrow[dr] & \\
\mathsf{T}+\mathsf{T} & & & \mathsf{T} & & & 1.
\end{tikzcd}$$

Note that the comonoidal structure is automatically multiplicative, for formal reasons: multiplication in $\mathbb{F}[\mathsf{T}]$ is dual to the diagonal functor $\mathbb{A}^{\mathsf{T}} \to \mathbb{A}^{\mathsf{T}} \times \mathbb{A}^{\mathsf{T}}$, and every monoidal structure on $\mathbb{A}^{\mathsf{T}}$ is compatible with this diagonal.

It is instructive to calculate Δ on a multiplicative generator of $\mathbb{F}[\mathsf{T}]$: these are the single-variable polynomials

$$\mathsf{T} \xleftarrow{\ulcorner T \urcorner} 1 \xrightarrow{=} 1 \xrightarrow{=} 1,$$

where the map $\ulcorner T \urcorner$ picks out the tree $T \in \mathsf{T}$. The composition in this case just amounts to taking fibres, so the result is

$$\begin{tikzcd}
 & (\overset{+}{\overline{\mathsf{T}}})_T + (\overline{\mathsf{T}})_T \arrow[rr, "\langle p_T{,}(\overline{\mathsf{T}})_T\rangle"] \arrow[dl, "f+r"'] & & (\overline{\mathsf{T}})_T \arrow[dr] & \\
\mathsf{T}+\mathsf{T} & & & & 1.
\end{tikzcd}$$

Here $(\overline{\mathsf{T}})_T$ is the set of cuts in the specific tree T, and the fibre over that is a two-component set where the left-hand component is the set of all pointed cuts on T, and the right-hand component is again the set of cuts on T. Spelling out what the polynomial does as a functor, we see that it is exactly the functorial version of the formula from 2.5: we have to sum over the set $(\overline{\mathsf{T}})_T$ (that's the set of cuts), then multiply the fibres for each cut c; the fibre has an element for each edge in the cut (and one extra): for each edge we must apply f, which gives the branch factor of P_c, and multiply all these, getting altogether what is denoted as P_c, and for the extra point of the fibre we must apply r, which gives the tree R_c below the cut, which is to be placed in the right-hand factor.

The counit associates to each tree T a polynomial in zero variables, i.e. just a finite set. If the tree is represented as $\ulcorner T \urcorner\colon 1 \to \mathsf{T}$, we need to compose

$$\begin{tikzcd}
 & 0 \arrow[r] \arrow[dl] & I \arrow[dr] & & 1 \arrow[dl] \arrow[r] & 1 \arrow[dr] & \\
0 & & & \mathbf{T} & & & 1.
\end{tikzcd}$$

The result is

$$\begin{tikzcd}
 & 0 \arrow[r] \arrow[dl] & I \times_{\mathbf{T}} 1 \arrow[dr] & \\
0 & & & 1,
\end{tikzcd}$$

i.e. the set $I \times_{\mathbf{T}} 1$, which is the singleton set 1 if T belongs to I and the empty set 0 otherwise. Again it is clear that this is the functorial analogue of the specification in Section 2.

7. Beyond P-trees

7.1. Trees (irrespective of any P)

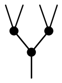
We have worked with P-trees for two reasons: the first is to allow the connection with the free-monad construction. The second is the fact that P-trees (in the set-based setting we are working in) are rigid. This guarantees that taking sets of isomorphism classes is well-behaved.

Abstract trees, in the sense of 5.1, are not P-trees for any polynomial endofunctor over **Set**. The 'terminal endofunctor' in the category of polynomial endofunctors and their cartesian morphisms (3), which should make the abstract trees P-trees, does not exist. Also, trees clearly may have non-trivial automorphisms, so taking sets of isomorphism classes may be prone to errors, and care is needed.

7.2. Reinterpretation of the sets and maps in Section 6

In spite of the above remarks, it is nevertheless possible to mimic all the constructions of the previous section, if just the involved sets and set maps are defined a little bit differently.

The starting point is the same: we now let $\mathbf{T}$ denote the set of isomorphism classes of abstract trees. Hence the basis for the bialgebra is the same as used already in Section 2. But while for P-trees, the set $\overline{\mathbf{T}}$ can be described as isomorphism classes of root-preserving inclusions, this description does not work for abstract trees. Indeed, it is crucial that the fibres of the projection $m\colon \overline{\mathbf{T}} \to \mathbf{T}$ have the correct cardinality: for example, the abstract tree



certainly has five different cuts, but two of them are isomorphic as abstract cuts. So instead of defining $\overline{\mathbf{T}}$ as the set of isomorphism classes of cuts, it is necessary first to choose a representative for each iso-class in $\mathbf{T}$, and then for each such representative T define m^{-1} to be the set of cuts of that specific tree T, and finally let $\overline{\mathbf{T}}$ be the disjoint union of all those fibres. Similar care is needed to define $\mathbf{T}'$ and $\overset{+}{\overline{\mathbf{T}}}$, and the other symbols involved. The fact that all the symbols are defined fibrewise over $\mathbf{T}$, instead of being defined abstractly in terms of isomorphism classes, guarantees that all the pullback and lowerstar operations (which are fibrewise operations) yield the expected results, and one can check that the proof of associativity and the unit axiom work equally well for abstract trees, with these provisos.

7.3. Groupoids instead of sets

Clearly the remedies just explained are rather ad hoc. A better and more conceptual way to account for abstract trees and P-trees on equal footing consists in upgrading the theory from sets to groupoids. This can be seen as one further step of categorification.

First of all it is necessary (and possible) to upgrade the theory of polynomial functors from sets to groupoids: this means that the representing diagrams for polynomial functors

$$I \longleftarrow E \longrightarrow B \longrightarrow J$$

should then be (suitably homotopically finite) groupoids, and the functors themselves go between slices of the category of groupoids (slices being taken in the homotopical sense). With the appropriate adjustments, everything works as for sets: pullbacks and fibres have to be homotopy pullbacks and homotopy fibres, if the maps are not groupoid fibrations; sums are replaced by slightly fancier colimits, and everything is up to equivalence of groupoids instead of isomorphism of sets. This theory correctly incorporates all questions of symmetries of objects [16].

In this setting, abstract trees are E-trees for the 'exponential functor'

$$E(X) = \sum_{n\in\mathbb{N}} X^n / \mathrm{Aut}[n],$$

represented by the groupoid diagram

$$1 \longleftarrow \mathbb{B}' \longrightarrow \mathbb{B} \longrightarrow 1,$$

where $\mathbb{B}$ is the groupoid of finite sets and bijections, $\mathbb{B}'$ the groupoid of finite pointed sets and basepoint-preserving bijections, and where the quotient is the weak groupoid quotient (sewing in paths instead of collapsing). See Baez–Dolan [2] for further discussion of such constructions, and the exponential functor in particular.

In the groupoid setting, one works with the groupoid of trees, instead of with its set of isomorphism classes. The practical benefit, in the presence of automorphisms of objects, is the same as working with moduli stacks rather than coarse moduli spaces in algebraic geometry.

The choice in this paper to work with sets is first of all that this theory is already available. Second, it is preferable to expose the essential ideas and constructions in the transparent rigid set-up, so as not to burden the arguments with homotopy theory. However, to ease the eventual upgrade to groupoids, the exposition is careful to perform the arguments in a clean, functorial language, favouring natural isomorphisms over element-based constructions.

In fact, the theory of polynomial functors is currently being worked out for infinity-groupoids, in joint work with David Gepner. One important insight is that ∞-groupoids play just the same role for ∞-categories as sets play for categories. Seeing how the theory goes for sets is therefore an important first step for the more general case of groupoids and ∞-groupoids.

Acknowledgements

The author has benefited greatly from many conversations with Kurusch Ebrahimi-Fard, and also thanks the anonymous referees for many pertinent remarks which led to improved exposition. Partial support from research grants MTM 2009-10359 and MTM 2010-20692 of Spain and 2009 SGR-1092 of Catalonia is gratefully acknowledged.

References

[1] Awodey S., Category Theory, Oxford Logic Guides, 49, Clarendon Press, Oxford University Press, New York, 2006

[2] Baez J.C., Dolan J., From finite sets to Feynman diagrams, In: Mathematics Unlimited—2001 and Beyond, Springer, Berlin, 2001, 29–50

[3] Bergbauer C., Kreimer D., Hopf algebras in renormalization theory: locality and Dyson–Schwinger equations from Hochschild cohomology, In: Physics and Number Theory, IRMA Lect. Math. Theor. Phys., 10, European Mathematical Society, Zürich, 2006, 133–164

[4] Brouder Ch., Runge–Kutta methods and renormalization, Eur. Phys. J. C Part. Fields, 2000, 12(3), 521–534

[5] Butcher J.C., An algebraic theory of integration methods, Math. Comp., 1972, 26(117), 79–106

[6] Carboni A., Lack S., Walters R.F.C., Introduction to extensive and distributive categories, J. Pure Appl. Algebra, 1993, 84(2), 145–158

[7] Chapoton F., Livernet M., Pre-Lie algebras and the rooted trees operad, Int. Math. Res. Not. IMRN, 2001, 8, 395–408

[8] Connes A., Kreimer D., Hopf algebras, renormalization and noncommutative geometry, Comm. Math. Phys., 1998, 199(1), 203–242

[9] Connes A., Kreimer D., Renormalization in quantum field theory and the Riemann–Hilbert problem I: The Hopf algebra structure of graphs and the main theorem, Comm. Math. Phys., 2000, 210(1), 249–273

[10] Crane L., Yetter D.N., Examples of categorification, Cahiers Topologie Géom. Différentielle Catég., 1998, 39(1), 3–25

[11] Figueroa H., Gracia-Bondía J.M., Combinatorial Hopf algebras in quantum field theory I, Rev. Math. Phys., 2005, 17(8), 881–976

[12] Gálvez-Carrillo I., Kock J., Tonks A., Groupoids and Faà di Bruno formulae for Green functions in bialgebras of trees, preprint available at http://arxiv.org/abs/1207.6404

[13] Gambino N., Kock J., Polynomial functors and polynomial monads, Math. Proc. Cambridge Philos. Soc. (in press), DOI: 10.1017/S0305004112000394

[14] Hartshorne R., Algebraic Geometry, Grad. Texts in Math., 52, Springer, New York–Heidelberg, 1977

[15] Kock J., Polynomial functors and trees, Int. Math. Res. Not. IMRN, 2011, 3, 609–673

[16] Kock J., Data types with symmetries and polynomial functors over groupoids, In: Proceedings of the 28th Conference on the Mathematical Foundations of Programming Semantics, Bath, 2012, Electronic Notes in Theoretical Computer Science, 286, preprint available at http://arxiv.org/abs/1210.0828

[17] Kock J., Categorical formalisms of graphs and trees in quantum field theory (in preparation)

[18] Kock J., Joyal A., Batanin M., Mascari J.-F., Polynomial functors and opetopes, Adv. Math., 2010, 224(6), 2690–2737

[19] Kreimer D., On the Hopf algebra structure of perturbative quantum field theories, Adv. Theor. Math. Phys., 1998, 2(2), 303–334

[20] Kreimer D., On overlapping divergences, Comm. Math. Phys., 1999, 204(3), 669–689

[21] Lawvere F.W., Menni M., The Hopf algebra of Möbius intervals, Theory Appl. Categ., 2010, 24(10), 221–265

[22] Mac Lane S., Categories for the Working Mathematician, 2nd ed., Grad. Texts in Math., 5, Springer, New York, 1998

[23] Moerdijk I., On the Connes–Kreimer construction of Hopf algebras, In: Homotopy Methods in Algebraic Topology, Boulder, June 20–24, 1999, Contemp. Math., 271, American Mathematical Society, Providence, 2001, 311–321

[24] Weber M., Polynomials in categories with pullbacks, preprint available at http://arxiv.org/abs/1106.1983