Higher Structures 1(1):22–55, 2017.

HIGHER STRUCTURES

A formal language for cyclic operads

Pierre-Louis Curien[a] **and Jovana Obradović**[a]

[a]*IRIF, Université Paris Diderot and πr^2 team, Inria, France*

Abstract

We propose a λ-calculus-style formal language, called the μ-syntax, as a lightweight representation of the structure of cyclic operads. We illustrate the rewriting methods behind the formalism by giving a complete step-by-step proof of the equivalence between the unbiased and biased definitions of cyclic operads.

Communicated by: Joachim Kock.
Received: 27th October, 2016. Accepted: 24th May, 2017.
MSC: 18D50, 68Q42.
Keywords: operad, cyclic operad, unrooted trees, syntax, rewriting system, formalisation.

Introduction

In the spirit of recent years' movement in bringing closer mathematics and computer science communities through formalisation of mathematics, this paper proposes a λ-calculus-style formal language, called the μ-syntax, as a lightweight representation of the cyclic operad structure.

The name and the language of the μ-syntax formalism were motivated by another formal syntactical tool, the $\mu\tilde{\mu}$-subsystem of the $\overline{\lambda}\mu\tilde{\mu}$-calculus, presented by Curien and Herbelin in [3]. In their paper, programs are described by means of expressions called commands, of the form

$$\langle \mu\beta.c_1 \,|\, \tilde{\mu}x.c_2 \rangle,$$

which exhibit a computation as the result of an interaction between a term $\mu\beta.c_1$ and an evaluation context $\tilde{\mu}x.c_2$, together with a symmetric reduction system

$$c_2[\mu\beta.c_1/x] \longleftarrow \langle \mu\beta.c_1 \,|\, \tilde{\mu}x.c_2 \rangle \longrightarrow c_1[\tilde{\mu}x.c_2/\beta],$$

accounting for the symmetry of calling mechanisms in programming languages. In our syntactical approach, we follow this idea and view operadic composition as such a program, i.e. as an interaction between two operations f and g, where f, considered as a context for g, provides an

Email addresses: curien@irif.fr (Pierre-Louis Curien)
jovana@irif.fr (Jovana Obradović)

DOI: 10.21136/HS.2017.02

input x (selected with $\tilde{\mu}$) for the output β of g (marked with μ). By moving this concept to the *entries-only framework of cyclic operads* [14, Definition 48], in which an operation, instead of having inputs and an output, now has only *entries*, and can be composed with another operation along any of them, the input/output distinction of the $\mu\tilde{\mu}$-subsystem goes away, leading to the existence of a *single binding operator* μ, whose purpose is to select the entries of two operations which are to be connected in this interaction.

Concretely, the pattern $\langle \mu x.\_ \,|\, \mu y.\_ \rangle$ encodes the partial composition operation $(-)_x{\circ}_y(-)$. Hence, from the tree-wise perspective, $\langle \mu x.\_ \,|\, \mu y.\_ \rangle$ encodes the unrooted tree obtained by grafting two unrooted trees along entries (or half-edges, or flags) x and y. For those combinatorially oriented, this construction (and, in particular, the syntactic concept of binding) can also be understood in terms of differentiation of species of Joyal [7], as a mapping $\partial S \cdot \partial S \to S$, where ∂S is the derivative of the species S and $\cdot$ denotes the product of species. In fact, in [18], cyclic operads are defined internally to the category of species by using precisely this mapping, and an equivalence with the representation by means of individual composition operations $(-)_x{\circ}_y(-)$ (and, therefore, with $\langle \mu x.\_ \,|\, \mu y.\_ \rangle$) is set up. In addition to commands of the form $\langle \mu x.\_ \,|\, \mu y.\_ \rangle$, which describe *partial* grafting of two unrooted trees, the μ-syntax features another kind of commands, whose shape is $(-)\{\mu x.\_, \ldots, \mu y.\_\}$, and which describe *simultaneous* grafting of unrooted trees. Such a command encodes the unrooted tree obtained by grafting to *all the entries* of the corolla $(-)$ the unrooted trees within the brackets, along their respective entries bound by μ. Therefore, the command $(-)\{\mu x.\_, \ldots, \mu y.\_\}$ is to the command $\langle \mu x.\_ \,|\, \mu y.\_ \rangle$ what the original notion of simultaneous operadic composition of [17] is to the notion of partial operadic composition of [16], in the framework of cyclic operads. The equations of the μ-syntax identify different constructions on unrooted trees that should be regarded as being the same, and the μ-syntax in whole is easilly mapped to the algebraic formalism of cyclic operads.

The advantage of the μ-syntax over the usual "mathematical" definitions of cyclic operads is tangible from two perspectives. On one hand, if one lays down the two usual ways of defining cyclic operads, the biased way (resulting in definitions via generators and relations [6, Theorem 2.2], [14, Definition 48]), and the unbiased way (leading to the definition via monads [6, Definition 2.1]), one would argue that these look quite formidable. This is due to the underlying intricate combinatorial structure of unrooted trees. The commands of the μ-syntax play the role of trees, but with the benefit of being rather simple in-line formulas. Accordingly, the equations of the μ-syntax make a crisp representation of the cumbersome laws defining the structure of cyclic operads. Summed up, the μ-syntax *makes the long story short(er).*

On the other hand, in the spirit of Leibniz's *characteristica universalis* and *calculus ratiocinator*, the usefulness of the μ-syntax arises when the question about the completeness, rigour and formalisability of mathematical proofs is asked. This especially concerns long and involved proofs, which are common in operad theory. Such a proof is, for example, the proof of the equivalence between the biased and unbiased definitions of cyclic operads, which is a well-known result (cf. [6, Theorem 2.2], [9, Section 5], [12, Section 4.2]). We shall make a syntactic reformulation of the monad of unrooted trees figuring in the unbiased definition, which, together with the μ-syntax, makes a syntactic framework well-suited for a complete step-by-step proof of the equivalence. The above requirements reflect through our syntactical proof, as follows. The *formalisability* property is met here by fixing a universal syntactic language in which the proof is presented. The internal structural patterns of this language are convenient for describing in a step-by-step fashion the transitions involved in this proof. In order to meet the *rigour* requirement, all the

involved structures are spelled out in detail. In particular, the correct treatment of the identities of the appropriate monad structure is given. Finally, as required by the *completeness* property, the proof that the laws satisfied by an algebra over the monad indeed come down to the axioms from the biased definition, is explicitly given.

Layout. In Section 1, we recall the biased entries-only definition [14, Definition 48] and the unbiased definition [6, Definition 2.1] of cyclic operads. For the latter definition, this involves a syntactic reformulation and a detailed description of the monad of unrooted trees. The section finishes with the statement of the theorem that expresses the equivalence between the two definitions. Section 2 will be devoted to the introduction and analysis of the μ-syntax. In Section 3, we employ the μ-syntax in crafting the proof of the equivalence from Section 1.

Notation and conventions. *About cyclic operads.* This paper is about non-skeletal set-based cyclic operads. Non-skeletality means that the entries of operations are labeled by arbitrary finite sets, rather than by natural numbers (as done in the skeletal approach). This is just a matter of convenience and a practice coming from computer science: we prefer the non-skeletal setting because we prefer formulas with "named" (rather than "numbered") variables, and we chose to work in **Set** (rather than in an arbitrary symmetric monoidal category) only to be able to (correctly) speak about operadic operations in terms of elements. We assume the existence of operadic units.

About finite sets and bijections. Conforming to the computer science practice, in this paper we assume that a sufficiently large universe of finite sets is fixed (denumerable is enough). Union will always be the *ordinary* union of *already* disjoint sets. For disjoint finite sets X and Y, $X \cup Y$ shall stand for the union of X and Y. For a bijection $\sigma : X' \to X$ and $Y \subseteq X$, we shall denote with $\sigma|^Y$ the corestriction of σ on $\sigma^{-1}(Y)$. For $y \notin X \cup X'$, we denote with σ_y the bijection $\sigma_y : X' \cup \{y\} \to X \cup \{y\}$, defined as σ on X', and such that $\sigma_y(y) = y$. If $\sigma(x') = x$, we denote with $\sigma^{y/x'}$ the bijection defined in the same way as σ, except that, instead of x', it contains y in its domain (the inverse image of x now being y). If $\tau : Y' \to Y$ is a bijection such that $X' \cap Y' = X \cap Y = \emptyset$, then $\sigma \cup \tau : X' \cup Y' \to X \cup Y$ denotes the bijection defined as σ on X' and as τ on Y'. Finally, if $\kappa : X\backslash\{x\} \cup \{x'\} \to X$ is the identity on $X\backslash\{x\}$ and $\kappa(x') = x$, we say that κ renames x to x' (notice the contravariant nature of this convention).

About type-theoretical notions. For a comprehensive account on the terminology of type theory and rewriting theory, whose basic notions we shall use in this paper, we refer to [19] and [1]. We list here the essentials.

We assume given an infinite set $\mathtt{V}$ of *variables*, or *names* (countable is enough). We say that a variable x is *fresh with respect to a set* X if $x \notin X$. The existence of $\mathtt{V}$ assures that for any finite set, there exists a variable which is fresh with respect to that set.

A *multi-sorted formal theory* is a formal theory for which variables, constant symbols and function symbols, as well as all the terms built from them, have a property called *sort* or *type*. Types serve to control the formation of terms and to classify them. A model of a multi-sorted formal theory, i.e. of a typed formal language, is a model in the usual sense, which additionally takes into account sorts of the symbols of the signature of the theory. In other words, the domain of such a model is a collection of sets $\{\mathcal{M}(s_i)\}_{i\in I}$, indexed by all sorts of the theory, and the interpretation function maps constant symbols of sort s_i to the set $\mathcal{M}(s_i)$, for all $i \in I$, and function symbols of sort $(s_1, \ldots, s_n; s)$ to functions of the form $\mathcal{M}(s_1) \times \cdots \times \mathcal{M}(s_n) \to \mathcal{M}(s)$.

An *abstract rewriting system* (a rewriting system for short) is a pair $(A, \to)$, where A is a

set and $\to$ is a binary relation on A. The name is supposed to indicate that an element (a, b) of $\to$ should be seen as a rewriting of a into b. We write $a \to b$ to denote that $(a, b) \in \to$. An element $a \in A$ is a *normal form* for $\to$ if there does not exist $a' \in A$, such that $a \to a'$. We say that a rewriting system $(A, \to)$ is *terminating* if there does not exist an infinite sequence $a_1 \to a_2 \to \cdots \to a_n \to \cdots$ of elements of A. We denote with $\xrightarrow{*}$ the reflexive and transitive closure of $\to$. A rewriting system $(A, \to)$ is *confluent* if, for any triple (a, a_1, a_2) of elements of A, such that $a \xrightarrow{*} a_1$ and $a \xrightarrow{*} a_2$, there exists $a' \in A$, such that $a_1 \xrightarrow{*} a'$ and $a_2 \xrightarrow{*} a'$. A rewriting system $(A, \to)$ is *locally confluent* if, for any triple (a, a_1, a_2) of elements of A, such that $a \to a_1$ and $a \to a_2$, there exists $a' \in A$, such that $a_1 \xrightarrow{*} a'$ and $a_2 \xrightarrow{*} a'$.

Fact 0.1. If $(A, \to)$ is terminating, then it is *normalising*, i.e. for any $a \in A$, there exists a normal form a', such that $a \xrightarrow{*} a'$.

Fact 0.2. If $(A, \to)$ is terminating and confluent, then for $a \in A$, there exists a *unique* normal form a', such that $a \xrightarrow{*} a'$.

Fact 0.3. If $(A, \to)$ is terminating, then it is confluent if and only if it is locally confluent.

In this paper, we shall examine certain *term rewriting systems*, i.e. abstract rewriting systems $(A, \to)$, for which the set A is the set of terms of some syntax, and the rewriting relation $\to$ is obtained by orienting some of the equations of the syntax.

1. Cyclic operads

Operads encode categories of algebras whose operations have multiple inputs and one output, such as associative algebras, commutative algebras, Lie algebras, etc. The interest in encoding more general algebraic structures was a part of the *renaissance of operads* in the early nineties of the last century, when various generalizations of operads came into existence. The formalism of cyclic operads was originally introduced by Getzler and Kapranov in [6]. The enrichment of the operad structure determined by the definition of a cyclic operad is provided by adding to the action of permuting the inputs of an operation an action of interchanging its output with one of the inputs. This feature essentially makes the distinction between the inputs and the output no longer visible, which is adequately captured by unrooted trees as pasting schemes for operations of a cyclic operad. In other words, cyclic operads can be seen as generalisations of operads for which an operation, instead of having inputs and an output, now has only "entries", and can be composed with another operation along any of them. As for the formal description of composition of such operations, the *unbiased* and *biased* frameworks provide two ways to complete the characterisation of a cyclic operad.

1.1 Biased definition of cyclic operads In the biased (entries-only) approach, the definition of a cyclic operad is biased towards "local" operadic compositions ${}_x\circ_y$, in the sense that these are the only explicitly defined concepts. The various ways to derive a global operadic composition are then equated by the appropriate axioms. We revisit below Markl's definition [14, Definition 48], for a particular case when the underlying functor is $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$, where **Bij** is the category of finite sets and bijections, and by adapting it further by also demanding operadic units. In the sequel, for $f \in \underline{\mathcal{C}}(X)$ and a bijection $\sigma : X' \to X$, we write f^σ instead of $\underline{\mathcal{C}}(\sigma)(f)$.

Definition 1.1. A *cyclic operad* is a functor $\mathcal{C} : \mathbf{Bij}^{op} \to \mathbf{Set}$, together with a distinguished element $id_{x,y} \in \mathcal{C}(\{x,y\})$ for each two-element set $\{x,y\}$, and a partial composition operation

$$ {}_x\circ_y : \mathcal{C}(X) \times \mathcal{C}(Y) \to \mathcal{C}(X\backslash\{x\} \cup Y\backslash\{y\}),$$

defined for arbitrary non-empty finite sets X and Y and elements $x \in X$ and $y \in Y$, such that $X\backslash\{x\} \cap Y\backslash\{y\} = \emptyset$. These data satisfy the axioms given below, wherein, for each of the axioms, we assume the set disjointness that ensures that all the partial compositions involved are well-defined.

Sequential associativity. For $f \in \mathcal{C}(X)$, $g \in \mathcal{C}(Y)$, $h \in \mathcal{C}(Z)$, $x \in X$, $y, u \in Y$ and $z \in Z$, the following equality holds:

(A1) $(f\ {}_x\circ_y\ g)\ {}_u\circ_z\ h = f\ {}_x\circ_y\ (g\ {}_u\circ_z\ h)$.

Commutativity. For $f \in \mathcal{C}(X)$, $g \in \mathcal{C}(Y)$, $x \in X$ and $y \in Y$, the following equality holds:

(CO) $f\ {}_x\circ_y\ g = g\ {}_y\circ_x\ f$.

Equivariance. For bijections $\sigma_1 : X' \to X$, $\sigma_2 : Y' \to Y$ and $\sigma = \sigma_1|^{X\backslash\{x\}} \cup \sigma_2|^{Y\backslash\{y\}}$, and $f \in \mathcal{C}(X)$ and $g \in \mathcal{C}(Y)$, the following equality holds:

(EQ) $f^{\sigma_1}\ {}_{\sigma_1^{-1}(x)}\circ_{\sigma_2^{-1}(y)}\ g^{\sigma_2} = (f\ {}_x\circ_y\ g)^{\sigma}$.

Right Unitality. For $f \in \mathcal{C}(X)$, $x \in X$ and a bijection σ that renames x to z, the following two equalities hold:

(U1) $f\ {}_x\circ_y\ id_{y,z} = f^{\sigma}$.

Moreover, the unit elements are preserved under the action of $\mathcal{C}(\sigma)$, i.e.

(U3) ${id_{x,y}}^{\sigma} = id_{u,v}$,

for any two two-element sets $\{x,y\}$ and $\{u,v\}$, and a bijection $\sigma : \{u,v\} \to \{x,y\}$.

For $f \in \mathcal{C}(X)$, the elements of the set X are called the *entries* of f. We say that a cyclic operad $\mathcal{C}$ is *constant-free* if $\mathcal{C}(\emptyset) = \mathcal{C}(\{x\}) = \emptyset$, for all singletons $\{x\}$. □

Note that we impose a slightly weaker condition on the sets X and Y and elements $x \in X$ and $y \in Y$ involved in partial composition than in [14, Definition 48]: instead of requiring X and Y to be disjoint, as Markl does, we allow the possibility that they intersect, provided that their intersection is a subset of $\{x,y\}$. This also means that we allow the possibility that $x = y$. Nevertheless, the characterizations of Definition 1.1 and [14, Definition 48], with units added, are equivalent. As for the units, here is a notational remark.

Notation 1.2. *It is understood that $id_{x,y} = id_{y,x}$. We reserve the notation $id_{\{x,y\}}$ for the identity bijection on the two-element set $\{x,y\}$.*

The lemma below gives basic properties of the partial composition operation.

Lemma 1.3. *The partial composition operation from Definition 1.1 satisfies the following laws.*

Parallel associativity. *For $f \in \mathcal{C}(X)$, $g \in \mathcal{C}(Y)$, $h \in \mathcal{C}(Z)$, $x, u \in X$, $y \in Y$ and $z \in Z$, the following equality holds:*

(A2) $(f\ {}_x\circ_y\ g)\ {}_u\circ_z\ h = (f\ {}_u\circ_z\ h)\ {}_x\circ_y\ g$.

Left unitality. *For $f \in \mathcal{C}(X)$, $x \in X$ and a bijection σ that renames x to z, the following equality holds:*

(U2) $id_{y,z}\ {}_y\circ_x\ f = f^{\sigma}$.

Proof. For (A2), combine (A1) and (CO). For (U2), combine (U1) and (CO). ■

Remark 1.4. *In the remainder of the article, merely for the sake of simplicity, we restrict ourselves to constant-free cyclic operads, to which we shall refer simply as cyclic operads.*

Definition 1.1 naturally incorporates the notion of *simultaneous composition*, as a sequence of partial compositions of the form as in the law (A2) from Lemma 1.3, that is, in which the entry involved in the next instance of a composition always comes from $f \in \mathcal{C}(X)$ and which, moreover, ends when *all* the entries of $f \in \mathcal{C}(X)$ are exhausted. In order to avoid writing explicitly such sequences, we introduce the following notation. For $f \in \mathcal{C}(X)$, let

$$\varphi : x \mapsto (Y_x, g_x, \underline{x})$$

be an assignment that associates to each $x \in X$ a finite set Y_x, an operation $g_x \in \mathcal{C}(Y_x)$ and an element $\underline{x} \in Y_x$, in such a way that

$$\bigcap_{x \in X} Y_x \backslash \{\underline{x}\} = \emptyset.$$

Let, moreover, $\sigma : X' \to X$ be an arbitrary bijection such that for all $x \in X$,

$$X' \backslash \{\sigma^{-1}(x)\} \cap Y_x \backslash \{\underline{x}\} = \emptyset.$$

Under these assumptions, the composite assignment

$$\varphi \circ \sigma : x' \mapsto (Y_{\sigma(x')}, g_{\sigma(x')}, \underline{\sigma(x')}),$$

defined for all $x' \in X'$, together with $f^\sigma \in \mathcal{C}(X')$, determines the composition

$$((f^\sigma \, {}_{x'}\circ_{\underline{\sigma(x')}} g_x) \, {}_{y'}\circ_{\underline{\sigma(y')}} g_y) \, {}_{z'}\circ_{\underline{\sigma(z')}} g_z \cdots,$$

consisting of a sequence of partial compositions indexed by the entries of f^σ. We will use the abbreviation $f^\sigma(\varphi \circ \sigma)$ to denote such a composition. Thanks to (A2), $f^\sigma(\varphi \circ \sigma)$ does not depend on the order in which the partial compositions were carried out. We finally set

$$f(\varphi) = f^\sigma(\varphi \circ \sigma), \tag{1.1}$$

and refer to $f(\varphi)$ as *the simultaneous composition determined by f and φ*. That $f(\varphi)$ does not depend on the choice of σ is a consequence of (EQ).

Notice that without the renaming role of σ, $f(\varphi)$ is not necessarily well-defined. For example, $f(\varphi) = (f \, {}_x\circ_{\underline{x}} g_x) \, {}_y\circ_{\underline{y}} g_y$, where $f \in \mathcal{C}(\{x, y\})$, $g_x \in \mathcal{C}(\{\overline{x}, y\})$ and $g_y \in \mathcal{C}(\{\overline{y}, v\})$, is not well-defined, although φ satisfies the required disjointness condition.

In relation to the above construction, the statements of the following lemma are easy consequences of the axioms from Definition 1.1.

Lemma 1.5. *The simultaneous composition $f(\varphi)$ has the following properties.*

a) Let $\psi : Z \to \bigcup_{x \in X}(Y_x \backslash \{\underline{x}\})$ be a bijection such that for all $x \in X$, $\underline{x} \notin \psi^{-1}(Y_x \backslash \{\underline{x}\})$. Denote with $\psi_{\underline{x}}$ the extension on Y_x of the bijection $\psi|^{Y_x \backslash \{\underline{x}\}}$, which is identity on $\underline{x}$, and let φ_ψ be defined as $\varphi_\psi : x \mapsto (g_x^{\psi_{\underline{x}}}, \underline{x})$, for all $x \in X$. Then $f(\varphi)^\psi = f(\varphi_\psi)$.

b) Let $\psi : y \mapsto (Z_y, h_y, \underline{y})$ be an assignment that associates to each $y \in \bigcup_{x \in X}(Y_x \backslash \{\underline{x}\})$ an operation $h_y \in \mathcal{C}(Z_y)$ and $\underline{y} \in Z_y$, in such a way that $f(\varphi)(\psi)$ is defined. If φ_ψ is the assigment defined as $\varphi_\psi : x \mapsto (g_x^{\psi_{\underline{x}}}, \underline{x})$, where $\psi_{\underline{x}}$ denotes the extension on Y_x of the assignment $\psi|_{Y_x \backslash \{\underline{x}\}}$, which is identity on $\underline{x}$, then $f(\varphi)(\psi) = f(\varphi_\psi)$.

The generators-and-relations nature of Definition 1.1 allows us to easily formalise cyclic operads as models of the multi-sorted equational theory which we now introduce.

The signature of this theory is determined by taking as sorts all finite sets, while, having denoted with s the sort of a constant symbol and with $(s_1, \ldots, s_n; s)$ the sort of an n-ary function symbol, as constant symbols we take the collection consisting of

$$id_{x,y} : \{x, y\}$$

and, as function symbols, we take the collection consisting of

$$\sigma : (Y; X) \text{ (of arity 1)} \quad \text{and} \quad {}_x\circ_y : (X, Y; X\backslash\{x\} \cup Y\backslash\{y\}) \text{ (of arity 2)},$$

where $x, y \in \mathtt{V}$ and σ ranges over all bijections of finite sets. Here, $\mathtt{V}$ is the infinite set of variables (i.e. names) whose existence we postulated in the Introduction.

Fixing a collection of *sorted variables*, or *parameters* P, and denoting with $P(X)$ the collection of parameters whose sort is X, the terms of the theory are built in the usual way:

$$\boxed{s, t ::= a \mid id_{x,y} \mid s\ {}_x\circ_y\ t \mid t^\sigma}$$

whereas the assignment of sorts to terms is done by the following rules:

$$\boxed{\frac{a \in P(X)}{a : X} \qquad \frac{}{id_{x,y} : \{x, y\}} \qquad \frac{s : X \quad t : Y}{s\ {}_x\circ_y\ t : X\backslash\{x\} \cup Y\backslash\{y\}} \qquad \frac{t : X \quad \sigma : (Y; X)}{t^\sigma : Y}}$$

where x and y are distinct variables in the second rule, while, in the third rule, $x \in X$, $y \in Y$ and $X\backslash\{x\} \cap Y\backslash\{y\} = \emptyset$. The equations of the theory are derived from the axioms of Definition 1.1, and there are two additional equations, namely

$$id^\sigma_{x,y} = id_{u,v} \qquad \text{and} \qquad (t^\sigma)^\tau = t^{\sigma\circ\tau}, \tag{1.2}$$

where, in the first equation, $\sigma : (\{u, v\}; \{x, y\})$.

Definition 1.6. A cyclic operad is a model of the equational theory from above. □

That this definition indeed describes the same structure as does Definition 1.1 is clear from the requirements that models of multi-sorted theories fulfill. The domain of such a model is a collection of sets $\mathcal{C}(X)$, arising by interpreting all sorts X, and the interpretation of the remaining of the signature in this universe exhibits the cyclic operad structure in the obvious way. Observe that the equations (1.2) ensure that the assignment $\mathcal{C} : \mathbf{Bij}^{op} \to \mathbf{Set}$, induced by the model, is functorial.

Let $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$ be a functor and let

$$P_{\underline{\mathcal{C}}} = \{a \in \underline{\mathcal{C}}(X) \mid X \text{ is a finite set}\} \tag{1.3}$$

be the collection of *parameters of* $\underline{\mathcal{C}}$. Observe that $P_{\underline{\mathcal{C}}}$ can be considered as a collection of sorted variables for the equational theory introduced above. In this regard, we call the syntax of terms built over $P_{\underline{\mathcal{C}}}$ the *combinator syntax generated by* $\underline{\mathcal{C}}$ and we refer to terms as *combinators*. We shall denote the set of all combinators induced by $\underline{\mathcal{C}}$ by $\mathtt{cTerm}_{\underline{\mathcal{C}}}$, and, for a finite set X, $\mathtt{cTerm}_{\underline{\mathcal{C}}}(X)$ will be used to denote the set of all combinators of type X.

In connection with Definition 1.6, if $\mathcal{C}$ is a cyclic operad (and, hence, a model of the equational theory from above), and writing $\underline{\mathcal{C}}$ for the underlying functor of $\mathcal{C}$, we shall denote with $[\_]_{\mathcal{C}} : \mathtt{cTerm}_{\underline{\mathcal{C}}} \to \mathcal{C}$ the induced interpretation of the combinator syntax.

1.2 Unbiased definition of cyclic operads Cyclic operads were originally introduced in unbiased manner in [6, Definition 2.1], as *algebras over a monad of unrooted trees*. In the operadic literature, incorporated in the structure of cyclic operads and similar definitions, one can find two formalisms of unrooted trees: in [6, Definition 2.1], the usual formalism of trees with "indivisible" edges is used, while in [5], [8], [9], trees with half-edges (or flags), due to [10], are used in the context of modular operads and Feynman categories. The operations decorating the nodes of an unrooted tree are "composed in one shot" through the structure morphism of the algebra. In this part, we syntactically reformulate [6, Definition 2.1]. The adaptations we make also include translating it to the non-skeletal setting, and reconstructing it within a formalism of unrooted trees that incorporates edges as pairs of half-edges, due to [10]. As it will be clear in Section 2, the formal language of unrooted trees that we present here is crafted in a way which reflects closely the formal language of the μ-syntax.

1.2.1 Graphs and unrooted trees Let $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$ be a functor such that $\underline{\mathcal{C}}(\emptyset) = \underline{\mathcal{C}}(\{x\}) = \emptyset$, for all singletons $\{x\}$, and let $P_{\underline{\mathcal{C}}}$ be as in (1.3). The syntax of unrooted trees generated by $P_{\underline{\mathcal{C}}}$ is obtained as follows. An *ordinary corolla* is a term

$$a(x, y, z, \dots),$$

where $a \in \underline{\mathcal{C}}(X)$ and $X = \{x, y, z, \dots\}$. We refer to a as the *head symbol* of $a(x_1, \dots, x_n)$. We call the elements of X the *free variables* of $a(x, y, z, \dots)$, and we write $FV(a) = X$ to denote this set. Whenever the set of free variables is irrelevant, we shall refer to an ordinary corolla only by its head symbol. In addition to ordinary corollas, we define *special corollas* to be terms of the shape

$$(x, y),$$

i.e. terms which do not have a parameter as a head symbol and which consist only of two distinct variables $x, y \in \mathtt{V}$. For a special corolla (x, y), we define $FV((x, y)) = \{x, y\}$.

Remark 1.7. *In both ordinary and special corollas, the order of appearance of free variables in the terms is irrelevant. In other words, we consider equal the terms, say, $a(x, y, z)$ and $a(z, x, y)$, as well as (x, y) and (y, x).*

A *graph* $\mathcal{V}$ is a non-empty, finite set of corollas with mutually disjoint free variables, together with an involution σ on the set

$$V(\mathcal{V}) = \bigcup_{i=1}^{k} FV(a_i) \cup \bigcup_{j=1}^{p} FV((u_j, v_j))$$

of all variables occuring in $\mathcal{V}$. We write

$$\mathcal{V} = \{a_1(x_1, \dots, x_n), \dots, a_k(y_1, \dots y_m), \dots, (u_1, v_1), \dots, (u_p, v_p); \sigma\}.$$

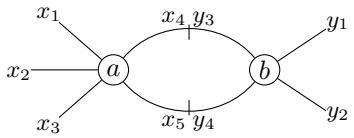
We denote with $Cor(\mathcal{V})$ the set of all corollas of $\mathcal{V}$, and we shall refer to an ordinary corolla by its parameter and denote special corollas with s_1, s_2, etc. The set of *edges* $Edge(\mathcal{V})$ of $\mathcal{V}$ consists of pairs (x, y) of variables such that $\sigma(x) = y$ (and, therefore, also $\sigma(y) = x$). Finally, we refer to the fixpoints of σ as the *free variables of* $\mathcal{V}$, the set of which we shall denote with $FV(\mathcal{V})$.

Remark 1.8. *The set of variables of a graph in our formalism corresponds to the set of* flags *(or half-edges) in the formalism of* [10]. *Such a formalism is inherent to operad theory. In graph theory in general, one does not usually encounter graphs with half-edges: graphs typically feature "indivisible" edges. The graphs considered here can be viewed as graphs with an* interface, *provided by the half-edges which are not paired to form edges (see [4]).*

Remark 1.9. *The condition* $\underline{\mathcal{C}}(\emptyset) = \underline{\mathcal{C}}(\{x\}) = \emptyset$*, imposed on* $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$*, corresponds to the convention to consider only constant-free cyclic operads, made in Remark 1.4. For the general case, the syntax of graphs is straightforwardly supplemented with appropriate corollas: ordinary corollas will additionally contain terms of the form* $a(x)$*, corresponding to elements* $a \in \mathcal{C}(\{x\})$*, and there will be another kind of corollas, corresponding to elements* $a \in \underline{\mathcal{C}}(\emptyset)$*, denoted simply with a (which, as "isolated corollas", can be repeated in a graph). If a graph contains only corollas of the latter kind, the involution of the graph is set to be the empty function.*

Here is an example.

EXAMPLE 1.10. The graph $\{a(x_1, x_2, x_3, x_4, x_5), b(y_1, y_2, y_3, y_4\}; \sigma\}$, where $\sigma = (x_4\ y_3)(x_5\ y_4)$, should be depicted as



This graph has two corollas, $a(x_1, x_2, x_3, x_4, x_5)$ and $b(y_1, y_2, y_3, y_4)$, two edges, (x_4, y_3) and (x_5, y_4), and five free variables, x_1, x_2, x_3, y_1, y_2. □

A non-empty graph is *connected* if for any two of its corollas there exists a path which connects them.

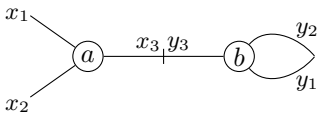
Remark 1.11. *Returning to Remark 1.9, if one allows corollas of the form* a*, where* $a \in \underline{\mathcal{C}}(\emptyset)$*, then a graph containing such a corolla is connected only if it does not contain any other corollas.*

A *subgraph of a graph* $\mathcal{V}$ *(with involution* σ*)* is any graph $\mathcal{V}'$, such that $Cor(\mathcal{V}') \subseteq Cor(\mathcal{V})$ and the involution τ of $\mathcal{V}'$ has the following property: if $\tau(x) \neq x$, then $\tau(x) = \sigma(x)$.

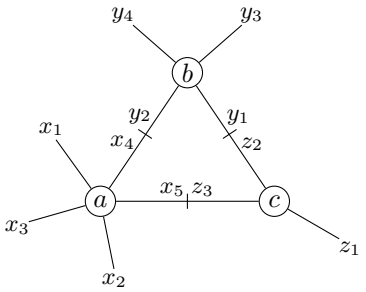
Starting from this notion of graph, an *extended unrooted tree* is defined as a *connected graph without loops, multiple edges and cycles.* As these requirements are standard in the terminology of graphs, we omit their formal definition and illustrate them with an example instead.

EXAMPLE 1.12. The graph from EXAMPLE 1.10 is not an extended unrooted tree, since it has two edges between corollas a and b.

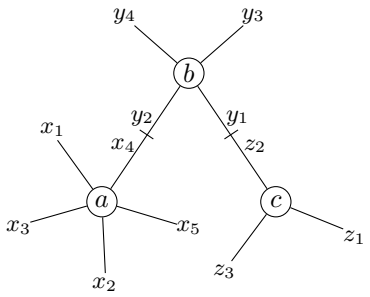
The graph $\{a(x_1, x_2, x_3), b(y_1, y_2, y_3); \sigma\}$, where $\sigma = (x_3\ y_3)(y_1\ y_2)$, is not an extended unrooted tree either, since the edge (y_1, y_2) connects the corolla b with itself, i.e. it is a loop:



The graph $\{a(x_1, x_2, x_3, x_4, x_5), b(y_1, y_2, y_3, y_4), c(z_1, z_2, z_3); \sigma\}$, with $\sigma = (x_4\ y_2)(y_1\ z_2)(z_3\ x_5)$, is another example of a graph which is not an extended unrooted tree, this time because of the presence of a cycle that connects its three corollas:



Finally, we get an example of a graph which is an extended unrooted tree by changing the involution σ of the previous graph to, say, $\sigma' = (x_4\ y_2)(y_1\ z_2)$, producing in this way the extended unrooted tree with graphical representation



□

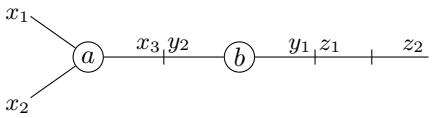
In moving from graphs to trees, we will additionally differentiate the classes of extended unrooted trees with respect to the shape of corollas they contain. Let $\mathcal{T}$ be a connected graph with no loops, multiple edges and cycles.

- If $Cor(\mathcal{T})$ consists only of ordinary corollas, then $\mathcal{T}$ is an *ordinary unrooted tree*.
- If $Cor(\mathcal{T})$ is a singleton with a special corolla, then $\mathcal{T}$ is an *exceptional unrooted tree*.
- An *unrooted tree* is either an ordinary unrooted tree or an exceptional unrooted tree.

Example 1.13. The last graph in Example 1.12 is an ordinary unrooted tree.

The graph $\{(x, y); id_{\{x,y\}}\}$ is an exceptional unrooted tree. We depict it as

The graph $\{a(x_1, x_2, x_3), b(y_1, y_2), (z_1, z_2); \sigma\}$, where $\sigma = (x_3\, y_2)(y_1\, z_1)$, depicted as



is an extended unrooted tree. It is neither ordinary, nor exceptional. □

A *subtree* of an (extended) unrooted tree $\mathcal{T}$ is a connected, non-empty subgraph of $\mathcal{T}$. We say that a subtree $\mathcal{S}$ of $\mathcal{T}$ is *proper* if $Cor(\mathcal{S}) \neq Cor(\mathcal{T})$.

We now define α-equivalence on extended unrooted trees. Suppose first that

$$\mathcal{T} = \{a(x_1, \ldots, x_n), \ldots; \sigma\}$$

is an ordinary unrooted tree, with $a \in \underline{\mathcal{C}}(X)$, $x_i \in FV(a) \backslash FV(\mathcal{T})$ and $\sigma(x_i) = y_j$. Let $\tau : X' \to X$ be a bijection that renames x_i to z, where z is fresh with respect to $V(\mathcal{T}) \backslash \{x_i\}$. The α-equivalence (for ordinary unrooted trees) is the smallest equivalence relation generated by equalities

$$(a(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n), \ldots; \sigma) =_\alpha (a^\tau(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n), \ldots; \sigma'),$$

where $\sigma' = \sigma$ on $V(\mathcal{T}) \backslash \{x_i, y_j\}$ and $\sigma'(z) = y_j$. This definition generalises in a natural way to extended unrooted trees: to the set of generators from above we add the clauses

$$\{(x, y), \ldots; \sigma\} =_\alpha \{(x, z), \ldots; \sigma'\},$$

where, for some variable x_i, $\sigma(y) = x_i$ (i.e. y is not a free variable of the tree on the left), z is fresh in the same sense as earlier, and σ' is the obvious modification of σ. In simple terms, we consider α-equivalent any two trees such that we can obtain one from another only by renaming variables which are not fixed points of the corresponding involutions.

We shall denote with $[\mathcal{T}]_\alpha$ the α-equivalence class determined by an (extended) unrooted tree $\mathcal{T}$. Finally, we shall denote with $\mathtt{T}_{\underline{\mathcal{C}}}(X)$ (resp. $\mathtt{eT}_{\underline{\mathcal{C}}}(X)$) the set of all α-equivalence classes of unrooted trees (resp. extended unrooted trees) whose parameters belong to $P_{\underline{\mathcal{C}}}$ and whose free variables are given by the set X. If X is a two-element set, this definition includes the possibility that an unrooted tree has 0 parameters, in which case the corresponding equivalence class is determined by the appropriate exceptional unrooted tree. We shall write $\mathtt{T}_{\underline{\mathcal{C}}}$ (resp. $\mathtt{eT}_{\underline{\mathcal{C}}}$) for the collection of all unrooted trees (resp. extended unrooted trees) generated by $P_{\underline{\mathcal{C}}}$.

1.2.2 The monad of unrooted trees The monad of unrooted trees is the monad $(\mathcal{M}, \mu, \eta)$ on the functor category $\mathbf{Set}^{\mathbf{Bij}^{op}}$, defined as follows. The endofunctor $\mathcal{M}$ is defined by

$$\mathcal{M}(\underline{\mathcal{C}})(X) = \mathtt{T}_{\underline{\mathcal{C}}}(X).$$

The component $\eta_{\underline{\mathcal{C}}X} : \underline{\mathcal{C}}(X) \to \mathcal{M}(\underline{\mathcal{C}})(X)$ of the monad unit associates to $a \in \underline{\mathcal{C}}(X)$ the isomorphism class of the unrooted tree $\{a(x_1, \ldots, x_n), id_X\}$, where $X = \{x_1, \ldots, x_n\}$.

The action of the monad multiplication, typically (and incompletely) described as "flattening" in the literature (which is acceptable only if one forgets about units), deserves more attention.

In order to obtain its complete description, we first build a rewriting system on $\mathtt{eT}_{\underline{\mathcal{C}}}$. The rewriting relation $\to$ on classes of $\mathtt{eT}_{\underline{\mathcal{C}}}$ is canonically induced by the reflexive and transitive closure of the union of the following reductions, defined on their representatives:

$$(a(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n), (y, z), \ldots; \sigma) \to (a^\tau(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n), \ldots; \sigma'),$$

where $\sigma(x_i) = y$, τ renames x_i to z, and σ' is the obvious restriction of σ, and

$$((x, y), (u, v), \ldots; \sigma) \to ((x, v), \ldots; \sigma'),$$

where $\sigma(y) = u$, and σ' is again the obvious restriction of σ.

Lemma 1.14. *The rewriting system* $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$ *is confluent and terminating.*

Proof. The termination of the system is obvious: in an arbitrary reduction sequence, each subsequent tree has one special corolla less, and the sequence finishes either when all of them are exhausted (in the case when the initial tree has at least one ordinary corolla), or when there is only one special corolla left (in the case when the initial tree consists only of special corollas). Due to the connectedness of unrooted trees, all special corollas (except one in the latter case) will indeed be exhausted. Clearly, the normal forms are precisely the unrooted trees of $\mathtt{T}_{\underline{\mathcal{C}}}$.

If $\mathcal{T}_1$ and $\mathcal{T}_2$ are reduced from $\mathcal{T} \in \mathtt{eT}_{\underline{\mathcal{C}}}$ in one step, and if s_1 and s_2 are the special corollas involved in the respective reductions, the local confluence is proved by case analysis, with respect to whether s_1 and s_2 are equal or not. By Fact 0.3, this establishes confluence. ■

By Lemma 1.14, an arbitrary normal form $nf(\mathcal{T})$ of an extended unrooted tree $\mathcal{T}$, with respect to $\to$, determines a unique α-equivalence class $[nf(\mathcal{T})]_\alpha$ in $\mathtt{T}_{\underline{\mathcal{C}}}$. It is easily seen that, for every finite set X, this assignment gives rise to the function $nf_X : \mathtt{eT}_{\underline{\mathcal{C}}}(X) \to \mathtt{T}_{\underline{\mathcal{C}}}(X)$, determined by

$$nf_X : [\mathcal{T}]_\alpha \mapsto [nf(\mathcal{T})]_\alpha. \tag{1.4}$$

We now formally define the flattening (which is still not the monad multiplication) on $\mathcal{MM}(\underline{\mathcal{C}})(X)$. Observe that the isomorphism classes of

$$\mathcal{MM}(\underline{\mathcal{C}})(X) = \mathcal{M}(\mathtt{T}_{\underline{\mathcal{C}}})(X) = \mathtt{T}_{\mathtt{T}_{\underline{\mathcal{C}}}}(X)$$

are determined by unrooted trees whose parameters are α-equivalence classes of unrooted trees themselves (with parameters from $P_{\underline{\mathcal{C}}}$), and whose set of free variables is X. We call these trees *two-level trees*. Syntactically, a two-level tree of $\mathtt{T}_{\mathtt{T}_{\underline{\mathcal{C}}}}$ can be either

- an exceptional unrooted tree $\{(x,y); id_{\{x,y\}}\}$, in which case we trivially have 0 parameters coming from $P_{\underline{\mathcal{C}}}$, or
- an ordinary unrooted tree

  $$\{[\{a(x_1,x_2,\dots), b(y_1,\dots),\dots;\sigma_1\}]_\alpha(x_1,x_2,y_1,\dots),\dots,[\{(z_1,z_2); id_{\{z_1,z_2\}}\}]_\alpha(z_1,z_2),\dots;\sigma\},$$

  whose parameters can be α-equivalence classes of both ordinary and exceptional unrooted trees of $\mathtt{T}_{\underline{\mathcal{C}}}$.

Notation 1.15. *Let $\mathcal{T}$ be a two-level unrooted tree. Suppose that, for $1 \leq i \leq n$, $[\mathcal{T}_i]_\alpha \in \mathtt{T}_{\underline{\mathcal{C}}}(Y_i)$ are the parameters of $\mathcal{T}$ and let C_i be their corresponding corollas. We then have $FV(C_i) = FV(\mathcal{T}_i) = Y_i$. The fact that the set of free variables of each corolla is recorded by the data of the corresponding parameter allows us to shorten the notation by writing $\mathcal{T}_i$ without listing explicitly the elements of $FV(\mathcal{T}_i)$. For example, for the tree from the latter case above, we shall write*

$$\{[\{a(x_1,x_2,\dots), b(y_1,\dots),\dots;\sigma_1\}]_\alpha,\dots,[\{(z_1,z_2); id_{\{z_1,z_2\}}\}]_\alpha,\dots;\sigma\}.$$

We extend this abbreviation to trees of $\mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}$, and when the form of the parameters of a two-level tree is irrelevant, we shall write $\{[\mathcal{T}_1]_\alpha,\dots,[\mathcal{T}_n]_\alpha, s_1,\dots,s_m;\sigma\}$, where s_i are special corollas.

The flattening of two-level unrooted trees is a familly of functions

$$\mathit{flat}_X : \mathtt{T}_{\mathtt{T}_{\underline{\mathcal{C}}}}(X) \to \mathtt{eT}_{\underline{\mathcal{C}}}(X),$$

indexed by finite sets, defined by the following two clauses:

- $\mathit{flat}_{\{x,y\}}([\{(x,y); id_{\{x,y\}}\}]_\alpha) = [\{(x,y); id_{\{x,y\}}\}]_\alpha$, and
- if $\mathcal{T} = \{[\{a(x_1,x_2,\dots), b(y_1,\dots),\dots;\sigma_1\}]_\alpha,\dots,[\{(z_1,z_2); id_{\{z_1,z_2\}}\}]_\alpha,\dots;\sigma\}$, then

  $$\mathit{flat}_X([\mathcal{T}]_\alpha) = [\{a(x_1,x_2,\dots), b(y_1,\dots),\dots,(z_1,z_2),\dots;\underline{\sigma}\}]_\alpha,$$

  where, having denoted with $\mathcal{T}_i$, $1 \leq i \leq n$, the corollas of $\mathcal{T}$, and with σ_i the corresponding involutions,

  $$\underline{\sigma}(x) = \begin{cases} \sigma(x) & \text{if } x \in \bigcup_{i=1}^n FV(\mathcal{T}_i) \\ \sigma_i(x) & \text{if } x \in V(\mathcal{T}_i)\backslash FV(\mathcal{T}_i)\,. \end{cases}$$

Observe that $\mathit{flat}_X([\mathcal{T}]_\alpha)$ is an α-equivalence class of an extended unrooted tree whenever $\mathcal{T}$ contains at least two corollas, one of which is special. These are the cases that make a gap between the flattening function and the action of the monad multiplication (which always results in an ordinary unrooted tree). In the same style as we presented the functions nf_X by (1.4), in what follows, we shall often denote the class $\mathit{flat}_X([\mathcal{T}]_\alpha)$ simply by $[\mathit{flat}(\mathcal{T})]_\alpha$.

The complete characterisation of the monad multiplication $\mu_{\underline{\mathcal{C}}} : \mathtt{T}_{\mathtt{T}_{\underline{\mathcal{C}}}} \to \mathtt{T}_{\underline{\mathcal{C}}}$ is given by

$$\mu_{\underline{\mathcal{C}}X} = nf_X \circ \mathit{flat}_X.$$

Therefore, for $[\mathcal{T}]_\alpha \in \mathtt{T}_{\mathtt{T}_{\underline{\mathcal{C}}}}(X)$, we have

$$\mu_{\underline{\mathcal{C}}X} : [\mathcal{T}]_\alpha \mapsto [nf(flat(\mathcal{T}))]_\alpha.$$

Hence, in the presence of units, this action is indeed more than just "flattening".

We now prepare the grounds for the proof that $(\mathcal{M}, \mu, \eta)$ is indeed a monad.

The domain of flattening is extended in a natural way to $\mathcal{M}'\mathcal{M}'(\underline{\mathcal{C}})$, where $\mathcal{M}'(\underline{\mathcal{C}})(X) = \mathtt{eT}_{\underline{\mathcal{C}}}(X)$. The clause that needs to be added to encompass $\mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}(X)$ concerns two-level trees of the form

$$\{[\{a(x_1, x_2, \dots), b(y_1, \dots), (z_1, z_2) \dots; \sigma_1\}]_\alpha, \dots, [\{(u_1, u_2); id_{\{u_1,u_2\}}\}]_\alpha, \dots, (v_1, v_2), \dots; \sigma\},$$

i.e. extended unrooted trees whose set of corollas allows special corollas and classes of extended unrooted trees. Let us denote with $\mathcal{T}$ the above tree, and let $Cor_s(\mathcal{T})$ be the set of its special corollas. The flattening of $[\mathcal{T}]_\alpha$ is defined simply as

$$flat_X([\mathcal{T}]_\alpha) = [\{a(x_1, x_2, \dots), b(y_1, \dots), (z_1, z_2), \dots, (u_1, u_2), \dots, (v_1, v_2), \dots; \underline{\sigma}\}]_\alpha,$$

with $\underline{\sigma}$ being defined exactly like before for the variables coming from $Cor(\mathcal{T}) \setminus Cor_s(\mathcal{T})$, while we set $\underline{\sigma}(x) = \sigma(x)$ for all variables $x \in \bigcup_{s \in Cor_s(\mathcal{T})} FV(s)$.

For $[\mathcal{T}]_\alpha$ and $[\mathcal{T}']_\alpha$ from $\mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}$, the following two lemmas give conditions that ensure that $flat([\mathcal{T}]_\alpha) \to flat([\mathcal{T}']_\alpha)$, in the instance $(\mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}, \to)$ of the rewriting system defined earlier.

Lemma 1.16. *For $[\mathcal{T}_1]_\alpha, [\mathcal{T}_2]_\alpha \in \mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}(X)$, if $[\mathcal{T}_1]_\alpha \to [\mathcal{T}_2]_\alpha$ in $(\mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}, \to)$, then $flat_X([\mathcal{T}_1]_\alpha) \to flat_X([\mathcal{T}_2]_\alpha)$ in $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$.*

Lemma 1.17. *For $[\{[\mathcal{T}_1]_\alpha, \dots, [\mathcal{T}_n]_\alpha, s_1, \dots, s_m; \sigma\}]_\alpha \in \mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}(X)$ and $1 \leq j \leq n$, if $[\mathcal{T}_j]_\alpha \to [\mathcal{T}'_j]_\alpha$ in $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$, then*

$$flat_X([\{[\mathcal{T}_1]_\alpha, \dots, [\mathcal{T}_j]_\alpha, \dots, [\mathcal{T}_n]_\alpha, s_1, \dots, s_m; \sigma\}]_\alpha) \to flat_X([\{[\mathcal{T}_1]_\alpha, \dots, [\mathcal{T}'_j]_\alpha, \dots, [\mathcal{T}_n]_\alpha, s_1, \dots, s_m; \sigma\}]_\alpha)$$
in $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$

Relying on Lemma 1.16 and Lemma 1.17, we obtain the following two equivalent characterisations of the monad multiplication.

Lemma 1.18. *For $[\mathcal{T}]_\alpha = [\{[\mathcal{T}_1]_\alpha, \dots, [\mathcal{T}_n]_\alpha, s_1, \dots, s_m; \sigma\}]_\alpha \in \mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}(X)$ the following claims hold:*

1. $nf(flat(\mathcal{T})) =_\alpha nf(flat(nf(\mathcal{T})))$,
2. $nf(flat(\mathcal{T})) =_\alpha nf(flat(\{[nf(\mathcal{T}_1)]_\alpha, \dots, [nf(\mathcal{T}_n)]_\alpha, s_1, \dots, s_m; \sigma\}))$.

Proof. By the termination of $(\mathtt{eT}_{\mathtt{eT}_{\underline{\mathcal{C}}}}, \to)$, we have $\mathcal{T} \to nf(\mathcal{T})$, and then, by Lemma 1.16 and the termination of $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$, we know that, in $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$,

$$flat(\mathcal{T}) \to flat(nf(\mathcal{T})) \to nf(flat(nf(\mathcal{T}))).$$

On the other hand, by the termination of $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$, we also have that $flat(\mathcal{T}) \to nf(flat(\mathcal{T}))$. Therefore, the first claim follows by the confluence of $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$.

As for the second claim, by the termination of $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$, we have $\mathcal{T}_i \to nf(\mathcal{T}_i)$, for all $i \in I$. Hence, by Lemma 1.17, and then again by the termination of $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$, we get that

$$\begin{array}{rcl} flat(\mathcal{T}) & \to & flat(\{[nf(\mathcal{T}_1)]_\alpha, \dots, [nf(\mathcal{T}_n)]_\alpha, s_1, \dots, s_m; \sigma\}) \\ & \to & nf(flat(\{nf(\mathcal{T}_1), \dots, nf(\mathcal{T}_n), s_1, \dots, s_m; \sigma\})) \end{array}$$

is a reduction sequence of $(\mathtt{eT}_{\underline{\mathcal{C}}}, \to)$. The conclusion follows as in the previous claim. ∎

On the other hand, by the very definition of flattening on extended unrooted trees, we have the following property.

Lemma 1.19. *For* $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_n; \sigma\} \in \mathtt{T}_{\mathtt{eT}_{\underline{\mathcal{C}}}}$ *the following equality holds:*

$$\mathit{flat}(\mathit{flat}(\mathcal{T})) = \mathit{flat}(\{[\mathit{flat}(\mathcal{T}_1)]_\alpha, \ldots, [\mathit{flat}(\mathcal{T}_n)]_\alpha; \sigma\}).$$

We now finally verify the laws of the monad $(\mathcal{M}, \mu, \eta)$.

Lemma 1.20. *For natural transformations* $\mu : \mathcal{M}\mathcal{M} \to \mathcal{M}$ *and* $\eta : 1 \to \mathcal{M}$*, the following diagrams commute for every functor* $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$ *and a finite set* X*:*

$$\begin{array}{ccc} \mathcal{M}\mathcal{M}\mathcal{M}(\underline{\mathcal{C}})(X) & \xrightarrow{\mathcal{M}\mu_{\underline{\mathcal{C}}X}} & \mathcal{M}\mathcal{M}(\underline{\mathcal{C}})(X) \\ \downarrow{\scriptstyle \mu\mathcal{M}_{\underline{\mathcal{C}}X}} & & \downarrow{\scriptstyle \mu_{\underline{\mathcal{C}}X}} \\ \mathcal{M}\mathcal{M}(\underline{\mathcal{C}})(X) & \xrightarrow{\mu_{\underline{\mathcal{C}}X}} & \mathcal{M}(\underline{\mathcal{C}})(X) \end{array} \qquad \begin{array}{ccc} \mathcal{M}(\underline{\mathcal{C}})(X) & \xrightarrow{\mathcal{M}\eta_{\underline{\mathcal{C}}X}} & \mathcal{M}\mathcal{M}(\underline{\mathcal{C}})(X) \\ & {\scriptstyle id_{\underline{\mathcal{C}}X}} \searrow \quad \swarrow {\scriptstyle \mu_{\underline{\mathcal{C}}X}} & \\ & \mathcal{M}(\underline{\mathcal{C}})(X) & \end{array} \qquad \begin{array}{ccc} \mathcal{M}(\underline{\mathcal{C}})(X) & \xrightarrow{\eta\mathcal{M}_{\underline{\mathcal{C}}X}} & \mathcal{M}\mathcal{M}(\underline{\mathcal{C}})(X) \\ & {\scriptstyle id_{\underline{\mathcal{C}}X}} \searrow \quad \swarrow {\scriptstyle \mu_{\underline{\mathcal{C}}X}} & \\ & \mathcal{M}(\underline{\mathcal{C}})(X) & \end{array}$$

Proof. We begin with the left diagram. Chasing the associativity of multiplication includes treating several cases, according to the shape of the unrooted tree of

$$\mathcal{M}\mathcal{M}\mathcal{M}(\underline{\mathcal{C}})(X) = \mathtt{T}_{\mathtt{T}_{\underline{\mathcal{C}}}}(X)$$

that we start from. The most interesting is the one starting from (a class determined by) an ordinary unrooted tree with corollas given by ordinary unrooted trees built upon $\mathtt{T}_{\underline{\mathcal{C}}}$ and we prove the associativity only for this case. Let, therefore, $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_n; \sigma\}$.

By chasing the diagram to the right-down, the action of $\mathcal{M}\mu_{\underline{\mathcal{C}}_X}$ corresponds to corolla-per-corolla flattening of $\mathcal{T}$, followed by taking the respective normal forms. Then μ flattens additionally the resulting tree and reduces it to a normal form. These actions make the following sequence of steps:

$$\begin{array}{rcl} \mathcal{T} & \mapsto & \{[\mathit{flat}(\mathcal{T}_1)]_\alpha, \ldots, [\mathit{flat}(\mathcal{T}_n)]_\alpha; \sigma\} \\ & \mapsto & \{[nf(\mathit{flat}(\mathcal{T}_1))]_\alpha, \ldots, [nf(\mathit{flat}(\mathcal{T}_n))]_\alpha; \sigma\} \\ & \mapsto & \mathit{flat}(\{[nf(\mathit{flat}(\mathcal{T}_1))]_\alpha, \ldots, [nf(\mathit{flat}(\mathcal{T}_n))]_\alpha; \sigma\}) \\ & \mapsto & nf(\mathit{flat}(\{[nf(\mathit{flat}(\mathcal{T}_1))]_\alpha, \ldots, [nf(\mathit{flat}(\mathcal{T}_n))]_\alpha; \sigma\})) = R. \end{array}$$

The action $\mu\mathcal{M}_{\underline{\mathcal{C}}_X}$ on the left-down side of the diagram corresponds to the action of μ on the tree $\mathcal{T}$ itself, which flattens it and reduces it to a normal form. Followed by μ again, this gives us the following sequence:

$$\begin{array}{rcl} \mathcal{T} & \mapsto & \mathit{flat}(\mathcal{T}) \\ & \mapsto & nf(\mathit{flat}(\mathcal{T})) \\ & \mapsto & \mathit{flat}(nf(\mathit{flat}(\mathcal{T}))) \\ & \mapsto & nf(\mathit{flat}(nf(\mathit{flat}(\mathcal{T})))) = L. \end{array}$$

Let $R' = nf(\mathit{flat}(\{[\mathit{flat}(\mathcal{T}_1)]_\alpha, \ldots, [\mathit{flat}(\mathcal{T}_n)]_\alpha; \sigma\}))$ and $L' = nf(\mathit{flat}(\mathit{flat}(\mathcal{T})))$. By Lemma 1.18, we have that $R = R'$ and $L = L'$, and, by Lemma 1.19, we have $R' = L'$.

We now verify the unit laws for the case when $[\mathcal{T}]_\alpha \in \mathcal{M}(\underline{\mathcal{C}})(X)$ is determined by an ordinary unrooted tree. Let, therefore, $\mathcal{T} = \{a_1(x_1, \ldots, x_k), \ldots, a_n(y_1, \ldots, y_r); \sigma\}$.

By going to the right-down in the first unit diagram (i.e. the diagram in the middle), the

action of $\mathcal{M}_{\eta \underline{\mathcal{C}}_X}$ turns each corolla a_i into a single-corolla unrooted tree $\mathcal{T}_i$, leading to a two-level unrooted tree, which is then flattened and reduced to a normal form by μ. Therefore, the right-down side sequence is as follows:

$$\begin{aligned}\mathcal{T} &\mapsto \{[\{a_1(x_1,\dots,x_k),id\}]_\alpha,\dots,[\{a_n(y_1,\dots,y_r);id\}]_\alpha;\sigma\}\\ &\mapsto \{a_1(x_1,\dots,x_k),\dots,a_n(y_1,\dots,y_r);\underline{\sigma}\}\\ &\mapsto \{a_1(x_1,\dots,x_k),\dots,a_n(y_1,\dots,y_r);\underline{\sigma}'\}\end{aligned}$$

the resulting tree being exactly $\mathcal{T}$, since

$$\underline{\sigma}'(x)=\underline{\sigma}(x)=\begin{cases}\sigma(x) & \text{if } x\in\bigcup_{i=1}^n FV(\mathcal{T}_i)\\ x & \text{if } x\in V(\mathcal{T}_i)\backslash FV(\mathcal{T}_i)\end{cases}=\begin{cases}\sigma(x) & \text{if } x\in V(\mathcal{T})\\ x & \text{if } x\in V(\mathcal{T}_i)\backslash FV(\mathcal{T}_i)\end{cases}=\sigma(x),$$

wherein the last equality holds since $V(\mathcal{T}_i)\backslash FV(\mathcal{T}_i)=\emptyset$, for all $1\leq i\leq n$.

By chasing the second unit diagram to the right, $\mathcal{T}$ will first be turned, by the action of $\eta\mathcal{M}_{\underline{\mathcal{C}}_X}$, into a single-corolla two-level tree, which will then be flattened and reduced to a normal form by the action of μ. Therefore, we have the sequence

$$\begin{aligned}\mathcal{T} &\mapsto \{[\{a_1(x_1,\dots,x_k),\dots,a_n(y_1,\dots,y_r);\sigma\}]_\alpha,id_X\}\\ &\mapsto \{a_1(x_1,\dots,x_k),\dots,a_n(y_1,\dots,y_r);\underline{id_X}\}\\ &\mapsto \{a_1(x_1,\dots,x_k),\dots,a_n(y_1,\dots,y_r);\underline{id_X}'\}\end{aligned}$$

For the resulting involution $\underline{id_X}'$ we have

$$\underline{id_X}'(x)=\underline{id_X}(x)=\begin{cases}x & \text{if } x\in FV(\mathcal{T})\\ \sigma(x) & \text{if } x\in V(\mathcal{T})\backslash FV(\mathcal{T})\end{cases}=\sigma(x).$$

Therefore, the resulting tree is exactly $\mathcal{T}$. ■

Finally, here is the original definition [6, Definition 2.1] of a cyclic operad, recasted in the new syntactic framework.

Definition 1.21. A cyclic operad is an algebra over the monad $(\mathcal{M},\mu,\eta)$.

And, under these syntactic glasses, here is the well-known result about the equivalence of the biased and unbiased definitions.

Theorem 1.22. *A functor $\underline{\mathcal{C}}:\mathbf{Bij}^{op}\to\mathbf{Set}$ is endowed with a cyclic operad structure (as described by Definition 1.6) if and only if it is endowed with a structure morphism of an $\mathcal{M}$-algebra.*

Before we introduce the μ-syntax in the following section, and ultimatelly prove Theorem 1.22, we indicate the biased structure "hiding" in the monad approach we just made. As we shall see, the exceptional unrooted trees will be used as pasting schemes of identities of cyclic operads.

1.2.3 The free cyclic operad structure implicit in $(\mathcal{M},\mu,\eta)$ A way to specify the free cyclic operad over a functor $\underline{\mathcal{C}}$ is given implicitly in §1.2.2. The functor $F(\underline{\mathcal{C}}):\mathbf{Bij}^{op}\to\mathbf{Set}$, underlying the free cyclic operad structure, is defined by $F(\underline{\mathcal{C}})(X)=\mathtt{T}_{\underline{\mathcal{C}}}(X)$. In the unbiased approach, the monad from §1.2.2 indeed arises from the adjunction $F\vdash U$, where U is the obvious forgetful functor. Before we spell out the biased free cyclic operad structure, we fix some notation.

Notation 1.23. *For an unrooted tree $\mathcal{T}$, a finite set V and a bijection $\vartheta : V \to V(\mathcal{T})$, we shall denote with $\mathcal{T}^{\vartheta}$ the unrooted tree obtained from $\mathcal{T}$ by renaming its variables in a way dictated by ϑ and adapting its corollas accordingly. More precisely, if $a \in Cor(\mathcal{T})$ is an ordiary corolla, $\mathcal{T}^{\vartheta}$ will, instead of a, contain the corolla $a^{\vartheta|^{FV(a)}}$, and, if $(x, y) \in Cor(\mathcal{T})$ is a special corolla, $\mathcal{T}^{\vartheta}$ will, instead of (x, y), contain the corolla $(\vartheta^{-1}(x), \vartheta^{-1}(y))$. The involution σ^{ϑ} of $\mathcal{T}^{\vartheta}$ is defined as $\sigma^{\vartheta}(v) = \vartheta^{-1}(\sigma(\vartheta(v)))$, for $v \in V$.*

For a bijection $\kappa : X' \to X$, the image $[\mathcal{T}]^{\kappa}_{\alpha}$ of $[\mathcal{T}]_{\alpha} \in \mathtt{T}_{\underline{\mathcal{C}}}(X)$ under $\mathtt{T}_{\underline{\mathcal{C}}}(\kappa) : \mathtt{T}_{\underline{\mathcal{C}}}(X) \to \mathtt{T}_{\underline{\mathcal{C}}}(X')$ is the equivalence class $[\mathcal{T}^{\kappa \cup \varepsilon}]_{\alpha}$, where $\varepsilon : V \to V(\mathcal{T}) \backslash X$ is an arbitrary bijection, such that $X' \cap V = \emptyset$.

Let X and Y be non-empty finite sets such that for some $x \in X$ and $y \in Y$ we have $X \backslash \{x\} \cap Y \backslash \{y\} = \emptyset$, and let $[\mathcal{T}_1]_{\alpha} \in \mathtt{T}_{\underline{\mathcal{C}}}(X)$, $[\mathcal{T}_2]_{\alpha} \in \mathtt{T}_{\underline{\mathcal{C}}}(Y)$. The partial composition operation

$$ {}_{x}\bullet_{y} : \mathtt{T}_{\underline{\mathcal{C}}}(X) \times \mathtt{T}_{\underline{\mathcal{C}}}(Y) \to \mathtt{T}_{\underline{\mathcal{C}}}(X \backslash \{x\} \cup Y \backslash \{y\}) $$

is given as

$$ [\mathcal{T}_1]_{\alpha} \, {}_{x}\bullet_{y} \, [\mathcal{T}_2]_{\alpha} = [nf(\mathcal{T})]_{\alpha}, $$

where $Cor(\mathcal{T})$ is obtained by taking the union of the sets of corollas of $\mathcal{T}_1$ and $\mathcal{T}_2$, after having previously adapted them in a way that makes this union disjoint with respect to the variables occuring in it. More precisely, if $\vartheta_1 : V_1 \to (V(\mathcal{T}_1) \backslash X) \cup \{x\}$ and $\vartheta_2 : V_2 \to (V(\mathcal{T}_2) \backslash Y) \cup \{y\}$ are bijections such that $V_1 \cap V_2 = \emptyset$, then

$$ Cor(\mathcal{T}) = \{C^{(\vartheta_1 \cup id_{X \backslash \{x\}})|^{FV(C)}} \mid C \in Cor(\mathcal{T}_1)\} \cup \{D^{(\vartheta_2 \cup id_{Y \backslash \{y\}})|^{FV(D)}} \mid D \in Cor(\mathcal{T}_2)\}. $$

If σ_i is the involution of $\mathcal{T}_i$, $i = 1, 2$, the involution σ of $\mathcal{T}$ is defined as follows:

$$ \sigma(v) = \begin{cases} \vartheta_1^{-1}(\sigma_1(\vartheta_1(v))) & \text{if } v \in V_1 \backslash \vartheta_1^{-1}(x) \\ \vartheta_2^{-1}(y) & \text{if } v = \vartheta_1^{-1}(x) \\ \vartheta_2^{-1}(\sigma_2(\vartheta_2(v))) & \text{if } v \in \vartheta_2^{-1}(y) \\ \vartheta_1^{-1}(x) & \text{if } v = \vartheta_2^{-1}(y) \\ v & \text{if } v \in X \backslash \{x\} \cup Y \backslash \{y\}\,. \end{cases} $$

For an arbitrary two-element set $\{y, z\}$, we set $id_{y,z} = [\{(y, z); id_{\{y,z\}}\}]_{\alpha}$.

Remark 1.24. *Observe that the partial composition structure on classes of unrooted trees does not violate the constant-freeness requirement. The condition $\underline{\mathcal{C}}(\{x\}) = \emptyset$ plays an indispensable role here: requiring that $\underline{\mathcal{C}}(\emptyset) = \emptyset$, but allowing the possibility that $\underline{\mathcal{C}}(\{x\}) \neq \emptyset$, is inconsistent. Consider, for example, $a \in \underline{\mathcal{C}}(\{x\})$, $b \in \underline{\mathcal{C}}(\{y\})$ and their composition $[a(x)]_{\alpha} \, {}_{x}\bullet_{y} \, [b(y)]_{\alpha}$.*

2. μ-syntax

Backed up with the graphical intuition of the biased cyclic operad structure on classes of unrooted trees described in §1.2.3, in this section we introduce the μ-syntax.

2.1 The language and the equations For a functor $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$, such that $\underline{\mathcal{C}}(\emptyset) = \underline{\mathcal{C}}(\{x\}) = \emptyset$, the language of the μ-syntax is built over the collection of parameters $P_{\underline{\mathcal{C}}}$ (see (1.3)) and the set of variables $\mathtt{V}$. Unlike the combinator syntax $\mathtt{cTerm}_{\underline{\mathcal{C}}}$ from §1.1, which has only one kind of expressions, the μ-syntax features two different kinds of typed expressions

COMMANDS	TERMS
$c ::= \langle s \,|\, t \rangle \;\mid\; \underline{a}\{t_1, \ldots, t_n\}$	$s, t ::= x \;\mid\; \mu x.c$

where $a \in P_{\underline{\mathcal{C}}}$ and $x \in \mathtt{V}$, whose respective typing judgments we denote with $c : X$ and $X \,|\, s$, where X ranges over finite sets. In expressions $c : X$ and $X \,|\, s$, the set X is the type of the command c and of the term s, respectively, and the backward typing judgment $X \,|\, s$ is used merely to further distinguish the representation of terms and commands.

The assignment of types to commands and terms is done by the following rules:

$$\frac{}{\{x\} \,|\, x} \qquad \frac{a \in \mathcal{C}(\{x_1, \ldots, x_n\}) \quad Y_i \,|\, t_i \text{ for all } i \in \{1, \ldots, n\}}{\underline{a}\{t_1, \ldots, t_n\} : \bigcup_{i=1}^{n} Y_i} \qquad \frac{X \,|\, s \quad Y \,|\, t}{\langle s \,|\, t \rangle : X \cup Y} \qquad \frac{c : X \;\; x \in X}{X \backslash \{x\} \,|\, \mu x.c}$$

where, in the second rule, the sets Y_i are pairwise disjoint, as are X and Y in the third rule.

Remark 2.1. *Observe that, thanks to the disjointness assumptions in the two rules for typing commands, for each term $\mu x.c$, where $c : X$, the variable x bound by μ has a unique occurence among the variables of X.*

Intuitively, commands mimick operations of the free cyclic operad over the functor $\underline{\mathcal{C}}$, and, thereby, a judgement $c : X$ should be thought of as describing an unrooted tree whose free variables are precisely the elements of X. On the other hand, terms represent operations with one selected entry and the role of the set X in a judgement $X \,|\, s$ is to label all entries *except* the selected one. From the tree-wise perspective, this is represented by an unrooted tree whose set of free variables is $X \cup \{x\}$, where x is precisely the variable bound by μ (i.e. the variable placed immediatelly on the right of the symbol μ).

Remark 2.2. *It is easily seen that, for any command $c : X$, the set X contains at least two elements. In particular, an expression of the form $\mu y.(\mu x.c)$ is not allowed by the typing rules. This reflects the constant-freeness requirement we imposed for cyclic operads.*

Notation 2.3. *We shall sometimes denote the commands introduced by the second typing rule as $\underline{a}\{t_x \,|\, x \in X\}$ (for $a \in \underline{\mathcal{C}}(X)$), or as $\underline{a}\{\sigma\}$, where σ assigns to every $x \in X$ a term t_x. The order of appearance of the t_x's in $\underline{a}\{t_x \,|\, x \in X\}$ is irrelevant. Whenever we use the notation, say $\underline{a}\{t, s\}$, for $a \in \underline{\mathcal{C}}(\{x, y\})$, it will be clear from the context whether we mean $\underline{a}\{t, s\} = \underline{a}\{\sigma\}$, with $\sigma(x) = t$ and $\sigma(y) = s$, or with σ defined in the other way around.*

The way commands are constructed is motivated by the action of the simultaneous and partial grafting of unrooted trees, formally defined through the composition operation ${}_x\bullet_y$ from §1.2.3. The command $\underline{a}\{t_x \,|\, x \in X\}$, introduced by the second rule, should be imagined as the simultaneous grafting of the corolla a and the "surrounding" trees t_x, one for each free variable x of a, along the variables bound by μ in each t_x. In the special case when, for some $x \in X$, the corresponding term t_x is a variable, say u, this process of grafting reduces to the renaming of the variable x of the corolla a to u. Therefore, if all the terms corresponding, by the second typing rule, to the elements of X are variables from the set, say, $V = \{u, v, w, ...\}$, then the appropriate command is $\underline{a}\{\sigma\}$, where $\sigma : V \to X$ is the bijection determined by that correspondence, and it describes the unrooted tree $\{a^{\sigma}(u, v, w, \ldots); id_V\}$. The command $\langle s \,|\, t \rangle$ describes the grafting of unrooted trees represented by the terms s and t along their variables bound by μ. Therefore, the pattern $\langle \mu x.\_ \,|\, \mu y.\_ \rangle$ corresponds to the composition $(-)_x\bullet_y(-)$ on classes of unrooted trees.

The equations of the μ-syntax are

$\langle s \,\vert\, t\rangle = \langle t \,\vert\, s\rangle$	(MU1)	$\mu x.c = \mu y.c[y/x]$	(MU3)
$\langle \mu x.c \,\vert\, s\rangle = c[s/x]$	(MU2)	$\underline{a}\{t_x \,\vert\, x \in X\} = \underline{a^\sigma}\{t_{\sigma(y)} \,\vert\, y \in Y\}$	(MU4)

where, in (MU2), $c[s/x]$ denotes the command c in which the unique occurrence of the variable x in X (see Remark 2.1) has been replaced by the term s, in (MU3) y is fresh with respect to all variables of c except x, and in (MU4) $\sigma : Y \to X$ is an arbitrary bijection.

The equation (MU1) stipulates the symmetry of grafting of unrooted trees, i.e. the commutativity of composition operations ${}_x\bullet_y$.

The equations (MU3) and (MU4) are α-conversions (cf. §1.2.1). Intuitively, α-conversion tells that the name of the entry selected for the composition does not matter, which reflects a part of the equivariance of composition operations ${}_x\bullet_y$. In simpler terms, α-conversion tells that the function $f(x)$ is the same as the function $f(y)$.

The substitution $c[s/x]$, figuring in the equation (MU2) (as well as the substitution $c[y/x]$ from (MU3)), must be performed in the *capture-avoiding* manner. This means that the variables which were originally "free" (i.e. *not bound* by μ) in c cannot become "captured" (i.e. *bound* by μ) after the substitution is made. This is achieved by renaming, prior to the substitution, all the bound variables in c and s, so that they are all turned mutually distinct, and then performing the appropriate substitution. For example,

$$\mu x.\underline{a}\{x,y\}[x/y] \neq \mu x.\underline{a}\{x,x\}, \quad \text{but} \quad \mu x.\underline{a}\{x,y\}[x/y] = \mu z.\underline{a^\sigma}\{z,y\}[x/y] = \mu z.\underline{a^\sigma}\{z,x\},$$

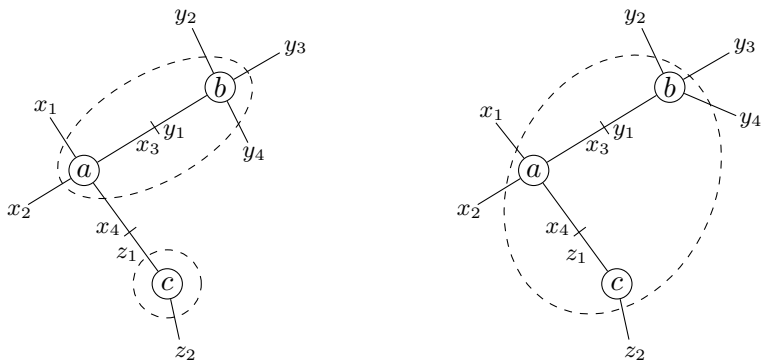
where σ renames x to z.

The equation (MU2)) is quite evidently reminescent of the β-reduction of λ-calculus, when considered as a rewriting rule $\langle \mu x.c \,\vert\, s\rangle \to c[s/x]$, and it essentially captures the same idea of function application as λ-calculus. The intuition becomes more tangible from the point of view of trees: the commands $\langle \mu x.c \,\vert\, s\rangle$ and $c[s/x]$, equated with (MU2), describe two ways to build (by means of grafting) the same unrooted tree. Here is an example.

Example 2.4. Consider the unrooted tree

$$\mathcal{T} = \{a(x_1,x_2,x_3,x_4), b(y_1,y_2,y_3,y_4), c(z_1,z_2); \sigma\},$$

where $\sigma = (x_3\ y_1)(x_4\ z_1)$. One way to build $\mathcal{T}$ is to graft along x_4 and z_1 unrooted trees $\mathcal{T}_1 = \{a(x_1,x_2,x_3,x_4), b(y_1,y_2,y_3,y_4); \sigma_1\}$, where $\sigma_1 = (x_3\ y_1)$, and $\mathcal{T}_2 = \{c(z_1,z_2); id_{\{z_1,z_2\}}\}$, singled out with dashed lines in the left picture below:



The unrooted tree $\mathcal{T}_1$ (in the upper part of the left picture) can itself be seen as a grafting, namely the simultaneous grafting of the corolla a and its surrounding trees: in this case this

involves explicit grafting only with the corolla b (along the free variables x_3 and y_1). This way of constructing $\mathcal{T}$ is described by the command

$$\langle \mu x_4.\underline{a}\{x_1, x_2, \mu y_1.\underline{b}\{y_1, y_2, y_3, y_4\}, x_4\} \,|\, \mu z_1.\underline{c}\{z_1, z_2\}\rangle \qquad (*)$$

that witnesses the fact that $\mathcal{T}_1$ and $\mathcal{T}_2$ are connected along their selected free variables x_4 and z_1, respectively: x_4 and z_1 are bound with μ in the terms corresponding to these two trees. The subterm $\underline{a}\{x_1, x_2, \mu y_1.\underline{b}\{y_1, y_2, y_3, y_4\}, x_4\}$ on the left-hand side is the command that accounts for the simultaneous grafting of the corolla a and its surrounding trees, while $\underline{c}\{z_1, z_2\}$ on the right-hand side stands for the corolla c. On the other hand, we could have chosen to build the tree $\mathcal{T}$ simply by making the simultaneous grafting of the corolla a and its surrounding trees, as indicated on the picture on the right. This way of building $\mathcal{T}$ is described with the command $\underline{a}\{x_1, x_2, \mu y_1.\underline{b}\{y_1, y_2, y_3, y_4\}, \mu z_1.\underline{c}\{z_1, z_2\}\}$, which is, up to substitution, exactly the command

$$\underline{a}\{x_1, x_2, \mu y_1.\underline{b}\{y_1, y_2, y_3, y_4\}, x_4\}[\mu z_1.\underline{c}\{z_1, z_2\}/x_4]$$

to which (*) reduces by applying the rewriting rule $\langle \mu x.c \,|\, s\rangle \to c[s/x]$.

We shall denote with $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}$ the set of all expressions of the μ-syntax induced by $P_{\underline{\mathcal{C}}}$, and we shall use $\mu\mathtt{Term}_{\underline{\mathcal{C}}}$ and $\mu\mathtt{Comm}_{\underline{\mathcal{C}}}$ to denote the subsets of terms and commands of $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}$, respectively. As in the case of the combinator syntax, the set of expressions (resp. terms and commands) of type X will be denoted by $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}(X)$ (resp. $\mu\mathtt{Term}_{\underline{\mathcal{C}}}(X)$ and $\mu\mathtt{Comm}_{\underline{\mathcal{C}}}(X)$).

2.2 μ-syntax as a rewriting system Let $\rightsquigarrow$ be the rewriting relation defined on $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}$ as (the reflexive and transitive closure of) the union of rewriting rules

$$\langle s \,|\, t\rangle \rightsquigarrow \langle t \,|\, s\rangle \quad \text{and} \quad \langle \mu x.c \,|\, s\rangle \rightsquigarrow c[s/x]$$

obtained by orienting from left to right the equations `(MU1)` and `(MU2)`, respectively, which is, moreover, congruent with respect to `(MU3)`, `(MU4)` and substitution[1].

The non-confluence of the rewriting system $(\mu\mathtt{Exp}_{\underline{\mathcal{C}}}, \rightsquigarrow)$ shows up immediately: for the reductions

$$c_2[\mu x.c_1/y] \leftsquigarrow \langle \mu x.c_1 \,|\, \mu y.c_2\rangle \rightsquigarrow c_1[\mu y.c_2/x]$$

arising due to `(MU1)` (which makes the whole reduction system symmetric), we do not have a way to exhibit a command c, such that $c_2[\mu x.c_1/y] \rightsquigarrow c$ and $c_1[\mu y.c_2/x] \rightsquigarrow c$. Nevertheless, all three commands above describe the same unrooted tree.

However, modulo the trivial commuting conversion, this rewriting system is terminating: the number of μ-binders in an expression is strictly decreasing at each reduction step of the form $\langle \mu x.c \,|\, s\rangle \rightsquigarrow c[s/x]$ (which makes it impossible to have an infinite sequence of reductions of this kind). It is straightforward to prove that the set $\mu\mathtt{Exp}^{nf}_{\underline{\mathcal{C}}} = \mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}} \cup \mu\mathtt{Term}^{nf}_{\underline{\mathcal{C}}}$ of normal forms is generated by the following rules:

$$\frac{}{x \in \mu\mathtt{Term}^{nf}_{\underline{\mathcal{C}}}} \qquad \frac{a \in \underline{\mathcal{C}}(X) \quad t_x \in \mu\mathtt{Term}^{nf}_{\underline{\mathcal{C}}} \text{ for all } x \in X}{\underline{a}\{t_x \,|\, x \in X\} \in \mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}} \qquad \frac{c \in \mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}}{\mu x.c \in \mu\mathtt{Term}^{nf}_{\underline{\mathcal{C}}}}$$

[1] Since the precautionary renaming which ensures that the substitution $c[s/x]$ is done in the capture-free manner is *non-deterministinc*, the rewriting relation $\rightsquigarrow$ is formally defined on the equivalence classes of expressions of the μ-syntax with respect to `(MU3)` and `(MU4)`, just as the usual rewriting systems in λ-calculus are actually defined on α-conversion classes.

In the next example, we examine the shape of normal forms in relation with unrooted trees.

EXAMPLE 2.5. Let $\mathcal{T}$ be the unrooted tree from EXAMPLE 2.4. Here is the list of commands in normal form that describe $\mathcal{T}$:

$$\underline{a}\{x_1, x_2, \mu y_1.\underline{b}\{y_1, y_2, y_3, y_4\}, \mu z_1.\underline{c}\{z_1, z_2\}\},$$

$$\underline{b}\{\mu x_3.\underline{a}\{x_1, x_2, x_3, \mu z_1.\underline{c}\{z_1, z_2\}\}, y_2, y_3, y_4\}, \quad \underline{b}\{\mu x_3.\underline{c}\{\mu x_4.\underline{a}\{x_1, x_2, x_3, x_4\}, z_2\}, y_2, y_3, y_4\},$$

$$\underline{c}\{\mu x_4.\underline{a}\{\mu y_1.\underline{b}\{y_1, y_2, y_3, y_4\}, x_2, x_3, x_4\}, z_2\}, \quad \underline{c}\{\mu x_4.\underline{b}\{\mu x_3.\underline{a}\{x_1, x_2, x_3, x_4\}, y_2, y_3, y_4\}, z_2\}.$$

Each of the commands records the free variables and corollas of $\mathcal{T}$: free variables are the variables not bound with μ (x_1, x_2, y_2, y_3, y_4 and z_2), and the corollas correspond to the underlined parameters (a, b and c). The variables involved in edges of $\mathcal{T}$ (x_3, y_1, x_4 and z_1) can also be recovered from the list, as the variables bound with μ. For example, in the first command we see that y_1 and z_1 are explicitly bound by μ, while for x_3 and x_4 we could say that they are implicitly bound, given that they are replaced with a non-variable term.

In general, the set $\mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}$ describes decompositions of unrooted trees of the following kind: pick a corolla a of a tree, and then proceed recursively so in all the connected components of the graph resulting from the removal of a . (We provide in §2.4.1 an algorithmic computation of these connected components).

Amusingly, one can show that, if `(MU1)` gets oriented in the other way around, the normal forms of the resulting rewriting system will be in one-to-one correspondence with the combinators of Section 1, and thus describe decompositions of unrooted trees of the following kind: pick an edge e of the tree, and then proceed recursively so in the two connected components of the graph resulting from the removal of e.

These two extremes substantiate our informal explanation of the μ-syntax as a mix of partial composition and simultaneous composition styles.

2.3 The interpretation of the μ-syntax in an arbitrary cyclic operad We next consider the semantic aspect of the μ-syntax relative to unrooted trees that we intuitively brought up in §2.1 and §2.2, by defining an interpretation function of the μ-syntax into a cyclic operad, as defined in Definition 1.6. We ascribe meaning to the μ-syntax by first translating it to the combinator syntax from Section 1.

The translation function

$$[[\_]] : \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \to \mathtt{cTerm}_{\underline{\mathcal{C}}}$$

is defined recursively as follows, wherein the assignment of a combinator to a term $t \in \mu\mathtt{Term}_{\underline{\mathcal{C}}}$ is indexed by a variable that is fresh relative to t:

- $[[x]]_y = id_{x,y}$,
- if, for each $x \in X$, $[[t_x]]_{\overline{x}}$ is a translation of the term t_x, then

  $$[[\underline{a}\{t_x \mid x \in X\}]] = a(\varphi),$$

  where $a(\varphi)$ denotes the combinator corresponding to the simultaneous composition determined by $a \in \mathcal{C}(X)$ and $\varphi : x \mapsto ([[t_x]]_{\overline{x}}, \overline{x})$ (see (1.1)),
- $[[\mu x.c]]_y = [[c[y/x]\,]]$, and
- $[[\langle s \mid t \rangle]] = [[s]]_{x\ x} \circ_y [[t]]_y$.

In order to show that $[[\_]]$ is well-defined, we introduce the following notational conventions. For a command $c : X$ (resp. term $X \mid t$) and a bijection $\sigma : X' \to X$, we define

$$c^\sigma := c[\dots, \sigma^{-1}(x)/x, \dots] \text{ (resp. } t^\sigma := t[\dots, \sigma^{-1}(x)/x, \dots])$$

as a simultaneous substitution (renaming) of the variables from the set X (guided by σ). One of the basic properties of the introduced substitution is the equality $(\mu a.c)^\sigma = \mu a.c^{\sigma_a}$ (for the definition of σ_a, see the paragraph Notation and conventions in the Introduction).

The way c^σ is defined indicates that its translation should be the combinator $[[c]]^\sigma : X'$. The following lemma ensures that this is exactly the case. In its statement, $[[\_]]_X$ denotes the restriction of $[[\_]]$ on $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}(X)$.

Lemma 2.6. *For a bijection $\sigma : X' \to X$, $t \in \mu\mathtt{Term}_{\underline{\mathcal{C}}}(X)$ and $c \in \mu\mathtt{Comm}_{\underline{\mathcal{C}}}(X)$, the following two equalities hold:*

$$[[t^\sigma]]_y = [[t]]_y^{\sigma_y} \qquad \text{and} \qquad [[c^\sigma]] = [[c]]^\sigma.$$

To verify that the definition of $[[\_]]$ is valid, we shall also need the following result.

Lemma 2.7 (Substitution lemma)**.** *Let $X \cap Y = \emptyset$, $t \in \mu\mathtt{Term}_{\underline{\mathcal{C}}}(Y)$ and $x \in X$. Then, for $s \in \mu\mathtt{Term}_{\underline{\mathcal{C}}}(X)$ and $c \in \mu\mathtt{Comm}_{\underline{\mathcal{C}}}(X)$, the following two equalities hold:*

$$[[s[t/x]]]_u = [[s]]_u \,{}_x\circ_v [[t]]_v \qquad \text{and} \qquad [[c[t/x]]] = [[c]] \,{}_x\circ_v [[t]]_v.$$

Proof. If t is a variable, say y, then, by `(U1)` and `(EQ)`, we get

$$[[s[y/x]]]_u = [[s^{id_X^{y/x}}]]_u = [[s]]_u{}^{id_X^{y/x}} = [[s]]_u \,{}_x\circ_v id_{v,y} = [[s]]_u \,{}_x\circ_v [[y]]_v\,,$$

and, analogously,

$$[[c[y/x]]] = [[c^{id_X^{y/x}}]] = [[c]]^{id_X^{y/x}} = [[c]] \,{}_x\circ_z id_{z,y} = [[c]] \,{}_x\circ_z [[y]]_z.$$

If $t = \mu y.c_1$, we proceed by induction on the structure of s, i.e. c.

- If $s = x$, then, again by `(U1)` and `(EQ)`, we get

$$\begin{aligned}
[[x[\mu y.c_1/x]]]_u &= [[\mu y.c_1]]_u = [[c_1[u/y]]] = [[c_1^{id_Y^{u/y}}]] \\
&= [[c_1]]^{id_Y^{u/y}} = [[c_1]] \,{}_y\circ_x id_{x,u} = [[c_1]] \,{}_y\circ_x [[x]]_u = [[c_1[u/y]]] \,{}_u\circ_x [[x]]_u.
\end{aligned}$$

- Next, assume that $c : X \cup \{z\}$ satisfies the equality and let $s = \mu z.c$. Denote $U = X\backslash\{x\} \cup \{z\} \cup Y$. By `(U1)` and `(EQ)`, we have

$$\begin{aligned}
[[\mu z.c[\mu y.c_1/x]]]_u &= [[\mu z.(c[\mu y.c_1/x])]]_u = [[c[\mu y.c_1/x][u/z]]] = [[c[\mu y.c_1/x]^{id_U^{u/z}}]] \\
&= [[c[\mu y.c_1/x]]]^{id_U^{u/z}} = ([[c]] \,{}_x\circ_y [[c_1]])^{id_U^{u/z}} = [[c]]^{id_U^{u/z}} \,{}_x\circ_y [[c_1]] \\
&= [[c[u/z]]] \,{}_x\circ_v [[c_1[v/y]]] = [[\mu v.c]]_u \,{}_x\circ_v [[c_1[v/y]]].
\end{aligned}$$

- Let $X = X_1 \cup X_2$ and suppose that $c = \langle t_1 \mid t_2\rangle$, where $X_1 \mid t_1$ and $X_2 \mid t_2$ satisfy the claim. Without loss of generality, we can assume that $x \in X_2$. By `(A1)`, we have

$$\begin{aligned}
[[\langle t_1 \mid t_2\rangle[\mu y.c_1/x]]] &= [[\langle t_1 \mid t_2[\mu y.c_1/x]\rangle]] = [[t_1]]_u \,{}_u\circ_v [[t_2[\mu y.c_1/x]]]_v \\
&= [[t_1]]_u \,{}_u\circ_v ([[t_2]]_v \,{}_x\circ_w [[\mu y.c_1]]_w) = ([[t_1]]_u \,{}_u\circ_u [[t_2]]_u) \,{}_x\circ_w [[\mu y.c_1]]_w \\
&= [[\langle t_1 \mid t_2\rangle]] \,{}_x\circ_v [[\mu y.c_1]]_v.
\end{aligned}$$

- Finally, let $X = \bigcup_{z\in Z} Y_z$ and suppose that $c = \underline{a}\{t_z \,|\, z \in Z\}$, where for all $z \in Z$, $Y_z \,|\, t_z$ satisfy the claim. Suppose, moreover, that for $u \in Z$, $x \in Y_u$. Then, on one hand, we have

$$[[\underline{a}\{t_z \,|\, z \in Z\}[\mu y.c_1/x]]] = [[\underline{a}\{\{t_z \,|\, z \in Z\backslash\{u\}\} \cup \{t_u[\mu y.c_1/x]\}\}]] = a(\varphi),$$

  where $\varphi : z \mapsto ([[t_z]]_{\overline{z}}, \overline{z})$, for all $z \in Z\backslash\{u\}$, and $\varphi : u \mapsto ([[t_u[\mu y.c_1/x]]]_{\overline{u}}, \overline{u})$. On the other hand,

$$[[\underline{a}\{t_z \,|\, z \in Z\}]] \,{}_x\circ_v\, [[\mu y.c_1]]_v = a(\psi_1) \,{}_x\circ_v\, [[\mu y.c_1]]_v,$$

  where $\psi_1 : z \mapsto ([[t_z]]_{\overline{z}}, \overline{z})$, for all $z \in Z$. By Lemma 1.5,

$$a(\psi_1) \,{}_x\circ_v\, [[\mu y.c_1]]_v = a(\psi_2),$$

  where $\psi_2 = \psi_1$ on $Z\backslash\{a\}$, and $\psi_2 : u \mapsto ([[t_u]]_{\overline{u}} \,{}_x\circ_v\, [[\mu y.c_1]]_v, \overline{u})$. Hence, we need to prove that

$$[[t_u[\mu y.c_1/x]]]_{\overline{u}} = [[t_u]]_{\overline{u}} \,{}_x\circ_v\, [[\mu y.c_1]]_v,$$

  but this equality is exactly the induction hypothesis for the term t_u. ■

Let $=_\mu$ (resp. $=$) be the smallest equivalence relation on $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}$ (resp. $\mathtt{cTerm}_{\underline{\mathcal{C}}}$) generated by the equations of μ-syntax (resp. by the equations of Definition 1.1).

Theorem 2.8. *The translation function* $[[\_]] : \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \to \mathtt{cTerm}_{\underline{\mathcal{C}}}$ *is well-defined, i.e., it induces a map from* $\mu\mathtt{Exp}_{\underline{\mathcal{C}}}/_{=_\mu}$ *to* $\mathtt{cTerm}_{\underline{\mathcal{C}}}/_{=}$*. Moreover, the induced map is a bijection.*

Proof. The equation `(MU1)` is valid in the world of combinators, as it gets translated to `(CO)`. As for `(MU2)`, for a command $c : X$, by Lemma 2.7, we get:

$$[[\langle \mu x.c \,|\, t\rangle]] = [[\mu x.c]]_u \,{}_u\circ_v\, [[t]]_v = [[c[u/x]]] \,{}_u\circ_v\, [[t]]_v = [[c]]^{id_X^{u/x}} \,{}_u\circ_v\, [[t]]_v = [[c]] \,{}_x\circ_v\, [[t]]_v = [[c[t/x]]].$$

For `(MU3)` and `(MU4)`, we have

$$[[\mu x.c]]_u = [[c[u/x]]] = [[c[y/x][u/y]]] = [[\mu y.c[y/x]]]_u$$

and

$$[[\underline{a}^\sigma\{t_{\sigma(y)} \,|\, y \in Y\}]] = a^\sigma(\varphi') = a^\sigma(\varphi \circ \sigma) = a(\varphi) = [[\underline{a}\{t_x \,|\, x \in X\}]],$$

where $\varphi' : y \mapsto ([[t_{\sigma(y)}]]_{\overline{\sigma(y)}}, \overline{\sigma(y)})$ and $\varphi : \sigma(y) \mapsto ([[t_{\sigma(y)}]]_{\overline{\sigma(y)}}, \overline{\sigma(y)})$, respectively. The inverse translation is obtained via the correspondence $(-)_x\circ_y(-) \;\mapsto\; \langle \mu x.\_ \,|\, \mu y.\_\rangle$. ■

We define the interpretation of the μ-syntax in an arbitrary cyclic operad $\mathcal{C}$, as the composition

$$[\,[[\_]]\,]_{\mathcal{C}} : \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \to \mathcal{C},$$

where the interpretation $[\_]_{\mathcal{C}} : \mathtt{cTerm}_{\underline{\mathcal{C}}} \to \mathcal{C}$ arises as explained after Definition 1.6.

2.4 μ-syntax does the job! The theorem below puts the μ-syntax in line with already established frameworks for defining a cyclic operad.

Theorem 2.9. *The quotient set of the commands of the* μ*-syntax relative to the relation* $=_\mu$*, is in one-to-one correspondence with the one of unrooted trees relative to the* α*-conversion. In other words, for every finite set* X*, there exists a bijection*

$$\Phi_X : \mu\mathtt{Comm}_{\underline{\mathcal{C}}}(X)_{/=_\mu} \to \mathtt{T}_{\underline{\mathcal{C}}}(X).$$

The proof of Theorem 2.9 goes through a new equality $='$ on $\mu\mathtt{Comm}^{nf}_{\underline{\mathbb{C}}}(X)$, as well as suitably tailored decompositions of unrooted trees, necessary for establishing the injectivity of Φ_X. We first describe these decompositions and the equality $='$ and then prove the theorem.

Remark 2.10. *By putting together theorems 2.8 and 2.9, we get that the combinator syntax and its equations provide a presentation of* $\mathtt{T}_{\underline{\mathbb{C}}}(X)$. *This remark gets all its flavour when trees, or, more precisely, forests, are set to be* morphisms *of a suitable category, as proposed in [2]. A forest is a graph with no cycles, but not necessarily connected. Here we remove the node labelling, i.e. corollas are just subsets of flags/variables. The objects of this category are aggregates (in the terminology of [9]), i.e., sets of (ordinary) corollas, with no involution, and the morphisms are forests. Each forest determines its domain and codomain: the domain is obtained by removing the edge information, the codomain is obtained by taking as corollas the connected components of the forest, with its free variables (the construction of course works for graphs in general). Composition is performed by replacing one forest's corollas with connected components of another forest, with appropriate identification of half-edges.*

We sketch here a route to Theorem 1.22 that is more abstract than the one that we followed here:

1. *Theorem 2.9 can be reformulated to the statement that Costello's category* **Forests** *of forests is presented by the combinator syntax.*
2. *The authors of [9] argue that, for general reasons, the unbiased definition of (cyclic) operads is equivalent to a third definition as monoidal functors from* **Forests**.

Now, thanks to (1), such monoidal functors are easily seen to be equivalent to the data of a biased structure of (cyclic) operad. Hence we get Theorem 1.22 by transitivity. In the sequel, we follow a more "pedestrian" route, but what we sketched here is arguably more conceptual.

2.4.1 "Pruning" of unrooted trees We describe an algorithm that takes an ordinary unrooted tree $\mathcal{T}$, a corolla $a \in Cor(\mathcal{T})$ and a variable $v \in FV(a)\backslash FV(\mathcal{T})$, and returns a proper subtree $\mathcal{T}_v$ of $\mathcal{T}$, the subtree "plucked" from a at the junction of v and $\sigma(v)$, where σ is the involution of $\mathcal{T}$. In the sequel, for an arbitrary corolla $b \in Cor(\mathcal{T})$ and $w \in FV(b)\backslash FV(\mathcal{T})$, $S_w(b)$ will denote the corolla adjacent to b along the edge $(w, \sigma(w))$, if such a corolla exists.

We first specify how to generate the set $Cor(\mathcal{T}_v)^+$ of pairs of a corolla of $\mathcal{T}_v$ and one of its free variables, by the following formal rules:

$$\frac{}{(S_v(a), \sigma(v)) \in Cor(\mathcal{T}_v)^+} \qquad \frac{(b,u) \in Cor(\mathcal{T}_v)^+ \quad w \in FV(b)\backslash(FV(\mathcal{T}) \cup \{u\})}{(S_w(b), \sigma(x)) \in Cor(\mathcal{T}_v)^+}$$

This formal system has the following properties.

Remark 2.11. *Each element* $(S_w(b), \sigma(w)) \in Cor(\mathcal{T}_v)^+$ *is such that* $S_w(b)$ *is adjacent to* b *in* $\mathcal{T}$. *For each* $(b,u) \in Cor(\mathcal{T}_v)^+$, *we have* $b \neq a$. *Intuitively, by iterative application of the second rule, a way to traverse a branch of the tree* $\mathcal{T}$ *is determined.*

We obtain the set of corollas of $\mathcal{T}_v$ by erasing from the elements of $Cor(\mathcal{T}_v)^+$ the data about the distinguished free variables, i.e. we define

$$Cor(\mathcal{T}_v) = \{b \,|\, (b,u) \in Cor(\mathcal{T}_v)^+ \text{ for some } u \in FV(b)\}.$$

The involution $\sigma_{\mathcal{T}_v}$ of $\mathcal{T}_v$ is defined as

$$\sigma_{\mathcal{T}_v}(z) = \begin{cases} \sigma(z) & \text{if } z \in \left(\bigcup_{b \in Cor(\mathcal{T}_v)} FV(b)\right) \backslash \sigma(v) \\ z & \text{if } z = \sigma(v)\,. \end{cases}$$

We shall denote the algorithm with $\mathcal{P}$, and the result $\mathcal{P}(\mathcal{T}, a, v)$ of instatiating $\mathcal{P}$ on a tree $\mathcal{T}$, a corolla $a \in Cor(\mathcal{T})$, and a variable $v \in FV(a) \backslash FV(\mathcal{T})$ will often be denoted as $\mathcal{T}_v$, as we have just done above. The following claim guarantees that $\mathcal{P}$ is correct.

Lemma 2.12. *For un unrooted tree $\mathcal{T}$, $a \in Cor(\mathcal{T})$ and $v \in FV(a) \backslash FV(\mathcal{T})$, $\mathcal{T}_v$ is a proper subtree of $\mathcal{T}$.*

Proof. By the construction, we have that $Cor(\mathcal{T}_v) \subseteq Cor(\mathcal{T})$ and that $\mathcal{T}_v$ is connected. By Remark 2.11, it follows that $Cor(\mathcal{T}_v)$ is a proper subset of $Cor(\mathcal{T})$. Finally, since $\sigma_{\mathcal{T}_v} = \sigma$ on $V(\mathcal{T}_v) \backslash FV(\mathcal{T}_v)$, we can conclude that $\mathcal{T}_v$ is indeed a subtree of $\mathcal{T}$. ■

For an unrooted tree $\mathcal{T}$ and $a \in Cor(\mathcal{T})$, we shall refer to the set of unrooted trees $\mathcal{P}(\mathcal{T}, a)$, defined by

$$\mathcal{P}(\mathcal{T}, a) = \{\{a(y_1, \ldots, y_n); id_Y\}\} \cup \{\mathcal{T}_v \,|\, v \in Y \backslash FV(\mathcal{T})\},$$

where $Y = \{y_1, \ldots, y_n\}$, as the *decomposition* of $\mathcal{T}$ *relative to the corolla* a. Observe that, since, by the connectedness, every corolla of $\mathcal{T}$ is connected to the corolla a via a path, and hence belongs to the subtree $\mathcal{T}_v$, where v is the free variable of a through which the path enters a, the whole tree $\mathcal{T}$ can be "recovered" in the obvious way from $\mathcal{P}(\mathcal{T}, a)$. From the point of view of the appropriate composition of classes of trees, this gives us the following result.

Lemma 2.13. *Let $\mathcal{T}$ be an unrooted tree and let $a \in Cor(\mathcal{T})$. Suppose that $FV(\mathcal{T}) = X$ and $FV(a) = Y$, where $Y = \{y_1, \ldots, y_n\}$. Let $I = \{i_1, \ldots i_k\} = \{i \in \{1, \ldots, n\} \,|\, y_i \in FV(a) \backslash X\}$. Then, if $\mathcal{P}(\mathcal{T}, a) = \{\{a(y_1, \ldots, y_n); id_Y\}\} \cup \{\mathcal{T}_{y_i} \,|\, i \in I\}$, we have that*

$$[\mathcal{T}]_\alpha = (([\{a(y_1, \ldots, y_n); id_Y\}]_\alpha \, {}_{y_{i_1}}\!\bullet_{\sigma(y_{i_1})} [\mathcal{T}_{y_{i_1}}]_\alpha) \cdots) \, {}_{y_{i_k}}\!\bullet_{\sigma(y_{i_k})} [\mathcal{T}_{y_{i_k}}]_\alpha\,.$$

Proof. Given that the trees from $\mathcal{P}(\mathcal{T}, a)$ have mutually disjoint sets of corollas and variables, the equality holds since the composition on the right-hand can be "calculated" without any "precautionary" renaming. ■

Lemma 2.14. *If an unrooted tree $\mathcal{T}$ has at least two corollas, then there exists $c \in Cor(\mathcal{T})$, such that $FV(c) \backslash FV(\mathcal{T})$ is a singleton.*

Proof. Suppose that $FV(\mathcal{T}) = X$ and let σ be the involution of $\mathcal{T}$. We proceed by induction on the number n of corollas of $\mathcal{T}$. For the base case, suppose that $Cor(\mathcal{T}) = \{a, b\}$. Then there exist $x \in FV(a)$ and $y \in FV(b)$ such that $\sigma(x) = y$, while all other variables of $\mathcal{T}$ are fixpoints of σ. Hence, $FV(a) \backslash FV(\mathcal{T}) = \{x\}$ and $FV(b) \backslash FV(\mathcal{T}) = \{y\}$, i.e. a and b both satisfy the claim.

Assume now that $\mathcal{T}$ has n corollas, where $n > 2$. Let $a \in Cor(\mathcal{T})$, $FV(a) = Y$, be such that there exists $v \in Y \backslash X$. If v is the unique such variable we are done. If not, let $\{C; id_Y\} \cup \{\mathcal{T}_u \,|\, u \in Y \backslash X\}$ be the decomposition of $\mathcal{T}$ obtained by applying $\mathcal{P}$ on a. Then, if $Cor(\mathcal{T}_v) = \{S_v(a)\}$, by the definition of $\mathcal{P}$, we know that $FV(S_v(a)) \backslash X = \{\sigma(v)\}$. Therefore, since $Cor(\mathcal{T}_v) \subseteq Cor(\mathcal{T})$, $S_v(a)$ is a corolla that satisfies the claim. On the other hand, if $\mathcal{T}_v$ contains more than one corolla, by the induction hypothesis on $\mathcal{T}_v$, we get $b \in Cor(\mathcal{T}_v)$ such that $FV(b) \backslash FV(\mathcal{T}_v) = \{w\}$. Since $FV(b) \backslash X \subseteq FV(b) \backslash FV(\mathcal{T}_v)$, we know that either $FV(b) \backslash X = \{w\}$, or $FV(b) \backslash X = \emptyset$. The latter is impossible because b would be the only corolla of $\mathcal{T}$. ■

Let $\mathcal{T}$ and c be as in the previous lemma, and let $FV(c)\backslash FV(\mathcal{T}) = \{v\}$. We shall denote with $\mathcal{T}_{/c}$ the unrooted tree such that $Cor(\mathcal{T}_{/c}) = Cor(\mathcal{T})\backslash\{c\}$ and whose involution $\sigma_{/c}$ agrees with the involution σ of $\mathcal{T}$ everywhere, except on $\sigma(v)$, which is a fixpoint of $\sigma_{/c}$. Lemma 2.14 guarantees that $\mathcal{T}_{/c}$ is well-defined.

We now establish a non-inductive characterisation of the output of the algorithm $\mathcal{P}$.

Lemma 2.15. *Let $\mathcal{T}$ be an unrooted tree with involution σ and let $a \in Cor(\mathcal{T})$ and $v \in FV(a)\backslash FV(\mathcal{T})$. The following properties are equivalent for a subtree $\mathcal{T}'$ of $\mathcal{T}$:*

1. *$\mathcal{T}' = \mathcal{P}(\mathcal{T}, a, v)$,*
2. *$\sigma(v) \in FV(\mathcal{T}')$ and $FV(\mathcal{T}')\backslash\{\sigma(v)\} \subseteq FV(\mathcal{T})$.*

Proof. That (1) implies (2) is clear.

We prove that (2) implies (1) by induction on the number n of corollas of $\mathcal{T}'$. If $n = 1$, then, since $\sigma(v) \in FV(\mathcal{T}')$, $S_v(a)$ is the only corolla of $\mathcal{T}'$ and the conclusion follows since, by the assumption, $FV(\mathcal{T}')\backslash\{\sigma(v)\} = FV(S_v(a))\backslash\{\sigma(v)\} \subseteq X$, i.e. $FV(S_v(a))\backslash\{X \cup \{\sigma(v)\}\} = \emptyset$.

Suppose that $n \geq 2$, and let, by Lemma 2.14, $c \in Cor(\mathcal{T}')$ be such that $FV(c)\backslash FV(\mathcal{T}') = \{u\}$. If $c = S_v(a)$, then it follows easily that $\mathcal{T}' = \mathcal{T}_v$. If not, by applying the induction hypothesis on $\mathcal{T}'_{/c}$, we get that $\mathcal{T}'_{/c} = \mathcal{P}(\mathcal{T}_{/c}, a, v)$. Observe that $(S_u(c), w) \in Cor(\mathcal{T}'_{/c})^+$, for some $w \in FV(S_u(c))$ different from $\sigma(u)$. By continuing $\mathcal{P}$ on $(S_u(c), w)$ and $\sigma(u)$, we get the pair (c, u), and the claim follows since $FV(c)\backslash(FV(\mathcal{T}) \cup \{u\}) = \emptyset$ (i.e. the algorithm stops) and since $\mathcal{T}'_{/c}$ and the single-corolla unrooted tree determined by c make a decomposition of $\mathcal{T}'$. ∎

For the following two lemmas, recall the definition of the simultaneous composition (1.1) for entries-only cyclic operads. We shall instantiate it on the cyclic operad of classes of unrooted trees, described in §1.2.3.

Lemma 2.16. *Let $a \in \underline{\mathcal{C}}(X)$, where $X = \{x_1, \ldots, x_n\}$, and let, for all $x_i \in X$, $\gamma : x_i \mapsto ([\mathcal{T}_{x_i}]_\alpha, \overline{x_i})$ be an assignment for which the simultaneous composition $[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma)$ is well-defined. Then the following properties hold.*

a) The α-equivalence class $[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma)$ admits a representative $\mathcal{T}$, such that $a \in Cor(\mathcal{T})$.

b) If $\mathcal{T}$ is a representative of $[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma)$, such that $a \in Cor(\mathcal{T})$, and if σ is the involution of $\mathcal{T}$, then each class $[\mathcal{T}_{x_i}]_\alpha$ admits the unrooted tree $\mathcal{P}(\mathcal{T}, a, x_i)^{\rho_i}$, where ρ_i renames $\sigma(x_i)$ to $\overline{x_i}$, as a representative.

Proof. Observe that there are two stages of renaming involved in forming the simultaneous composition $[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma)$. By (1.1), we first rename the free variables of the corolla a, obtaining in this way the composition

$$(\cdots([\{a^\sigma(x'_1, \ldots, x'_n); id_{X'}\}]_\alpha \, {}_{x'_1}\bullet_{\overline{x_1}} \, [\mathcal{T}_{x_1}]_\alpha)\cdots) \, {}_{x'_n}\bullet_{\overline{x_n}} \, [\mathcal{T}_{x_n}]_\alpha,$$

where $X' = \{x'_1, \ldots, x'_n\}$ and $\sigma : X' \to X$ is defined by $\sigma(x'_i) = x_i$, which is then "calculated" by the definition of ${}_x\bullet_y$ from §1.2.3. This calculation involves the renaming of variables of all the trees from the above composition, in such a way that the resulting trees have mutually disjoint sets of variables, i.e. it goes though the simultaneous composition

$$(\cdots([\{a^{\sigma\circ\tau}(y_1, \ldots, y_n); id_Y\}]_\alpha \, {}_{y'_1}\bullet_{\overline{y_1}} \, [\mathcal{T}_{x_1}^{\tau_1 \cup id_{FV(\mathcal{T}_{x_i})\backslash\{\overline{x_i}\}}}]_\alpha)\cdots) \, {}_{y'_n}\bullet_{\overline{y_n}} \, [\mathcal{T}_{x_n}^{\tau_n \cup id_{FV(\mathcal{T}_{x_n})\backslash\{\overline{x_n}\}}}]_\alpha,$$

where $Y = \{y_1, \ldots, y_n\}$, $\tau : Y \to X'$ is defined by $\tau(y_i) = x'_i$ and each $\tau_i : V_i \to (V(\mathcal{T}_{x_i}) \backslash FV(\mathcal{T}_{x_i})) \cup \{\overline{x_i}\}$ is such that $\tau_i(\overline{y_i}) = \overline{x_i}$. The resulting class now has as a representative the tree $\mathcal{T}'$, such that

$$Cor(\mathcal{T}') = \{a^{\sigma \circ \tau}(y_1, \ldots, y_n)\} \cup \bigcup_{1 \leq i \leq n} Cor(\mathcal{T}_{x_i}^{\tau_i})$$

and whose involution σ' is defined in the obvious way.

The first claim holds, since, thanks to the equivariance axiom (EQ) for ${}_x\bullet_y$, we can turn $\mathcal{T}'$ into an unrooted tree $\mathcal{T}$ that has a as a corolla, by "undoing" the renaming $\sigma \circ \tau$. Clearly, if some variable x_i appears in $\mathcal{T}'$, but did not originally come from the corolla a, this variable has to be renamed too, in order to ensure that all the variables of $\mathcal{T}$ are distinct. Therefore,

$$Cor(\mathcal{T}) = \{a(x_1, \ldots, x_n)\} \cup \bigcup_{1 \leq i \leq n} Cor((\mathcal{T}_{x_i}^{\tau_i})^{\kappa_i \cup id_{FV(\mathcal{T}_{x_i}) \backslash \{\overline{x_i}\}}}),$$

where $\kappa_i : U_i \cup \{\overline{z_i}\} \to V_i \cup \{\overline{y_i}\}$ is such that $\kappa_i(\overline{z_i}) = \overline{y_i}$ and the distinctness requirement for the variables of $\mathcal{T}$ is satisfied. The involution σ of $\mathcal{T}$ is defined from σ' in the obvious way.

For the second claim, fix an $i \in \{1, \ldots, n\}$. Observe that we have that

$$(\mathcal{T}_{x_i}^{\tau_i})_{x_i}^{\nu_i} =_\alpha \mathcal{T}_{x_i},$$

where ν_i renames $\overline{y_i}$ to $\overline{x_i}$. Also, we have that

$$\mathcal{T}_{x_i}^{\tau_i} =_\alpha ((\mathcal{T}_{x_i}^{\tau_i})^{\kappa_i \cup id_{FV(\mathcal{T}_{x_i}) \backslash \{\overline{x_i}\}}})^{\pi_i},$$

where π_i renames $\overline{z_i}$ to $\overline{y_i}$. Therefore,

$$(((\mathcal{T}_{x_i}^{\tau_i})^{\kappa_i \cup id_{FV(\mathcal{T}_{x_i}) \backslash \{\overline{x_i}\}}})^{\pi_i})^{\nu_i} =_\alpha \mathcal{T}_{x_i},$$

i.e. each class $[\mathcal{T}_{x_i}]_\alpha$ admits as a representative $((\mathcal{T}_{x_i}^{\tau_i})^{\kappa_i \cup id_{FV(\mathcal{T}_{x_i}) \backslash \{\overline{x_i}\}}})^{\rho_i}$, where ρ_i renames $\overline{z_i} = \sigma(x_i)$ to $\overline{x_i}$. Observe that $(\mathcal{T}_{x_i}^{\tau_i})^{\kappa_i \cup id_{FV(\mathcal{T}_{x_i}) \backslash \{\overline{x_i}\}}}$ is a subtree of $\mathcal{T}$. That we indeed have that

$$(\mathcal{T}_{x_i}^{\tau_i})^{\kappa_i \cup id_{FV(\mathcal{T}_{x_i}) \backslash \{\overline{x_i}\}}} = \mathcal{P}(\mathcal{T}, a, x_i)$$

is clear by the non-inductive criterion from Lemma 2.15: $\sigma(x_i)$ is a free variable of the tree on the left-hand side, and all the remaining free variables of that tree are free variables of $\mathcal{T}$. ■

Lemma 2.17. *Let $a \in \underline{\mathcal{C}}(X)$, where $X = \{x_1, \ldots, x_n\}$, and let, for all $x_i \in X$, $\gamma : x_i \mapsto ([\mathcal{T}_{x_i}]_\alpha, \overline{x_i})$ and $\tau : x_i \mapsto ([\mathcal{T}'_{x_i}]_\alpha, \tilde{x}_i)$ be assignments for which the simultaneous compositions*

$$[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma) \quad \text{and} \quad [\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\tau)$$

are well-defined. Then, if $[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma) = [\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\tau)$, we have that $[\mathcal{T}_{x_i}]_\alpha^\kappa = [\mathcal{T}'_{x_i}]_\alpha$ for all $x_i \in X$, where κ renames $\overline{x_i}$ to $\tilde{x}_i$.

Proof. By Lemma 2.16(a), for

$$[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\gamma) = [\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\tau) = [\mathcal{T}]_\alpha,$$

we can assume that the representative $\mathcal{T}$ is such that it has $a \in Cor(\mathcal{T})$. Let σ be the involution of $\mathcal{T}$. By applying twice Lemma 2.16(b), we get that

$$[\mathcal{T}_{x_i}]_\alpha^\kappa = [\mathcal{P}(\mathcal{T}, a, x_i)^{\rho_i}]_\alpha^\kappa = [\mathcal{T}'_{x_i}]_\alpha,$$

where ρ_i renames $\sigma(x_i)$ to $\overline{x_i}$, which proves the claim. ■

2.4.2 The equivalence relation $='$ on $\mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}$ Let $a \in \underline{\mathcal{C}}(X)$ and let $\sigma : x \mapsto t_x$ be an association of terms to variables from X, such that the command $\underline{a}\{\sigma\}$ is well-typed. The equivalence relation $='$ is the smallest equivalence relation generated by equalities

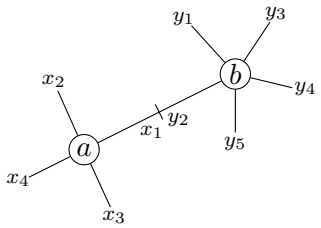
$$\boxed{\underline{a}\{\sigma\} =' c[\mu x.\underline{a}\{\sigma[x/x]\}/y]}$$

where $\sigma(x) = \mu y.c$ and $\sigma[x/x]$ denotes the same association as σ, except for x, to which it associates x itself. We, moreover, assume that $='$ is congruent with respect to `(MU3)`, `(MU4)` and substitution.

Remark 2.18. *Observe that, if $\underline{a}\{\sigma\} =' c[\mu x.\underline{a}\{\sigma[x/x]\}/y]$, and if $\underline{a}\{\sigma\}$ is a normal form, then this is also true for the command $c[\mu x.\underline{a}\{\sigma[x/x]\}/y]$. Therefore, $='$ is well-defined on $\mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}$.*

The intuition behind these equalities is again about equating commands that reflect two ways to build the same unrooted tree.

Example 2.19. Consider the unrooted tree $\mathcal{T} = \{a(x_1, x_2, x_3, x_4), b(y_1, y_2, y_3, y_4, y_5); \sigma\}$, where $\sigma = (x_1\, y_2)$, represented pictorially as



The commands equated by $='$ reflect the two possible ways to build $\mathcal{T}$ by means of simultaneous grafting: we could pick either the corolla a and graft to it the surrounding trees, or we can do the same by choosing first the corolla b. In the language of the μ-syntax, the two constructions are described by the left hand side and the right hand side of the equality

$$\underline{a}\{\mu y_2.\underline{b}\{y_1, y_2, y_3, y_4, y_5\}, a, b, c\} =' \underline{b}\{y_1, \mu x_1.\underline{a}\{x_1, x_2, x_3, x_4\}, y_3, y_4, y_5\},$$

respectively. Observe that, from the tree-wise perspective, $='$ enables us to "move between two adjacent corollas", i.e. it enables us to "move along a path in a tree". As we shall see, this feature will be crucial for the proof of injectivity of Theorem 2.9. □

The proof of the following lemma shows that $='$ is a "macro" derivable from $=_\mu$.

Lemma 2.20. *For any $c_1, c_2 \in \mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}$, if $c_1 =' c_2$, then $c_1 =_\mu c_2$.*

Proof. If $\underline{a}\{\sigma\} =' c[\mu x.\underline{a}\{\sigma[x/x]\}/y]$, then $\sigma(x) = \mu y.c$, which justifies the following sequence of equalities:

$$\underline{a}\{\sigma\} =_\mu \langle \mu x.\underline{a}\{\sigma[x/x]\} \,|\, \mu y.c\rangle =_\mu \langle \mu y.c \,|\, \mu x.\underline{a}\{\sigma[x/x]\}\rangle =_\mu c[\mu x.\underline{a}\{\sigma[x/x]\}/y].$$

■

The equality $='$ (denoted differently) appears in the work [11] of Lamarche, where it is called Adjunction and used in the context of the so-called *reversible terms*. Although the Adjunction rule materialises the same intuition about unrooted trees, there, unlike in our work, it is not derived from a more primitive notion of equality.

2.4.3 The proof of Theorem 2.9 The correspondence $\Phi_X : \mu\mathtt{Comm}_{\underline{\mathcal{C}}}(X)_{/=_\mu} \to \mathtt{T}_{\underline{\mathcal{C}}}(X)$ is canonically induced from the correspondence

$$\underline{\Phi} : \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \to \mathtt{T}_{\underline{\mathcal{C}}},$$

defined as the composition of the translation function $[[\_]] : \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \to \mathtt{cTerm}_{\underline{\mathcal{C}}}$ (see §2.3) with the interpretation function $[\_]_{\mathtt{T}_{\underline{\mathcal{C}}}} : \mathtt{cTerm}_{\underline{\mathcal{C}}} \to \mathtt{T}_{\underline{\mathcal{C}}}$ (that arises by considering the free cyclic operad defined in §1.2.3 through Definition 1.6). We show explicitly the definition of $\underline{\Phi}$ below, wherein the assignment of an α-equivalence class of unrooted trees to a term $t \in \mu\mathtt{Term}_{\underline{\mathcal{C}}}$ will be indexed by a fresh variable y involved in the corresponding interpretation $[[t]]_y$:

- $\underline{\Phi}_y(x) = [\{(x,y); id_{\{x,y\}}\}]_\alpha$,
- if, for each $x_i \in \{x_1, \ldots, x_n\}$, $\underline{\Phi}_{\overline{x}_i}(t_{x_i}) = [\mathcal{T}_{x_i}]_\alpha$, then

$$\underline{\Phi}(\underline{a}\{t_{x_1}, \ldots, t_{x_n}\}) = [\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\varphi),$$

  where $\varphi : x_i \mapsto ([\mathcal{T}_{x_i}]_\alpha, \overline{x}_i)$ (see (1.1)),
- $\underline{\Phi}_y(\mu x.c) = (\underline{\Phi}(c))^\kappa$, where κ renames x to y, and
- if $\underline{\Phi}_x(s) = [\mathcal{T}_s]_\alpha$ and $\underline{\Phi}_y(t) = [\mathcal{T}_t]_\alpha$, then $\underline{\Phi}(\langle s \,|\, t\rangle) = [\mathcal{T}_s]_\alpha \, {}_x\bullet_y \, [\mathcal{T}_t]_\alpha$.

By Theorem 2.8, $\underline{\Phi}$ is well-defined. We prove that it is both injective and surjective.

Surjectivity. Suppose given an α-equivalence class $[\mathcal{T}]_\alpha \in \mathtt{T}_{\underline{\mathcal{C}}}(X)$. If $\mathcal{T} = \{(x,y); id_{\{x,y\}}\}$, then it is easily seen that $\underline{\Phi}(\langle x \,|\, y\rangle) = [\{(x,y); id_{\{x,y\}}\}]_\alpha$.

Suppose now that $\mathcal{T}$ is an ordinary unrooted tree. We proceed by induction on the number k of corollas of $\mathcal{T}$. Let $a \in Cor(\mathcal{T})$ be such that $FV(a) = Y$, where $Y = \{y_1, \ldots, y_n\}$.

If a is the only corolla of $\mathcal{T}$, then $\underline{\Phi}(\underline{a}\{y_1, \ldots, y_n\}) = [\{a(y_1, \ldots, y_n); id_Y\}]_\alpha$.

Suppose that a is not the only corolla of $\mathcal{T}$, i.e. that $k \geq 2$, and let σ be the involution of $\mathcal{T}$. Let $I = \{i \in \{1, \ldots, n\} \,|\, y_i \in FV(a) \backslash X\}$ and $J = \{1, \ldots, n\} \backslash I$. By the induction hypothesis for each $\mathcal{P}(\mathcal{T}, a, y_i) = \mathcal{T}_{x_i}$ (recall from §2.4.1 that $\mathcal{P}$ is the "pruning" algorithm), for $i \in I$, we get a set

$$\{c_i \in \mu\mathtt{Comm}_{\underline{\mathcal{C}}} \,|\, i \in I \text{ and } \underline{\Phi}(c_i) = [\mathcal{T}_{y_i}]_\alpha\}.$$

We now set for all $i \in I$, $t_{y_i} = \mu\sigma(y_i).c_i$, and for all $j \in J$, $t_{y_j} = y_j$, and we claim that $\underline{\Phi}(a\{t_y \,|\, y \in Y\}) = [\mathcal{T}]_\alpha$. We have $\underline{\Phi}(a\{t_{y_k} \,|\, k \in \{1, \ldots, n\}\}) = a(\varphi)$, where

$$\varphi : y_k \mapsto \begin{cases} ([\mathcal{T}_{y_i}]_\alpha^{\kappa_i}, z_i) & \text{if } k = i \text{ for some } i \in I \\ ([\{(y_j, \underline{y_j}); id_{\{y_j, \underline{y_j}\}}\}]_\alpha, \underline{x_j}) & \text{if } k = j \text{ for some } j \in J \end{cases}$$

with $[\mathcal{T}_{y_i}]_\alpha^{\kappa_i} = \underline{\Phi}_{z_i}(\mu\sigma(y_i).c_i)$ being the class associated to the term $\mu\sigma(y_i).c_i$ with respect to the interpretation under the fresh variable z_i. Therefore, if $I = \{i_1, \ldots, i_{m_I}\}$ and $J = \{j_1, \ldots, j_{m_J}\}$, by the axiom (U1), $\underline{\Phi}(a\{t_{y_k} \,|\, k \in \{1, \ldots, n\}\})$ is equal to

$$(\cdots([\{a(y_1, \ldots, y_n); id_Y\}]_\alpha^{\kappa_{j_1}\kappa_{j_2}\cdots\kappa_{j_{m_J}}} \, {}_{y_{i_1}}\bullet_{z_{i_1}} [\mathcal{T}_{y_{i_1}}]_\alpha^{\kappa_{i_1}}) \cdots) \, {}_{y_{i_{m_I}}}\bullet_{z_{i_{m_I}}} [\mathcal{T}_{y_{i_{m_I}}}]_\alpha^{\kappa_{i_{m_I}}}$$

where each κ_{j_m}, $1 \leq m \leq m_J$ is the renaming of y_{j_k} to y_{j_k}, i.e. the identity on Y, and each κ_{i_m}, $1 \leq m \leq m_I$, is the renaming of z_{i_k} to $\sigma(x_{i_k})$. Finally, by (EQ), we have

$$\underline{\Phi}(a\{t_{y_k} \,|\, k \in \{1, \ldots, n\}\}) = (([\{a(y_1, \ldots, y_n); id_Y\}]_\alpha \, {}_{y_{i_1}}\bullet_{\sigma(y_{i_1})} [\mathcal{T}_{y_{i_1}}]_\alpha) \cdots) \, {}_{y_{i_{m_I}}}\bullet_{\sigma(y_{i_{m_I}})} [\mathcal{T}_{y_{i_{m_I}}}]_\alpha,$$

and, consequently, by Lemma 2.13, that $\underline{\Phi}(a\{t_{y_k} \,|\, k \in \{1, \ldots, n\}\}) = [\mathcal{T}]_\alpha$.

Injectivity. Notice that, in order to establish the injectivity of $\underline{\Phi}$, it suffices to prove it for

commands $c_1, c_2 \in \mu\mathtt{Comm}^{nf}_{\underline{c}}$. By Lemma 2.20, the injectivity for normal forms follows if we show that, if $\underline{\Phi}(c_1) = \underline{\Phi}(c_2)$, then $c_1 =' c_2$.

If c_1 and c_2 have the same head symbol, we proceed by induction on the structure of c_1 and c_2. Suppose that $c_1 = \underline{a}\{s_x | x \in X\} = \underline{a}\{\sigma\}$ and $c_2 = \underline{a}\{t_x | x \in X\} = \underline{a}\{\sigma'\}$. The assumption $\underline{\Phi}(c_1) = \underline{\Phi}(c_2)$ means that

$$[\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\varphi) = [\{a(x_1, \ldots, x_n); id_X\}]_\alpha(\psi),$$

where $\varphi : x \mapsto (\underline{\Phi}_{\tilde{x}}(s_x), \tilde{x})$ and $\psi : x \mapsto (\underline{\Phi}_{\overline{x}}(t_x), \overline{x})$, and consequently, by Lemma 2.17, that for all $x \in X$, $\underline{\Phi}_{\tilde{x}}(s_x)^\kappa = \underline{\Phi}_{\overline{x}}(t_x)$, where κ renames $\tilde{x}$ to $\overline{x}$. The claim holds by the reflexivity of $='$ if all s_x and t_x are variables: if $s_x = u$ and $t_x = v$, then

$$[\{(u, \overline{x}); id_{\{u,\overline{x}\}}\}]_\alpha = (\underline{\Phi}_{\tilde{x}}(u))^\kappa = \underline{\Phi}_{\overline{x}}(v) = [\{(v, \overline{x}); id_{\{v,\overline{x}\}}\}]_\alpha,$$

and, therefore, it must be the case that $u = v$.

Suppose, therefore, that $s_x = \mu u.c_x$ and $t_x = \mu v.c'_x$. We then have

$$[[c_x^{\tau_1}]] = [[c_x]]^{\tau_1} = [[s_x]]^\kappa_{\tilde{x}} = [[t_x]]_{\overline{x}} = [[c'_x]]^{\tau_2} = [[c_x'^{\tau_2}]],$$

and, consequently, that $\underline{\Phi}(c_x^{\tau_1}) = \underline{\Phi}(c_x'^{\tau_2})$, where τ_1 renames u to $\overline{x}$ and τ_2 renames v to $\overline{x}$. By the induction hypothesis we now have $c_x^{\tau_1} =' c_x'^{\tau_2}$ and, consequently, we get that

$$\begin{array}{rcccl} \underline{a}\{\sigma\} & =' & c_x[\mu x.\underline{a}\{\sigma[x/x]\}/u] & = & c_x^{\tau_1}[\mu x.\underline{a}\{\sigma[x/x]\}/\overline{x}] \\ & =' & c_x'^{\tau_2}[\mu x.\underline{a}\{\sigma[x/x]\}/\overline{x}] & = & c'_x[\mu x.\underline{a}\{\sigma[x/x]\}/v] \quad =' \quad \underline{a}\{\sigma'\}. \end{array}$$

Suppose now that c_1 and c_2 do not have the same head symbol, i.e. that $c_1 = \underline{a}\{s_x | x \in X\} = \underline{a}\{\sigma_1\}$ and $c_2 = \underline{b}\{t_y | y \in Y\} = \underline{b}\{\sigma_2\}$, and let $\underline{\Phi}(c_1) = [\mathcal{T}_{c_1}]_\alpha$ and $\underline{\Phi}(c_2) = [\mathcal{T}_{c_2}]_\alpha$. Let $\mathcal{T}$ be a representative of $[\mathcal{T}_{c_1}]_\alpha = [\mathcal{T}_{c_2}]_\alpha$. Observe that two groups of renamings feature in the transitions from c_1 and c_2 to $\mathcal{T}$: the first one contains the renamings specified by the definitions of the simultaneous compositions $\underline{\Phi}(c_1)$ and $\underline{\Phi}(c_2)$, and the second one contains the renamings given by the α-equivalence of $\mathcal{T}_{c_1}$ and $\mathcal{T}$, and $\mathcal{T}_{c_2}$ and $\mathcal{T}$. However, by (MU4), all the renamings of parameters and variables of c_1 and c_2 made in defining $\mathcal{T}$ can be also performed on c_1 and c_2 themselves, leading to commands $c'_1 =_\mu c_1$ and $c'_2 =_\mu c_2$, such that $\underline{\Phi}(c'_1) = \underline{\Phi}(c'_2) = [\mathcal{T}]_\alpha$ and such that $\mathcal{T}$ shares the same sets of parameters and variables with both c'_1 and c'_2. Hence, we can assume that $\mathcal{T}$ already shares the same sets of parameters and variables with c_1 and c_2. This, in particular, means that $a, b \in Cor(\mathcal{T})$.

Let $x \in X$ be such that $b \in Cor(\mathcal{P}(\mathcal{T}, a, x))$. By the construction of $\mathcal{T}$, the parameter $\underline{b}$ appears in $\sigma_1(x) = \mu u.c$. We define the *distance between* $\underline{a}$ *and* $\underline{b}$ *in* c_1 as the natural number $d_{c_1}(a, b)$ determined as follows.

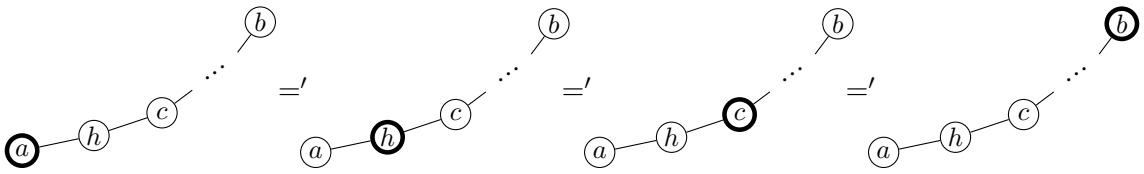
- If $\underline{b}$ is the head symbol of c, then $d_{c_1}(a, b) = 1$.
- If $\underline{h}$ is the head symbol of c, $h \neq b$, then $d_{c_1}(a, b) = d_c(h, b) + 1$.

We prove that $c_1 =' c_2$ by induction on $d_{c_1}(a, b)$. If $d_{c_1}(a, b) = 1$, then, for some $y \in FV(b)$, we have that $\sigma_1(x) = \mu y.\underline{b}\{\sigma_2[y/y]\}$. Therefore,

$$\begin{array}{rcl} \underline{a}\{\sigma_1\} & =' & \underline{b}\{\sigma_2[y/y]\}[\mu x.\underline{a}\{\sigma_1[x/x]\}/y] \\ & = & \underline{b}\{\sigma_2[\mu x.\underline{a}\{\sigma_1[x/x]\}/y]\} \\ & =' & \underline{b}\{\sigma_2\}. \end{array}$$

If $d_{c_1}(a, b) \geq 2$, then, since $d_{c_1}(a, h) = 1$ (where h is as above), we have that $c_1 =' c[\mu x.\underline{a}\{\sigma_1[x/x]\}/u]$. On the other hand, by the induction hypothesis for $d_c(h, b) < n$, we have

that $c_2 =' c[\mu x.\underline{a}\{\sigma_1[x/x]\}/u]$, and the conclusion follows by the transitivity of $='$. The iterative application of the equality $='$, implicit in the induction argument, which reduces the distance between $\underline{a}$ and $\underline{b}$, can be illustrated as follows



This completes the proof of Theorem 2.9.

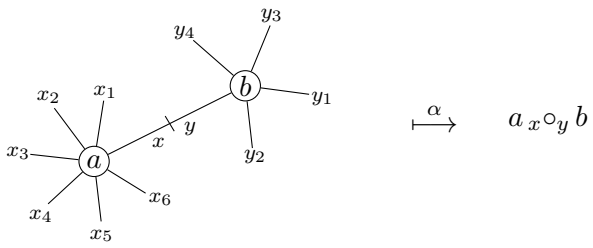
Note that we have in fact *two* bijections: $\mu\mathtt{Comm}_{\underline{\mathcal{C}}}(X)/_{=_\mu} \simeq \mu\mathtt{Comm}^{nf}_{\underline{\mathcal{C}}}(X)/_{='} \simeq \mathtt{T}_{\underline{\mathcal{C}}}(X)$, the first one being induced via normal forms of $\rightsquigarrow$: we have that $nf(c_1) =' nf(c_2)$ implies $c_1 =_\mu c_2$, and conversely, if $c_1 =_\mu c_2$, then $\Phi(nf(c_1) = \Phi(nf(c_2))$ implies $nf(c_1) =' nf(c_2)$.

3. The equivalence established

We finally show how the μ-syntax, together with the syntactic formalism of unrooted trees suited to it, allows us to prove Theorem 1.22 in a genuinely constructive, and, thereby, algorithmic way. In both directions, the proof we give elaborates calculations to be made at each step of the transition, relying on the constructions made in the proof of Theorem 2.9. Let $\underline{\mathcal{C}} : \mathbf{Bij}^{op} \to \mathbf{Set}$ be a functor.

Suppose that $(\underline{\mathcal{C}}, \delta)$ is an $\mathcal{M}$-algebra. We build a cyclic operad, as in Definition 1.1, as follows.

We distinguish the identities, by setting $id_{x,y} = \delta_{\{x,y\}}([\{(x,y); id_{\{x,y\}}\}]_\alpha)$. The definition of the partial composition operation ${}_x\circ_y$ is derived by considering restrictions of δ to unrooted trees with two corollas:



Formally, for $a \in \underline{\mathcal{C}}(X)$ and $b \in \underline{\mathcal{C}}(Y)$, the partial composition operation

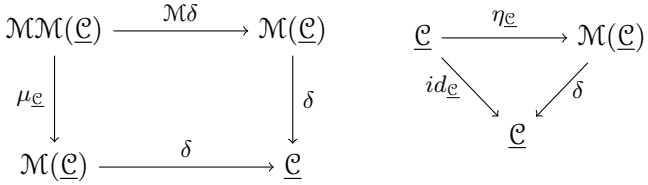
$$ {}_x\circ_y : \underline{\mathcal{C}}(X) \times \underline{\mathcal{C}}(Y) \to \underline{\mathcal{C}}(X\backslash\{x\} \cup Y\backslash\{y\}) $$

is characterised via $\delta_{X\backslash\{x\}\cup Y\backslash\{y\}} : \mathcal{M}(\underline{\mathcal{C}})(X\backslash\{x\} \cup Y\backslash\{y\}) \to \underline{\mathcal{C}}(X\backslash\{x\} \cup Y\backslash\{y\})$ as

$$ a\ {}_x\circ_y\ b = \delta_{X\backslash\{x\}\cup Y\backslash\{y\}}([\{a(x,\dots); id_X\}]_\alpha\ {}_x\bullet_y\ [\{b(y,\dots); id_Y\}]_\alpha), $$

where ${}_x\bullet_y$ is the operation on (classes of) unrooted trees defined in §1.2.3.

As a structure morphism of $\mathcal{M}$-algebra $(\underline{\mathcal{C}}, \delta)$, δ satisfies the coherence conditions given by commutations of the following two diagrams:



called the multiplication and the unit law for δ, which allows us to verify the axioms from Definition 1.1 as follows.

For the proof of (A1), let a and b be as above, let $c \in \underline{\mathcal{C}}(Z)$, $z \in Z$ and $u \in Y$. Suppose that a, b and c are all different from identity and that X, Y and Z are mutually disjoint (only to avoid the renaming technicalities). We will chase the multiplication diagram above two times, starting with two-level unrooted trees

$$\mathcal{T}_1 = \{[\{a(x,\dots), b(y,u,\dots); \sigma_1'\}]_\alpha, [\{c(z,\dots); id_Z\}]_\alpha; \sigma_1\}$$

and

$$\mathcal{T}_2 = \{[\{a(x,\dots); id_X\}]_\alpha, [\{b(y,u,\dots), c(z,\dots); \sigma_2'\}]_\alpha; \sigma_2\},$$

where $\sigma_1' = (x\ y)$, $\sigma_1 = (u\ z)$, $\sigma_2' = (u\ z)$ and $\sigma_2 = (x\ y)$. If we start with $\mathcal{T}_1$, then, by chasing the diagram to the right-down, the action of $\mathcal{M}\delta$ corresponds to the action of δ on $[\{a(x,\dots), b(y,u,\dots); \sigma_1'\}]_\alpha$ and $[\{c(z,\dots); id_Z\}]_\alpha$ separately. Followed by the action of δ again, we get the following sequence

$$[\mathcal{T}_1]_\alpha \overset{\mathcal{M}\delta}{\longmapsto} [\{(a\ {}_x\!\circ_y\ b)(u,\dots), c(z,\dots); \sigma\}]_\alpha \overset{\delta}{\longmapsto} (a\ {}_x\!\circ_y\ b)\ {}_u\!\circ_z\ c.$$

In the other direction, the action of the monad multiplication flattens $\mathcal{T}_1$, the resulting tree already being in normal form. Followed by the action of δ, we obtain the sequence:

$$[\mathcal{T}_1]_\alpha \overset{\mu_{\underline{\mathcal{C}}}}{\longmapsto} [\{a(x,\dots), b(y,u,\dots), c(z,\dots); \underline{\sigma}\}]_\alpha \overset{\delta}{\longmapsto} \delta([\{a(x,\dots), b(y,u,\dots), c(z,\dots); \underline{\sigma}\}]_\alpha).$$

Hence,

$$(a\ {}_x\!\circ_y\ b)\ {}_u\!\circ_z\ c = \delta([\{a(x,\dots), b(y,u,\dots), c(z,\dots); \underline{\sigma}\}]_\alpha).$$

The claim follows, since the diagram chasing with respect to $\mathcal{T}_2$ results in

$$a\ {}_x\!\circ_y\ (b\ {}_u\!\circ_z\ c) = \delta([\{a(x,\dots), b(y,u,\dots), c(z,\dots); \underline{\sigma}\}]_\alpha).$$

The axiom (CO) follows directly by the commutativity of ${}_x\!\bullet_y$.

The axiom (EQ) holds by the equivariance of ${}_x\!\bullet_y$ and the naturality of η and δ. For σ_1, σ_2 and σ as in (EQ), and denoting $Z = X'\backslash\{\sigma_1^{-1}(x)\} \cup Y'\backslash\{\sigma_2^{-1}(y)\}$, we have

$$\begin{aligned}
a^{\sigma_1}\ {}_{\sigma_1^{-1}(x)}\!\circ_{\sigma_2^{-1}(y)}\ b^{\sigma_2} &= \delta_Z(\eta_{\underline{\mathcal{C}}X'}(a^{\sigma_1})\ {}_{\sigma_1^{-1}(x)}\!\bullet_{\sigma_2^{-1}(y)}\ \eta_{\underline{\mathcal{C}}Y'}(b^{\sigma_2})) \\
&= \delta_Z(\eta_X(a)^{\sigma_1}\ {}_{\sigma_1^{-1}(x)}\!\bullet_{\sigma_2^{-1}(y)}\ \eta_Y(b)^{\sigma_2}) \\
&= \delta_Z((\eta_X(a)\ {}_x\!\bullet_y\ \eta_Y(b))^{\sigma}) \\
&= \delta_Z(\eta_X(a)\ {}_x\!\bullet_y\ \eta_Y(b))^{\sigma} \\
&= (a\ {}_x\!\circ_y\ b)^{\sigma}.
\end{aligned}$$

To prove (U1), we chase the multiplication law of δ, starting from the two-level tree

$$\mathcal{T} = \{[\{a(x,\dots); id_X\}]_\alpha, [\{(y,z); id_{y,z}\}]_\alpha; \sigma\},$$

where $\sigma = (x\ y)$. By going to the right-down, we get the sequence

$$\mathcal{T} \overset{\mathcal{M}\delta}{\longmapsto} [\{a(x,\dots), id_{y,z}(y,z); \sigma\}]_\alpha \overset{\delta}{\longmapsto} a\ {}_x\!\circ_y\ id_{y,z},$$

and, in the other direction, denoting $X' = X\backslash\{x\} \cup \{z\}$, we get

$$\mathcal{T} \overset{\mu_{\underline{\mathcal{C}}}}{\longmapsto} [\{a^\kappa(z,\dots); id_{X'}\}]_\alpha \overset{\delta}{\longmapsto} \delta([\{a^\kappa(z,\dots); id_{X'}\}]_\alpha),$$

where κ t renames x to z. The equality $a\,{}_x{\circ}_y\, id_{y,z} = a^\kappa$ follows, since, by the definition of the unit of the monad $\mathcal{M}$, and by the unit law of δ, for the result of the second sequence, we have

$$\delta([\{a^\kappa(z,\dots); id_{X'}\}]_\alpha) = \delta(\eta_{\underline{\mathcal{C}}X'}(a^\kappa)) = a^\kappa.$$

The axiom (U3) follows simply by the naturality of δ.

In the other direction, we define $\delta : \mathcal{M}(\underline{\mathcal{C}}) \to \underline{\mathcal{C}}$ as the map induced by the interpretation of the μ-syntax in the cyclic operad $\underline{\mathcal{C}}$, i.e. by the composition of $[[\_]] : \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \to \mathtt{cTerm}_{\underline{\mathcal{C}}}$ and $[\_]_{\mathcal{C}} : \mathtt{cTerm}_{\underline{\mathcal{C}}} \to \underline{\mathcal{C}}$. Therefore, with Φ being defined as in the proof of Theorem 2.9, we set

$$\delta(\mathcal{T}) = [\,[[c]]\,]_{\underline{\mathcal{C}}}, \quad \text{where } c \text{ is any command of } \mu\mathtt{Exp}_{\underline{\mathcal{C}}} \text{ such that } \underline{\Phi}(c) = [\mathcal{T}]_\alpha.$$

Note that this definition is valid by Theorem 2.9. We verify that δ satisfies the equations of an $\mathcal{M}$-algebra on simple examples. The general case follows naturally. Let

$$\mathcal{T} = \{[\{a(x_1,\dots,x_n), b(y_1,\dots,y_m); \sigma_1\}]_\alpha, [\{d(z_1,\dots,z_p); id_Z\}]_\alpha; \sigma\}$$

be a two-level unrooted tree such that $\sigma_1(x_i) = y_j$, and $\sigma(y_k) = z_l$, and suppose, say, that

$$\underline{\Phi}(\underline{a}\{t_1,\dots,t_n\}) = [\{a(x_1,\dots,x_n), b(y_1,\dots,y_m); \sigma_1\}]_\alpha$$

and

$$\underline{\Phi}(\underline{d}\{s_1,\dots,s_p\}) = [\{d(z_1,\dots,z_p); id_Z\}]_\alpha.$$

By chasing the multiplication diagram to the right-down, the action of $\mathcal{M}\delta$ provides the interpretations of the commands that correspond to each of the corollas of $\mathcal{T}$. Thus, setting $[[\underline{a}\{t_1,\dots,t_n\}]] = a(\varphi)$ and $[[\underline{d}\{s_1,\dots,s_p\}]] = d(\tau)$, we get that

$$\mathcal{M}\delta([\mathcal{T}]_\alpha) = \{[a(\varphi)]_{\underline{\mathcal{C}}}(x_1,\dots,x_{i-1},x_{i+1},\dots,x_n,y_1,\dots,y_{j-1},y_{j+1},\dots,y_m), [d(\tau)]_{\underline{\mathcal{C}}}(z_1,\dots,z_p); \sigma\}.$$

If now $\underline{\Phi}(\underline{[a(\varphi)]_{\mathcal{C}}}\{k_1,\dots,k_{n+m-2}\}) = \mathcal{M}\delta([\mathcal{T}]_\alpha)$, then, by setting $[[\underline{[a(\varphi)]_{\mathcal{C}}}\{k_1,\dots,k_{n+m-2}\}]] = [a(\varphi)]_{\underline{\mathcal{C}}}(\psi)$, we get

$$\delta(\mathcal{M}\delta([\mathcal{T}]_\alpha)) = [a(\varphi)(\psi)]_{\underline{\mathcal{C}}}.$$

By chasing the multiplicaiton diagram to the down-right, we first get

$$\mu_{\underline{\mathcal{C}}}([\mathcal{T}]_\alpha) = [\{a(x_1,\dots,x_n), b(y_1,\dots,y_m), d(z_1,\dots,z_p); \underline{\sigma}\}]_\alpha$$

We shall construct a command c, such that $\underline{\Phi}(c) = \mu_{\underline{\mathcal{C}}}(\mathcal{T})$, in the way guided by the choices we made in chasing the diagram to the right-down. More precisely, in that direction, a was the corolla of $\{a(x_1,\dots,x_n), b(y_1,\dots,y_m); \sigma_1\}$ chosen in constructing the corresponding command, and d was the one for $\{d(z_1,\dots,z_p); id_Z\}$, and then, in the next step, $[a(\varphi)]_{\underline{\mathcal{C}}}$ was the chosen corolla of $\mathcal{M}\delta([\mathcal{T}]_\alpha)$. Therefore, we set $c = \underline{a}\{\sigma\}$, where

$$\sigma(x_i) = \mu y_j.\underline{b}\{y_1,\dots,y_{k-1},\mu z_l.\underline{d}\{z_1,\dots,z_p\},y_{k+1},\dots,y_m\}.$$

Thus, setting $[[\underline{a}\{\sigma\}]] = a(\xi)$, we get $\delta(\mu_{\underline{\mathcal{C}}}(\mathcal{T})) = [a(\xi)]_{\underline{\mathcal{C}}}$ as a result of chasing the diagram to the down-right. The equality $a(\varphi)(\psi) = a(\xi)$ follows by Lemma 1.5.(b).

As for the unit diagram, if $a \in \underline{\mathcal{C}}(X)$, where $X = \{x_1,\dots,x_n\}$, then $\eta_{\underline{\mathcal{C}}X}(a) = \{a(x_1,\dots,x_n); id_X\}$, and, since $[\{a(x_1,\dots,x_n); id_X\}]_\alpha = \underline{\Phi}(\underline{a}\{x_1,\dots,x_n\})$, we have that

$$\delta_X(\eta_{\underline{\mathcal{C}}X}(f)) = [\,[[\underline{a}\{x_1,\dots,x_n\}]]\,]_{\underline{\mathcal{C}}} = a.$$

This completes the proof.

Acknowledgements

We are very grateful to the referees for their detailed remarks and for the references to related works that they brought to our attention.

References

[1] F. Baader, T. Nipkow, *Term Rewriting and All That*, Cambridge University Press, 1999.

[2] K.Costello, *The A-infinity operad and the moduli space or curves*, arXiv:math/0402015, 2004.

[3] P. -L. Curien, H. Herbelin, *The duality of computation*, ACM SIGPLAN Notices, Volume 35 Issue 9, 233-243, September 2000.

[4] M. P. Fiore, M. Devesas Campos, *The Algebra of Directed Acyclic Graphs*, In Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky, Volume 7860 of Lecture Notes in Computer Science, 2013.

[5] E. Getzler, *Operads revisited*, Algebra, arithmetics, and geometry: in honor of Yu. I. Manin, Vol. I, volume 269 of Progr. Math. p. 675-698. Birkhäuser Boston Inc., Boston MA, 2009.

[6] E. Getzler, M. Kapranov, *Cyclic operads and cyclic homology*, Geom., Top., and Phys. for Raoul Bott, International Press, Cambridge, MA, 167-201, 1995.

[7] A. Joyal, Une théorie combinatoire des séries formelles, *Advances in Mathematics*, **42** 1-82, 1981.

[8] A. Joyal, J. Kock, *Feynman Graphs, and Nerve Theorem for Compact Symmetric Multicategories* (Extended Abstract), Electronic Notes in Theoretical Computer Science 270 (2) 105-113, 2011.

[9] R. M. Kaufmann, B. C. Ward, *Feynman categories*, arXiv:1312.1269v3, 2017.

[10] M. Kontsevich, Y. Manin. *Gromov-Witten Classes, Quantum Cohomology, and Enumerative Geometry*, Comm. Math. Phys., 164:525-562, February 1994. Preprint, hep-th/9402147.

[11] F. Lamarche, *On the algebra of structural contexts*, Mathematical Structures in Computer Science, Cambridge University Press, 51 p, 2003.

[12] Y. I. Manin, *Frobenius manifolds, quantum cohomology, and moduli spaces*, volume 47 of AMS Colloquium Publications, American mathematical Society, Providence, RI, 1999.

[13] M. Markl, *Models for operads*, Comm. Algebra, 24(4):1471–1500, 1996.

[14] M. Markl, *Modular envelopes, OSFT and nonsymmetric (non-Σ) modular operads*, J. Noncommut. Geom. 10, 775-809, 2016.

[15] M. Markl, *Operads and PROPs*, Elsevier, Handbook for Algebra, Vol. 5, 87-140, 2008.

[16] M. Markl, S. Schnider, J. Stasheff, Operads in Algebra, Topology and Physics, American Mathematical Society, Providence, 2002.

[17] J. P. May, *The geometry of iterated loop spaces*, volume 271 of Lectures Notes in Mathematics. Springer-Verlag, Berlin, 1972.

[18] J. Obradović, *Monoid-like definitions of cyclic operads*, Theory and Applications of Categories, Vol. 32, No. 12, pp. 396-436, 2017.

[19] B. C. Pierce, *Types and Programming Languages*, The MIT Press, 2002.