C-SYSTEM OF A MODULE OVER A Jf-RELATIVE MONAD

VLADIMIR VOEVODSKY

Abstract. Let $\mathbb{F}$ be the category with the set of objects $\mathbf{N}$ and morphisms given by the functions between the standard finite sets of the corresponding cardinalities. Let $Jf : \mathbb{F} \to Sets$ be the obvious functor from this category to the category of sets. In this paper we construct, for any Jf-relative monad $\mathbf{RR}$ and any left $\mathbf{RR}$-module $\mathbf{LM}$, a C-system $C(\mathbf{RR}, \mathbf{LM})$ and explicitly compute the action of the four B-system operations on its B-sets.

In the introduction we explain in detail the relevance of this result to the construction of the term C-systems of type theories.

Contents

1. Introduction

The first few steps in all approaches to the set-theoretic semantics of dependent type theories remain insufficiently understood. The constructions which have been worked out in detail in the case of a few particular type systems by dedicated authors are being extended to the wide variety of type systems under consideration today by analogy. This is not acceptable in mathematics. Instead we should be able to obtain the required results for new type systems by *specialization* of general theorems and constructions formulated for abstract objects the instances of which combine together to produce a given type system.

An approach that follows this general philosophy was outlined in [37]. In this approach the connection between the type theories, which belong to the concrete world of logic and programming, and abstract mathematical concepts such as sets or homotopy types is constructed through the intermediary of C-systems.

C-systems were introduced in [12] (see also [13]) under the name "contextual categories". A modified axiomatics of C-systems and the construction of new C-systems as sub-objects and regular quotients of the existing ones in a way convenient for use in type-theoretic applications are considered in [45]. A C-system equipped with additional operations corresponding to the inference rules of a type theory is called a model or a C-system model of these rules or of this type theory. There are other classes of objects on which one can define operations corresponding to inference rules of type theories most importantly categories with families or CwFs. They lead to other classes of models.

In the approach of [37], in order to provide an interpretation for a type theory one first constructs two C-systems. One C-system, which we will call the proximate or term C-system of a type theory, is constructed from formulas of the type theory using the main construction of the present paper. The second C-system is constructed from the category of abstract mathematical objects using the results of [39]. Both C-systems are then equipped with additional operations corresponding to the inference rules of the type theory making them into models of type theory. The model whose underlying C-system is the term C-system is called the term model.

A crucial component of this approach is the expected result that for a particular class of inference rules the term model is an initial object in the category of models. This is known as the Initiallity Conjecture. In the case of the pure Calculus of Constructions with a "decorated" application operation this conjecture was proved in 1988 by Thomas Streicher [34]. The problem of finding an appropriate formulation of the general version of the conjecture and of proving this general version will be the subject of future work.

For such inference rules, then, there is a unique homomorphism from the term C-system to the abstract C-system that is compatible with the corresponding systems of operations. Such homomorphisms are called representations or interpretations of the type theory. More generally, any functor from the category underlying the term C-system of the type theory to another category may be called a representation of the type theory in that category. Since objects and morphisms of term models are built from formulas of the type theory and objects and morphisms of abstract C-systems are built from mathematical objects such as sets or homotopy types and the corresponding functions, such representations provide a mathematical meaning to formulas of type theory.

The existence of these homomorphisms in the particular case of the "standard univalent models" of Martin-Löf type theories and of the Calculus of Inductive Constructions (CIC) provides the only known justification for the use of the proof assistants such as Coq for the formalization of mathematics in the univalent style [40], [47].

Only if we know that the initiallity result holds for a given type theory can we claim that a model defines a representation. A similar problem also arises in the predicate logic but there, since one considers only one fixed system of syntax and inference rules, it can and had been solved once without the development of a general theory. The term models for a class of type theories can be obtained by considering slices of the term model of the type theory called Logical Framework (LF), but unfortunately it is unclear how to extend this approach to type theories that have more substitutional (definitional) equalities than LF itself.

A construction of a model for the version of the Martin-Löf type theory that is used in the UniMath library [47], [40] is sketched in [24]. At the time when that paper was written it was unfortunately assumed that a proof of the initiallity result can be found in the existing body of work on type theory which is reflected in [24, Theorem 1.2.9] (cf. also [24, Example 1.2.3] that claims as obvious everything that is done in tens of different papers by computer scientists, the present paper and in [45]). Since then it became clear that this is not the case and that a mathematical theory leading to the initiallity theorem and providing a proof of such a theorem is lacking and needs to be developed.

As the criteria for what constitutes an acceptable proof were becoming more clear as a result of continuing work on formalization, it also became clear that more detailed and general proofs need to be given to many of the theorems of [24] that are related to the model itself. For the two of the several main groups of inference rules of current type theories it is done in [44], [46] and [42]. Other groups of inference rules will be considered in further papers of that series.

That work concerned the construction of the second, “abstract”, C-system model used in the construction of a representation.

The work done in this paper provides the first step in the construction of the “concrete” term C-system model. The result of our construction is equivalent to the results of constructions sketched by earlier authors [23]. The main innovation, other than the first careful mathematical proofs of all the required assertions, is the observations that one can take *all* raw judgements as the source for the construction and build from them a C-system. The term C-system of a type theory with a given raw syntax is then a sub-quotient of the raw syntax C-system. The raw syntax C-system can be either defined directly in a way that allows for a straightforward rigorous verification of all the axioms, see Remark **??**, or understood from the perspective of the abstract mathematical theory of C-systems as a particular case of a more general construction of the presheaf extensions of C-systems. In this paper we follow the second path that also allows us to connect our construction to the main constructions of [43] and [41].

The description of a type theory in a modern paper is usually given in terms of five kinds of “sequents” originally introduced by Per Martin-Löf in [28, p.161][1] . If we consider the type theory as a language then sequents are complete sentences in this language - the smallest units that have semantical meaning when an interpretation

[1]This paper is highly recommended. It is a foundational one for many ideas of type theory and for the modern approach to constructive mathematics in general.

is chosen. Below we use the word "sentence" instead of the word "sequent" to avoid confusion with the meaning of the word "sequent" in logic. The five kinds of sentences considered by Martin-Löf are sequences of expressions of the following forms

(1) [**2017.02.06.eq1**]$x_0 : T_0, \ldots, x_{n-1} : T_{n-1} \rhd ok$

(2) [**2017.02.06.eq2**]$x_0 : T_0, \ldots, x_{n-1} : T_{n-1} \rhd T\ type$

(3) [**2017.02.06.eq3**]$x_0 : T_0, \ldots, x_{n-1} : T_{n-1} \rhd t : T$

(4) [**2017.02.06.eq4**]$x_0 : T_0, \ldots, x_{n-1} : T_{n-1} \rhd T \equiv T'$

(5) [**2017.02.06.eq5**]$x_0 : T_0, \ldots, x_{n-1} : T_{n-1} \rhd t \equiv t' : T$

Here $x_0, \ldots, x_{n-1}$ are names of variables, T_i is an expression with free variables from the set $\{x_0, \ldots, x_{i-1}\}$, and T and t are expressions with free variables from the set $\{x_0, \ldots, x_{n-1}\}$. If one wants to emphasize that a variable x may appear as a free variable in the expression T one writes $T(x)$, but *in most cases the set of allowed free variables in an expression should be inferred from its position in the sentence.*

In many modern papers on type theory the symbol $\vdash$ is used where we use the triangle symbol $\rhd$. We made this choice because the meaning of the former symbol in type theory may conflict with its meaning in logic.

The part of a sentence to the left of $\rhd$ is called the context and the part to the right of this symbol is called the judgement. When the names of variables and the expressions of the context are not important or can be inferred from some data or conventions, it is customary to denote the context by a capital Greek letter such as Γ or Δ.

There are some equivalent versions of the Martin-Löf's approach. For example, Martin Hofmann, in [23], considers six kinds of judgements adding the equality of contexts, $\rhd \Gamma \equiv \Delta$, as a separate kind.

In any approach sentences are sequences of expressions with some restrictions on allowed free variables. Therefore

The first step towards mathematical theory of type theories is to find a way to view "expressions" as mathematical objects.

In practice, what are "expressions" from which the sentences of a type theory are formed is most likely to be specified in detail when this type theory is used as a basis of a computer proof assistant. Depending on the programming language on which the proof assistant is written and on the personal tastes of the developers "expressions" will be represented as elements of different datatypes. They may be represented as actual strings of characters or as trees with additional labels at nodes and edges or as something else entirely. While each of these representations can be given a precise mathematical form it would be clearly wrong to make the mathematical theory of type theories dependent on which of the representations is chosen. Therefore, we need a concept of an abstract expression, or, as we will see below, two concepts one for abstract element expressions and one for abstract type expressions.

This problem has been addressed by many authors, first in the context of algebraic expressions and later in the context of expressions with binders, that is, expressions that may contain bound variables. As far as we know, the first mathematical abstraction in the case of expressions with binders was described by Fiore, Plotkin and Turi in [16]. Later a different and more convenient for mathematicians abstraction was described by Hirschowitz and Maggesi in a series of papers including [21]. The two approaches were shown to be equivalent in [7], [8] using the concept of a well behaved functor. The proof of equivalence in [7], [8] was based on the important observation that the monoids of [16] are particular cases of *relative monads*.

These results would have closed the issue and we would have probably used in this paper the Hirschowitz and Maggesi concept of an abstract expression based on the concept of a monad if not for the fact that the proof of the equivalence uses the axiom of choice.

This may be the place to say a few words about the intended meta-theories of the present paper. By an intended meta-theory of a paper we mean a foundation of mathematics in which its results can be stated and proved. It has become customary not to specify intended meta-theories for papers in pure mathematics due to the tacit assumption that this meta-theory is the Zermelo-Fraenkel set theory with Axiom of Choice (ZFC). Here ZFC, or any other foundation that we may mention, is considered as a body of mathematical knowledge that includes both the underlying formal theory and the dictionary that is needed to translate informal mathematics into the statements about this formal system. In the case of the ZFC, the formal deduction system is classical predicate logic with a distinguished theory and statements of informal mathematics are translated into the provability statement applied to various formulas of this theory.

We intend two meta-theories for the present paper. The first one is the usual ZFC with a Grothendieck universe U. The second one is a univalent foundation called UniMath. We consider ZFC to be the primary meta-theory and UniMath a secondary one. That is, we take care to adapt our definitions and proofs precisely for the ZFC while accepting that UniMath formalization will require some small degree of modification.

Having UniMath as a secondary meta-theory imposes a number of strong restrictions on the features of the ZFC that we can and can not use. Most notably, we can not use the law of excluded middle and we can not use the axiom of choice.

Let us go back to the concept of an abstract expression. Let us consider the case of algebraic expressions first. Modern mathematical theory of algebraic expressions has been developed at least as far back as the 1963 Ph.D. thesis of Bill Lawvere [26]. However, our interest in having the theory extended later to operations that bound variables, such as the $\forall$ quantifier, together with the need to have our approach adapted for a constructive meta-theory bring forward aspects of this theory that are easy to miss otherwise.

Systems of algebraic expressions are specified by algebraic signatures - pairs, consisting of a set Op, called the set of operations, and a function $Ar : Op \to \mathbf{N}$, called arity. Given a signature Sig and a set V such that $V \cap Op = \emptyset$ one can define, for any $X \subset V$, the set $Exp(X)$ of expressions relative to Sig with (free) variables from

X. Note that when we write $Exp(X)$ we assume that the signature Sig and the set V have been fixed.

There are many different families of sets $Exp(X)$ whose elements may be called expressions. For example, one can use a subset of the set of sequences (lists) of elements of $Op \cup X$. Alternatively, one can use some axiomatization of planar rooted trees with labels from $Op \cup X$ on the nodes. Translating from a representation of the first kind into a representation of the second is done, in the concrete word of computer programs, by programs called parsers and there is a beautiful mathematical theory behind it that many of us are robbed of the joy of learning.

Let U be a universe that contains Op, V, and $Exp(X)$ for all $X \subset V$. Let $Sets(2^V)$ be the category of sets whose set of objects is 2^V, $Sets(U)$ the category of sets whose set of objects is U, and $J_V : Sets(2^V) \to Sets(U)$ the inclusion functor.

We may consider Exp as a function $2^V \to U$. In the chosen representation of expressions one can construct the substitution operation that for any $X, Y \subset V$ and $f : X \to Exp(Y)$, defines a function $f^* : Exp(X) \to Exp(Y)$. In addition, for any $X \subset V$ one can define a function $\eta_X : X \to Exp(X)$. The triple $(Exp, \eta, -^*)$ satisfies the conditions of Definition 2.5 making it into a J_V-relative monad.

This is how relative monads appear in the theory of expressions with variables.

Next, following [16], we let $\mathbb{F}$ denote the category with the set of objects $\mathbf{N}$ and the set of morphisms

$$[\mathbf{2017.02.24.eq1}]Mor(\mathbb{F}) = \cup_{m,n} Fun(stn(m), stn(n)) \tag{6}$$

where $stn(m) = \{i \in \mathbf{N} \mid i < m\}$ is our choice for the standard set with m elements and where for two sets X and Y, $Fun(X, Y)$ is the set of functions from X to Y defined as in [11, p.81] such that each function has a well defined domain and codomain. Then the union in (6) is disjoint and

$$Mor_{\mathbb{F}}(m, n) = Fun(stn(m), stn(n))$$

It is sometimes convenient to distinguish natural numbers used as objects of $\mathbb{F}$ from their other uses. For this purpose we will write $\underline{n}$ for n used as an object of $\mathbb{F}$.

Let $Jf : \mathbb{F} \to Sets(U)$ be the functor given by $\underline{n} \mapsto stn(n)$ on objects and by the inclusion of the sets of morphisms.

Assume that $\mathbf{N} \cap Op = \emptyset$. Then our previous construction applies to $V = \mathbf{N}$. Consider the functor $\Phi : \mathbb{F} \to Sets(2^{\mathbf{N}})$ that takes $\underline{n}$ to $stn(n)$ and that is again the inclusion of the sets of morphisms.

Relative monads on a functor $\mathcal{C}_1 \to \mathcal{C}_2$ can be precomposed with functors $\mathcal{C}_0 \to \mathcal{C}_1$. See Construction 2.15. Precomposing the monad of expressions $\mathbf{Exp}^{\mathbf{N}}$ with Φ and observing that $\Phi \circ J_{\mathbf{N}} = Jf$ we obtain, for any algebraic signature Sig such that $Op \cap \mathbf{N} = \emptyset$, a Jf-relative monad that we denote by $\mathbf{Exp}_{Sig}$.

This is how the Jf-relative monads appear in the theory of algebraic expressions.

Note that that up to this point our constructions were completely elementary.

Suppose now that we want to associate with the family of sets $Exp(X)$ not a relative monad but a usual monad. First we would need to extend the function $Exp : 2^V \to U$ to a function $U \to U$. There is no way of doing it in the UniMath and I do not know of any way of doing it in any constructive foundation. The best one could achieve is to construct a function $Exp' : U \to U$ and a family of isomorphisms $\phi_V : Exp(X) \to Exp'(X)$ for $X \subset V$. This requires developing a constructive theory of filtered colimits and functors that commute with such colimits and it is not an obvious task.

Alternatively, one can build a monad Exp'' corresponding to a signature directly by constructing the set $Exp''(X)$ as an initial algebra over the functor $F_{Sig,X} : Sets(U) \to Sets(U)$ given on objects by the formula

$$F_{Sig,X}(A) := X \coprod (\coprod_{O \in Op} A^{Ar(O)})$$

Constructing initial algebras for $F_{Sig,X}$ also requires the use of colimits, but only ω-colimits, that is, colimits of sequences, see e.g. [2]. The monad structure on the family of sets $Exp''(X)$ can be constructed from the initial algebra structures, see [10] or [29, Th.3, p.161]. One is then left with the task of establishing a family of bijections between $Exp''(X)$ and $Exp(X)$ for $X \subset V$ that are compatible with the substitution which can be done but requires extra work.

In general, a monad on $Sets(U)$ is a richer object than a Jf-relative monad and there may be situations when a monad associated with a signature is required as an intermediary between the syntax and an abstract mathematical construction. However, in our case, when we want to construct from the syntax a C-system, a Jf-relative monad is precisely what we need, so that even when we have a monad at our disposal we have to restrict it to a Jf-relative monad first in order to perform our construction.

This is why we use Jf-relative monads and not the usual monads.

Let us now explain another very important point. At the very start of our explanation of how the Jf-relative monads are related to expressions we said that we will consider *algebraic* expressions. However, the expressions that appear in the sentences of type theories are often not algebraic because they contain operations that bound some of the variables in their arguments. For example, the expression $\prod x : A, B$ that appear in (11) can be rewritten as $\prod(A, x.B)$ which makes it visible that it is the result of an operation $\prod$ applied to two arguments A and B and that this operation binds one variable, here called x, in its second argument.

Expressions that contain operations that may bound variables in their arguments are called expressions with binders.

Expressions with binders are specified by binding signatures - pairs consisting of a set of operations Op and the arity function $Ar : Op \to Fseq(\mathbf{N})$. Here $Fseq(\mathbf{N})$ is the set of finite sequences of elements of $\mathbf{N}$. The set $\mathbf{N}$ is considered as a subset of $Fseq(\mathbf{N})$ through the embedding taking d to the sequence $(0, \ldots, 0)$ of length d. This defines an inclusion of algebraic signatures into binding signatures. The earliest mention of the concept equivalent to the binding arity that we know of is in [1]. The meaning of an operation E with the algebraic arity d is that E has d arguments.

The meaning of an operation E with the binding arity $(i_1,\ldots,i_d)$ is that E has d arguments and binds i_k variables in its k-th argument.

To apply an operation Op with arity $(n_0,\ldots,n_{d-1})$ to expressions $E_0,\ldots,E_{d-1}$ one has to specify, in addition to the expressions themselves, d sequences, of lengths $n_0,\ldots,n_{d-1}$ respectively, of names of variables. These sequences will show which of the variables are bound in each argument.

The best known examples of operations with binders are the quantifiers $\forall$ and $\exists$ of predicate logic and the λ-abstraction of the (untyped) lambda calculus [14],[9]. All three of these operations have arity (1), that is, they have one argument in which they bind one variable.

To get an example with arity (2) one may consider the operation that one gets by applying an operation of arity (1) twice.

Consider expressions formed by operations with binders applied to variables. For example, consider expressions generated by one operation of arity (1) that we will call λ. Every such expression is of the form

$$E = \lambda\, x_{n-1}.\lambda\, x_{n-2}.\ldots.\lambda\, x_0.x$$

Here $x_n,\ldots,x_0$ are bound variables. We do not assume that $x_i \neq x_j$ for $i \neq j$ or that $x_i \neq x$. In particular x is a free variable if $x_i \neq x$ for all $i \geq 0$ and a bound one otherwise. The usual, but hard to formulate precisely, rules of α-equivalence (see e.g. [9, Def. 2.1.11, p.26]) imply that if we rename the bound variables in any way that preserves the rightmost occurrence of x among the x_i's then the resulting expression will be α-equivalent to the original one. In particular, we can always rename x_i such that $x_i \neq x_j$ for $i \neq j$ and there is an most one k such that $x_k = x$. If such a k exists then E has no free variables and if it does not then E has one free variable x.

If x is a free variable then we can substitute another expression E' of the same form for x. However, we can not do it directly. Instead we have to use something called the *capture avoiding substitution* to avoid the "capture" of variable names by binders. For example, let $E = \lambda\, x_0.x$, where x is free, and $E' = x'$. Then we have two cases - if $x_0 \neq x'$ then we can directly substitute E' for x and $E[E'/x] = \lambda\, x_0.x'$. If $x_0 = x'$ we have first to rename x_0 into x_0' such that $x_0' \neq x'$ and then to substitute, obtaining $E[E'/x] = \lambda\, x_0'.x'$. If we used direct substitution the resulting operation would not respect the α-equivalence. The capture avoiding substitution does.

One shows, and it should be clear from the above that it is not easy, that for any binding signature (Op, Ar) one can define, for expressions constructed using operations of this signature and names of variables from a given set V, which occurrences of variable names among the arguments of the operations are free and which are bound. From this one can define, for any subset X of V, the set $Exp^{\cdot}(X)$ of expressions with free variables from X. Next one can define the concept of α-equivalence on each of the sets $Exp^{\cdot}(X)$ and define the sets $Exp^{\alpha}(X)$ of α-equivalence classes of expressions with free variables from X. Most definitions of α-equivalence require V to be a set with an additional operation that for every finite subset of V gives an element in the complement to this subset. Let us call it a freshness operation. Some approaches to the α-equivalence and further constructions discussed below, notably the approach through the nominal sets [31], may only require that for any finite subset of V there exist an element in the complement to this subset. In the latter case we will say that V has the freshness property. In the ZFC a set has the freshness property if and only if it is infinite. In constructive meta-theories the situation may be more involved and it is convenient to have a special name for this particular property.

If V has the freshness property one can define, and again it is not at all easy, the simultaneous capture avoiding substitution of expressions $E_x \in Exp^{\cdot}(Y)$, $x \in X$, for the free variables of an expression $E' \in Exp^{\cdot}(X)$ such that it is compatible with the α-equivalence. After the passage to the α-equivalence classes these constructions become equivalent and one obtains, for any function $X \to Exp^{\alpha}(Y)$ and an element of $Exp^{\alpha}(X)$, an element of $Exp^{\alpha}(Y)$. In addition one has, for any $X \subset V$, a function $X \to Exp^{\alpha}(X)$.

This brings us again to a structure of the same form as we obtained in the case of algebraic operations - a J_V-relative monad, where J_V is the obvious functor from $Sets(2^V)$ to $Sets(U)$. Performing the same construction as the one described above in the case of algebraic expressions one obtains from a $J_{\mathbf{N}}$-relative monad, a Jf-relative monad $\mathbf{Exp}_{Sig}$. This is a direct generalization of the construction that we described previously from algebraic expressions to expressions with binders. The main idea behind this generalization goes back to Fiore, Plotkin and Turi [16] where structures equivalent to the Jf-relative monads are introduced under the name of abstract clones.

This is how the Jf-relative monads appear in the theory of expressions with binders.

Let us calculate what we get from this construction when the signature is given by one operation λ with the arity (1). The expressions then are the expressions that we considered above. We have seen that

$$Exp^{\alpha}(\emptyset) = \{a_{n,k} \,|\, n \in \mathbf{N}, k = 0, \ldots, n-1\}$$

where $a_{n,k}$ is (the equivalence class of) the expression with n λ-abstractions such that k is the smallest index satisfying $x_k = x$.

Next, we know that

$$Exp^{\alpha}(\{x\}) = Exp^{\alpha}(\emptyset) \cup \{b_n(x) \,|\, n \in \mathbf{N}\}$$

where $b_n(x)$ is the expression with n λ-abstractions ending with x and such that $x_i \neq x$ for all $n-1 \geq i \geq 0$. We have to add $Exp^{\alpha}(\emptyset)$ because an expression without free variables is an expression with free variables from the set $\{x\}$.

Finally, for a general $X \subset V$ we have

$$Exp^{\alpha}(X) = Exp^{\alpha}(\emptyset) \cup (\cup_{x \in X}\{b_n(x) \,|\, n \in \mathbf{N}\})$$

and the union on the right hand side is disjoint.

The capture avoiding substitution in the case of one free variable is of the form

$$b_n(a_{n',k'}/x) = a_{n+n',k'}$$

$$b_n(b_{n'}(x')/x) = b_{n+n'}(x')$$

For many free variables the substitution is determined by the case of one free variable because in any one expression there is at most one free variable.

It is easy to see that the Jf-monad that we obtain in this case is isomorphic to the Jf-monad defined by the algebraic signature with operations a_k, $k \in \mathbf{N}$ and b where the arity of a_k is 0 and the arity of b is 1. The elements corresponding under this isomorphism to $a_{n,k}$ are $b^n(a_k)$ and the elements corresponding to $b_n(x)$ are $b^n(x)$.

Church's famous λ-calculus starts with the system of abstract expressions corresponding to two operations ap and λ with the arity of ap being $(0,0)$ and the arity of λ being (1). Operation ap is called application and is usually denoted using the infix notation with the empty operation symbol, that is, $ap(E,E')$ is denoted $E\,E'$.

I do not know of an algebraic representation similar to what we have described above for the free Church's λ-expressions, that is, for the Jf-monad corresponding to the binding signature

$$Sig_\Lambda = (\{ap, \lambda\}, Ar(ap) = (0,0), Ar(\lambda) = (1))$$

More generally, one may ask if for any binding signature Sig one may construct an algebraic signature $Alg(Sig)$ and an isomorphism between the Jf-relative monads corresponding to Sig and $Alg(Sig)$ as we have done in the case when $Sig = (\{\lambda\}, Ar(\lambda) = (1))$.

To obtain the actual λ-calculus, or more specifically, the $\lambda_{\eta\beta}$-calculus, one has to add to the system of expressions defined by Sig_Λ two relations that are called the β- and the η-reductions. The fact that one still gets a Jf-monad structure after passing to the equivalence classes under the equivalence relation generated by these "reductions" requires a proof.

It appears that the Jf-monad, corresponding to the $\lambda_{\eta\beta}$-calculus has an algebraic presentation closely related to the combinatory logic of Schönfinkel [32] (translated in [35]) and Curry [15]. However many subtle difficulties arise in making it precise (cf. [33]) and we know of no theorem asserting such a presentation in terms of relative monads or monads.

This is how the Jf-relative monads corresponding to binding signatures relate to the Jf-relative monads corresponding to algebraic signatures.

What we said about the direct extension of $\mathbf{Exp}_{Sig}$ from a Jf-relative monad to a monad immediately generalizes from the algebraic case to the case of operations with binders.

Also generalizes the discussion about the possibility to construct a monad corresponding to the signature directly using category theory. The beginnings of this generalization can be see in [16]. It is highly non-trivial. Operations that bind variables change the set of free variables e.g for $x \in X$, the operation $\lambda\, x$ can be seen as an operation from $Exp(X \coprod \{x\})$ to $Exp(X)$. Because of this, the individual sets $Exp^\alpha(X)$ do not have universal characterization. Instead, a universal characterization can be given to a functor $Exp : Sets(U) \to Sets(U)$ that will be later given a monad structure. This functor has an initial algebra structure for $\underline{Id} + H_{Sig}$ where H_{Sig} is a *functor of the second order* - a functor from functors to functors and $\underline{Id}$ is the functor of second order that takes any F to the identity functor of $Sets(U)$. The functor H_{Sig} can be directly constructed from the binding signature Sig. Bindings correspond to the operation on functors $F \mapsto F'$ where $F'(X) = F(X \coprod pt)$. The general theory of initial algebras for ω-cocontinuous functors from [2] is applicable here as well and an initial algebra Exp'' for $\underline{Id} + H_{Sig}$ can be constructed as the colimit of the sequence of functors $(\underline{Id} + H_{Sig})^n(\underline{\emptyset})$ where $\underline{\emptyset}$ is the functor $X \mapsto \emptyset$. Since the initial algebras are unique up to a unique algebra isomorphism the sets $Exp''(X)$ constructed by the colimit construction are in a bijective correspondence with the sets $Exp(X)$ of α-equivalence classes of expressions. The set Exp'' are closely related to the sets that one obtains representing α-equivalence classes using de Bruijn levels or indexes. There is more story to tell here, but it is too much outside of the scope of the present paper.

Next one needs to construct a monad structure on Exp''. The corresponding theory is developed in [29], [4] and [5]. An outline of the theory that allows one to give a universal characterization to the monad structure itself can be found in [22]. Not all is understood yet and it remains an active area of research. Much of the work that is being done today is being simultaneously formalized in the UniMath. The key question here is what structure on H has to be specified in order to obtain a monad structure on the initial algebra of $\underline{Id}+H$. The main idea was introduced in [29]. In [4] a functor with this structure is called an (abstract) signature. As became understood later in [5], the additional condition of H being ω-cocontinuous allows one to remove the condition of the existence of the right adjoints from the main theorem [29, Th. 15, p.170] leading to [5, Th. 48].

The case that is most important for us, that of the monads defined by the binding signatures, has been fully formalized in the UniMath. There remains the problem of showing that the families of sets of the J_V-relative monads corresponding to this monad are isomorphic to the sets of expressions modulo the α-equivalence and that the monad structure that one obtains satisfies the universality conditions of [22].

The preceding discussion shows how the monads corresponding to the binding signatures can be constructed by methods of category theory.

The raw syntax of a type theory can be specified by a binding signature[2]. For example, the raw syntax of Streicher's formulation of the Calculus of Constructions of G. Huet and T. Coquand (CC-S), when brought into the standard form, consists of six operations $\prod$, $Prop$, $Proof$, λ, app and $\forall$ with the corresponding arities $(0,1)$, $()$, (0), $(0,1)$, $(0,1,0,0)$ and $(0,1)$, see [34, p.157].

In view of the preceding discussion, this suggests that the class of abstract mathematical objects that can be used to most directly model the raw syntax of type theories is the class of Jf-relative monads. However, in this paper we use pairs of a Jf-relative monad **RR** and a left module **LM** (see Definition 2.31) over this monad. Let us explain why we need such pairs and how one can generate them from data similar to binding signatures.

[2]The type theories whose syntax can be specified by an algebraic signature correspond to the "generalized algebraic theories" of John Cartmell [13], [12], [18].

To obtain the binding signature of the raw syntax from the usual presentation of a type theory by a list of inference rules such as (11) one should make the list of operations that these inference rules introduce with their names and their binding arities. Often operations will be given in a non-standard form such as $\prod x : A, B$ instead of $\prod(A, x.B)$, but for unambiguous inference rules it should be easy to see what the corresponding standard form should be.

Among those operations will be operations that introduce *types* and operations that introduce *elements* (also called *objects*) of types.

For example, in the type theory CC-S the operations $\prod$, $Prop$ and $Proof$ introduce types while operations λ, *app* and $\forall$ introduce elements. In addition, some arguments of each operation must be types and some elements. However, *only element variables can be bound.*

Define a restricted 2-sorted binding signature as a signature where arities of operations are given by sequences $((n_0, \epsilon_0), \ldots, (n_{d-1}, \epsilon_{d-1}), \epsilon)$ where $\epsilon \in \{0, 1\}$ with 0 corresponding to elements and 1 to types. Such two sorted arities of the six operations of CC-S are, correspondingly, $((0,1),(1,1),1)$, (1), $((0,0),1)$, $((0,1),(1,0),0)$, $((0,1),(1,1),(0,0),(0,0),0)$ and $((0,1),(1,0),0)$.

Any restricted 2-sorted binding signature defines the usual, 1-sorted one, where the set of operations is the same and the arity function is the composition of the original arity function with the function that maps $((n_0, \epsilon_0), \ldots, (n_{d-1}, \epsilon_{d-1}), \epsilon)$ to $(n_0, \ldots, n_d)$.

Let Sig_2 be a (restricted) 2-sorted binding signature and Sig_1 the corresponding 1-sorted one. Let Z be a set such that the set of expressions with respect to Sig_1 with variables from Z is defined. Let us fix two subsets $V, Y \subset Z$ such that $V \cap Y = \emptyset$. Consider the subset $Exp_{Sig_2}[V, Y]$ of expressions that conform to the additional rules defined by the sequences $(\epsilon_0, \ldots, \epsilon_{n-1}, \epsilon)$ of the 2-sorted arities of the operations of Sig_2 under the assumption that a variable can be used as an element variable if and only if it is in V and as a type variable if and only if it is in Y. This subset will be the disjoint union of two smaller subsets $ElExp_{Sig_2}[V, Y]$ and $TyExp_{Sig_2}[V, Y]$ where the first one consists of expressions of sort "element" and the second one of expressions of sort "type".

Next, for a subset X of V let $Exp^{\cdot}_{Sig_2}(X, Y)$ be the subset of $Exp_{Sig_2}[V, Y]$ that consists of expressions where an element variable is free if and only if it belongs to X with a similar notation for $ElExp$ and $TyExp$. Let us assume in addition that V has the freshness property. Then one can define the α-equivalence relation on $Exp^{\cdot}_{Sig_2}[V, Y]$ and therefore on $ElExp^{\cdot}_{Sig_2}(X, Y)$ and $TyExp^{\cdot}_{Sig_2}(X, Y)$. Let $ElExp^{\alpha}_{Sig_2}(X, Y)$ and $TyExp^{\alpha}_{Sig_2}(X, Y)$ be the corresponding sets of equivalence classes.

Let us fix a set $PrTy \subset Z$ such that $V \cap PrTy = \emptyset$. This set will eventually play the role of the set of primitive types that we add to the base type theory. Consider X as a variable, writing $RR_V(X)$ and $LM_V(X)$ instead of $ElExp^{\alpha}_{Sig_2}(X, PrTy)$ and $TyExp^{\alpha}_{Sig_2}(X, PrTy)$.

The structures that we get on the families of sets $RR_V(-)$ and $LM_V(-)$ are slightly different. On RR we get the J_V-relative monad structure - for any $X \subset V$ we have a function $\eta_X : X \to RR_V(X)$ and for any $X, Y \subset V$ and a function $f : X \to RR_V(Y)$

we have a function

$$rr_{X,Y}(f) : RR_V(X) \to RR_V(Y)$$

On the other hand, on the LM_V we do not have η_X since variables from X are not type expressions and substitution defines for any $X, Y \subset V$ and a function $f : X \to RR_V(Y)$, a function

$$lm_{X,Y}(f) : LM_V(X) \to LM_V(Y)$$

This operation makes the family of sets $LM_V(X)$ into a left module $\mathbf{LM}_V = (LM_V, lm)$ over the J_V-monad $\mathbf{RR}_V = (RR_V, \eta, rr)$.

Precomposing $\mathbf{RR_N}$ and $\mathbf{LM_N}$ with the obvious functor $\Phi : \mathbb{F} \to Sets(2^{\mathbf{N}})$ using Constructions 2.15 and 2.35 we obtain a pair $(\mathbf{RR}, \mathbf{LM})$ of a J_f-relative monad and a left module over it.

In some type theories all types are elements of universes and moreover element expressions are not syntactically distinguishable from type expressions. For example, it is the case in the very important type theory MLTT79 - the Martin-Löf type theory from [28]. The inference rules related to the universes [28, p.172] make all type expressions also element expressions and an element expression of any form may be used as a type. In our notation it means that $LM(X) = RR(X)$.

The preceding discussion shows how pairs of a Jf-relative monad $\mathbf{RR}$ *and a left module* $\mathbf{LM}$ *over it correspond to the raw syntax of type theories because some expressions are type expressions and some are element expressions.*

To construct the pair $(\mathbf{RR}, \mathbf{LM})$ by methods of category theory without a reference to expressions one can proceed as follows.

A restricted 2-sorted binding signature defines a monad on the category $Sets(U) \times Sets(U)$. See [48] for a much more general case of multi sorted signatures. For the formalization of this construction in UniMath see [6].

Choosing an object $PrTy$ of $Sets(U)$ and applying Construction 2.24 we obtain a monad on $Sets(U)$. Let us denote it by $\mathbf{RR}^*$. Applying Construction 2.42 we obtain a module $\mathbf{LM}^*$ over this monad with values in $Sets(U)$. Precomposing with the functor $Jf : \mathbb{F} \to Sets(U)$ using Constructions 2.15 and 2.35 one obtains a pair $(\mathbf{RR}, \mathbf{LM})$ of a Jf-monad and a module over it.

This is how the pairs $(\mathbf{RR}, \mathbf{LM})$ *can be obtained from a restricted 2-sorted binding signature by methods of category theory.*

To make it easier to compare the constructions that follow with the earlier constructions let us recall that the substitution structure on elements of $RR(n)$ and $LM(n)$ can be easily expressed in terms of the structures of the relative monad and a module of a relative monad. Namely, if $E_1, \ldots, E_k \in RR(\underline{m})$, $E \in RR(\underline{n})$ and $0 \leq i_1, \ldots, i_k \leq n-1$ then $E[E_1/\underline{i_1}, \ldots, E_k/\underline{i_k}] \in RR(\underline{m})$ is the element $rr(f)(E)$ where $f : stn(n) \to RR(\underline{m})$ is given by $f(i) = \eta i$ for $i \neq i_1, \ldots, i_k$ and $f(i_j) = E_j$. Exactly the same formula with the replacement of $RR(n)$ by $LM(n)$ describes $T[E_1/\underline{i_1}, \ldots, E_k/\underline{i_k}]$ for $T \in LM(n)$.

This completes a large section of our introduction. We can now view the α-equivalence classes of the type and element expressions with the given sets of free variables as mathematical objects.

Let us return to sentences of type theory that can be of the five types (1)-(5). The expressions in the sentences are considered modulo the α-equivalence. Moreover, the sentences themselves are also considered modulo the α-equivalence, that is, modulo the renaming of the variables $x_0, \ldots, x_{n-1}$ introduced by the context. Using this α-equivalence we may assume that $(x_0, \ldots, x_{n-1}) = (0, \ldots, n-1)$. Then T_i has free variables from $\{0, \ldots, i-1\}$ and T, T', t and t' free variables from $\{0, \ldots, n-1\}$.

When we are given a pair $(\mathbf{RR}, \mathbf{LM})$ we know that an α-equivalence class of element expressions with free variables from $\{0, \ldots, n-1\}$ is given by an element of $\mathbf{RR}(n)$ and an α-equivalence class of type expressions with free variables from $\{0, \ldots, n-1\}$ is given by an element of $\mathbf{LM}(n)$. Therefore,

(1) a sentence of the form (1) is an element of

$$B(\mathbf{RR}, \mathbf{LM}) = \coprod_{n\geq 0} \prod_{i=0}^{n-1} LM(i)$$

(2) a sentence of the form (2) is an element of

$$Bt(\mathbf{RR}, \mathbf{LM}) = \coprod_{n\geq 0} (\prod_{i=0}^{n-1} LM(i)) \times LM(n)$$

(3) a sentence of the form (3) is an element of

$$\widetilde{B}(\mathbf{RR}, \mathbf{LM}) = \coprod_{n\geq 0} (\prod_{i=0}^{n-1} LM(i)) \times RR(n) \times LM(n)$$

(4) a sentence of the form (4) is an element of

$$Beq(\mathbf{RR}, \mathbf{LM}) = \coprod_{n\geq 0} (\prod_{i=0}^{n-1} LM(i)) \times LM(n) \times LM(n)$$

(5) a sentence of the form (5) is an element of

$$\widetilde{Beq}(\mathbf{RR}, \mathbf{LM}) = \coprod_{n\geq 0} (\prod_{i=0}^{n-1} LM(i)) \times RR(n) \times RR(n) \times LM(n)$$

In any approach to Martin-Lof type theory a sentence of the form $0 : T_0, \ldots, n-1 : T_{n-1} \rhd T\ type$ is equivalent to the sentence $0 : T_0, \ldots, n-1 : T_{n-1}, n : T \rhd ok$. This allows one not to consider sentences of the form (2).

Next, we need to construct from $(\mathbf{RR}, \mathbf{LM})$, B, $\widetilde{B}$, Beq and $\widetilde{Beq}$ a C-system CC. This construction should be compatible with the constructions outlined in earlier papers, such as the construction of the category with families outlined in [23]. In particular, the set of objects of CC should be defined together with an isomorphism to the quotient set $B/\sim$ of the set B by the equivalence relation defined by the set

Beq according to the rue that $(T_0, \ldots, T_{n-1})$ is equivalent to $(T'_0, \ldots, T'_{n'-1})$ if and only if $n' = n$ and the sequences defined by the table (7) are in Beq.

Hofmann and some other authors suggest to directly construct the set of morphisms and all the required structures using the subsets $\widetilde{B}$ and $\widetilde{Beq}$. Already the first step, the definition from $\widetilde{B}$ of a set that will later have to be factorized by an equivalence relation coming from $\widetilde{Beq}$ to produce the set of morphisms is non-trivial, c.f [23, Def. 2.11, p.97]. Constructing the composition and proving its properties such as the associativity represents additional difficulties.

We proceed in a different manner. Instead of starting with $B/\sim$ and building the C-system structure on it, we will construct a C-system $C(\mathbf{RR}, \mathbf{LM})$, which knows nothing about the subsets B,$\widetilde{B}$,Beq,$\widetilde{Beq}$, and then use the results of [45] to show how any quadruple of subsets B,$\widetilde{B}$,Beq,$\widetilde{Beq}$, satisfying certain properties, defines a sub-quotient C-system of $C(\mathbf{RR}, \mathbf{LM})$. This sub-quotient will be the term model C-system of our type theory.

The conditions on the B-subsets will be seen to be the conditions many of which have been long known as the "structural properties" that the valid sentences of all type theories must satisfy. By approaching them from the direction of [45] we will see why these particular conditions are necessary and sufficient for a type theory to "make sense".

The main property of a sentence is its validity. Validity of sentences is determined on the basis of the inference rules.

The validity of a sentence of the form $0 : T_0 \rhd ok$ asserts that T_0, which must be a closed expression according to the rules stated above, is a valid expression that describes *a type* in the system. For example, in any of the Martin-Löf type theories, there is a type $\mathbf{N}$, which is called the type of natural numbers. The formal equivalent of this assertion is that $\mathbf{N}$ is an expression and the sentence $0 : \mathbf{N} \rhd ok$ is valid.

One often uses the words "type", "type expression" and "expression that describes a type" interchangeably. The same applies to "element", "element expression" and "expression that describes an element".

A sentence of the form $0 : T_0, 1 : T \rhd ok$ asserts that T_0 is a valid, closed, type expression and T is an expression with the only possible free variable being 0 that describes a family of types parametrized by T_0. For example, in any of the Martin-Löf type theories, given a type T and two elements $t, t' : T$ of T, there is a type $Id\,T\,t\,t'$ whose elements are to be thought of as constructions of equalities between t and t' in T. Correspondingly, the sentence $0 : T, 1 : Id\,T\,0\,0 \rhd ok$ is valid if and only if the sentence $0 : T \rhd ok$ is.

Sentences of the form (2) with $n > 1$ describe "iterated type families". For example, for $n = 2$, T_0 is a type, T_1 is a type family parametrized by T_0 and T_2, which is an expression that may contain x_0 and x_1 as free variables, is a type family with two parameters $x_0 : T_0$ and $x_1 : T_1(x_0)$.

Let $\Gamma = (x_0 : T_0, \ldots, x_{n-1} : T_{n-1})$. If the sentence (3) is valid then so is the sentence $\Gamma \rhd T\ type$. A sentence (3) with $n = 0$, that is a sentence of the form

$\rhd t : T$, asserts that T is a valid (closed) type expression and t is a valid (closed) expression that describes an element of type T. For example, the element 0 of $\mathbf{N}$ in the Martin-Löf type theories is denoted by O so that the sentence $\rhd O : \mathbf{N}$ is valid in all these theories. A sentence of the form (3) with $n = 1$ describes a family T of types parametrized by T_0 together with a "section" of this family, that is, a family of elements $t(x_0)$ of types $T(x_0)$ for all x_0. If T does not contain x_0 then the family of types is constant and the sentence (3) is a syntactic representation of a function from T_0 to T.

If the sentence (4) is valid than so are the sentences $\Gamma \rhd T$ *type* and $\Gamma \rhd T'$ *type*. The validity of (4) asserts that the type expressions T and T' are *definitionally equal* in the context Γ. The similar meaning is assigned to sentences of the form (5). Definitional equality of type expressions can be used to define definitional equality of contexts. Namely, one defines two contexts $x_0 : T_0, \ldots, x_{n-1} : T_{n-1}$ and $x_0 : T'_0, \ldots, x_{n-1} : T'_{n-1}$ to be definitionally equal if the sentences in the following sequence are valid

$$[\mathbf{2017.04.07.eq1}] \rhd \ T_0 \equiv T'_0 \tag{7}$$

$$x_0 : T_0 \rhd \ T_1 \equiv T'_1 \tag{8}$$

$$\ldots \tag{9}$$

$$x_0 : T_0, \ldots, x_{n-2} : T_{n-2} \rhd \ T_{n-1} \equiv T'_{n-1} \tag{10}$$

This provides us with the concept of definitional equality of sentences of the form (1). Two such sentences are called definitionally equal if their contexts are.

Two sentences of the form (2) $\Gamma \rhd T$ *type* and $\Gamma' \rhd T'$ *type* are definitionally equal if $\Gamma \equiv \Gamma'$ and $\Gamma \rhd T \equiv T'$.

Similarly, one can use definitional equality of type expressions and definitional equality of element expressions defined by sentences of the form (4), and (5) to define definitional equality of sentences of the form (3). Two such sentences $\Gamma \rhd t : T$ and $\Gamma' \rhd t' : T'$ are called definitionally equal if $\Gamma \equiv \Gamma'$ and the following two sentences are valid:

$$\Gamma \rhd \ T \equiv T'$$

$$\Gamma \rhd \ t \equiv t' : T$$

It should be immediately clear from the asymmetry of these rules that in order for the definitional equality relations on sentences of the form (1), (2) and (3) to be equivalence relations the sets of sentences of various kinds should satisfy more conditions than the ones that we have mentioned so far. These conditions are among the conditions **??** whose mathematical meaning is established in this paper.

@@@If we write $x \notin \Gamma$ for the condition that x is a name of free variable not contained in the subset of names declared in Γ, then this assertion can be expressed in the form of a pair of inference rules

$$\frac{\Gamma \rhd T \ type}{x \notin \Gamma \quad \Gamma, x : T \rhd ok} \qquad \frac{\Gamma, x : T \rhd ok}{\Gamma \rhd T \ type}$$

@@@

@@@the form of a list of "inference rules" that may look like, for example, this one:

$$[\mathbf{2017.03.02.eq1}]\frac{\Gamma, x : A \vdash B\, type}{\Gamma \vdash \prod x : A, B\, type} \tag{11}$$

These inference rules are formulated in@@@

To speak about the mathematical meaning of the conditions that the valid sentences should satisfy, we need to find a way to view the structure formed by the five kinds of sentences as a mathematical object and to describe, and analyze, a construction that generates a C-system from such an object.

... that were specified through their raw syntax and typing algorithms or, extensionally speaking, the subsets of well-typed context-judgement pairs.

For $p : X \to Y$ in a category, let $sec(p) = \{s : Y \to X \mid s \circ p = Id_X\}$. Elements of $sec(p)$ are called sections of p. To any C-system CC one associates a pair of sets $(Ob(CC), \widetilde{Ob}(CC))$ where $Ob(CC)$ is the set of objects of the category underlying CC and where

(12)

$$[\mathbf{2017.02.04.eq1}]\widetilde{Ob}(CC) = \{s \in Mor(CC) \mid s \in sec(p_{codom(s)}), l(codom(s)) > 0\}$$

or, in words, where $\widetilde{Ob}(CC)$ is the set of sections of the non-trivial p-morphisms of CC. When CC is clear from the context we will abbreviate $Ob(CC)$ and $\widetilde{Ob}(CC)$ to Ob and $\widetilde{Ob}$ respectively.

The sets $(Ob, \widetilde{Ob})$ are equipped with a system of operations $(l, ft, \partial, T, S, \widetilde{T}, \widetilde{S}, \delta)$. Pairs of sets equipped with operations of such form are called (unital) pre-B-systems (c.f. [**?**]). The first main result of [45], Proposition 4.2, shows how to construct from a sub-pre-B-system $(B, \widetilde{B})$ of $(Ob, \widetilde{Ob})$ a sub-C-system of CC whose associated pre-B-system is $(B, \widetilde{B})$. The second main result of that paper, Proposition 5.4, shows how to construct from a pair of equivalence relations $(\sim, \approx)$ on Ob and $\widetilde{Ob}$ respectively that satisfies certain conditions, a quotient C-system of CC whose associated pre-B-system is $(Ob/\sim, \widetilde{Ob}/approx)$. These conditions are shows in Proposition **??** of the present paper to be compatible with isomorphisms of pre-B-systems.

After defining $C(\mathbf{RR}, \mathbf{LM})$ in Sections **??** and **??** we define, in Construction **??**, a pre-B-system structure on the sets $(B(\mathbf{RR}, \mathbf{LM}), \widetilde{B}(\mathbf{RR}, \mathbf{LM}))$ and, in Constrcution **??**, an isomorphism between this pre-B-system and the pre-B-system of $C(\mathbf{RR}, \mathbf{LM})$.

Using this isomorphism we formulate the conditions on subsets $B \subset B(\mathbf{RR}, \mathbf{LM})$ and $\widetilde{B} \in \widetilde{B}(\mathbf{RR}, \mathbf{LM})$ that are necessary and sufficient for such a pair to correspond to a sub-C-system of $C(\mathbf{RR}, \mathbf{LM})$.

Next, we construct, for any subsets B, $\widetilde{B}$ and Beq, $\widetilde{Beq}$ a pair of relations $\sim$ and $\approx$ on B and $\widetilde{B}$ respectively and show under which condition on Beq, $\widetilde{Beq}$ the transport of these relations to the subsets of $(Ob(C(\mathbf{RR}, \mathbf{LM})), \widetilde{Ob}(C(\mathbf{RR}, LM)))$ corresponding to B, $\widetilde{B}$, satisfy the conditions of [45, Prop. 5.4] and therefore define a (regular) quotient of the C-system corresponding to $(B, \widetilde{B})$.

Summing it up in Construction **??** we obtain a construction, for any four subsets B,$\widetilde{B}$,Beq,$\widetilde{Beq}$ satisfying certain condition, of a C-system. This is the final construction of the paper.

It should be possible to approach this construction in a more direct way. First, one would define the concept of a B-system as a pre-B-system whose operations satisfy some axioms. Next one would show that for a C-system CC the pre-B-system $(Ob(CC), \widetilde{Ob}(CC))$ is a B-system and that the mapping

$$CC \mapsto (Ob(CC), \widetilde{Ob}(CC))$$

is the object component of a functor from the category of C-systems to the category of B-systems with obviously defined homomorphisms as morphisms. Next, one would construct, using in particular an abstract analog of [23, Def. 2.11], a functor in the opposite direction, from B-systems to C-systems. Finally, one would construct two functor isomorphisms extending this pair of functors to an equivalence between the categories of C-systems and B-systems.

Then one would prove that for any $\mathbf{RR}$, $\mathbf{LM}$ the pre-B-system structure that we define on $(B(\mathbf{RR}, \mathbf{LM}), \widetilde{B}(\mathbf{RR}, \mathbf{LM}))$ is a B-system structure. A sub-pre-B-system of a B-system is a B-system. Therefore the subsets $(B, \widetilde{B})$ should define a B-system and the relations $(\sim, \approx)$ constructed from the subsets $(Beq, \widetilde{Beq})$ should be a congruence relation such that the quotients $(B/\sim, \widetilde{B}/\approx)$ again carry a structure of a B-system. Finally, applying the inverse functor to this B-system we would obtain a C-system that will be the C-system corresponding to the quadruple of sets B,$\widetilde{B}$,Beq and $\widetilde{Beq}$.

Such a direct construction would probably be more satisfying than the one that we provide. However, it would require a lot of non-trivial work and, as far as the goal of constructing a C-system from the subsets B,$\widetilde{B}$,Beq and $\widetilde{Beq}$, will give the same result as our less direct, but more simple approach. Still, defining B-systems and constructing an equivalence between the categories of C-systems and B-systems is important and we plan to address it in future papers. The approach that we take here, using the results of [45] instead, gives us a rigorous construction that can be completed today.

Note: Ignore the rest of the introduction

To construct the regular sub-quotient of $C(\mathbf{RR}, \mathbf{LM})$ that corresponds to the type system generated by the given system of inference rules one uses the main results, Propositions 4.3 and 5.4, of [45]. To use these propositions to obtain a sub-quotient of a C-system CC one should provide a pair of subsets $B \subset Ob(CC)$, $\widetilde{B} \subset \widetilde{Ob}(CC)$ that is closed under operations $(pt, ft, \partial, \widetilde{T}, \widetilde{S}, \delta)$ that are defined on the pair of sets $(Ob(CC), \widetilde{Ob}(CC))$ associated with any C-system [45, Prop. 4.3] and a pair of equivalence relations on $(B, \widetilde{B})$ that satisfy the conditions of [45, Prop. 5.4]. The most important conditions of [45, Prop. 5.4] are connected to the behavior of the equivalence relations under the restriction of the operations $(pt, ft, \partial, \widetilde{T}, \widetilde{S}, \delta)$ to $(B, \widetilde{B})$.

We defer the detailed descriptions both of the step preceding the one described here and of the one following it to future papers. In the remaining part of the introduction we describe the content of the paper without further references to type theory.

We start the paper with two sections where we introduce some constructions applicable to general C-systems.

On the sets of objects of any C-system one can consider the partial ordering defined by the condition that $X \leq Y$ if and only if $l(X) \leq l(Y)$ and $X = ft^{l(Y)-l(X)}(Y)$. In the first section we re-introduce some of the objects and constructions defined in [45] using the length function using this partial ordering instead. This allows to avoid the use of natural numbers in some of the arguments that significantly simplifies the proofs.

In the second section we construct for any C-system CC and a presheaf F on the category underlying CC a new C-system $CC[F]$ that we call the F-extension of CC. The C-systems of this form remind in some way the affine spaces over schemes in algebraic geometry. While the geometry of affine spaces in itself is not very interesting their sub-spaces encompass all affine algebraic varieties of finite type . Similarly, while the C-systems $CC[F]$ look to be not very different from CC their sub-systems and more generally regular sub-quotients, even in the case of the simplest C-systems $CC = C(\mathbf{RR})$ corresponding to Lawvere theories (see Section 4.2), include all of the term C-systems of type theories.

Regular sub-quotients of any C-system CC are classified by quadruples $(B, \widetilde{B}, \sim, \simeq)$ of the following form.

Let $\widetilde{Ob}(CC)$ be the set of sections of the p-morphisms of CC, i.e., the subset in $Mor(CC)$ that consists of morphisms s such that $dom(s) = ft(codom(f))$ and $s \circ p_{codom(f)} = Id_{dom(f)}$. The sets $Ob(CC)$ and $\widetilde{Ob}(CC)$ are called the B-sets of a C-system and can also be denoted as $B(CC)$ and $\widetilde{B}(CC)$.

The first two components B and $\widetilde{B}$ of the quadruple are subsets in the sets $Ob(CC)$ and $\widetilde{Ob}(CC)$ respectively. The next two components are equivalence relations on B and $\widetilde{B}$. To correspond to a regular sub-quotient the pair $(B, \widetilde{B})$ should be closed under the eight B-system operations on $(B(CC), \widetilde{B}(CC))$ and the equivalence relations of the pair $(\sim, \simeq)$ should be compatible with the restrictions of these eight operations to $(B, \widetilde{B})$ as well as to satisfy three additional simple conditions (see [45, Proposition 5.4]) that involve the length function $l : B \to \mathbf{N}$ on B.

Therefore, in order to be able to describe regular sub-quotients of a C-system one needs to know the B-sets of this C-system, the length function and the action of the eight B-system operations on these sets. Such a collection of data is called a pre-B-system (see [38]). The main result of this paper is a detailed description of the pre-B-systems of the form $(B(CC[F]), \widetilde{B}(CC[F]))$ for an important particular class of "coefficient" C-systems CC (see below).

In Section 4.1 we first remind the notion of a relative monad on a functor $J : C \to D$ that was introduced in [7, Def.1, p.299] and considered in more detail in [8]. Then we focus our attention on relative monads over the functor Jf that is defined as follows.

For two sets X and Y let $Fun(X,Y)$ be the set of functions from X to Y. Let $stn(n)$ be the standard set with n elements that we take to be the subset of $\mathbf{N}$ that consists of numbers $< n$. Consider the category F such that $Ob(F) = \mathbf{N}$ and

$$Mor(F) = \cup_{m,n} Fun(stn(m), stn(n))$$

The functor Jf is the obvious functor from F to the category of sets.

In [43] we constructed an equivalence between the category of Jf-relative monads and the category of Lawvere theories whose component functor from the relative monads to Lawvere theories is denoted RML. A key component of this equivalence is the construction of the Kleisli category $K(\mathbf{RR})$ of a relative monad $\mathbf{RR}$ given in [8]. Most of Section 4.1 is occupied by simple computations in $K(\mathbf{RR})$ for Jf-relative monads $\mathbf{RR}$.

In [41] we constructed an isomorphism between the category of Lawvere theories and the category of l-bijective C-systems - the C-systems CC where the length function $Ob(CC) \to \mathbf{N}$ is a bijection. In Section 4.2 we consider the C-system $C(\mathbf{RR})$ corresponding to the Lawvere theory $RML(\mathbf{RR})$ defined by a Jf-relative monad $\mathbf{RR}$. The underlying category of this C-system is $K(\mathbf{RR})^{op}$. The main result of this section is the description of the B-sets of $C(RR)$ and of the actions of the B-system operations on these sets.

In the final Section 4.3 we apply the construction of Section 3.2 to $C(\mathbf{RR})$ taking into account that the functors $\mathbf{LM} : C(\mathbf{RR})^{op} \to Sets$ are the same as the functors $K(\mathbf{RR}) \to Sets$ that are the same as the relative (left) modules over Jf. In (64) and Construction 4.34 we compute the B-sets $B(C(\mathbf{RR}, \mathbf{LM}))$ and $\widetilde{B}(C(\mathbf{RR}, \mathbf{LM}))$ and in Theorem 4.36 the action of the B-system operations on these sets.

In the next paper we will connect these computations to the conditions that the valid judgements of a type theory must satisfy in order for the term C-system of this type theory to be defined.

We use neither the axiom of excluded middle nor the axiom of choice. The paper is written in the formalization-ready style and should be easily formalizable both in the UniMath and in the ZF.

We use the diagrammatic order of composition, i.e., for morphisms $f : X \to Y$ and $g : Y \to Z$ we write their composition as $f \circ g$.

A category $\mathcal{C}$ is always understood as a pair of sets $Ob(\mathcal{C})$, $Mor(\mathcal{C})$ connected by the operations of domain, codomain, identity and composition where composition is a partially defined operation. A functor $F : \mathcal{C} \to \mathcal{D}$ is a pair of functions $F_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{D})$, $F_{Mor} : Mor(\mathcal{C}) \to Mor(\mathcal{D})$ satisfying the well known conditions. We emphasize it here because it is also possible to define a category starting with a set $Ob(\mathcal{C})$ and a *family* of sets $Mor_{\mathcal{C}}(X,Y)$ parametrized by $X, Y \in Ob(\mathcal{C})$ where a family is understood in the sense of [44, Remark 3.9]. These two concepts are a little different from each other. For example there exists a category with $Ob(\mathcal{C}) = \{0,1\}$ and $Mor_{\mathcal{C}}(a,b) = \{0\}$ for all $a, b \in Ob(\mathcal{C})$, in the sense of the second definition, but not in the sense of the first. Indeed, any category in the sense of the first definition has the property that

$$Mor(X,Y) \cap Mor(X',Y') = \emptyset$$

if $X \neq X'$ or $Y \neq Y'$. This property makes it necessary sometimes to perform additional constructions such as ...

Note that the expression "for all x in A, a $y(x)$ in $B(x)$" is a form of saying "a family y, parametrized by x in A, such that $y(x)$ is in $B(x)$". For a way to define a family of sets in the ZFC without a universe see [44, Remark 3.9]. Families of sets do not form a set. However, families of sets with a given parameter set and such that all the sets of the family are subsets of given set do. A formulation in terms of iterated families can be used for "for all" expressions where there are several parameters such as in Definition 2.5(3). Similarly, a "collection of data", is understood as an n-tuple, which is understood as an iterated pair $(\ldots(-,-),-)\ldots,-)$.

We fix a universe U without making precise what conditions on the set U we require. It is clear that it is sufficient for all constructions of this paper to require U to be a Grothendieck universe. However, it is likely that a much weaker set of conditions on U is sufficient for our purposes. In all that follows we write $Sets$ instead of $Sets(U)$.

This is one the papers extending the material which I started to work on in [36]. I would like to thank the Institute Henri Poincare in Paris and the organizers of the "Proofs" trimester for their hospitality during the preparation of the first version of this paper. The work on this paper was facilitated by discussions with Benedikt Ahrens, Richard Garner and Egbert Rijke.

2. Relative monads and left modules over relative monads

2.1. **Monads and relative monads.** Let us start by reminding the definition of a monad in the form that became standard after MacLane's textbook [27, p.133] and that we will call the *monoidal* form. For a category $\mathcal{C}$ we often write $X \in \mathcal{C}$ instead of $X \in Ob(C)$. For a functor $F = (F_{Ob}, F_{Mor})$ from $\mathcal{C}$ to $\mathcal{D}$ we often write $F(X)$ instead of $F_{Ob}(X)$ for $X \in Ob(\mathcal{C})$ and $F(f)$ instead of $F_{Mor}(f)$ for $f \in Mor(\mathcal{C})$. We emphasize these standard conventions here because below we will sometimes work with the object and morphism components of a functor separately in which case the subscripts Ob and Mor will need to be enforced.

Definition 2.1. **[2017.04.01.def1]** *A monad in the monoidal form on a category $\mathcal{C}$ is a triple $\mathbf{R} = (R, \eta, \mu)$ where $R : \mathcal{C} \to \mathcal{C}$ is a functor and $\eta : Id_{\mathcal{C}} \to R$, $\mu : R \circ R \to R$ are natural transformations such that for any $X \in \mathcal{C}$ one has*

1. **[2017.04.19.eq7]** $R(\mu_X) \circ \mu_X = \mu_{R(X)} \circ \mu_X$ *("associativity"),*
2. **[2017.04.19.eq8]** $\eta_{R(X)} \circ \mu_X = Id_{R(X)}$ *and* $R(\eta_X) \circ \mu_X = Id_{R(X)}$ *(two "unity axioms").*

We will omit the qualification "in the monoidal form" when it is clear from the context.

The following definition specifies objects that we will call "monads in the Kleisli form". To the best of my knowledge it explicitly appeared in the first time in the groundbreaking paper by Eugenio Moggi [30].

Definition 2.2. *[**2017.04.13.def1**] A monad in the Kleisli form on a category $\mathcal{C}$ is a triple $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ where*

*(1) [**2017.04.19.eq1**] $RR_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{C})$ is a function,*
*(2) [**2017.04.19.eq2**] η is a family, parametrized by $X \in Ob(\mathcal{C})$, of morphisms $\eta_X : X \to RR_{Ob}(X)$,*
*(3) [**2017.04.19.eq3**] rr is a family, parametrized by pairs $X, Y \in Ob(\mathcal{C})$, of functions*

$$rr_{X,Y} : Mor_{\mathcal{C}}(X, RR_{Ob}(Y)) \to Mor_{\mathcal{C}}(RR_{Ob}(X), RR_{Ob}(Y))$$

such that

*(4) [**2017.04.19.eq4**] for all $X \in \mathcal{C}$, $rr_{X,X}(\eta_X) = Id_{RR_{Ob}(X)}$,*
*(5) [**2017.04.19.eq5**] for all X, Y, $f : X \to RR_{Ob}(Y)$, $\eta_X \circ rr_{X,Y}(f) = f$,*
*(6) [**2017.04.19.eq6**] for all X, Y, Z, $f : X \to RR_{Ob}(Y)$, $g : Y \to RR_{Ob}(Z)$,*

$$rr_{X,Y}(f) \circ rr_{Y,Z}(g) = rr_{X,Z}(f \circ rr_{Y,Z}(g))$$

We will omit the qualification "in the Kleisli form" when it is clear from the context. Definition 2.1 is equivalent to the Definition 2.2 in the precise sense that is specified by Problem 2.3 and following it Construction 2.4.

Problem 2.3. *[**2017.01.04.prob1**] Given a function $RR_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{C})$ and a family η, parametrized by $X \in Ob(\mathcal{C})$, of morphisms $\eta_X : X \to RR_{Ob}(X)$, to construct a bijection between the following two sets:*

(1) the set of pairs of the form
(1.1) a function $RR_{Mor} : Mor(\mathcal{C}) \to Mor(\mathcal{C})$,
(1.2) a family, parametrized by $X \in Ob(\mathcal{C})$, of morphisms

$$\mu_X : RR_{Ob}(RR_{Ob}(X)) \to RR_{Ob}(X)$$

such that $((RR_{Ob}, RR_{Mor}), \eta, \mu)$ is a monad on $\mathcal{C}$ in the monoidal form, that is,
(1.3) $RR = (RR_{Ob}, RR_{Mor})$ is a functor,
(1.4) (RR_{Ob}, μ) satisfies condition (1) of Definition 2.1,
(1.5) (RR_{Ob}, η, μ) satisfies condition (2) of Definition 2.1,
(1.6) η is a natural transformation $Id_{\mathcal{C}} \to RR$,
(1.7) μ is a natural transformation $RR \circ RR \to RR$,
(2) the set of families, parametrized by $X, Y \in Ob(\mathcal{C})$, of functions

$$rr_{X,Y} : Mor_{\mathcal{C}}(X, RR_{Ob}(Y)) \to Mor_{\mathcal{C}}(RR_{Ob}(X), RR_{Ob}(Y))$$

such that (RR_{Ob}, η, rr) is a monad on $\mathcal{C}$ in the Kleisli form, that is,
(2.1) (RR_{Ob}, η, rr) satisfies condition (4) of Definition 2.2,
(2.2) (RR_{Ob}, η, rr) satisfies condition (5) of Definition 2.2,
(2.3) (RR_{Ob}, rr) satisfies condition (6) of Definition 2.2.

We have intentionally expanded the definitions of monads in the monoidal form and monads in the Kleisli form to show that that the expanded definition of the latter is much shorter than that of the former. Monads are used extensively in computer science, but almost always in the Kleisli form and the fact that Kleisli form is much more concise than the monoidal form may be one of the reasons.

On the other hand, it is likely that monads in the Kleisli form have not been widely known among mathematicians because they are defined not as functors with a structure, but as functions between sets of objects with a structure. In particular, it is not obvious from their definition that monads in the Kleisli form can be transported along equivalences of categories.

We devote so much attention to these two definitions because they provide one of the clearest examples of how the same objects can have different and often mutually incomprehensible definitions in the parallel realities of mathematics and that part of theoretical computer science which is known as Theoretical Computer Science B.

We now outline the construction for Problem 2.3

Construction 2.4. **[2017.01.04.constr1]** To go from the monoidal form to the Kleisli form, one defines, for $X, Y \in Ob(\mathcal{C})$ and $f : X \to RR_{Ob}(Y)$

$$[\mathbf{2017.04.17.eq1}] rr_{X,Y}(f) = RR_{Mor}(f) \circ \mu_Y \tag{13}$$

To go from the Kleisli form to the monoidal form one defines

(1) for $f : X \to Y$ in $Mor(\mathcal{C})$

$$[\mathbf{2017.04.17.eq2}] RR_{Mor}(f) = rr_{X,Y}(f \circ \eta_Y) \tag{14}$$

(2) for $X \in Ob(\mathcal{C})$

$$[\mathbf{2017.04.17.eq2}] \mu_X = rr_{RR_{Ob}(X),X}(Id_{RR_{Ob}(X)}) \tag{15}$$

We leave the verification of the conditions and the proof that these functions are mutually inverse to the formally verified version of the paper. □

For a monad $\mathbf{RR}$ in the Kleisli form we let $\mathbf{RR}^M$ denote the corresponding monad in the monoidal form and a monad $\mathbf{R}$ in the monoidal form we let $\mathbf{R}^K$ denote the corresponding monad in Kleisli form.

The notion of a relative monad arises very naturally for monads in the Kleisli form. It was introduced in [7, Def.1, p.299] and considered in more detail in [8]. Let us remind it here.

Definition 2.5. ***[2015.12.22.def1]*** *Let $J : \mathcal{C} \to \mathcal{D}$ be a functor. A relative monad* $\mathbf{RR}$ *on J or a J-relative monad or a J-monad is a triple (RR_{Ob}, η, rr) where*

(1) $RR_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{D})$ is a function,
(2) for all X in $\mathcal{C}$, a morphism $\eta_X : J(X) \to RR_{Ob}(X)$ in $\mathcal{D}$,
(3) for all X, Y in $\mathcal{C}$ and $f : J(X) \to RR_{Ob}(Y)$ in $\mathcal{D}$, a morphism in $\mathcal{D}$

$$rr_{X,Y}(f) : RR_{Ob}(X) \to RR_{Ob}(Y)$$

such that the following conditions hold:

(4) for all $X \in \mathcal{C}$, $rr_{X,X}(\eta_X) = Id_{RR_{Ob}(X)}$,
(5) for all $X, Y \in \mathcal{C}$ and $f : J(X) \to RR_{Ob}(Y)$, $\eta_X \circ rr_{X,Y}(f) = f$,

(6) for all $X, Y, Z \in \mathcal{C}$, $f : J(X) \to RR_{Ob}(Y)$ and $g : J(Y) \to RR_{Ob}(Z)$,

$$rr_{X,Y}(f) \circ rr_{Y,Z}(g) = rr_{X,Z}(f \circ rr_{Y,Z}(g))$$

Sometimes one writes f^* instead of $rr_{X,Y}(f)$. It makes long computations look nicer, but one should remember that the notation f^* is under-specified because f^* depends not only on f but also on X and Y and it is possible that for example $RR_{Ob}(Y_1) = RR_{Ob}(Y_2)$ while $Y_1 \neq Y_2$.

Problem 2.6. *[**2016.01.15.prob1**] Given a J-relative monad $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ to construct a function $RR_{Mor} : Mor(\mathcal{C}) \to Mor(\mathcal{D})$ such that $\mathbf{RR}^f = (RR_{Ob}, RR_{Mor})$ is a functor.*

Construction 2.7. [**2016.01.15.constr1**]For $f : X \to Y$ in $\mathcal{C}$ set

$$[\mathbf{2017.04.05.eq3}]RR_{Mor}(f) = rr_{X,Y}(J(f) \circ \eta_Y) \tag{16}$$

The proofs of the composition and the identity axioms of a functor are straightforward. □

An $Id_{\mathcal{C}}$-relative monad will be called an endo-monad. An endo-monad is precisely a monad on the corresponding category given in the Kleisli form. This permits us to call, sometimes, the endo-monads simply monads. As the explicit form of functions $(-)^K$ and $(-)^M$ defined in Construction 2.4 shows that for $\mathbf{R} = (R, \eta, \mu)$ we have $\mathbf{R}^K = (R_{Ob}, \eta, rr)$ and for $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ we have $\mathbf{RR}^M = (\mathbf{RR}^f, \eta, \mu)$. In particular, since $(\mathbf{R}^K)^M = \mathbf{R}$, we have

$$[\mathbf{2017.04.05.eq2}](\mathbf{R}^K)^f = R \tag{17}$$

The construction of the Kleisli category of a monad[3] was extended to the case of relative monads in [8, p.8] (see also [43, Constr. 2.9]). Since it plays an important role in what follows let us remind the definition of the corresponding category data here without proving that it actually defines a category.

Problem 2.8. *[**2017.03.12.prob3**] Given a functor $J : \mathcal{C} \to \mathcal{D}$ and a J-monad $\mathbf{RR} = (RR, \eta, rr)$ to construct a category $K(\mathbf{RR})$ that will be called the Kleisli category of $\mathbf{RR}$.*

Construction 2.9. [**2017.03.12.constr3**] We set $Ob(K(\mathbf{RR})) = Ob(\mathcal{C})$ and

$$Mor(K(\mathbf{RR})) = \amalg_{X,Y \in K(\mathbf{RR})} Mor(J(X), RR(Y))$$

For $X, Y \in K(\mathbf{RR})$, we will identify the set of morphisms in $K(\mathbf{RR})$ from X to Y with the set $Mor(J(X), RR(Y))$ by means of the obvious bijections.

For $X \in Ob(\mathcal{C})$ we set $Id_{X,K(\mathbf{RR})} = \eta_X$.

For $f \in Mor_{\mathcal{D}}(J(X), RR(Y))$ and $g \in Mor_{\mathcal{D}}(J(Y), RR(Z))$ we set

$$[\mathbf{2017.04.05.eq1}] f \circ_{K(\mathbf{RR})} g := f \circ_{\mathcal{D}} rr_{Y,Z}(g) \tag{18}$$

□

[3] Actually Kleisli, in [25], introduced the corresponding category for what we would today call a comonad.

Problem 2.10. *[**2017.04.05.prob2**] In the context of Problem 2.8, to construct a functor*

$$Et_{\mathbf{RR}} : \mathcal{C} \to K(\mathbf{RR})$$

Construction 2.11. [**2017.04.05.constr2**] We omit the index **RR** at Et. We set $Et_{Ob} = Id$. For a morphism $f : X \to Y$ in $\mathcal{C}$ we set

$$[\mathbf{2017.04.09.eq2}]Et_{Mor}(f) = J(f) \circ \eta_Y \tag{19}$$

For $f = Id_X$ we have

$$Et(Id_X) = J(Id_X) \circ \eta_X = Id_{J(X)} \circ \eta_X = \eta_X = Id_{X,K(\mathbf{RR})}$$

This proves the identity axiom. For $f : X \to Y$, $g : Y \to Z$ we have

$$Et(f \circ g) = J(f \circ g) \circ \eta_Z = J(f) \circ J(g) \circ \eta_Z$$

and

$$Et(f) \circ Et(g) = (J(f) \circ \eta_Y) \circ_{K(\mathbf{RR})} (J(g) \circ \eta_Z) =$$
$$J(f) \circ \eta_Y \circ rr_{Y,Z}(J(g) \circ \eta_Z) = J(f) \circ J(g) \circ \eta_Z$$

where the second equality is by (18) and the third by condition (2) of Definition 2.5. This proves the composition axiom. □

Problem 2.12. *[**2017.04.11.prob1**] Let $J : \mathcal{C} \to \mathcal{D}$ be a functor and $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ a J-relative monad. To construct a functor $\mathbf{RR}^{lm} : K(\mathbf{RR}) \to \mathcal{D}$.*

Construction 2.13. [**2017.04.09.constr1**] We set $\mathbf{RR}^{lm}_{Ob} = RR_{Ob}$. For

$$f \in Mor_{K(\mathbf{RR})}(X, Y) = Mor_{\mathcal{D}}(J(X), RR_{Ob}(Y))$$

where the equality is actually the bijection mentioned in the definition of $K(\mathbf{RR})$, we set $\mathbf{RR}^{lm}_{Mor}(f) = rr_{X,Y}(f)$.

The verification of the composition and identity axioms for $(\mathbf{RR}^{lm}_{Ob}, \mathbf{RR}^{lm}_{Mor})$ is straightforward and is left for the formalized version of the paper. □

So far our only examples of relative monad were provided by the "usual" monads in the Kleisli form, in particular, they all were endo-monads. The following construction allows one to obtain a large class of relative monads that are not endo-monads.

Problem 2.14. *[**2017.02.24.prob1**] Given functors $F : \mathcal{C}_0 \to \mathcal{C}_1$, $J : \mathcal{C}_1 \to \mathcal{D}$ and a J-relative monad $\mathbf{RR}$ to construct a $(F \circ J)$-relative monad $F^{\circ}(\mathbf{RR})$.*

Construction 2.15. [**2017.02.24.constr1**]We omit the indexes Ob and Mor at F and J. Let $\mathbf{RR} = (RR_{Ob}, \eta, rr)$. We set

$$[\mathbf{2017.04.11.eq3}]F^{\circ}(\mathbf{RR}) = (F^{\circ}(RR_{Ob}), F^{\circ}(\eta), F^{\circ}(rr)) \tag{20}$$

where, for for $X \in \mathcal{C}_0$

$$[\mathbf{2017.04.11.eq1}]F^{\circ}(RR_{Ob})(X) = RR_{Ob}(F(X)) \qquad F^{\circ}(\eta)_X = \eta_{F(X)} \tag{21}$$

and for $X, Y \in \mathcal{C}_0$ and $f : J(F(X)) \to RR_{Ob}(F(Y))$,

$$[\mathbf{2017.04.11.eq2}]F^{\circ}(rr)_{X,Y}(f) = rr_{F(X),F(Y)}(f) \tag{22}$$

Let us verify conditions (1)-(3) of Definition 2.5. We write RR instead of RR_{Ob} :

(1) Let $X \in \mathcal{C}_0$. Then
$$F^{\circ}(rr)_{X,X}(F^{\circ}(\eta)_X) = rr_{F(X),F(X)}(\eta_{F(X)}) = Id_{RR(F(X))} = Id_{F^{\circ}(RR)(X)}$$
(2) Let $X, Y \in \mathcal{C}_0$ and $f : J(F(X)) \to RR(F(Y))$. Then
$$F^{\circ}(\eta)_X \circ F^{\circ}(rr)_{X,Y}(f) = \eta_{F(X)} \circ rr_{F(X),F(Y)}(f) = f$$
(3) Let $X, Y, Z \in \mathcal{C}_0$ and
$$f : J(F(X)) \to RR(F(Y)) \qquad g : J(F(Y)) \to RR(F(Z))$$
are morphisms in $\mathcal{D}$. Then
$$F^{\circ}(rr)_{X,Y}(f) \circ F^{\circ}(rr)_{Y,Z}(g) = rr_{F(X),F(Y)}(f) \circ rr_{F(Y),F(Z)}(g) =$$
$$rr_{F(X),F(Z)}(f \circ rr_{F(Y),F(Z)}(g)) = F^{\circ}(rr)_{X,Z}(f \circ F^{\circ}(rr)_{Y,Z}(g))$$

This completes Construction 2.15. □

Lemma 2.16. **[2017.04.09.l1]** *In the context of Problem 2.14 one has*

(23) $$[\mathbf{2017.04.17.eq6}](F^{\circ}(\mathbf{RR}))^f = F \circ RR^f$$

Proof. In what follows we omit the indexes Ob and Mor at F and J. Both the left and right hand side of (23) are functors $\mathcal{C}_0 \to \mathcal{D}$. Let $\mathbf{RR} = (RR_{Ob}, \eta, rr)$. Then both of these functors on objects are given, by construction, by $F \circ RR_{Ob}$. It remains to show that for $f :\to Y$ in $\mathcal{C}_0$ we have

(24) $$[\mathbf{2017.04.09.eq5}]F^{\circ}(\mathbf{RR})_{Mor}(f) = (F_{Mor} \circ RR_{Mor})(f)$$

We have
$$F^{\circ}(\mathbf{RR})_{Mor}(f) = F^{\circ}(rr)((F \circ J)(f) \circ F^{\circ}(\eta)_Y) = rr_{F(X),F(Y)}(J(F(f)) \circ \eta_{F(Y)})$$
where the first equality is by (16) and (20) and the second by (21) and (22). On the other hand
$$(F_{Mor} \circ RR_{Mor})(f) = RR_{Mor}(F(f)) = rr_{F(X),F(Y)}(J(F(f)) \circ \eta_{F(Y)})$$
where the second equality is by (16). This completes the proof of the lemma. □

Remark 2.17. **[2017.04.09.rem1]** Note that in the construction of $F^{\circ}(\mathbf{RR})$ there participate only F_{Ob}, but not F_{Mor}. On the other hand we have (24). The explanation for this seeming contradiction is that $F^{\circ}(\mathbf{RR})$ is a $(F \circ J)$-relative monad and $(F \circ J)_{Mor}$, and, therefore F_{Mor}, participates in the definition of $F^{\circ}(\mathbf{RR})_{Mor}$.

Problem 2.18. **[2017.03.12.prob1]** *Given functors $F : \mathcal{C}_0 \to \mathcal{C}_1$ and $J : \mathcal{C}_1 \to \mathcal{D}$, and a J-monad $\mathbf{RR}$ to construct a functor $F_{\mathbf{RR}} : K(F^{\circ}(\mathbf{RR})) \to K(\mathbf{RR})$.*

Construction 2.19. **[2017.03.12.constr2]** We set $F_{\mathbf{RR},Ob} = F_{Ob}$. For $X, Y \in \mathcal{C}_0$ and a morphism $f : J(F(X)) \to RR(F(Y))$ in $K(F^{\circ}(\mathbf{RR}))$ from X to Y, we set

(25) $$[\mathbf{2017.03.12.eq2}]F_{\mathbf{RR},X,Y}(f) = f$$

that is, $F_{\mathbf{RR},Mor}$ is given by the function
$$\amalg_{X_0,Y_0 \in \mathcal{C}_0} Mor_{\mathcal{D}}(J(F(X_0)), RR(F(Y_0))) \to \amalg_{X_1,Y_1 \in \mathcal{C}_1} Mor_{\mathcal{D}}(J(X_1), RR(Y_1))$$
of the form $((X_0, Y_0), f) \mapsto ((F(X_0), F(Y_0)), f)$.

Let us show that $(F_{\mathbf{RR},Ob}, F_{\mathbf{RR},Mor})$ is a functor.

Note first that for $X \in \mathcal{C}_0$ one has

$$[\mathbf{2017.03.12.eq1}] Id_{K(F^{\circ}(\mathbf{RR})),X} = F^{\circ}(\eta)_X = \eta_{F(X)} \tag{26}$$

where the first equality is by Construction 2.9 and the second by Construction 2.15. The data $(F_{\mathbf{RR},Ob}, F_{\mathbf{RR},Mor})$ satisfies the identity axiom because of the equalities

$$F_{\mathbf{RR},X,X}(Id_{K(F^{\circ}(\mathbf{RR})),X}) = F_{\mathbf{RR},X,X}(\eta_{F(X)}) = \eta_{F(X)} = Id_{K(\mathbf{RR}),F(X)}$$

where the first equality is by (26), the second by (25) and the third by Construction 2.9.

Next, for $X, Y, Z \in \mathcal{C}_0$ and $f : J(F(X)) \to RR(F(Y))$, $g : J(F(Y)) \to RR(F(Z))$ one has

$$[\mathbf{2017.03.12.eq3}] f \circ_{KK(F^{\circ}(\mathbf{RR}))} g = f \circ_{\mathcal{D}} F^{\circ}(rr)_{Y,Z}(g) = f \circ_{\mathcal{D}} rr_{F(Y),F(Z)}(g) \tag{27}$$

where, again, the first equality is by Construction 2.9 and the second by Construction 2.15. The data $(F_{\mathbf{RR},Ob}, F_{\mathbf{RR},Mor})$ satisfies the composition axiom because of the equalities

$$F_{\mathbf{RR},X,Z}(f \circ_{K(F^{\circ}(\mathbf{RR}))} g) = F_{\mathbf{RR},X,Z}(f \circ_{\mathcal{D}} rr_{F(Y),F(Z)}(g)) =$$
$$f \circ_{\mathcal{D}} rr_{F(Y),F(Z)}(g) = f \circ_{K(\mathbf{RR})} g = F_{\mathbf{RR},X,Y}(f) \circ_{K(RR)} F_{\mathbf{RR},Y,Z}(g)$$

where the first equality is by (27), the second by (25), the third by Construction 2.9 and the fourth again by (25).

This completes Construction 2.19. □

The following construction together with Construction **??** are used to extract pairs $(\mathbf{RR}, \mathbf{LM})$ of a relative monad and a left module over it from two-sorted relative monads such as the ones that arise from two-sorted second-order signatures, see [6].

For two functors $F : \mathcal{C} \to \mathcal{D}$, $F' : \mathcal{C}' \to \mathcal{D}'$ we let $F \boxtimes F'$ denote the corresponding functor from $\mathcal{C} \times \mathcal{C}'$ to $\mathcal{D} \times \mathcal{D}'$. If $\mathcal{C} = \mathcal{C}'$ we let $F \times F'$ denote the composition of $F \boxtimes F'$ with the diagonal $\mathcal{C} \to \mathcal{C} \times \mathcal{C}$, that is, the functor given on objects by $X \mapsto (F(X), F'(X))$. For a functor $FF : \mathcal{C} \to \mathcal{D} \times \mathcal{D}'$ we let $FF_{\mathcal{D}}$ and $FF_{\mathcal{D}'}$ denote the compositions of FF with the projections from $\mathcal{D} \times \mathcal{D}'$ to $\mathcal{D}$ and $\mathcal{D}'$ respectively. We will also use the notation $FF_{\mathcal{D}}$ when FF is only a function on objects.

Definition 2.20. *[***2017.04.21.def1***] An object constancy structure on a functor $F : \mathcal{C} \to \mathcal{D}$ is a family, parametrized by $X, Y \in \mathcal{C}$, of morphisms $\nu_{X,Y} : F(X) \to F(Y)$ such that:*

1. *[***2017.04.21.eq3***] for all X, $\nu_{X,X} = Id_{F(X)}$,*
2. *[***2017.04.21.eq4***] for all X, Y, Z, $\nu_{X,Z} = \nu_{X,Y} \circ \nu_{Y,Z}$.*

Note that for an object constancy structure we have, for all $X, Y \in \mathcal{C}$,

$$\nu_{X,Y} \circ \nu_{Y,X} = \nu_{X,X} = Id_{F(X)}$$

$$\nu_{Y,X} \circ \nu_{X,Y} = \nu_{Y,Y} = Id_{F(Y)}$$

that is, $\nu_{X,Y}$ and $\nu_{Y,X}$ are mutually inverse and, in particular, $\nu_{X,Y}$ is an isomorphism.

For any object A of $\mathcal{D}$ the functor $\underline{A}_{\mathcal{C}} : \mathcal{C} \to \mathcal{D}$ given on objects by $\underline{A}_{\mathcal{C}}(X) = A$ and on morphisms by $\underline{A}_{\mathcal{C}}(f) = Id_A$ has an obvious object constancy structure with

$\nu_{X,Y} = Id_A$ for all X, Y. We will call it the identity object constancy structure corresponding to A.

Since the object constancy structure does not impose any condition on ν with respect to morphisms of $\mathcal{C}$ or $\mathcal{D}$, there are many other examples. Indeed, for A as above, the identity object constancy structure is defined for any functor F, however non-trivial on morphisms, but such that $F(X) = A$ for all $X \in \mathcal{C}$.

Problem 2.21. *[***2017.04.21.prob1***] Let $\mathcal{C}, \mathcal{D}, \mathcal{D}'$ be categories, $J : \mathcal{C} \to \mathcal{D} \times \mathcal{D}'$ a functor and $\mathbf{RR} = (RR, \eta, rr)$ a J-relative monad. Assume in addition that we are given an object constancy structure ν on $J_{\mathcal{D}'} : \mathcal{C} \to \mathcal{D}'$.*

To construct a structure of a $J_{\mathcal{D}}$-monad on $RR_{\mathcal{D}}$. This $J_{\mathcal{D}}$-monad will be denoted $\mathbf{RR}_{\mathcal{D}}$.

Construction 2.22. [**2017.04.21.constr1**] We need to construct:

(1) a family $\eta^{\mathcal{D}}$, parametrized by $X \in \mathcal{C}$, of morphisms $\eta^{\mathcal{D}}_X : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(X)$,
(2) a family parametrized by $X, Y \in \mathcal{C}$ and $f : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$, of morphisms $rr^{\mathcal{D}}_{X,Y}(f) : RR_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$,

and to prove that $\mathbf{RR}_{\mathcal{D}}$, defined as $(RR_{\mathcal{D}}, \eta^{\mathcal{D}}, rr^{\mathcal{D}})$, satisfies conditions (4), (5) and (6) of Definition 2.2.

Let $X \in \mathcal{C}$. Then η_X is a morphism from $J(X)$ to $RR(X)$. We set

$$[\mathbf{2017.04.21.eq1}]\eta^{\mathcal{D}}_X = pr_{\mathcal{D}}(\eta_X) : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(X) \tag{28}$$

We will also consider $pr_{\mathcal{D}'}(\eta_X)$ that we will denote by $\eta^{\mathcal{D}'}_X$.

Let $X, Y \in \mathcal{C}$ and $f : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$. We need to construct a morphism $RR_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$. We have $\nu_{X,Y} : J_{\mathcal{D}'}(X) \to J_{\mathcal{D}'}(Y)$ and therefore we may consider the morphism

$$(f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y) : (J_{\mathcal{D}}(X), J_{\mathcal{D}'}(X)) = J(X) \to RR(Y) = (RR_{\mathcal{D}}(Y), RR_{\mathcal{D}'}(Y))$$

and applying to it $rr_{X,Y}$ we obtain

$$rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y)) : RR(X) \to RR(Y)$$

We set

$$[\mathbf{2017.04.21.eq2}]rr^{\mathcal{D}}_{X,Y}(f) = pr_{\mathcal{D}}(rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y))) \tag{29}$$

Let $X \in \mathcal{C}$. Then

$$[\mathbf{2017.04.23.eq1}]rr_{X,X}((\eta^{\mathcal{D}}_X, \nu_{X,X} \circ \eta^{\mathcal{D}'}_X)) = rr_{X,X}((\eta^{\mathcal{D}}_X, \eta^{\mathcal{D}'}_X)) = rr_{X,X}(\eta_X) = Id_{RR(X)} \tag{30}$$

where the first equality holds by Definition 2.20(1), the second by definition of $\eta^{\mathcal{D}}$ and $\eta^{\mathcal{D}'}$, and the fourth by the property 2.2(4) of $\mathbf{RR}$.

Therefore,

$$rr^{\mathcal{D}}_{X,X}(\eta^{\mathcal{D}}_X) = pr_{\mathcal{D}}(rr_{X,X}((\eta^{\mathcal{D}}_X, \nu_{X,X} \circ \eta^{\mathcal{D}'}_X))) = pr_{\mathcal{D}}(Id_{RR(X)}) = Id_{RR_{\mathcal{D}}(X)}$$

where the first equality holds by (29), the second by (30) and the third by the definition of $pr_{\mathcal{D}}$. This proves the property 2.2(4) for $\mathbf{RR}_{\mathcal{D}}$.

Let $X, Y \in \mathcal{C}$ and $f : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$. Then

$$[\mathbf{2017.04.23.eq2}]\eta^{\mathcal{D}}_X \circ rr^{\mathcal{D}}_{X,Y}(f) =$$

$$pr_{\mathcal{D}}(\eta_X) \circ pr_{\mathcal{D}}(rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y))) = pr_{\mathcal{D}}(\eta_X \circ rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y))) =$$

$$pr_{\mathcal{D}}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y)) = f \tag{31}$$

where the first equality holds by (28) and (29), the second since $pr_{\mathcal{D}}$ commutes with compositions, the third by the property 2.2(5) of $\mathbf{RR}$, and the fourth by definition of $pr_{\mathcal{D}}$. This proves the property 2.2(5) for $\mathbf{RR}_{\mathcal{D}}$.

Let $X, Y, Z \in \mathcal{C}$, $f : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$, $g : J_{\mathcal{D}}(Y) \to RR_{\mathcal{D}}(Z)$. Then

$$rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y)) \circ rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z)) =$$

$$rr_{X,Z}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y) \circ rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))) =$$

$$rr_{X,Z}((f \circ pr_{\mathcal{D}}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))), \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y \circ pr_{\mathcal{D}'}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z)))))) =$$

$$rr_{X,Z}((f \circ pr_{\mathcal{D}}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))), \nu_{X,Y} \circ pr_{\mathcal{D}'}(\eta_Y \circ rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))))) =$$

$$rr_{X,Z}((f \circ pr_{\mathcal{D}}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))), \nu_{X,Y} \circ pr_{\mathcal{D}'}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z)))) =$$

$$rr_{X,Z}((f \circ pr_{\mathcal{D}}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))), \nu_{X,Y} \circ \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z)) =$$

$$rr_{X,Z}((f \circ pr_{\mathcal{D}}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))), \nu_{X,Z} \circ \eta^{\mathcal{D}'}_Z))) =$$

$$[\mathbf{2017.04.21.eq5}]rr_{X,Z}((f \circ rr^{\mathcal{D}}_{Y,Z}(g), \nu_{X,Z} \circ \eta^{\mathcal{D}'}_Z)) \tag{32}$$

where the first equality holds by the property 2.2(6) of $\mathbf{RR}$, the second by the definition of composition in $\mathcal{D} \times \mathcal{D}'$, the third since $pr_{\mathcal{D}'}$ commutes with compositions, the fourth by the property 2.2(5) of $\mathbf{RR}$, the fifth by the definition of $pr_{\mathcal{D}'}$, the sixth by Definition 2.20(2), and the seventh by (29).

Therefore,

$$rr^{\mathcal{D}}_{X,Y}(f) \circ rr^{\mathcal{D}}_{Y,Z}(g) = pr_{\mathcal{D}}(rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y))) \circ pr_{\mathcal{D}}(rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))) =$$

$$pr_{\mathcal{D}}(rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y)) \circ rr_{Y,Z}((g, \nu_{Y,Z} \circ \eta^{\mathcal{D}'}_Z))) =$$

$$pr_{\mathcal{D}}(rr_{X,Z}((f \circ rr^{\mathcal{D}}_{Y,Z}(g), \nu_{X,Z} \circ \eta^{\mathcal{D}'}_Z))) =$$

$$rr^{\mathcal{D}}_{X,Z}(f \circ rr^{\mathcal{D}}_{Y,Z}(g))$$

where the first equality holds by (29), the second since $pr_{\mathcal{D}}$ commutes with compositions, the third by (32), and the fourth again by (29). This proves the property 2.2(6) for $\mathbf{RR}_{\mathcal{D}}$ and completes Construction 2.22. □

We want to emphasize the following particular case of Construction 2.22.

Problem 2.23. *[***2017.04.23.prob1***] Let $\mathcal{C}$,$\mathcal{D}$ be categories. Let* $\mathbf{RR}$ *be an endo-monad on $\mathcal{C} \times \mathcal{D}$. Let $A \in Ob(D)$. To construct an endo-monad* $\mathbf{RR}_{1,A}$ *on $\mathcal{C}$.*

Construction 2.24. **[2017.04.23.constr1]** Let $J : \mathcal{C} \to \mathcal{C} \times \mathcal{D}$ be the functor given by $X \mapsto (X, A)$ on objects and $f \mapsto (f, Id_A)$ on morphisms. Consider the J-monad $J^{\circ}(\mathbf{RR})$. It satisfies the conditions of Problem 2.21 with respect to the identity object constancy structure corresponding to A. Therefore, Constructions 2.22 applies and we obtain a $J \circ pr_{\mathcal{C}}$-monad $(J^{\circ}(\mathbf{RR}))^{\mathcal{D}}$. Since $J \circ pr_{\mathcal{C}} = Id_{\mathcal{C}}$ it is an endo-monad on $\mathcal{C}$ that we denote by $\mathbf{RR}_{1,A}$. Explicitly, for $\mathbf{RR} = (RR, \eta, rr)$, we have $\mathbf{RR}_{1,A} = (RR_{1,A}, \eta^{1,A}, rr^{1,A})$ where:

(1) for $X \in \mathcal{C}$, $RR_{1,A}(X) = pr_{\mathcal{C}}(RR((X, A)))$,
(2) for $X \in \mathcal{C}$, $\eta^{1,A}_X = pr_{\mathcal{C}}(\eta_{(X,A)})$,
(3) for $X, Y \in \mathcal{C}$, $f : X \to Y$, $rr^{1,A}_{X,Y}(f) = pr_{\mathcal{C}}(rr_{(X,A),(Y,A)}(f, pr_{\mathcal{D}}(\eta_{(Y,A)})))$.

□

Remark 2.25. The notation $\mathbf{RR}_{1,A}$ emphasizes that the monad is on the first projection of the product $\mathcal{C} \times \mathcal{D}$. We can also construct, for $A \in Ob(C)$, an endo-monad $\mathbf{RR}_{2,A}$ on $\mathcal{D}$.

2.2. **Left modules over monads and relative monads.** A left module in the monoidal form over a monad in the monoidal form is defined as follows (cf. [19, p.222]).

Definition 2.26. **[2017.04.01.def2]** *Let $\mathcal{C}$,$\mathcal{E}$ be categories, $\mathbf{R} = (R, \eta, \mu)$ be a monad on $\mathcal{C}$, and $L : \mathcal{C} \to \mathcal{E}$ a functor. A (left) $\mathbf{R}$-module structure on L is a natural transformation $\rho : R \circ L \to L$ such that for all $X \in \mathcal{C}$ one has:*

(1) $L(\mu_X) \circ \rho_X = \rho_{R(X)} \circ \rho_X$,
(2) $L(\eta_X) \circ \rho_X = Id_{L(X)}$.

A left $\mathbf{R}$-module in the monoidal form with values in $\mathcal{E}$ is a pair $\mathbf{L} = (L, \rho)$ where $L : \mathcal{C} \to \mathcal{E}$ is a functor and ρ an $\mathbf{R}$-module structure on L.

Example 2.27. **[2017.04.15.ex1]** For a monad $\mathbf{R} = (R, \eta, \rho)$ the pair $\mathbf{R}^{lm} = (R, \rho)$ is a left module over $\mathbf{R}$.

Left modules can also be defined in Kleisli form.

Definition 2.28. **[2017.04.15.def1]** *Let $\mathcal{C}$,$\mathcal{E}$ be categories and $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ a monad on $\mathcal{C}$. A (left) $\mathbf{RR}$-module with values in $\mathcal{E}$ in the Kleisli form is a pair (LM_{Ob}, lm) where $LM_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{E})$ is a function and lm is a family, parametrized by $X, Y \in Ob(\mathcal{C})$ of functions*

$$lm_{X,Y} : Mor_{\mathcal{C}}(X, RR_{Ob}(Y)) \to Mor_{\mathcal{E}}(LM_{Ob}(X), LM_{Ob}(Y))$$

such that

(1) for all $X \in \mathcal{C}$, $lm_{X,X}(\eta_X) = Id_{LM_{Ob}(X)}$,
(2) for all $X, Y, Z \in \mathcal{C}$, $f : X \to RR_{Ob}(Y)$, $g : Y \to RR_{Ob}(Z)$,

$$lm_{X,Y}(f) \circ lm_{Y,Z}(g) = lm_{X,Z}(f \circ rr_{Y,Z}(g))$$

As in the case of monads the monoidal and Kleisli forms of left modules are equivalent in the following sense.

Problem 2.29. *[**2017.04.03.prob1**] Given categories $\mathcal{C}$, $\mathcal{E}$, a monad $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ on $\mathcal{C}$, and a function $LM_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{E})$ to construct a bijection between the following two sets:*

1. *the set of pairs (LM_{Mor}, ρ) where $LM_{Mor} : Mor(\mathcal{C}) \to Mor(\mathcal{E})$ is a function such that $LM = (LM_{Ob}, LM_{Mor})$ is a functor and (LM, ρ) is a left $\mathbf{RR}^M$-module in the monoidal form,*
2. *the set of families lm, parametrized by $X, Y \in Ob(\mathcal{C})$, of functions*
$$lm_{X,Y} : Mor_{\mathcal{C}}(X, RR_{Ob}(Y)) \to Mor_{\mathcal{C}}(LM_{Ob}(X), LM_{Ob}(Y))$$
such that (LM_{Ob}, lm) is a left $\mathbf{RR}$-module in the Kleisli form.

Construction 2.30. [**2017.04.03.constr1**] In one direction, given LM_{Mor}, a family $\rho_X : LM(RR_{Ob}(X)) \to LM(X)$ parametrized by $X \in Ob(\mathcal{C})$, and $f : X \to R(Y)$, one defines

$$[\mathbf{2017.04.09.eq4}] lm_{X,Y}(f) = LM(f) \circ \rho_Y \tag{33}$$

In the other direction, given LM_{Ob} and lm, one defines, for $f : X \to Y$,

$$[\mathbf{2017.04.09.eq3}] LM_{Mor}(f) = lm_{X,Y}(f \circ \eta_Y) \tag{34}$$

and for X,

$$\rho_X = lm_{RR_{Ob}(X),X}(Id_{RR_{Ob}(X)})$$

We leave the verification of the conditions and the proof that these functions are mutually inverse to the formally verified version of the paper. □

Left $\mathbf{RR}$-modules in the Kleisli form with values in $\mathcal{E}$ are precisely the (covariant) functors from the Kleisli category of $\mathbf{RR}$ to $\mathcal{E}$, see below.

Left modules over relative monads were introduced in [3, Definition 9]. One can observe by direct comparison of unfolded definitions that there is a bijection between the set of modules over a relative monad $\mathbf{RR}$ with values in a category $\mathcal{E}$ and the set of functors from the Kleisli category $K(\mathbf{RR})$ of $\mathbf{RR}$ to $\mathcal{E}$. Whether this bijection is the identity bijection or not depends on how the expressions such as "collection of data" or "family of functions" are translated into the formal constructions of set theory. We assume that in this particular case they have been translated in a such a way that this bijection is the identity and left modules over $\mathbf{RR}$ with values in $\mathcal{E}$ are actually and precisely the same as (covariant) functors from $K(\mathbf{RR})$ to $\mathcal{E}$.

Definition 2.31. *[**2017.03.16.def1**] Let $J : \mathcal{C} \to \mathcal{D}$ be a functor and $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ a J-monad. A left module over $\mathbf{RR}$ with values in a category $\mathcal{E}$ is a functor $\mathbf{LM} : K(\mathbf{RR}) \to \mathcal{E}$, that is, a pair (LM_{Ob}, lm) where $LM_{Ob} : Ob(\mathcal{C}) \to Ob(\mathcal{E})$ is a function and lm is a family, parametrized by $X, Y \in Ob(\mathcal{C})$, of functions*

$$lm_{X,Y} : Mor_{\mathcal{D}}(J(X), RR_{Ob}(Y)) \to Mor_{\mathcal{E}}(LM_{Ob}(X), LM_{Ob}(Y))$$

such that

1. *for all $X \in \mathcal{C}$, $lm_{X,X}(\eta_X) = Id_{LM_{Ob}(X)}$,*
2. *for all $X, Y, Z \in \mathcal{C}$, $f : X \to RR_{Ob}(Y)$, $g : Y \to RR_{Ob}(Z)$,*
$$lm_{X,Y}(f) \circ lm_{Y,Z}(g) = lm_{X,Z}(f \circ rr_{Y,Z}(g))$$

We will say that **LM** *is an* **RR***-module if it is a left* **RR***-module. We will say that* **LM** *is a module over* **RR** *without specifying* $\mathcal{E}$ *if* $\mathcal{E} = \mathcal{D}$*.*

From the unfolded definition we see that the left modules over an Id-relative monad are exactly the same as the left modules in the Kleisli form over the corresponding monad. Following the notation $\mathbf{R}^K$ for the monad in the Kleisli form corresponding to a monad $\mathbf{R}$ in the monoidal form, we let $\mathbf{L}^K$ denote the module over $\mathbf{R}^K$ in the Kleisli form corresponding to a module $\mathbf{L}$ over $\mathbf{R}$ in the monoidal form.

Definition 2.32. **[2017.04.05.def1]** *Let* $J : \mathcal{C} \to \mathcal{D}$ *be a functor,* **RR** *a* J*-monad and* **LM** *a left module over* **RR** *with values in a category* $\mathcal{E}$*. We define the functor* $\mathbf{LM}^f : \mathcal{C} \to \mathcal{E}$ *corresponding to* **LM** *as the composition* $Et_{\mathbf{RR}} \circ \mathbf{LM}$*.*

Explicitly, for $\mathbf{LM} = (LM_{Ob}, lm)$, we have

$$\mathbf{LM}^f_{Ob} = LM_{Ob}$$

which follows from $Et_{\mathbf{RR},Ob} = Id_{Ob(\mathcal{C})}$, and for $f : X \to Y$

$$[\mathbf{2017.04.11.eq5}]\mathbf{LM}^f_{Mor}(f) = lm(J(f) \circ \eta_Y) \tag{35}$$

which follows from (19).

As in the case of $\mathbf{RR}^f$ we will use the notation $\mathbf{LM}^f_{Ob}$ and $\mathbf{LM}^f_{Mor}$, with or without the subscripts Ob and Mor as our preferential notation for the corresponding objects.

If $\mathbf{R}$ and $\mathbf{L} = (L, \rho)$ are a monad on $\mathcal{C}$ and a left module over it with values in $\mathcal{E}$ given in the monoidal form then we have

$$[\mathbf{2017.04.17.eq4}](\mathbf{L}^K)^f = L \tag{36}$$

On objects we have $(\mathbf{L}^K)^f_{Ob} = L_{Ob}$ by construction. It remains to show that

$$[\mathbf{2017.04.09.eq1}](\mathbf{L}^K)^f_{Mor} = L_{Mor} \tag{37}$$

Indeed, for $f : X \to Y$ in $Mor(\mathcal{C})$ we have

$$(\mathbf{L}^K)^f_{Mor}(f) = \mathbf{L}^K_{Mor}(Et_{\mathbf{R}^K,Mor}(f)) =$$
$$\mathbf{L}^K_{Mor}(f \circ \eta_Y) = L_{Mor}(f \circ \eta_Y) \circ \rho_Y = L_{Mor}(f) \circ (L_{Mor}(\eta_Y) \circ \rho_Y) =$$
$$L_{Mor}(f) \circ Id_{L(Y)} = L_{Mor}(f)$$

where the first equality is by Definition 2.32, the second by (19), the third by (33), the fourth by the composition axiom for L and associativity of composition of $\mathcal{E}$, the fifth by Definition 2.26(1) and the sixth by the right unity axiom of $\mathcal{E}$.

Example 2.33. **[2017.04.15.ex2]** Construction 2.13 gives us, for any $J : \mathcal{C} \to \mathcal{D}$ and any J-monad **RR** a left module $\mathbf{RR}^{lm}$ over **RR** with values in $\mathcal{D}$. If $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ then $\mathbf{RR}^{lm} = (RR_{Ob}, rr)$. This is the same relationship as in the case of monads in the monoidal form where for $\mathbf{R} = (R, \eta, \mu)$ we have $\mathbf{R}^{lm} = (R, \mu)$.

We have

$$(\mathbf{RR}^{lm})^f = \mathbf{RR}^f$$

Indeed, for $\mathbf{RR} = (RR_{Ob}, \eta, \mu)$ both functors are given by RR_{Ob} on objects and on morphisms they also coincide by construction because (35) becomes (16) when $lm = rr$.

When $J = Id_{\mathcal{C}}$ we also have

$$(\mathbf{R}^K)^{lm} = (\mathbf{R}^{lm})^K$$

Indeed, for $\mathbf{R} = (R, \eta, \mu)$ we have

$$(\mathbf{R}^K)^{lm} = (R_{Ob}, \eta, rr(R_{Mor}, \mu))^{lm} = (R_{Ob}, rr(R_{Mor}, \mu))$$

and

$$(\mathbf{R}^{lm})^K = (R, \mu)^K = (R_{Ob}, lm(R_{Mor}, \mu))$$

and $rr(R_{Mor}, \mu)$ and $lm(R_{Mor}, \mu)$ coincide by construction because formulas (13) and (33) become the same when $RR_{Mor} = LM_{Mor}$ and $\mu = \rho$.

Problem 2.34. *[**2017.03.12.prob2**] Given functors $F : \mathcal{C}_0 \to \mathcal{C}_1$ and $J : \mathcal{C}_1 \to \mathcal{D}$, a J-monad $\mathbf{RR}$ and an $\mathbf{RR}$-module $\mathbf{LM}$ with values in $\mathcal{E}$ to construct an $F^\circ(\mathbf{RR})$-module $F^\circ(\mathbf{LM})$ with values in $\mathcal{E}$.*

Construction 2.35. [**2017.03.12.constr1**] We need to construct a functor $K(F^\circ(\mathbf{RR})) \to \mathcal{E}$. We define this functor as the composition $F_{\mathbf{RR}} \circ \mathbf{LM}$, where $F_{\mathbf{RR}}$ is defined in Construction 2.19. Explicitly, for $\mathbf{RR} = (RR_{Ob}, \eta, rr)$ and $\mathbf{LM} = (LM_{Ob}, lm)$, we let $F^\circ(\mathbf{LM}) = (F^\circ(LM_{Ob}), F^\circ(lm))$. In this notation we have

$$[\mathbf{2017.04.17.eq7}] F^\circ(LM_{Ob}) = F \circ LM_{Ob} \tag{38}$$

and

$$[\mathbf{2017.04.17.eq8}] F^\circ(lm)_{X,Y} = lm_{F(X),F(Y)} \tag{39}$$

□

Lemma 2.36. *[**2017.04.17.l1**] In the context of Problem 2.34 we have*

$$[\mathbf{2017.04.13.eq2}] (F^\circ(\mathbf{LM}))^f = F \circ \mathbf{LM}^f \tag{40}$$

Proof. The equality

$$(F^\circ(\mathbf{LM}))^f_{Ob} = (F \circ \mathbf{LM}^f)_{Ob}$$

is by construction and to prove the equality

$$[\mathbf{2017.04.13.eq1}] (F^\circ(\mathbf{LM}))^f_{Mor} = (F \circ \mathbf{LM}^f)_{Mor} \tag{41}$$

we have, for $f : X \to Y$ in $\mathcal{C}_0$,

$$(F^\circ(\mathbf{LM}))^f_{Mor}(f) = F^\circ(lm)_{X,Y}((F \circ J)(f) \circ F^\circ(\eta)_Y) =$$

$$F^\circ(lm)_{X,Y}(J(F(f)) \circ \eta_{F(Y)}) = lm_{F(X),F(Y)}(J(F(f)) \circ \eta_{F(Y)}) =$$

$$\mathbf{LM}^f_{Mor}(F(f)) = (F \circ \mathbf{LM}^f_{Mor})(f)$$

where the first equality is by (35), the second by definition of $F \circ J$ and (21), the third by (39), the fourth by (35) and the fifth by the definition of $F \circ \mathbf{LM}^f_{Mor}$. This completes the proof of Lemma 2.36. □

Combining the previous results we obtain a solution to the following problem that we find convenient to formulate for the future reference.

Problem 2.37. *[2017.04.05.prob1] Let $\mathcal{C}_0, \mathcal{C}_1, \mathcal{E}$ be categories. Let $\mathbf{R} = (R, \eta, \mu)$ be a monad on $\mathcal{C}_1$ and $\mathbf{L} = (L, \rho)$ a left $\mathbf{R}$-module with values in $\mathcal{E}$. Let further $J : \mathcal{C}_0 \to \mathcal{C}_1$ be a functor. To construct a pair of a J-relative monad module with values in $\mathcal{E}$ over it.*

Construction 2.38. **[2017.04.05.constr1]** We take $(J^\circ(\mathbf{R}^K), J^\circ(\mathbf{L}^K))$. □

Note that in the notation of Problem 2.37, we have

$$[\mathbf{2017.04.17.eq9}] J^\circ(\mathbf{R}^K)^f = J \circ (\mathbf{R}^K)^f = J \circ R \tag{42}$$

where the first equality is by (23) and the second by (17). Similarly,

$$[\mathbf{2017.04.17.eq10}] J^\circ(^K)^f = J \circ (\mathbf{L}^K)^f = J \circ L \tag{43}$$

where the second equality is by (40) and the third by (36).

Constructions 2.15 and 2.34 show that both relative monads and left modules over them can be "precomposed" with any functor. The left modules can also be "post-composed" with any functor. It is done by literal post-composition. Since a left module is a functor $\mathbf{LM} : K(\mathbf{RR}) \to \mathcal{E}$ we can post-compose it with any functor $F : \mathcal{E} \to \mathcal{E}'$ and obtain a new module that we will denote $\mathbf{LM} \circ F$. Explicitly, for $\mathbf{LM} = (LM_{Ob}, lm)$ one has

$$\mathbf{LM} \circ F = (LM_{Ob} \circ F, F(lm))$$

where, for $X, Y \in \mathcal{C}$ and $f : J(X) \to RR_{Ob}(Y)$, one has

$$F(lm)_{X,Y}(f) = F(lm_{X,Y}(f))$$

There is an analog of Construction 2.22 for modules.

Problem 2.39. *[2017.04.23.prob2] Let $\mathcal{C}, \mathcal{D}, \mathcal{D}'$ be categories, $J : \mathcal{C} \to \mathcal{D} \times \mathcal{D}'$ a functor and $\mathbf{RR} = (RR, \eta, rr)$ a J-relative monad. Assume in addition that we are given an object constancy structure ν on $J_{\mathcal{D}'} : \mathcal{C} \to \mathcal{D}'$. Let $\mathbf{RR}_{\mathcal{D}}$ be the $J_{\mathcal{D}}$-monad specified in Construction 2.22.*

To construct a structure of a $\mathbf{RR}_{\mathcal{D}}$-module with values in $\mathcal{D} \times \mathcal{D}'$ on RR.

Construction 2.40. **[2017.04.23.constr2]** We need to construct a family $lm^{\mathcal{D}}$, parametrized by $X, Y \in \mathcal{C}$ and $f : J_{\mathcal{D}}(X) \to RR_{\mathcal{D}}(Y)$, of morphisms $lm^{\mathcal{D}}_{X,Y}(f) : RR(X) \to RR(Y)$, and to prove that $\mathbf{LM}_{\mathcal{D}}$, defined as $(RR, lm^{\mathcal{D}})$, satisfies conditions (1) and (2) of Definition 2.28.

We set:

$$lm^{\mathcal{D}}_{X,Y}(f) = rr_{X,Y}((f, \nu_{X,Y} \circ \eta^{\mathcal{D}'}_Y))$$

The proof of condition (1) of Definition 2.28 is given by (30). The proof of condition (2) is given by (32). This completes Construction 2.40. □

We also have an analog of the special case described in Construction 2.24. Let $\mathcal{C}$,$\mathcal{D}$ be categories and $A \in Ob(\mathcal{D})$. Let $J_{1,A} : \mathcal{C} \to \mathcal{C} \times \mathcal{D}$ be the functor given by $X \mapsto (X, A)$ on objects and $f \mapsto (f, Id_A)$ on morphisms. In Construction 2.24 we wrote J instead of $J_{1,A}$.

Problem 2.41. *[2017.04.23.prob3] Let $\mathcal{C}$,$\mathcal{D}$ be categories. Let $\mathbf{RR} = (RR, \eta, rr)$ be an endo-monad on $\mathcal{C} \times \mathcal{D}$. Let $A \in Ob(D)$. To construct an $\mathbf{RR}_{1,A}$-module structure on $J_{1,A} \circ RR$.*

Construction 2.42. *[2017.04.23.constr3]* Consider the $J_{1,A}$-monad $J^{\circ}_{1,A}(\mathbf{RR})$. It satisfies the conditions of Problem 2.39 with respect to the identity object constancy structure corresponding to A. Therefore, Constructions 2.40 applies and we obtain a $(J^{\circ}_{1,A}(\mathbf{RR}))_{\mathcal{C}}$-module structure on $J_{1,A} \circ RR$. Since, by Construction 2.24, we have $\mathbf{RR}_{1,A} = (J^{\circ}_{1,A}(\mathbf{RR}))_{\mathcal{C}}$, this provides a construction for Problem 2.41. Explicitly, if we let $(J_{1,A} \circ RR, lm^{1,A})$ denote this module, we have, for $X, Y \in \mathcal{C}$ and $f : X \to Y$

$$lm^{1,A}_{X,Y}(f) = rr_{(X,A),(Y,A)}(f, pr_{\mathcal{D}}(\eta_{(Y,A)}))$$

□

3. C-systems

3.1. **Generalities.** [**onCsystems**] The definition of a C-system is given in [45, Def. 2.1, 2.3]. Homomorphisms of C-systems are defined in [39, Def. 3.1, p 1188]. In [39, Lemma 3.4, p. 1190] it is shown that for two C-systems CC_1, CC_2 a homomorphism between the underlying C0-systems is always a homomorphism of C-systems.

Further study of C-systems can be found in [44, Sec. 2, pp. 1048-1064].

For Γ', Γ in a C-system let us write $\Gamma' \geq \Gamma'$ and say that Γ' is over Γ if $l(\Gamma') \geq l(\Gamma)$ and $ft^{l(\Gamma')-l(\Gamma)}(\Gamma') = \Gamma$. We write $\Gamma' > \Gamma$ if $\Gamma' \geq \Gamma$ and $l(\Gamma') > l(\Gamma)$.

If Γ' is over Γ we will denote by $p_{\Gamma',\Gamma}$ the obvious composition of the p-morphisms starting with Γ' and ending in Γ.

If Γ'', Γ' are over Γ then we say that a morphism $f : \Gamma'' \to \Gamma'$ is over Γ if

$$f \circ p_{\Gamma',\Gamma} = p_{\Gamma'',\Gamma}$$

If Δ' is an object over Δ and $f : \Gamma \to \Delta$ is a morphism then we denote simply by $f^*(\Delta')$ the object $f^*(\Gamma', n)$ where $n = l(\Gamma') - l(\Delta)$. Note that n can always be inferred from f and Γ'.

Similarly we will write simply $q(f, \Gamma')$ for $q(f, \Gamma', n)$ since n can be inferred as $l(\Gamma') - l(codom(f))$.

Lemma 3.1. *[2015.08.23.l1a] Let Γ', Γ'' be objects over Δ, $a : \Gamma' \to \Gamma''$ a morphism over Δ and $f : \Gamma \to \Delta$ a morphism. Then there is a unique morphism $f^*(a) : f^*(\Gamma') \to f^*(\Gamma'')$ over Γ such that the square*

$$\begin{array}{ccc} f^*(\Gamma') & \xrightarrow{q(f,\Gamma')} & \Gamma' \\ {\scriptstyle f^*(a)}\downarrow & & \downarrow{\scriptstyle a} \\ f^*(\Gamma'') & \xrightarrow{q(f,\Gamma'')} & \Gamma'' \end{array}$$

commutes.

Proof. See [44, Lemma 2.13]. □

Lemma 3.2. *[2015.08.29.l2] Let $a : \Gamma' \to \Gamma''$ be a morphism over Γ''' and Γ''' an object over Δ. Then a is a morphism over Δ and for any $f : \Gamma \to \Delta$ one has*

(44) $$[\mathbf{2015.08.29.eq2}] f^*(a) = q(f, \Gamma''')^*(a)$$

Proof. See [44, Lemma 2.15]. □

We will also need the following facts about homomorphisms of C-systems. For the last item of the lemma recall that for Γ such that $l(\Gamma) > 0$, one defines $\delta(\Gamma) \in Mor(CC)$ as s_{Id_Γ}

Lemma 3.3. *[2015.09.03.l2] Let $F : CC \to CC'$ be a homomorphism of C-systems. Then one has:*

(1) for $\Gamma', \Gamma \in CC$, $\Gamma \geq \Gamma'$ implies $F(\Gamma) \geq F(\Gamma')$ and similarly for $>$,
(2) for $\Gamma' \geq \Gamma$ in CC one has $F(p_{\Gamma',\Gamma}) = p_{F(\Gamma'),F(\Gamma)}$,
(3) for $\Gamma' \geq \Delta$ and $f : \Gamma \to \Delta$ one has

$$F(f^*(\Gamma')) = (F(f))^*(F(\Gamma'))$$

$$F(q(f, \Gamma')) = q(F(f), F(\Gamma'))$$

(4) for $a : \Gamma' \to \Gamma''$ over Γ and $f : \Delta \to \Gamma$, $F(a)$ is a morphism over $F(\Gamma)$ and one has

$$F(f^*(a)) = (F(f))^*(F(a))$$

(5) for Γ such that $l(\Gamma) > 0$ one has

$$F(\delta(\Gamma)) = \delta(F(\Gamma))$$

Proof. For (1),(2),(3) see [44, Lemma 2.5]. For (4) see [44, Lemma 2.14]. For (5) recall that we have $\delta(\Gamma) = s_{Id_\Gamma}$ by the definition of δ in [45, p. 131]. Next, we have $F(s_f) = s_{F(f)}$ by [39, Lemma 3.4]. Therefore

$$F(\delta(\Gamma)) = F(s_{Id_\Gamma}) = s_{F(Id_\Gamma)} = s(Id_{F(\Gamma)}) = \delta(F(\Gamma))$$

This completes the proof of Lemma 3.3. □

3.2. **The presheaf extension of a C-system.** [**Fext**] Let CC be a C-system and $F : CC^{op} \to Sets$ a presheaf on the category underlying CC. In this section we construct a new C-system $CC[F]$ which we call the F-extension of CC and describe a unital pre-B-system $B(CC, F)$ and an isomorphism $B(CC[F]) \to B(CC, F)$.

We will first construct a C0-system $CC[F]$ and then show that it is a C-system. For the definition of a C0-system see [45, Definition 2.1].

Problem 3.4. *[2016.01.19.prob1] Given a C-system CC and a presheaf $F : CC^{op} \to Sets$ to construct a C0-system that will be denoted $CC[F]$ and called the F-extension of CC.*

Construction 3.5. [**2016.01.19.constr1**] We set
(45)
$$[\mathbf{2016.01.19.eq1}] Ob(CC[F]) = \amalg_{X \in CC} F(ft^{l(X)}(X)) \times \cdots \times F(ft^2(X)) \times F(ft(X))$$

where the product of the empty sequence of factors is a 1-point set. We will write elements of $Ob(CC[F])$ as (X,Γ) where $X \in CC$ and $\Gamma = (T_0, \dots, T_{l(X)-1})$. Note that $ft^{l(X)}(X) = pt$ for any X and therefore all the products in (45) start with $F(pt)$.

We set

$$Mor(CC[F]) = \amalg_{(X,\Gamma),(Y,\Gamma')} Mor_{CC}(X,Y)$$

We will write elements of $Mor(CC[F])$ as $((X,\Gamma),(Y,\Gamma'),f)$. When the domain and the codomain of a morphism are clear from the context we may write f instead of $((X,\Gamma),(Y,\Gamma'),f)$.

We define the composition function by the rule

$$((X,\Gamma),(Y,\Gamma'),f)) \circ ((Y,\Gamma'),(Z,\Gamma''),g) = ((X,\Gamma),(Z,\Gamma''),f \circ g)$$

We define the identity morphisms by the rule

$$Id_{CC[F],(X,\Gamma)} = ((X,\Gamma),(X,\Gamma),Id_{CC,X})$$

The associativity and the identity conditions of a category follow easily from the corresponding properties of CC. This completes the construction of a category $CC[F]$.

We define the length function as

$$l((X,\Gamma)) = l(X)$$

If $l((X,\Gamma)) = 0$ then $X = pt$ and $\Gamma = ()$ where $()$ is the unique element of the one point set that is the product of the empty sequence. We will often write $(pt,())$ as pt.

We define the ft-function on (X,Γ) such that $l(X) > 0$ as

$$ft((X,(T_0,\dots,T_{l(X)-1})) = (ft(X),(T_0,\dots,T_{l(X)-2}))$$

which is well defined because $l(ft(X)) = l(X) - 1$, and set $ft((pt,())) = (pt,())$. We will write $ft(\Gamma)$ for $(T_0,\dots,T_{l(X)-2})$ so that $ft((X,\Gamma)) = (ft(X), ft(\Gamma))$.

We define the p-morphisms as

$$p_{(X,\Gamma)} = ((X,\Gamma), ft(X,\Gamma), p_X)$$

For (Y,Γ') such that $l((Y,\Gamma')) > 0$ and $f : (X,\Gamma) \to ft(Y,\Gamma')$ where $\Gamma = (T_0,\dots,T_{l(X)-1})$ and $\Gamma' = (T'_0,\dots,T'_{l(Y)-1})$ we set

(46) $\quad$ [**2016.01.31.eq1**] $f^*((Y,\Gamma')) = (f^*(Y),(T_0,\dots,T_{l(X)-1},F(f)(T'_{l(Y)-1})))$.

In the same context as above we define the q-morphism as

$$q(f,(Y,\Gamma')) = (f^*((Y,\Gamma')),(Y,\Gamma'),q(f,Y))$$

This completes the construction of the elements of the structure of a C0-system. Let us verify that these elements satisfy the axioms of a C0-system.

The uniqueness of an object of length 0 is obvious.

The condition that $l(ft(X,\Gamma)) = l((X,\Gamma)) - 1$ if $l((X,\Gamma)) > 0$ is obvious.

The condition that $ft((pt,())) = (pt,())$ is obvious.

The fact that pt is a final object in $CC[F]$ follows from the fact that pt is a final object of CC.

The fact that for (Y,Γ') such that $l((Y,\Gamma'))>0$ and $f:(X,\Gamma)\to ft(Y,\Gamma')$ one has $q(f,(Y,\Gamma'))\circ p_{(Y,\Gamma')} = p_{f^*((Y,\Gamma'))}\circ f$ follows from the corresponding fact in CC.

The fact that for (Y,Γ') such that $l((Y,\Gamma'))>0$ one has $Id^*_{ft(Y,\Gamma)}((Y,\Gamma'))=(Y,\Gamma')$ follows from the corresponding fact for CC and the identity axiom of the functor F.

The fact that for (Y,Γ') such that $l((Y,\Gamma'))>0$ one has $q(Id_{(Y,\Gamma)},(Y,\Gamma))=Id_{(Y,\Gamma)}$ follows from the previous assertion and the corresponding fact in CC.

The fact that (Y,Γ') such that $l((Y,\Gamma'))>0$, $f:(X,\Gamma)\to ft(Y,\Gamma')$ and $g:(W,\Delta)\to(X,\Gamma)$ one has $g^*(f^*((Y,\Gamma')))=(g\circ f)^*((Y,\Gamma'))$ follows from the composition axiom for the functor F and the corresponding fact for CC.

The fact that in the same context as in the previous assertion one has

$$q(g,f^*((Y,\Gamma')))\circ q(f,(Y,\Gamma'))=q((g\circ f),(Y,\Gamma'))$$

follows from the previous assertion and the corresponding fact for CC.

This completes Construction 3.5 □

Lemma 3.6. *[2016.01.19.l2] The functions $Ob(CC[F])\to Ob(F)$ and $Mor(CC(F))\to Mor(CC)$ given by*

$$(X,\Gamma)\mapsto X$$

and

$$((X,\Gamma),(Y,\Gamma'),f)\mapsto f$$

form a functor $tr_F:CC[F]\to CC$ and this functor is fully faithful.

Proof. Straightforward from the construction. □

Lemma 3.7. *[2016.01.19.l1] The C0-system of Construction 3.5 is a C-system.*

Proof. By [45, Proposition 2.4] it is sufficient to prove that the canonical squares of $CC[F]$, i.e., the squares formed by morphisms $q(f,(Y,\Gamma')), p_{(Y,\Gamma')}$ and $p_{f^*((Y,\Gamma'))}, f$ are pull-back squares. The functor of Lemma 3.6 map these square to canonical squares of the C-system CC that are pull-back squares. Since this functor is fully faithful we conclude that the canonical squares in $CC[F]$ are pull-back squares. The lemma is proved. □

This completes the construction of the presheaf extension of a C-system.

Remark 3.8. **[2015.09.01.rem1]** For any two objects of $C[F]$ of the form $(X,\Gamma),(X,\Gamma')$ the formula

$$can_{X,\Gamma,\Gamma'}=((X,\Gamma),(X,\Gamma'),Id_X)$$

defines a morphism which is clearly an isomorphism with $can_{X,\Gamma',\Gamma}$ being a canonical inverse. Therefore, all objects of $CC[F]$ with the same image in CC are "canonically isomorphic".

Remark 3.9. **[2015.09.01.rem2]** If $F(pt) = \emptyset$ then $CC[F] = \{pt\}$. On the other hand, the choice of an element y in $F(pt)$ defines distinguished elements $y_X = F(\pi_X)(y)$ in all sets $F(X)$ and therefore distinguished objects $(X, \Gamma_{X,y}) = (X, (y, \dots, y_{ft^2(X)}, y_{ft(X)}))$ in the fibers of the object component of tr_F over all X.

Mapping X to $(X, \Gamma_{X,y})$ and $f : X \to Y$ to $((X, \Gamma_{X,y}), (Y, \Gamma_{Y,y}), f)$ defines, as one can immediately prove from the definitions, a functor $tr^!_{F,y} : CC \to CC[F]$.

This functor clearly satisfies the conditions $tr^!_{F,y} \circ tr_F = Id_{CC}$.

One verifies easily that the morphisms

$$can_{X,\Gamma,\Gamma_{(X,y)}} : (X, \Gamma) \to tr^!_{F,y}(X, \Gamma)$$

form a natural transformation. We conclude that tr_F and $tr^!_{F,y}$ is a pair of mutually inverse equivalences of categories.

However these equivalences are not isomorphisms unless $F(X) \cong unit$ for all X and as a C-system $CC[F]$ is often very different from CC, for example, in that that it may have many more C-subsystems.

We provide the following lemma without a proof because the proof is immediate from the definitions and [39, Lemma 3.4] that asserts that a functor that satisfies all conditions of the definition of a homomorphism except possibly the s-morphisms condition is a homomorphism.

Lemma 3.10. *[**2015.08.22.l4**] The functor $tr : CC[F] \to CC$ is a homomorphism of C-systems.*

Remark 3.11. **[2015.08.22.rem1]** Let $y \in F(pt)$. Then for $f : X \to Y$ one has $F(f)(y_Y) = y_X$ and therefore for $f : X \to ft(Y)$ one has

$$(tr^!_y(f))^*(Y) = (f^*(Y), \Gamma_{f^*(Y),y}) = f^*((Y, \Gamma_Y)) = tr^!_y(f)^*(tr_y(Y))$$

The rest of the conditions that one needs to prove in order to show that tr_y is a homomorphism of C-systems is immediate from definitions and we obtain that

$$tr^!_y : CC \to CC[F]$$

is a homomorphism of C-systems.

Recall that by definition $(X, \Gamma) \le (Y, \Gamma')$ if and only if $l(X, \Gamma) \le l(Y, \Gamma')$ and

$$(X, \Gamma) = ft^{l(Y,\Gamma') - l(X,\Gamma)}(Y, \Gamma').$$

From construction we conclude that $(X, \Gamma) \le (Y, \Gamma')$ if and only if $X \le Y$ in CC and

$$(X, \Gamma) = ft^{l(Y) - l(X)}((Y, \Gamma')).$$

Lemma 3.12. *[**2016.01.31.l1**] Let $i \ge 0$, (Y, Γ') be such that $l(Y) \ge i$. Let $f : (X, \Gamma) \to ft^i(Y, \Gamma')$. Let $lx = l(X)$, $ly = l(Y)$ and*

$$\Gamma = (T_0, \dots, T_{lx-1})$$
$$\Gamma' = (T'_0, \dots, T'_{ly-1})$$

Then

$$f^*((Y, \Gamma'), i) = (f^*(Y, i), (T_0, \dots, T_{lx-1}, F(q(f, ft^i(Y), 0))(T'_{ly-i}), \dots, F(q(f, ft(Y), i-1))(T'_{ly-1}))$$

Proof. By induction on i.

For $i = 0$ we have

$$f^*((Y,\Gamma'),0) = (X,\Gamma) = (f^*(Y,0),(T_0,\dots,T_{lx-1}))$$

For the successor of i we need to show that

$$[\mathbf{2016.01.31.eq3}] f^*((Y,\Gamma'),i+1) =$$

(47)
$$(f^*(Y,i+1),(T_0,\dots,T_{lx-1},F(q(f,ft^{i+1}(Y),0))(T'_{ly-i-1}),\dots,F(q(f,ft(Y),i))(T'_{ly-1}))$$

We have by (**??**),

$$f^*((Y,\Gamma'),i+1) = q(f,ft((Y,\Gamma')),i)^*((Y,\Gamma'))$$

By the inductive assumption, $q(f,ft((Y,\Gamma')),i)$ is a morphism with the domain

$$f^*(ft(Y,\Gamma'),i) =$$
$$(f^*(ft(Y),i),(T_0,\dots,T_{lx-1},F(q(f,ft^i(ft(Y)),0))(T'_{ly-1-i}),\dots,F(q(f,ft(ft(Y))),i-1)(T'_{ly-2})))$$

By (46) we get

$$q(f,ft((Y,\Gamma')),i)^*((Y,\Gamma')) =$$
$$(q(f,ft(Y),i)^*(Y),$$
$$(T_0,\dots,T_{lx-1},F(q(f,ft^i(ft(Y)),0))(T'_{ly-1-i}),\dots,F(q(f,ft(ft(Y))),i-1)(T'_{ly-2}),$$
$$F(q(f,ft(Y),i))(T'_{ly-1})))$$

which coincides with our goal (3.2). □

4. ???

4.1. Some computations with Jf-relative monads. [Jfrel]

For two sets X and Y we let $Fun(X,Y)$ to denote the set of functions from X to Y.

Next, following [16] we let $\mathbb{F}$ denote the category with the set of objects $\mathbf{N}$ and the set of morphisms from m to n being $Fun(stn(m),stn(n))$, where $stn(m) = \{i \in \mathbf{N} \mid i < m\}$ is our choice for the standard set with m elements (cf. [41]) and where for two sets X and Y, $Fun(X,Y)$ is the set of functions from X to Y defined as in [11, p.81] such that each function has a well defined domain and codomain.

For any set U there is a category $Sets(U)$ of the following form. The set of objects of $Sets(U)$ is U. The set of morphisms is

$$Mor(Sets(U)) = \cup_{X,Y\in U} Fun(X,Y)$$

Since a function from X to Y is defined as a triple (X,Y,G) where G is the graph subset of this function the domain and codomain functions are well defined on $Mor(Sets(U))$ such that

$$Mor_{Sets(U)}(X,Y) = Fun(X,Y)$$

and a composition function can be defined that restricts to the composition of functions function on each $Mor_{Sets(U)}(X,Y)$. Finally the identity function $U \to Mor(Sets(U))$

is obvious and the collection of data that one obtains satisfies the axioms of a category. This category is called the category of sets in U and denoted $Sets(U)$.

We will only consider the case when U is a universe. As was mentioned in the introduction we fix U and omit it from our notations below.

Following [7] we let $Jf : F \to Sets$ denote the functor that takes n to $stn(n)$ and that is the identity on morphisms between two objects (on the total sets of morphisms the morphism component of this functor is the inclusion of a subset).

As the following construction shows any monad on sets defines a Jf-relative monad. Combined with our construction of $C(\mathbf{RR})$ this gives a construction of a C-system for any monad on sets.

Problem 4.1. *[**2016.01.13.prob1**] Given a monad $\mathbf{R} = (R, \eta, \mu)$ (cf.* [27]*[p.133]) on the category of sets to construct a Jf-relative monad* $\mathbf{RR}$.

Construction 4.2. [**2016.01.13.constr1**] We set

(1) $R(n) = R(stn(n))$,
(2) $\eta_n = \eta_{stn(n)}$,
(3) for $f : stn(m) \to R(n)$ we set $f^* = R(f) \circ \mu_{stn(n)}$.

The verification of the relative monad axioms is easy. □

Remark 4.3. [**2016.01.03.rem1**] It seems to be possible to provide a construction of a monad from a Jf-relative monad without the use of the axioms of choice and excluded middle. This construction will be considered in a separate note.

Remark 4.4. [**2016.01.17.rem1**] The set of Jf-relative monads is in an easy to construct bijection with the set of abstract clones as defined in [16, Section 3].

In [43] we constructed for any Jf-relative monad $\mathbf{RR} = (RR, \eta, -^*)$ a Lawvere theory $(T, L) = RML(\mathbf{RR})$. Most of this section is occupied by simple computations in T that will be used in the later sections.

Recall that the category T has as the set of objects the set of natural numbers and as the set of morphisms the set

$$Mot_T = \amalg_{m,n} Fun(stn(m), RR(n))$$

Therefore the set of morphisms in T from m to n is the set of iterated pairs $((m, n), f)$ where $f \in Fun(stn(m), RR(n))$. We fix the obvious bijection between this set and $Fun(stn(m), RR(n))$ and use the corresponding functions in both directions as coercions. A coercion, in the terminology of the proof assistant Coq, is a function $f : X \to Y$ such that when an expression denoting an element x of the set X occurs in a position where an element of Y should be it is assumed that x is replaced by $f(x)$.

Let us introduce the following notation:

$$F(m, n) = Fun(stn(m), stn(n))$$

and, for a Jf-relative monad $\mathbf{RR}$,

$$RR(m, n) = Fun(stn(m), RR(n))$$

Then for $f \in RR(l,m)$ and $g \in RR(m,n)$ the composition $f \circ_T g$ in T is defined as $f^* \circ g$ and for $m \in \mathbf{N}$ the identity morphism Id_m in T is defined as η_m.

The functor $L : F \to T$ is defined as the identity on objects and as the function on morphisms corresponding to the functions $f \mapsto f \circ \eta_n$ from $F(m,n)$ to $RR(m,n)$.

We also obtain the extension of RR to a functor $F \to Sets$ according to Construction 2.7. For a morphism $f \in F(m,n)$ we have $RR(f) = (f \circ \eta_n)^* = L(f)^*$.

We are going to use the functions $f \mapsto RR(f)$ as coercions so that when an element f of $F(m,n)$ occurs in a position where an element of $Fun(RR(m), RR(n))$ is expected it has to be replaced by $RR(f)$.

Remark 4.5. [2015.11.20.rem4] We can not replace $\amalg$ by $\cup$ in our definition of the set of morphisms of T because for a general $\mathbf{RR}$ the sets $RR(m,n)$ are not disjoint. For example, if $RR(m) = pt$ where pt is a fixed one element set then RR has a (unique) structure of a Jf-relative monad and $RR(m,n) = RR(m,n')$ for all m,n,n'. Therefore no function to $\mathbf{N}$ from the union of these sets can distinguish the codomain of a morphism. In particular, in this case there is no category with the sets of morphisms from m to n being equal $RR(m,n)$.

Since we will have to deal with elements of the sets of functions $Fun(stn(m), RR(n))$ and of similar sets such as the sets $Ob_n(C(\mathbf{RR}, \mathbf{LM}))$ introduced later we need to choose some way to represent them. For the purpose of the present paper we will write such elements as sequences, i.e., to denote the function, which in the notation of λ-calculus is written as $\lambda\, i : stn(n), f_i$, we will write $(f_0, \ldots, f_{n-1})$. In particular, for an element x of a set X, the expression (x) denotes the function $stn(1) \to X$ that takes 0 to x.

Lemma 4.6. *[**2016.01.15.l4**] Let $f = (f(0), \ldots, f(l-1))$ be a morphism in T from l to m and $g = (g(0), \ldots, g(m-1))$ a morphism from m to n. Then one has*

$$f \circ_T g = (g^*(f(0)), \ldots, g^*(f(l-1)))$$

Proof. We have

$$(f \circ_T g)(i) = (f \circ g^*)(i) = g^*(f(i)).$$

The lemma is proved. □

Lemma 4.7. *[**2015.08.30.l1**] Let $f \in F(l,m)$, $g \in RR(m,n)$ and $i \in stn(l)$. Then one has*

$$[\mathbf{2015.08.26.eq4}](L(f) \circ_T g)(i) = g(f(i)) \tag{48}$$

Proof. Rewriting the left hand side we get

$$(L(f) \circ_T g)(i) = ((f \circ \eta_m) \circ g^*)(i) = (f \circ (\eta_m \circ g^*))(i) = (f \circ g)(i) = g(f(i)).$$

which completes the proof. □

For $n \in \mathbf{N}$ and $i = 0, \ldots, n-1$ let

$$x_i^n = \eta_n(i) \in RR(n)$$

Observe also that for $f \in RR(m,n)$ one has

(49) $$[\mathbf{2015.08.24.eq5}] f^*(x_i^m) = (\eta_m \circ f^*)(i) = f(i)$$

and for $f \in F(m,n)$ one has

(50) $$[\mathbf{2016.01.15.eq1}] f(x_i^m) = RR(f)(\eta_m(i)) = (\eta_m \circ (f \circ \eta_n))^*(i) = (f \circ \eta_n)(i) = \eta_n(f(i)) = x^n_{f(i)}$$

Let

$$\partial_n^i : stn(n) \to stn(n+1)$$

for $0 \le i \le n$ be the increasing inclusion that does not take the value i and

$$\sigma_n^i : stn(n+2) \to stn(n+1)$$

for $0 \le i \le n$ be the non-decreasing surjection that takes the value i twice. Taking into account that, in the notation of [17], $[n] = stn(n+1)$ these are the standard generators of the simplicial category Δ together with $\partial_0^0 : stn(0) \to stn(1)$.

In our sequence notation we have

(51) $$[\mathbf{2015.08.24.eq7}] L(\partial_n^i) = (x_0^{n+1}, \ldots, x_{i-1}^{n+1}, x_{i+1}^{n+1}, \ldots, x_n^{n+1})$$

and

(52) $$[\mathbf{2015.08.24.eq8}] L(\sigma_n^i) = (x_0^{n+1}, \ldots, x_{i-1}^{n+1}, x_i^{n+1}, x_i^{n+1}, x_{i+1}^{n+1}, \ldots, x_n^{n+1})$$

in particular

(53) $$[\mathbf{2015.07.12.eq5}] L(\partial_n^n) = (x_0^{n+1}, \ldots, x_{n-1}^{n+1})$$

Let

$$\iota_n^i : stn(n) \to stn(n+i)$$

be the function given by $\iota_n^i(j) = j$ for $j = 0, \ldots, n-1$. Then we have

(54) $$[\mathbf{2015.08.22.eq7}] \iota_n^1 = \partial_n^n$$

and (50) implies that

(55) $$[\mathbf{2015.08.22.eq8}] \iota_n^i(x_j^n) = x_j^{n+i}$$

Lemma 4.8. *[**2015.08.26.l1**] Let $f = (f(0), \ldots, f(m))$ be a morphism from $m+1$ to n in T. Then*

(56) $$[\mathbf{2016.01.15.eq3}] L(\iota_m^1) \circ_T f = (f(0), \ldots, f(m-1))$$

In particular, if $f \in RR(n+1,n)$ then $L(\iota_n^1) \circ_T f = Id_{T,n}$ if and only if $f(i) = x_n^i$ for $i = 0, \ldots, n-1$.

Proof. Both sides of the required equality are elements of $Fun(stn(m), RR(n))$. Therefore, the equality holds if and only if for all $i = 0, \ldots, n-1$ we have $(L(\iota_m^1) \circ_T f)(i) = f(i)$. The assertion of the lemma follows now from Lemma 4.7.

Since $Id_{T,n} = (x_0^n, \ldots, x_{n-1}^n)$ the second assertion immediately follows from the first one. $\square$

For $f\in RR(n,m)$, $f=(f(0),\dots,f(n-1))$ define an element $qq(f)\in RR(n+1,m+1)$ by the formula:

(57) $$[\mathbf{2015.08.26.eq9}]qq(f)=(\iota^1_m(f(0)),\dots,\iota^1_m(f(n-1)),x^{m+1}_m)$$

Lemma 4.9. *[**2015.08.26.l2**] For $i\in\mathbf{N}$ and $f=(f(0),\dots,f(n-1))$ in $RR(n,m)$ one has*

$$qq^i(f)=(\iota^i_m(f(0)),\dots,\iota^i_m(f(n-1)),x^{m+i}_m,\dots,x^{m+i}_{m+i-1})$$

Proof. Straightforward by induction on i. □

Lemma 4.10. *[**2015.08.26.l3a**] For $n,i\in\mathbf{N}$ one has*

$$qq^i(L(\iota^1_n))=L(\partial^n_{n+i})$$

Proof. We have $L(\iota^1_n)=L(\partial^n_n)=(x^{n+1}_0,\dots,x^{n+1}_{n-1})$. By Lemma 4.9 and (55) we get

$$qq^i(L(\iota^1_n))=(\iota^i_{n+1}(x^{n+1}_0),\dots,\iota^i_{n+1}(x^{n+1}_{n-1}),x^{n+1+i}_{n+1},\dots,x^{n+1+i}_{n+i})=$$
$$=(x^{n+1+i}_0,\dots,x^{n+1+i}_{n-1},x^{n+1+i}_{n+1},\dots,x^{n+1+i}_{n+i})=L(\partial^{n+i}_n)$$

where the last equality is (51). □

Lemma 4.11. *[**2015.08.28.l1**] For $i,m\in\mathbf{N}$ and $r\in RR(m)$ one has*

$$qq^i(x^m_0,\dots,x^m_{m-1},r)=(x^{m+i}_0,\dots,x^{m+i}_{m-1},\iota^i_m(r),x^{m+i}_m,\dots,x^{m+i}_{m+i-1})$$

Proof. One has

$$qq^i(x^m_0,\dots,x^m_{m-1},r)=(\iota^i_m(x^m_0),\dots,\iota^i_m(x^m_{m-1}),\iota^i_m(r),x^{m+i}_m,\dots,x^{m+i}_{m+i-1})=$$
$$(x^{m+i}_0,\dots,x^{m+i}_{m-1},\iota^i_m(r),x^{m+i}_m,\dots,x^{m+i}_{m+i-1})$$

where the first equality is by Lemma 4.9 and the second one by (55). □

4.2. The C-system $C(\mathbf{RR})$. [CRR]

In [41] we constructed for any Lawvere theory (T,L) a C-system $LC((T,L))$. For $(T,L)=RML(\mathbf{RR})$ we denote the C-system $LC((T,L))$ by $C(\mathbf{RR})$. In this section we first provide a more explicit description of $C(\mathbf{RR})$ and then compute the action of the operations $T,\widetilde{T},S,\widetilde{S}$ and δ on the B-sets $(Ob(C(\mathbf{RR})),\widetilde{Ob}(C(\mathbf{RR})))$ of this C-system (cf. Definition 4.22).

Recall that as a category $C(\mathbf{RR})$ is the opposite category to T. Its set of objects is the set of natural numbers $\mathbf{N}$. To distinguish the positions in formulas where natural numbers are used as objects of $C(\mathbf{RR})$ we will write in such places $\widehat{m}$ instead of m, $\widehat{n}$ instead of n etc.

The sets of morphisms of $C(\mathbf{RR})$ are given by the formula

$$Mor_{C(\mathbf{RR})}(\widehat{m},\widehat{n})=\{((n,m),f)\,|\,f\in Fun(stn(n),RR(m))\}$$

and we apply to them the same convention as to morphisms of T, that is, we identify the sets $Mor_{C(\mathbf{RR})}(\widehat{m},\widehat{n})$ with the sets $Fun(stn(n),RR(m))$ by means of the obvious bijection.

We consider L as a functor

$$L:F^{op}\to C(\mathbf{RR})$$

i.e., as a contravariant functor from F to $C(\mathbf{RR})$ and keep the conventions introduced in the previous section the most important of which is that for $f \in F(m,n)$ and $x \in RR(m)$ we write $f(x)$ for $RR(f)(x) = (f \circ \eta_n)^*(x)$.

The ft function on $C(\mathbf{RR})$ is defined by the formula $ft(\widehat{n+1}) = \widehat{n}$ and $ft(\widehat{0}) = \widehat{0}$.

The p-morphisms are defined by setting $p_{\widehat{0}} = Id_{\widehat{0}}$ and $p_{\widehat{n+1}} : \widehat{n+1} \to \widehat{n}$ to be the morphism $L(\iota_n^1)$. In the sequence notation we have

$$[\mathbf{2015.08.24.eq6}] p_{\widehat{n+1}} = (x_0^{n+1}, \ldots, x_{n-1}^{n+1}) \tag{58}$$

For a morphism $f : \widehat{m} \to \widehat{n}$ in $C(\mathbf{RR})$ we have $f^*(\widehat{n+1}) = \widehat{m+1}$.

Before giving an explicit description of q-morphisms we will prove the following lemma.

Lemma 4.12. **[2015.07.24.l1]** *One has:*

1. *Let $f = (f(0), \ldots, f(n))$ be a morphism $\widehat{m+1} \to \widehat{n+1}$. Then*
$$f \circ_C p_{\widehat{n+1}} = (f(0), \ldots, f(n-1))$$
2. *Let $f = (f(0), \ldots, f(n-1))$ be a morphism $\widehat{m} \to \widehat{n}$. Then*
$$p_{\widehat{m+1}} \circ_C f = (\iota_m^1(f(0)), \ldots, \iota_m^1(f(n-1)))$$

Proof. Both sides of the first equality are elements of $Fun(stn(n), RR(m+1))$ and for $i \in stn(n)$ we have

$$(f \circ_C p_{\widehat{n+1}})(i) = (L(\iota_n^1) \circ_T f)(i) = f(i)$$

where the second equality is by (48).

Both sides of the second equality are again elements of $Fun(stn(n), RR(m+1))$ and for $i \in stn(n)$ we have:

$$(p_{\widehat{m+1}} \circ_C f)(i) = (f \circ_T L(\iota_m^1))(i) = (f \circ (L(\iota_m^1))^*)(i) = (f \circ RR(\iota_m^1))(i) = \iota_m^1(f(i))$$

□

The q-morphisms were defined in [41] in a somewhat implicit manner. We give their explicit description in the following lemma.

Lemma 4.13. **[2016.01.15.l3]** *Let $f : \widehat{m} \to \widehat{n}$ be a morphism in $C(\mathbf{RR})$. Then one has*

$$q(f, \widehat{n+1}) = qq(f)$$

Proof. The morphism $q(f) = q(f, \widehat{n+1})$ was defined in [41] as the unique morphism such that

$$q(f) \circ_C p_{\widehat{n+1}} = p_{\widehat{m+1}} \circ_C f$$

and

$$q(f) \circ_C (x_n^{n+1}) = (x_m^{m+1})$$

For the first equation we have

$$qq(f) \circ_C p_{\widehat{n+1}} = (\iota_m^1(f(0)), \ldots, \iota_m^1(f(n-1)))$$

by Lemma 4.12(1) and (57) and
$$p_{\widehat{m+1}}\circ_C f = (\iota^1_m(f(0)),\dots,\iota^1_m(f(n-1)))$$
by Lemma 4.12(2).

Both sides of the second equation are elements of $Fun(stn(1), RR(m+1))$ and it is sufficient that their values on 0 coincide. We have
$$(q(f)\circ_C(x^{n+1}_n))(0) = ((x^{n+1}_n)\circ_T qq(f))(0) = ((x^{n+1}_n)\circ qq(f)^*) =$$
$$qq(f)^*(x^{n+1}_n) = qq(f)(n) = x^{m+1}_m$$
where the fourth equality is by (49) and the fifth by (57). This completes the proof of Lemma 4.13. $\square$

Let us describe the constructions introduced in Section 3.1 in the case of $C(\mathbf{RR})$. Note that our wide-hat notation that distinguishes the places in formulas where natural numbers are used as objects of $C(\mathbf{RR})$ allows us to avoid the ambiguity that might have arisen otherwise. For example $p_{m,n}$ could be understood either as the canonical morphism $m\to n$ using the notation $p_{\Gamma',\Gamma}$ introduced in Section 3.1 or as the canonical morphism $m\to m-n$ using the notation $p_{\Gamma,i}$ that we have used in [45]. The use of the wide-hat diacritic allows to distinguish between $p_{\widehat{m},\widehat{n}}$ - a morphism $\widehat{m}\to\widehat{n}$, and $p_{\widehat{m},n}$ - a morphism $\widehat{m}\to\widehat{m-n}$.

Lemma 4.14. **[2015.08.22.l6]** *Let $n,i\in\mathbf{N}$.*

(1) One has
(1.1) $p_{\widehat{n+i},i} = L(\iota^{n+i}_n) = (x^{n+i}_0,\dots,x^{n+i}_{n-1})$,
(1.2) for $m\in\mathbf{N}$ and $g=(g(0),\dots,g(n+i-1))$ from $\widehat{m}$ to $\widehat{n+i}$ one has
$$g\circ p_{\widehat{n+i},i} = (g(0),\dots,g(n-1)),$$
(2) for $f:\widehat{m}\to\widehat{n}$ one has
$$f^*(\widehat{n+i},i) = m+i$$
and
$$q(f,\widehat{n+i},i) = qq^i(f)$$

Proof. All three assertions a proved by induction on i. For the first assertion both parts are proved by induction simultaneously. One has

(1) in the case $i=0$ the first assertion follows from the identity axiom of the functor defined by $\mathbf{RR}$ as in Construction 2.7 and second from the identity axiom of the category $C(\mathbf{RR})$,
(2) for the successor of i we have
$$p_{\widehat{n+i+1},i+1} = p_{\widehat{n+i+1}}\circ p_{\widehat{n+i},i} = (x^{n+i+1}_0,\dots,x^{n+i+1}_{n-1})$$
where the second equality is by the second part of the inductive assumption. For the inductive step in the second part we have
$$(g(0),\dots,g(n+i))\circ p_{\widehat{n+i+1},i+1} = (g(0),\dots,g(n+i))\circ p_{\widehat{n+i+1}}\circ p_{\widehat{n+i},i} =$$
$$(g(0),\dots,g(n+i-1))\circ p_{\widehat{n+i},i} = (g(0),\dots,g(n-1))$$

The proof of the first part of the second assertion is obvious. For the second part we have:

(1) for $i = 0$ the assertion is obvious,
(2) for the successor of i we have

$$q(f, \widehat{n+i+1}, i+1) = qq(q(f, \widehat{n+i}, i)) = qq(qq^i(f)) = qq^{i+1}(f)$$

□

Lemma 4.15. *[**2015.08.22.l7**] Let $f = (f(0), \dots, f(n))$ be a morphism from $\widehat{n}$ to $\widehat{n+1}$. Then $f \circ p_{\widehat{n+1}} = Id_{\widehat{n}}$ if and only if $f(i) = x_i^n$ for $i = 0, \dots, n-1$.*

Proof. It follows immediately from Lemma 4.8. □

Lemma 4.16. *[**2015.09.09.l1**] Let $f = (f(0), \dots, f(n-1))$ be a morphism from $\widehat{m}$ to $\widehat{n}$ where $n > 0$. Then one has*

$$s_f = (x_0^m, \dots, x_{m-1}^m, f(n-1))$$

Proof. By [45, Definition 2.3(2)] we have that

$$s_f \circ p_{\widehat{m+1}} = Id_{\widehat{m}}$$

Therefore, by Lemma 4.15, s_f is of the form $(x_0^m, \dots, x_{m-1}^m, sf)$ for some $sf \in RR(m)$. By [45, Definition 2.3(3)] we have $f = s_f \circ q(ft(f), \widehat{n})$ where $ft(f) = f \circ p_{\widehat{n}}$. By Lemma 4.12(1) we have $ft(f) = (f(0), \dots, f(n-2))$ and by Lemma 4.13 and (57) we have

$$q(ft(f), \widehat{n}) = qq(ft(f)) = (\iota_m^1(f(0)), \dots, \iota_m^1(f(n-2)), x_m^{m+1})$$

Therefore we should have

$$(f(0), \dots, f(n-1)) = (\iota_m^1(f(0)), \dots, \iota_m^1(f(n-2)), x_m^{m+1}) \circ_T (x_0^m, \dots, x_{m-1}^m, sf)$$

which is equivalent to, by Lemma 4.6,

$$[\mathbf{2016.01.15.eq6}] f(i) = (x_0^m, \dots, x_{m-1}^m, sf)^*(\iota_m^1(f(i))) \tag{59}$$

for $i = 0, \dots, n-2$ and

$$[\mathbf{2016.01.15.eq7}] f(n-1) = (x_0^m, \dots, x_{m-1}^m, sf)^*(x_m^{m+1}) \tag{60}$$

For the first series of equalities we get, by inserting the coercion RR and rewriting of the right hand side, the following

$$(x_0^m, \dots, x_{m-1}^m, sf)^*(\iota_m^1(f(i))) = ((L(\iota_m^1))^* \circ (x_0^m, \dots, x_{m-1}^m, sf)^*)(f(i)) =$$
$$(L(\iota_m^1) \circ (x_0^m, \dots, x_{m-1}^m, sf)^*)^*(f(i)) = (L(\iota_m^1) \circ_T (x_0^m, \dots, x_{m-1}^m, sf))^*(f(i)) =$$
$$(x_0^m, \dots, x_{m-1}^m)^*(f(i)) = \eta_m^*(f(i)) = Id_{RR(m)}(f(i)) = f(i)$$

where the fourth equality is by (56).

Equality (60) gives us, by (49) that $sf = f(n-1)$. □

Recall from [45] that for a C-system CC one defines $\widetilde{Ob}(CC)$ as the subset of $Mor(CC)$ which consists of morphisms s of the form $ft(X) \to X$ such that $l(X) > 0$ and $s \circ p_X = Id_{ft(X)}$.

Lemma 4.17. **[2015.08.24.l1]** *Let* $f : \widehat{m} \to \widehat{n}$ *and let* $s : \widehat{n} \to \widehat{n+1}$ *be an element of* $\widetilde{Ob}$*. Then one has*

$$f^*(s) = (x_0^m, \dots, x_{m-1}^m, f^*(s(n)))$$

Proof. The fact that the first m terms of the sequence representation of $fs = f^*(s)$ have the required form follows from Lemma 4.15. It remains to prove that

$$fs(m) = f^*(s(n)) = (s \circ_T f)(n)$$

The morphism $f^*(s)$, as a morphism over $\widehat{m}$ is defined by the equation

$$f^*(s) \circ_C q(f, \widehat{n+1}) = f \circ_C s$$

which is equivalent, by Lemma 4.13, to $qq(f) \circ_T fs = s \circ_T f$. Therefore

$$(s \circ_T f)(n) = (qq(f) \circ_T fs)(n) = (fs)^*(qq(f)(n)) = (fs)^*(x_m^{m+1}) = (fs)^*(\eta_{m+1}(m)) = (\eta_{m+1} \circ (fs)^*)(m) = fs(m)$$

The lemma is proved. □

Lemma 4.18. **[2015.08.29.l1]** *Let* $f : \widehat{m} \to \widehat{n}$ *and let* $s : \widehat{n+i} \to n\widehat{+i+}1$ *be an element of* $\widetilde{Ob}$*. Then one has*

(61) $$[\mathbf{2015.08.29.eq1}] f^*(s) = (x_0^{m+i}, \dots, x_{m+i-1}^{m+i}, qq^i(f)^*(s(n+i)))$$

Proof. The morphisms involved in the proof can be seen on the following diagram

$$\begin{array}{ccc} \widehat{m+i} & \xrightarrow{qq^i(f)} & \widehat{n+i} \\ {\scriptstyle f^*(s)}\downarrow & & \downarrow{\scriptstyle s} \\ m\widehat{+i+}1 & \xrightarrow{qq^{i+1}(f)} & n\widehat{+i+}1 \\ {\scriptstyle p_{m+i+1,i+1}}\downarrow & & \downarrow{\scriptstyle p_{n+i+1,i+1}} \\ \widehat{m} & \xrightarrow{f} & \widehat{n} \end{array}$$

The morphism s is a morphism from $Id_{\widehat{n+i}}$ to $p_{\widehat{n+i+1}}$ over $\widehat{n+i}$. Therefore, we may apply Lemma 3.2 obtaining the equality

$$f^*(s) = (qq^i(f))^*(s)$$

On the other hand by Lemma 4.17 we have

$$qq^i(f)^*(s) = (x_0^{m+i}, \dots, x_{m+i-1}^{m+i}, qq^i(f)^*(s(n+i))).$$

The lemma is proved. □

Another operation that we would like to have an explicit form of is operation δ. For a C-system CC and an object Γ in CC such that $l(\Gamma) > 0$ one defines δ_Γ as $s_{Id(\Gamma)}$ (cf. [45, Section 3]).

Lemma 4.19. **[2015.08.24.l5]** *In* $C(\mathbf{RR})$ *one has:*

$$\delta_{\widehat{n}} = (x_0^n, \dots, x_{n-1}^n, x_{n-1}^n)$$

Proof. It follows from Lemma 4.16 since $Id_{\widehat{n}} = (x_0^n, \dots, x_{n-1}^n)$. □

Problem 4.20. *To construct a bijection*

$$[\mathbf{2015.08.24.eq9}]mb_{\mathbf{RR}} : \widetilde{Ob}(C(\mathbf{RR})) \to \amalg_{n\in\mathbf{N}} RR(n) \tag{62}$$

Construction 4.21. [**2015.08.22.constr3**] For $s : \widehat{n} \to \widehat{n+1}$ define

$$mb_{\mathbf{RR}}(s) = (n, s(n))$$

To show that this is a bijection let us construct the inverse bijection. For $n \in \mathbf{N}$ and $o \in RR(n)$ set

$$mb^!_{\mathbf{RR}}(n, o) = (x^n_0, \ldots, x^n_{n-1}, o)$$

The fact that these functions are mutually inverse follows easily from Lemma 4.15. □

Our next goal is to describe operations T', $\widetilde{T}'$, S', $\widetilde{S}'$ and δ' obtained from operations T, $\widetilde{T}$, S, $\widetilde{S}$ and δ that were introduced at the end of Section 3 in [45] through transport by means of the bijection (62).

Let us first recall the definition of operations T, $\widetilde{T}$, S, $\widetilde{S}$ and δ associated with a general C-system CC.

Definition 4.22. *[***2015.08.26.def1***] Let CC be a C-system. We will write Ob for $Ob(CC)$ and $\widetilde{Ob}$ for $\widetilde{Ob}(CC)$.*

(1) Operation T is defined on the set

$$T_{dom} = \{\Gamma, \Gamma' \in Ob \,|\, l(\Gamma) > 0 \text{ and } \Gamma' > ft(\Gamma)\}$$

and takes values in Ob. For $(\Gamma, \Gamma') \in T_{dom}$ one defines

$$T(\Gamma, \Gamma') = p^*_\Gamma(\Gamma')$$

(2) Operation $\widetilde{T}$ is defined on the set

$$\widetilde{T}_{dom} = \{\Gamma \in Ob, s \in \widetilde{Ob} \,|\, l(\Gamma) > 0 \text{ and } \partial(s) > ft(\Gamma)\}$$

and takes values in $\widetilde{Ob}$. For $(\Gamma, s) \in \widetilde{T}_{dom}$ one defines

$$\widetilde{T}(\Gamma, s) = p^*_\Gamma(s)$$

(3) Operation S is defined on the set

$$S_{dom} = \{r \in \widetilde{Ob}, \Gamma \in Ob \,|\, \Gamma > \partial(r)\}$$

and takes values in Ob. For $(r, \Gamma) \in S_{dom}$ one defines

$$S(r, \Gamma) = r^*(\Gamma)$$

(4) Operation $\widetilde{S}$ is defined on the set

$$\widetilde{S}_{dom} = \{r, s \in \widetilde{Ob} \,|\, \partial(s) > \partial(r)\}$$

and takes values in $\widetilde{Ob}$. For $(r, s) \in \widetilde{S}_{dom}$ one defines

$$S(r, s) = r^*(s)$$

(5) *Operation* δ *is defined on the set*

$$\delta_{dom} = \{\Gamma \in Ob \,|\, l(\Gamma) > 0\}$$

and takes values in $\widetilde{Ob}$. *For* $\Gamma \in \delta_{dom}$ *one defines* $\delta(\Gamma)$ *as* s_{Id_Γ}.

Define, for any Jf-relative monad **RR** operations $\theta_{m,n} = \theta^{\mathbf{RR}}_{m,n}$ such that for $m, n \in \mathbf{N}$, $n > m$ and $r \in RR(m)$, $s \in RR(n)$ one has

$$[\mathbf{2015.09.07.eq1}]\theta_{m,n}(r,s) = (qq^{n-m-1}(x^m_0, \ldots, x^m_{m-1}, r))^*(s) =$$

$$(x^{n-1}_0, \ldots, x^{n-1}_{m-1}, \iota^{n-m-1}_m(r), x^{n-1}_m, \ldots, x^{n-1}_{n-2})^*(s) \tag{63}$$

Theorem 4.23. *[***2015.08.26.th1***] Let* $Ob = Ob(C(\mathbf{RR}))$ *and let* $\widetilde{Ob}'$ *be the right hand side of (62). One has:*

(1) *Operation* T' *is defined on the set*

$$T'_{dom} = \{\widehat{m}, \widehat{n} \in Ob \,|\, m > 0 \text{ and } n > m-1\}$$

and is given by

$$T'(\widehat{m}, \widehat{n}) = \widehat{n+1}$$

(2) *Operation* $\widetilde{T}'$ *is defined on the set*

$$\widetilde{T}'_{dom} = \{\widehat{m} \in Ob, (n,s) \in \widetilde{Ob}' \,|\, m > 0 \text{ and } n+1 > m-1\}$$

and is given by

$$\widetilde{T}'(\widehat{m}, (n,s)) = (n+1, \partial^{m-1}_n(s))$$

(3) *Operation* S' *is defined on the set*

$$S'_{dom} = \{(m,r) \in \widetilde{Ob}', \widehat{n} \in Ob \,|\, n > m+1\}$$

and is given by

$$S'((m,r), \widehat{n}) = \widehat{n-1}$$

(4) *Operation* $\widetilde{S}'$ *is defined on the set*

$$\widetilde{S}'_{dom} = \{(m,r) \in \widetilde{Ob}', (n,s) \in \widetilde{Ob}' \,|\, n > m\}$$

and is given by

$$\widetilde{S}'((m,r),(n,s)) = \theta_{m,n}(r,s)$$

(5) *Operation* δ' *is defined on the subset*

$$\delta'_{dom} = \{\widehat{n} \in Ob \,|\, n > 0\}$$

and is given by

$$\delta'(\widehat{n}) = (n, x^n_{n-1})$$

Proof. We have:

(1) Operation T' is the same as operation T for $C(\mathbf{RR})$ since $\widetilde{Ob}$ is not involved in it. The form of T'_{dom} is obtained by unfolding definitions and the formula for the operation itself follows from Lemma 4.14(2).

(2) Operation $\widetilde{T}'$ is defined on the set of pairs $(\widehat{m} \in Ob, (n,s) \in \widetilde{Ob}')$ such that $m > 0$ and $\partial(mb^!_{\mathbf{RR}}(n,s)) > m-1$. Since $\partial(mb^!_{\mathbf{RR}}(n,s)) = n+1$ we obtain the required domain of definition. The formula by the operation itself is obtained immediately by combining Lemma 4.18 and Lemma 4.10.

(3) Operation S' is defined on the set of pairs $((m,r) \in \widetilde{Ob}', \widehat{n} \in Ob)$ where $n > \partial(mb^!_{\mathbf{RR}}(m,r))$. Since $\partial(mb^!_{\mathbf{RR}}(m,r)) = m+1$ we obtain the required domain of definition. The operation itself is given by

$$S'((m,r),n) = (mb^!_{\mathbf{RR}}(m,r))^*(\widehat{n}) = (x_0^m, \dots, x_{m-1}^m, r)^*(\widehat{n}) = \widehat{n + m - (m+1)} = \widehat{n-1}$$

(4) Operation $\widetilde{S}'$ is defined on the set of pairs $(m,r),(n,s) \in \widetilde{Ob}'$ such that $\partial(mb^!_{\mathbf{RR}}(n,s)) > \partial(mb^!_{\mathbf{RR}}(m,r))$ which is equivalent to $n > m$. The formula for the operation itself is obtained immediately by combining Lemma 4.18 with $i = n-m-1$ and Lemma 4.11.

(5) Operation δ' is defined on the subset $\widehat{n} \in Ob$ such that $n > 0$ and is given by

$$\delta'(\widehat{n}) = mb_{\mathbf{RR}}(\delta(\widehat{n})) = mb_{\mathbf{RR}}((x_0^n, \dots, x_{n-1}^n, x_{n-1}^n)) = (n, x_{n-1}^n)$$

The theorem is proved. □

The length function on $Ob = \mathbf{N}$ is the identity. Of the remaining three operations that define the pre-B-system structure on the pair of sets $(Ob, \widetilde{Ob}')$ - pt, ft and ∂', the first two are described above and ∂' is given by $\partial'((m,r)) = m+1$.

This completes the description of the pre-B-system structure on $(Ob, \widetilde{Ob}')$ that is obtained by the transport of structure from the standard pre-B-system structure on $(Ob, \widetilde{Ob})$ by means of the pair of isomorphisms Id and $mb_{\mathbf{RR}}$.

Remark 4.24. **[2015.08.29.rem2]** Conjecturally, a C-system can be reconstructed (up to an isomorphism) from the sets Ob and $\widetilde{Ob}$ equipped with the length function $l : Ob \to \mathbf{N}$, the distinguished object $pt \in Ob$ and operations $ft, \partial, T, \widetilde{T}, S, \widetilde{S}$ and δ. Combining this conjecture with Theorem 4.23 we conclude that the C-system $C(\mathbf{RR})$ and, therefore, the relative monad $\mathbf{RR}$, can be reconstructed from the sets $RR(n)$ with distinguished elements x_i^n and equipped with operations ∂_n^i and $\theta_{m,n} : RR(m) \times RR(n) \to RR(n-1)$ for $n > m$.

Using Remark 4.4 this can be compared with the assertion of [16, Theorem 3.3] that the category of abstract clones is equivalent to the category of substitution systems of [16, Definition 3.1]. In such a comparison the operation ζ of substitution systems of the form $RR(n+1) \times RR(n) \to RR(n)$ is the same as the operation $(s,r) \mapsto \theta_{n,n+1}(r,s)$.

Remark 4.25. **[2015.08.29.rem1]** Let lRR be the disjoint union of $RR(n)$ for all n. Then we can sum up all of the operations that we need to consider as follows:

(1) a function $l : lRR \to \mathbf{N}$,
(2) a function $\eta : \mathbf{N} \to lRR$ that takes n to $x_0^n = \eta_n(0)$,
(3) a function $\partial : \{r \in lRR, i \in \mathbf{N} \,|\, l(r) \geq i\} \to lRR$,
(4) a function $\theta : \{r, s \in lRR, \,|\, l(r) > l(s)\} \to lRR$,

such that

(1) for all $n \in \mathbf{N}$, $l(\eta_n) = n+1$,
(2) for all $r \in lRR$, $i \in \mathbf{N}$ such that $l(r) \geq i$, $l(\partial(r,i)) = l(r)+1$,
(3) for all $r, s \in lRR$ such that $l(s) > l(r)$ one has $l(\sigma(r,s)) = l(s)-1$.

It should be possible to describe, by a collection of further axioms on these operations, a full subcategory in the category whose objects are sets lRR with operations of the form l, η, ∂ and θ that is equivalent to the category of Jf-relative monads or, equivalently, the category of Lawvere theories or Fiore-Plotkin-Turi substitution algebras.

Remark 4.26. **[2015.08.29.rem1b]** It seems at first unclear why it should be possible to realize the action of the symmetric group on $RR(n)$ using operations of Remark 4.24 since they all seem to respect, in some sense, the linear ordering of the sets $stn(n)$.

In the substitution notation of Remark 4.27, given r in $RR(m)$ and E in $RR(n)$,

$$\theta_{m,n}(r, E) = E[r/x_m, x_m/x_{m+1}, \ldots, x_{n-2}/x_{n-1}],$$

i.e., the operation $\theta_{m,n}$ corresponds to the substitution of an expression in variables $x_0, \ldots, x_{m-1}$ for the variable x_m in an expression in variables $x_0, \ldots, x_n$ followed by a downshift of the indexes of the variables with the higher index.

The operation ∂^i_n and the constants $x_n := x^{n+1}_n$ are similarly defined in terms of linear orderings.

To see how it is, nevertheless, possible to realize, for example, the permutation of x_0 and x_1 consider the following. First let, for all $i, n \in \mathbf{N}$,

$$\iota^i_n = \partial^{n+i-1}_{n+i-1} \circ \cdots \circ \partial^n_n : RR(n) \to RR(n+i)$$

Then define for all $i, n \in \mathbf{N}$, $n \geq i+1$ an element $x^n_i \in RR(n)$ by the formula

$$x^n_i = \iota^{n-i-1}_{i+1}(x_i)$$

such that, in particular, $x^{n+1}_n = x_n$.

Define now a function $\psi : RR(2) \to RR(2)$ by the formula

$$\psi = \partial^0_2 \circ \partial^0_3 \circ \theta_{3,4}(x^3_0, -) \circ \theta_{2,3}(x^2_1, -)$$

One can verify that for any Jf-relative monad RR, $\psi = \sigma$ where σ is the permutation of 0 and 1 in $stn(2)$.

In the substitution notation this can be seen as follows:

$$\psi(E(x^2_0, x^2_1)) = \theta_{2,3}(x^2_1, \theta_{3,4}(x^3_0, \partial^0_3(\partial^0_2(E(x^2_0, x^2_1))))) = \theta_{2,3}(x^2_1, \theta_{3,4}(x^3_0, \partial^0_3(E(x^3_1, x^3_2)))) =$$
$$\theta_{2,3}(x^2_1, \theta_{3,4}(x^3_0, E(x^4_2, x^4_3))) = \theta_{2,3}(x^2_1, E(x^3_2, x^3_0)) = E(x^2_1, x^2_0)$$

4.3. The C-system $C(\mathbf{RR}, \mathbf{LM})$. [CRRLM]

In this paper we are interested in the Jf-relative monads $\mathbf{RR}$. The corresponding Kleisli categories are the categories opposite to the categories $C(\mathbf{RR})$ underlying the C-systems considered above. Therefore, left modules over a Jf-monad $\mathbf{RR}$ with values in $Sets$ are the presheaves on $C(RR)$, i.e., the contravariant functors from $C(RR)$ to $Sets$.

Let $\mathbf{LM} = (LM, LM_{Mor})$ be such a presheaf. The morphism component LM_{Mor} of $\mathbf{LM}$ is a function that sends a morphism f from $\widehat{m}$ to $\widehat{n}$ in $C(\mathbf{RR})$ to a function $LM_{Mor}(f) \in Fun(LM(\widehat{n}), LM(\widehat{m}))$, i.e., we have for each $m, n \in \mathbf{N}$ a function

$$RR(n,m) \to Fun(LM(\widehat{n}), LM(\widehat{m}))$$

We will use this function as a coercion so that, for $f \in RR(n,m)$ and $E \in LM(\widehat{n})$ the expression $f(E)$ is assumed to be expanded into $LM_{Mor}(f)(E)$ when needed.

Remark 4.27. [2015.08.18.rem1] If we think of $E \in LM(\widehat{n})$ as of an expression in variables $0, \dots, n-1$ then the action of $RR(n,m)$ on $LM(\widehat{n})$ can be thought of as the substitution. This analogy can be used to introduce the notation when for $f = (f(0), \dots, f(n-1)) \in RR(n,m)$ and $E \in LM(\widehat{n})$ one writes $f(E)$ as

$$f(E) = E[f(0)/0, \dots, f(n-1)/n-1]$$

For example, in this notation we have

$$\partial_n^i(E) = E[0/0, \dots, i-1/i-1, i+1/i, \dots, n/n-1]$$

Similarly, for $E \in LM(\widehat{n+2})$ one has

$$\sigma_n^i(E) = E[0/0, \dots, i/i, i/i+1, \dots, n/n+1]$$

and $\iota_n^i(E)$ is "the same expression" but considered as an expression of $n+i$ variables.

Example 4.28. [2015.09.07.rem3] An important example of $\mathbf{LM}$ is given by the functor defined on objects by $\widehat{n} \mapsto RR(n)$ and on morphisms by

$$f \mapsto (s \mapsto f^*(s))$$

for $f : \widehat{m} \to \widehat{n}$ and $s \in RR(n)$. We will denote this functor by the same symbol $\mathbf{RR}$ as the underlying Jf-relative monad.

This functor is isomorphic to the (contravariant) functor represented by the object $\widehat{1}$ but it is not equal to this functor since the set of elements of the form $((\widehat{n}, \widehat{1}), r')$ where $r' \in RR(1,n)$ is isomorphic but not equal to the set $RR(n)$.

Let $C(\mathbf{RR}, \mathbf{LM}) = C(\mathbf{RR})[\mathbf{LM}]$ be the $\mathbf{LM}$-extension of the C-system $C(\mathbf{RR})$. The role of these C-systems in the theory of type theories is that the term C-systems of the raw syntax of dependent type theories are of this form and therefore the term C-systems of dependent type theories are regular sub-quotients of such C-systems and can be studied using the description of the regular sub-quotients given in [45].

By construction,

$$[\mathbf{2016.01.21.eq3}] Ob(C(\mathbf{RR}, \mathbf{LM})) = \amalg_{n \in \mathbf{N}} Ob_n(\mathbf{RR}, \mathbf{LM}) \tag{64}$$

where

$$Ob_n(\mathbf{RR}, \mathbf{LM}) = \mathbf{LM}(\widehat{0}) \times \dots \times \mathbf{LM}(\widehat{n-1})$$

and therefore objects of $C(\mathbf{RR}, \mathbf{LM})$ are pairs of the form (n, Γ) where Γ is a sequence $(T_0, \dots, T_{n-1})$ where $T_i \in LM(\widehat{i})$. While the number n in a pair (n, Γ) is an object of $C(\mathbf{RR})$ we will not add the $\widehat{}$ diacritic to it since no confusion of the kind possible with objects of $C(RR)$ and objects of F can arise. We may sometimes omit n from our notation altogether since it can be recovered from Γ. Similarly, while the morphisms of $C(\mathbf{RR}, \mathbf{LM})$ are given by iterated pairs of the form $(((m, \Gamma), (n, \Gamma')), ((\widehat{m}, \widehat{n}), f))$

where $f \in \mathbf{RR}(n,m)$ we will sometimes write them as $f : (m,\Gamma) \to (n,\Gamma')$ or $f : \Gamma \to \Gamma'$ or even just as f.

Let us also recall that for two objects $X = (m,(T_0,\dots,T_{m-1})))$ and $Y = (n+1,(T'_0,\dots,T'_n))$ and a morphism $f : X \to ft(Y)$ the object $f^*(Y)$ is given by the formula

$$[\mathbf{2015.09.09.eq3old}]f^*(Y) = (m+1,(T_0,\dots,T_{m-1},f(T'_n))) \tag{65}$$

and the morphism $q(f,Y) : f^*(Y) \to Y$ by the formula $q(f,Y) = qq(f)$.

Lemma 4.29. **[2015.08.26.l8]** *Let* $X = (m,(T_0,\dots,T_{m-1}))$ *and* $Y = (n,(T_0,\dots,T_{n-2},T))$ *where* $m > n-1$. *Then one has*

$$p_Y^*(X) = (m+1,(T_0,\dots,T_{n-2},T,\partial^{n-1}_{n-1}(T_{n-1}),\dots,\partial^{n-1}_{m-1}(T_{m-1})))$$

Proof. We want to apply Lemma 3.12. We have $lx = n$, $ly = m$. The morphism p_Y is of the form

$$p_Y = p_{\widehat{n}} : (n,(T_0,\dots,T_{n-2},T)) \to (n-1,(T_0,\dots,T_{n-2}))$$

and

$$(n-1,(T_0,\dots,T_{n-2})) = ft^i((m,(T_0,\dots,T_{m-1})))$$

where $i = m-n+1$. Therefore,

$$p_Y^*(X) = p_Y^*(X,i) =$$

$$(p_{\widehat{n}}^*(\widehat{m},i),(T_0,\dots,T_{n-2},T,q(p_{\widehat{n}},ft^i(\widehat{m}),0)(T_{m-i}),\dots,q(p_{\widehat{n}},ft(\widehat{m}),i-1)(T_{m-1})) =$$

$$(m+1,(T_0,\dots,T_{n-2},T,\iota^1_{n-1}(T_{n-1}),\dots,qq^{i-1}(L(\iota^1_{n-1}))(T_{m-1}))) =$$

$$(m+1,(T_0,\dots,T_{n-2},T,\partial^{n-1}_{n-1}(T_{n-1}),\dots,\partial^{n-1}_{m-1}(T_{m-1})))$$

where the third equality is by Lemma 4.14(2) and the fourth one by Lemma 4.10. □

Lemma 4.30. **[2015.08.22.l5]** *A morphism* $f : X \to Y$, *where* $l(Y) = n+1$ *and* $f \in R(n+1,n)$ *belongs to* $\widetilde{Ob}(C(\mathbf{RR},\mathbf{LM}))$ *if and only if* $X = ft(Y)$ *and* $f(i) = x^n_i$ *for* $i = 0,\dots,n-1$.

Proof. It follows immediately from Lemma 4.8. □

The following analog of Lemma 4.16 for the C-system $C(\mathbf{RR},\mathbf{LM})$ provides us with the explicit form of the operation $f \mapsto s_f$.

Lemma 4.31. **[2015.09.09.l2]** *Let* $f : X \to Y$, $f = (f(0),\dots,f(n-1))$ *where* $n > 0$. *Then* $s_f : X \to (ft(f))^*(Y)$,

$$[\mathbf{2015.09.09.eq1}]s_f = (x^m_0,\dots,x^m_{m-1},f(n-1)) \tag{66}$$

where $ft(f) = f \circ p_Y$ *and* $m = l(X)$.

Proof. By definition s_f is a morphism from X to $(ft(f))^*(Y)$. Therefore it is sufficient to show that the left hand side of (66) agrees with the right hand side after application of the homomorphism $tr_{\mathbf{LM}}$ and our goal follows from Lemma 4.16. □

Lemma 4.32. *[**2015.09.03.l1**] For $i > 0$, $f : X \to ft^i(Y)$ and $s : ft(Y) \to Y$ in $\widetilde{Ob}(C(\mathbf{RR}, \mathbf{LM}))$ one has $s : f^*(ft(Y)) \to f^*(Y)$,*

$$f^*(s) = (x_0^{m+i-1}, \ldots, x_{m+i-2}^{m+i-1}, qq^{i-1}(f)^*(s(n+i-1)))$$

where $m = l(\Gamma')$ and $n = l(\Gamma)$.

Proof. Since $tr_{\mathbf{LM}}$ is fully faithful, it is sufficient, in order to verify the equality of two morphisms to verify that their domain and codomain are equal and that their images under $tr_{\mathbf{LM}}$ are equal. For the domain and codomain it follows from the definition of f^* on morphisms. For the images under $tr_{\mathbf{LM}}$ it follows from the fact that $tr_{\mathbf{LM}}$ is a homomorphism of C-systems, Lemma 3.3(4) and Lemma 4.18. □

Problem 4.33. *[**2015.08.22.prob1**] To construct a bijection*

(67) $$[\mathbf{2009.10.15.eq2}]mb_{\mathbf{RR},\mathbf{LM}} : \widetilde{Ob}(C(\mathbf{RR}, \mathbf{LM})) \to \coprod_{n\in\mathbf{N}} Ob_{n+1}(\mathbf{RR}, \mathbf{LM}) \times R(n)$$

Construction 4.34. **[2015.08.22.constr1] [2014.06.30.l2]** Let $s \in \widetilde{Ob}(C(\mathbf{RR}, \mathbf{LM}))$. Then $s : ft(X) \to X$, $s \in R(n, n+1)$ and $X = (n+1, \Gamma)$. We set:

$$mb_{\mathbf{RR},\mathbf{LM}}(s) = (n, (\Gamma, s(n)))$$

To show that this is a bijection let us construct an inverse. For $n \in \mathbf{N}$, $\Gamma \in Ob_{n+1}(\mathbf{RR}, \mathbf{LM})$ and $o \in R(n)$ let

$$mb^!_{\mathbf{RR},\mathbf{LM}}(n, (\Gamma, o)) = ((ft((n+1, \Gamma)), (n+1, \Gamma)), (x_0^n, \ldots, x_{n-1}^n, o))$$

This is a morphism from $ft(X)$ to X where $X = (n+1, \Gamma)$. The equation $mb^!_{\mathbf{RR},\mathbf{LM}}(n, (\Gamma, o)) \circ p_X = Id_{ft(X)}$ follows from Lemma 4.30.

Let us show now that $mb_{\mathbf{RR},\mathbf{LM}}$ and $mb^!_{\mathbf{RR},\mathbf{LM}}$ are mutually inverse bijections. Let $s \in \widetilde{Ob}$ be as above, then:

$$mb^!_{\mathbf{RR},\mathbf{LM}}(mb_{\mathbf{RR},\mathbf{LM}}(s)) = mb^!_{\mathbf{RR},\mathbf{LM}}(n, (\Gamma, s(n))) = ((ft(X), X), (x_0^n, \ldots, x_{n-1}^n, s(n))) = s$$

where the last equality follows from the assumption that $s \in \widetilde{Ob}$ and Lemma 4.30.

On the other hand for $\Gamma \in Ob_{n+1}(\mathbf{RR}, \mathbf{LM})$ and $o \in R(n)$ we have

$$mb_{\mathbf{RR},\mathbf{LM}}(mb^!_{\mathbf{RR},\mathbf{LM}}(n, (\Gamma, o))) = mb_{\mathbf{RR},\mathbf{LM}}(ft((n+1, \Gamma)), ((n+1, \Gamma), (x_0^n, \ldots, x_{n-1}^n, o))) = (n, (\Gamma, o))$$

This completes Construction 4.34. □

Lemma 4.35. *[**2015.09.09.l3**] Let $f : X \to Y$, $f = (f(0), \ldots, f(n-1))$ where $X = (m, (T_0, \ldots, T_{m-1}))$, $Y = (n, (T'_0, \ldots, T'_{n-1}))$. Then one has*

$$mb_{\mathbf{RR},\mathbf{LM}}(s_f) = (m, ((T_0, \ldots, T_{m-1}, (f(0), \ldots, f(n-2))(T'_{n-1})), f(n-1)))$$

Proof. It follows immediately from Lemma 4.31 and the formula for $mb_{\mathbf{RR},\mathbf{LM}}$. □

Consider operations T', $\widetilde{T}'$, S', $\widetilde{S}'$ and δ' obtained by transport by means of the bijection of Construction 4.34 from the operations T, $\widetilde{T}$, S and $\widetilde{S}$ and δ corresponding to the C-system $C(\mathbf{RR}, \mathbf{LM})$ (cf. Definition 4.22). Let us give an explicit description of these operations.

Recall that we defined, for any Jf-relative monad $\mathbf{RR}$, operations

$$\theta^{\mathbf{RR}}_{m,n} : RR(m) \times RR(n) \to RR(n-1)$$

For $\mathbf{LM}$ as above and $n > m$ define operations $\theta^{\mathbf{LM}}_{m,n}$ of the form

$$\theta^{\mathbf{LM}}_{m,n} : RR(m) \times LM(n) \to LM(n-1)$$

by the formula

$$[\mathbf{2015.09.07.eq2}]\theta^{\mathbf{LM}}_{m,n}(r, E) =$$

(68)
$$(qq^{n-m-1}(x^m_0, \ldots, x^m_{m-1}, r))(E) = (x^{n-1}_0, \ldots, x^{n-1}_{m-1}, \iota^{n-m-1}_m(r), x^{n-1}_m, \ldots, x^{n-1}_{n-2})(E)$$

where the second equality is the equality of Lemma 4.11. As in the case of $\theta^{\mathbf{RR}}_{m,n}$ we will often write $\theta_{m,n}$ instead of $\theta^{\mathbf{LM}}_{m,n}$ since the whether we consider $\theta^{\mathbf{RR}}$ or $\theta^{\mathbf{LM}}$ can be inferred from the type of the arguments.

Theorem 4.36. *[***2015.08.26.th2***] Let $Ob = Ob(C(\mathbf{RR}, \mathbf{LM}))$ and let $\widetilde{Ob}' = \widetilde{Ob}'(\mathbf{RR}, \mathbf{LM})$ be the right hand side of (67). One has:*

(1) Operation T' is defined on the set T'_{dom} of pairs $(m, \Gamma), (n, \Gamma') \in Ob$ where $\Gamma = (T_0, \ldots, T_{m-1})$, $\Gamma' = (T'_0, \ldots, T'_{n-1})$ such that $m > 0$, $n > m-1$ and $T_i = T'_i$ for $i = 0, \ldots, m-2$. It takes values in Ob and is given by

$$T((m, \Gamma), (n, \Gamma')) =$$
$$(n + 1, (T'_0, \ldots, T'_{m-2}, T_{m-1}, \partial^{m-1}_{m-1}(T'_{m-1}), \ldots, \partial^{m-1}_{n-1}(T'_{n-1})))$$

(2) Operation $\widetilde{T}'$ is defined on the set $\widetilde{T}'_{dom}$ of pairs $(m, \Gamma) \in Ob$, $(n, (\Gamma', s)) \in \widetilde{Ob}'$ where $\Gamma = (T_0, \ldots, T_{m-1})$, $\Gamma' = (T'_0, \ldots, T'_{n-1})$ such that $m > 0$, $n+1 > m-1$ and $T_i = T'_i$ for $i = 0, \ldots, m-2$. It takes values in $\widetilde{Ob}'$ and is given by

$$\widetilde{T}'((m, \Gamma), (n, (\Gamma', s))) = (n + 1, (T((m, \Gamma), (n, \Gamma')), \partial^{m-1}_n(s)))$$

(3) Operation S' is defined on the set of pairs $(m, (\Gamma, r)) \in \widetilde{Ob}'$, $(n, \Gamma') \in Ob$ where $\Gamma = (T_0, \ldots, T_m)$, $\Gamma' = (T'_0, \ldots, T'_{n-1})$ such that $n > m + 1$ and $T_i = T'_i$ for $i = 0, \ldots, m$. It takes values in the set Ob and is given by

$$S'((m, (\Gamma, r)), (n, \Gamma')) =$$
$$(n - 1, (T'_0, \ldots, T'_{m-1}, \theta_{m,m+1}(r, T'_{m+1}), \theta_{m,m+2}(r, T'_{m+2}), \ldots, \theta_{m,n-1}(r, T'_{n-1})))$$

(4) Operation $\widetilde{S}'$ is defined on the set of pairs $(m, (\Gamma, r)) \in \widetilde{Ob}'$, $(n, (\Gamma', s)) \in \widetilde{Ob}'$ where $\Gamma = (T_0, \ldots, T_m)$, $\Gamma' = (T'_0, \ldots, T'_n)$ such that $n > m$ and $T_i = T'_i$ for $i = 0, \ldots, m$. It takes values in $\widetilde{Ob}'$ and is given by

$$\widetilde{S}'((m, (\Gamma, r)), (n, (\Gamma', s))) = (n - 1, (S'((m, (\Gamma, r)), (n + 1, \Gamma'))), \theta_{m,n}(r, s))$$

(5) Operation δ' is defined on the subset of (m, Γ) in Ob such that $m > 0$. It takes values in $\widetilde{Ob}'$ and is given by

$$\delta'((m, \Gamma)) = (m, (T((m, \Gamma), (m, \Gamma)), x^m_{m-1}))$$

Proof. In the proof we will write mb and $mb^!$ instead of $mb_{\mathbf{RR},\mathbf{LM}}$ and $mb^!_{\mathbf{RR},\mathbf{LM}}$. We have:

(1) Operation T' is the same as operation T for $C(\mathbf{RR}, \mathbf{LM})$ since $\widetilde{Ob}$ is not involved in it. The form of T'_{dom} is obtained by unfolding definitions.
The operation itself is given by

$$T'((m,\Gamma),(n,\Gamma')) = p^*_{(m,\Gamma)}((n,\Gamma')) =$$
$$(m,(T_0,\dots,T_{m-1},\partial^{m-1}_{m-1}(T'_{m-1}),\dots,\partial^{m-1}_{n-1}(T'_{n-1})))$$

where the first equality is by Definition 4.22(1) and the second by Lemma 4.29.

(2) Operation $\widetilde{T}'$ is defined on the set of pairs $(m,\Gamma)\in Ob$, $(n,(\Gamma',s))\in \widetilde{Ob}'$ such that $m>0$ and $\partial(mb^!(n,(\Gamma',s)))>ft(m,\Gamma)$ and takes values in $\widetilde{Ob}'$. Since $\partial(mb^!(n,(\Gamma',s)) = (n+1,\Gamma')$ we obtain the required domain by unfolding definitions.
To verify the formula for the operation itself consider the equalities:

$$\widetilde{T}'((m,\Gamma),(n,(\Gamma',s))) = mb(p^*_{(m,\Gamma)}(mb^!(n,(\Gamma',s)))) =$$
$$mb(p^*_{(m,\Gamma)}((ft((n+1,\Gamma')),((n+1,\Gamma'),(x^n_0,\dots,x^n_{n-1},s)))))$$

where the first equality is by Definition 4.22(2). By Lemma 4.32 we can extend these equalities as follows:

$$mb(p^*_{(m,\Gamma)}((ft((n+1,\Gamma')),((n+1,\Gamma'),(x^n_0,\dots,x^n_{n-1},s))))) =$$
$$mb(p^*_X(ft(Y)),(p^*_X(Y),(x^{n+1}_0,\dots,x^{n+1}_n,(qq^{n-m+1}(\iota^1_{m-1}))(s)))) =$$
$$(n+1,(p^*_X(Y),\partial^{m-1}_n(s))) = (n+1,(T((m,\Gamma),(n+1,\Gamma')),\partial^{m-1}_n(s)))$$

where $X=(m,\Gamma)$, $Y=(n+1,\Gamma')$, the first equality is by Lemma 4.32, the second by Lemma 4.10 and the third by Definition 4.22(1).

(3) Operation S' is defined on the set of pairs $((m,(\Gamma,r))\in\widetilde{Ob}',(n,\Gamma')\in Ob)$ such that $(n,\Gamma')>\partial(mb^!(m,(\Gamma,r)))$ and takes values in Ob. Since $\partial(mb^!(m,(\Gamma,r))) = (m+1,\Gamma)$ we obtained the required domain of definition. The operation itself is given by

(69) $$[\mathbf{2016.01.21.eq2}]S'((m,(\Gamma,r)),(n,\Gamma')) = (mb^!((m,(\Gamma,r))))^*((n,\Gamma'))$$

Next we have

$$(mb^!((m,(\Gamma,r))))^*((n,\Gamma')) =$$
$$((ft(A),A),(x^m_0,\dots,x^m_{m-1},r))^*(B) = ((ft(A),A),(x^m_0,\dots,x^m_{m-1},r))^*(B,i)$$

where $A=(m+1,\Gamma)$, $B=(n,\Gamma')$ and $i=n-m-1$. To apply Lemma 3.12 we should take $X=\widehat{m}$, $lx=m$ and $Y=\widehat{n}$, $ly=n$ and $f=((ft(A),A),(x^m_0,\dots,x^m_{m-1},r))$. Let further $rr=(x^m_0,\dots,x^m_{m-1},r)$. Then we can extend these equalities as follows

$$f^*((n,\Gamma'),i) = (rr^*(\widehat{n},i),(T_0,\dots,T_{m-1},q(rr,ft^i(\widehat{n}),0)(T'_{m+1}),\dots,q(rr,ft(\widehat{n}),i-1)(T'_{n-1})))$$
$$(n-1,(T_0,\dots,T_{m-1},rr(T'_{m+1}),\dots,qq^{n-m-2}(rr)(T'_{n-1}))) =$$
$$(n-1,(T'_0,\dots,T'_{m-1},rr(T'_{m+1}),qq(rr)(T'_{m+2}),\dots,qq^{n-m-2}(rr)(T'_{n-1})))$$

where the last equality holds by the assumption that $T_i=T'_i$ for $i=0,\dots,m$.
The required formula follows from the equality

$$qq^j(rr)(T'_{m+j+1}) = \theta_{m,m+j+1}(r,T'_{m+j+1}).$$

(4) Operation $\widetilde{S}'$ is defined on the set of pairs $(m,(\Gamma,r)) \in \widetilde{Ob}'$, $(n,(\Gamma',s)) \in \widetilde{Ob}'$ such that

$$[\mathbf{2016.01.21.eq1}]\partial(mb^!((n,(\Gamma',s)))) > \partial(mb^!(m,(\Gamma,r))) \tag{70}$$

and takes values in $\widetilde{Ob}'$. The inequality (70) is equivalent to

$$(n+1,\Gamma') > (m+1,\Gamma)$$

which is, in turn, equivalent to the conditions in the theorem. In the computation below let us sometimes abbreviate $((X,Y),f)$ to f. Let

$$rr = (x_0^m,\ldots,x_{m-1}^m,r)$$
$$ss = (x_0^n,\ldots,x_{n-1}^n,s)$$

Then the operation itself is given by:

$$\widetilde{S}'((m,(\Gamma,r)),(n,(\Gamma',s))) = mb((mb^!(m,(\Gamma,r)))^*(mb^!((n,(\Gamma',s))))) = mb(rr^*ss) =$$
$$mb((x_0^{n-1},\ldots,x_{n-2}^{n-1},(qq^{n-m-1}(rr))(s))) = (n-1,(rr^*((n+1,\Gamma')),(qq^{n-m-1}(rr))(s))) =$$
$$(n-1,(S'((m,(\Gamma,r)),(n+1,\Gamma'))),\theta_{m,n}(r,s))$$

where the third equality is by Lemma 4.32 and the fifth by (69) and the definition of $\theta_{m,n}(r,s)$.

(5) Operation δ' is defined on the subset $(m,\Gamma) \in Ob$ such that $m>0$ and is given by

$$\delta'((m,\Gamma)) = mb(\delta((m,\Gamma)))$$

Therefore it is sufficient to show that

$$\delta((m,\Gamma)) = (((m,\Gamma),p^*_{(m,\Gamma)}((m,\Gamma))),(x_0^m,\ldots,x_{m-1}^m,x_{m-1}^m))$$

By Definition 4.22(5), $\delta((m,\Gamma))$ is a morphism from (m,Γ) to $p^*_{(m,\Gamma)}((m,\Gamma))$. Therefore, since $tr_{\mathbf{LM}}$ is a fully faithful functor it is sufficient to show that

$$tr_{\mathbf{LM}}(\delta((m,\Gamma))) = ((\widehat{m},\widehat{m+1}),(x_0^m,\ldots,x_{m-1}^m,x_{m-1}^m))$$

which follows from Lemma 3.3(5) and Lemma 4.19.

This completes the proof of the theorem. □

The length function on Ob is described above. Of the remaining three operations that define the pre-B-system structure on the pair of sets $(Ob,\widetilde{Ob}')$ - pt, ft and ∂', the first two are described above as well and ∂' is given by $\partial'((m,(\Gamma,r))) = (m+1,\Gamma)$.

This completes the description of the pre-B-system structure on $(Ob,\widetilde{Ob}')$ that is obtained by the transport of structure from the standard pre-B-system structure on $(Ob,\widetilde{Ob})$ by means of the pair of isomorphisms Id and $mb_{\mathbf{RR},\mathbf{LM}}$.

Remark 4.37. [**2015.09.13.rem1**] Given an Jf-relative monad $\mathbf{RR}$ in the form $l\mathbf{RR} = (lR,l,\eta,\partial,\theta)$ of Remark 4.25 we can define a left l-module $l\mathbf{LM}$ over $\mathbf{RR}$ as a quadruple:

(1) a set lLM,
(2) a function $l : lLM \to \mathbf{N}$,
(3) a function $\partial : \{E \in lLM, i \in \mathbf{N} \mid l_{\mathbf{LM}}(E) \geq i\} \to lLM$,

(4) a function $\theta^{\mathbf{LM}} : \{r \in lR, E \in lLM \,|\, l_{\mathbf{LM}}(E) > l_{\mathbf{RR}}(r)\} \to lLM$

where operations l, ∂ and $\theta^{\mathbf{LM}}$ satisfy some conditions.

Once these conditions are properly established the category of such pairs $(l\mathbf{RR}, l\mathbf{LM})$ should be equivalent to the Hirschowitz-Maggesi "large module category" category (see [20, Definition 2.9]) and in particular the systems of expressions associated with binding signatures can be described as universal objects carrying some additional operations in this category.

These l-versions of the relative monads and their modules should be easier to formalize in systems such as HOL.

Acknowledgements:

(1) Work on this paper was supported by NSF grant 1100938.
(2) This material is based on research sponsored by The United States Air Force Research Laboratory under agreement number FA9550-15-1-0053. The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the United States Air Force Research Laboratory, the U.S. Government or Carnegie Mellon University.

References

[1] Peter Aczel. The type theoretic interpretation of constructive set theory. In *Logic Colloquium '77 (Proc. Conf., Wrocław, 1977)*, volume 96 of *Stud. Logic Foundations Math.*, pages 55–66. North-Holland, Amsterdam-New York, 1978.
[2] Jiří Adámek. Free algebras and automata realizations in the language of categories. *Comment. Math. Univ. Carolinae*, 15:589–602, 1974.
[3] Benedikt Ahrens. Modules over relative monads for syntax and semantics. *Math. Structures Comput. Sci.*, 26(1):3–37, 2016.
[4] Benedikt Ahrens and Ralph Matthes. Heterogeneous substitution systems revisited. *https://arxiv.org/abs/1601.04299*, 2016.
[5] Benedikt Ahrens, Ralph Matthes, and Anders Mörtberg. From signatures to monads in unimath. *https://arxiv.org/abs/1612.00693*, 2016.
[6] Benedikt Ahrens, Ralph Matthes, and Anders Mörtberg. Monads from multi-sorted binding signatures. *http://benedikt-ahrens.de/multisorted.pdf*, 2017.
[7] Thorsten Altenkirch, James Chapman, and Tarmo Uustalu. Monads need not be endofunctors. In *Foundations of software science and computational structures*, volume 6014 of *Lecture Notes in Comput. Sci.*, pages 297–311. Springer, Berlin, 2010.
[8] Thorsten Altenkirch, James Chapman, and Tarmo Uustalu. Monads need not be endofunctors. *Logical Methods in Computer Science*, 11(1:3):1–40, 2015.
[9] H. P. Barendregt. *The lambda calculus*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, revised edition, 1984. Its syntax and semantics.
[10] Michael Barr. Coequalizers and free triples. *Math. Z.*, 116:307–322, 1970.
[11] Nicolas Bourbaki. *Theory of sets*. Elements of Mathematics (Berlin). Springer-Verlag, Berlin, 2004. Reprint of the 1968 English translation [Hermann, Paris; MR0237342].
[12] John Cartmell. Generalised algebraic theories and contextual categories. *Ph.D. Thesis, Oxford University*, 1978. `http://www.cs.ru.nl/~spitters/Cartmell.pdf`.

[13] John Cartmell. Generalised algebraic theories and contextual categories. *Ann. Pure Appl. Logic*, 32(3):209–243, 1986.

[14] Alonzo Church. A set of postulates for the foundation of logic. *Ann. of Math. (2)*, 33(2):346–366, 1932.

[15] H. B. Curry. Grundlagen der Kombinatorischen Logik. *Amer. J. Math.*, 52(4):789–834, 1930.

[16] Marcelo Fiore, Gordon Plotkin, and Daniele Turi. Abstract syntax and variable binding (extended abstract) (Warning: the paper uses the name "presheaves" for covariant functors to the category of sets). In *14th Symposium on Logic in Computer Science (Trento, 1999)*, pages 193–202. IEEE Computer Soc., Los Alamitos, CA, 1999.

[17] P. Gabriel and M. Zisman. *Calculus of fractions and homotopy theory*. Ergebnisse der Mathematik und ihrer Grenzgebiete, Band 35. Springer-Verlag New York, Inc., New York, 1967.

[18] Richard Garner. Combinatorial structure of type dependency. *J. Pure Appl. Algebra*, 219(6):1885–1914, 2015.

[19] André Hirschowitz and Marco Maggesi. Modules over monads and linearity. In *Logic, language, information and computation*, volume 4576 of *Lecture Notes in Comput. Sci.*, pages 218–237. Springer, Berlin, 2007.

[20] André Hirschowitz and Marco Maggesi. Higher order theories. *http://arxiv.org/abs/0704.2900*, 2010.

[21] André Hirschowitz and Marco Maggesi. Modules over monads and initial semantics. *Inform. and Comput.*, 208(5):545–564, 2010.

[22] André Hirschowitz and Marco Maggesi. Nested abstract syntax in Coq. *J. Automat. Reason.*, 49(3):409–426, 2012.

[23] Martin Hofmann. Syntax and semantics of dependent types. In *Semantics and logics of computation (Cambridge, 1995)*, volume 14 of *Publ. Newton Inst.*, pages 79–130. Cambridge Univ. Press, Cambridge, 1997.

[24] Chris Kapulkin, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. The simplicial model of univalent foundations. *Available at* *http://arxiv.org/abs/1211.2851*, 2012, 2014.

[25] H. Kleisli. Every standard construction is induced by a pair of adjoint functors. *Proc. Amer. Math. Soc.*, 16:544–546, 1965.

[26] F. William Lawvere. Functorial semantics of algebraic theories and some algebraic problems in the context of functorial semantics of algebraic theories. *Repr. Theory Appl. Categ.*, (5):1–121, 2004. Reprinted from Proc. Nat. Acad. Sci. U.S.A. **50** (1963), 869–872 [MR0158921] and ıt Reports of the Midwest Category Seminar. II, 41–61, Springer, Berlin, 1968 [MR0231882].

[27] S. MacLane. *Categories for the working mathematician*, volume 5 of *Graduate texts in Mathematics*. Springer-Verlag, 1971.

[28] Per Martin-Löf. Constructive mathematics and computer programming. In *Logic, methodology and philosophy of science, VI (Hannover, 1979)*, volume 104 of *Stud. Logic Found. Math.*, pages 153–175. North-Holland, Amsterdam, 1982.

[29] Ralph Matthes and Tarmo Uustalu. Substitution in non-wellfounded syntax with variable binding. *Theoret. Comput. Sci.*, 327(1-2):155–174, 2004.

[30] Eugenio Moggi. Notions of computation and monads. *Inform. and Comput.*, 93(1):55–92, 1991. Selections from the 1989 IEEE Symposium on Logic in Computer Science.

[31] Andrew M. Pitts. *Nominal sets*, volume 57 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 2013. Names and symmetry in computer science.

[32] Moses Schönfinkel. Über die bausteine der mathematischen logik. *Math. Ann.*, 92:305 – 316; Jbuch 50, 23, 1924.

[33] Peter Selinger. The lambda calculus is algebraic. *J. Funct. Programming*, 12(6):549–566, 2002.

[34] Thomas Streicher. *Semantics of type theory*. Progress in Theoretical Computer Science. Birkhäuser Boston Inc., Boston, MA, 1991. Correctness, completeness and independence results, With a foreword by Martin Wirsing.

[35] Jean van Heijenoort. *From Frege to Gödel. A source book in mathematical logic, 1879–1931*. Harvard University Press, Cambridge, Mass., 1967.

[36] Vladimir Voevodsky. Notes on type systems. 2009–2012. https://github.com/vladimirias/old_notes_on_type_systems.

[37] Vladimir Voevodsky. The equivalence axiom and univalent models of type theory. *http://arxiv.org/abs/1402.5556*, pages 1–11, February 4, 2010. http://arxiv.org/abs/1402.5556.

[38] Vladimir Voevodsky. B-systems. *arXiv 1410.5389, submitted*, pages 1–17, 2014.

[39] Vladimir Voevodsky. A C-system defined by a universe category. *Theory Appl. Categ.*, 30(37):1181–1215, 2015. http://www.tac.mta.ca/tac/volumes/30/37/30-37.pdf.

[40] Vladimir Voevodsky. An experimental library of formalized mathematics based on the univalent foundations. *Math. Structures Comput. Sci.*, 25(5):1278–1294, 2015.

[41] Vladimir Voevodsky. Lawvere theories and C-systems. *arXiv 1512.08104*, pages 1–15, 2015. http://arxiv.org/abs/1512.08104, under review in Proceedings of the American Mathematical Society.

[42] Vladimir Voevodsky. Martin-Lof identity types in the C-systems defined by a universe category. *arXiv 1505.06446*, pages 1–51, 2015. http://arxiv.org/abs/1505.06446, under review in Publication IHES.

[43] Vladimir Voevodsky. Lawvere theories and Jf-relative monads. *arXiv 1601.02158*, pages 1–21, January 9, 2016. http://arxiv.org/abs/1601.02158.

[44] Vladimir Voevodsky. Products of families of types and (Π, λ)-structures on C-systems. *Theory Appl. Categ.*, 31:Paper No. 36, 1044–1094, 2016. http://www.tac.mta.ca/tac/volumes/31/36/31-36.pdf.

[45] Vladimir Voevodsky. Subsystems and regular quotients of C-systems. In *A panorama of mathematics: pure and applied*, volume 658 of *Contemp. Math.*, pages 127–137. Amer. Math. Soc., Providence, RI, 2016. prepublication version in http://arxiv.org/abs/1406.7413.

[46] Vladimir Voevodsky. The (Π, λ)-structures on the C-systems defined by universe categories. *Theory Appl. Categ.*, 32:Paper No. 4, 113–121, 2017. http://www.tac.mta.ca/tac/volumes/32/4/32-04.pdf.

[47] Vladimir Voevodsky, Benedikt Ahrens, Daniel Grayson, et al. *UniMath* - a library of formalized mathematics. Available at https://github.com/UniMath.

[48] Julianna Zsido. *Typed Abstract Syntax*. Theses, Université Nice Sophia Antipolis, June 2010.

School of Mathematics, Institute for Advanced Study, Princeton NJ, USA. e-mail: vladimir@ias.edu