Logical Methods in Computer Science
Volume 22, Issue 1, 2026, pp. 9:1–9:36
https://lmcs.episciences.org/

Submitted Mar. 04, 2024
Published Feb. 16, 2026

REPRESENTING GUARDEDNESS IN CALL-BY-VALUE AND GUARDED PARAMETERIZED MONADS

SERGEY GONCHAROV

University of Birmingham, Birmingham, UK
e-mail address: s.goncharov@bham.ac.uk

Abstract. Like the notion of computation via (strong) monads serves to classify various flavours of impurity, including exceptions, non-determinism, probability, local and global store, the notion of guardedness classifies well-behavedness of cycles in various settings. In its most general form, the guardedness discipline applies to general symmetric monoidal categories and further specializes to Cartesian and co-Cartesian categories, where it governs guarded recursion and guarded iteration, respectively. Here, even more specifically, we deal with the semantics of call-by-value guarded iteration. It was shown by Levy, Power and Thielecke that call-by-value languages can be generally interpreted in Freyd categories, but in order to represent effectful function spaces, such a category must canonically arise from a strong monad. We generalize this fact by showing that representing *guarded* effectful function spaces calls for certain parameterized monads (in the sense of Uustalu). This provides a description of guardedness as an intrinsic categorical property of programs, complementing the existing description of guardedness as a predicate on a category.

1. Introduction

A traditional way to model call-by-value languages is based on a clear-cut separation between computations and values. A computation can be *suspended* and thus turned into a value, and a value can be *executed*, and thus again be turned into a computation. The paradigmatic example of these conversions is the application and abstraction mechanisms of the λ-calculus. From the categorical modelling perspective, this view naturally requires two categories, suitably connected with each other. As essentially suggested by Moggi [Mog91], a minimal modelling framework requires a Cartesian category (i.e. a category with finite products) as a category of values and a Kleisli category of a strong monad over it, as a category of (side-effecting) computations (also called *producers* [Lev04]). A generic *computational metalanguage* thus arises as an internal language of strong monads. Levy, Power and Thielecke [LPT02] designed a refinement of Moggi's computational metalanguage, called *fine-grain call-by-value (FGCBV)*, whose models are not necessarily strong monads, but are more general *Freyd categories*. They have shown that a strong monad in fact always emerges from a Freyd category if certain function spaces (needed to interpret higher-order functions) are *representable* as objects of the value category – thus strong monads arise from first principles.

Support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) is gratefully acknowledged – project number 501369690.

DOI:10.46298/LMCS-22(1:9)2026

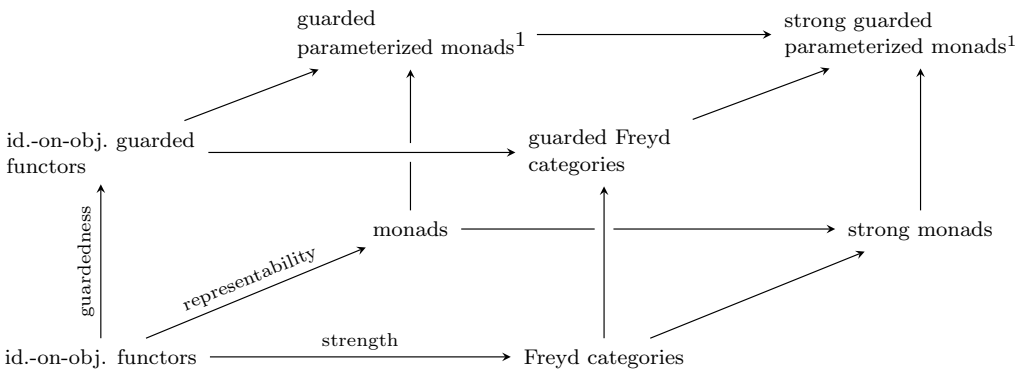


Figure 1: Three dimensions within call-by-value.

Here, we analyse an extension of the FGCBV paradigm with a notion of guardedness, which is a certain predicate on computations, certifying their well-behavedness, in particular that they can be iterated [GSRP17, LG19]. A typical example is guardedness in process algebra, where guardedness is often used to ensure that recursive systems of process definitions have unique solutions [Mil89].

FGCBV does not directly deal with fixpoints, since these are usually considered to be features orthogonal to computational effects and evaluation strategies. Analogously, even though the notion of guardedness is motivated by fixpoints, here we do not consider (guarded) fixpoints as a core language feature. In fact, in practically relevant cases guardedness is meaningful on its own as a suitable notion of *productivity* of computation, and need not be justified via fixpoints, which may or may not exist. In FGCBV, one typically regards general recursion to be supported by the category of values, and once the latter indeed does so (e.g. by being a suitable category of complete partial orders), it is obvious to add a corresponding fixpoint construct to the language.

Let us nevertheless outline the connection between guardedness and recursion in some more detail. General recursion entails partiality for programs, meaning that even if we abstract from it, the corresponding effect of *partiality* must be part of the computational effect abstraction (see e.g. [Fio04]). Recursion and computational effects are thus intimately connected. This connection persists under the restriction from general recursion to iteration, which is subject to a much broader range of models, and triggers the partiality effect just as well. Arguably, the largest class of monads, supporting iteration, are *Elgot monads* [AMV10, GRS15]. These are monads T, equipped with *Elgot iteration*:

$$\frac{f\colon X \to T(Y + X)}{f^\dagger\colon X \to TY} \tag{1.1}$$

and subject to established equational laws [BE93, SP00]. Intuitively, $f^\dagger$ is obtained from f by iterating away the right summand in the output type $Y + X$. For example, the maybe-monad $(-) + 1$ is an Elgot monad over the category of *classical* sets, which yields a model for a while-language with non-termination as the only computational effect. Now, *guarded Elgot monads* [LG19] refine Elgot monads in that the operator (1.1) needs only to be defined w.r.t. a custom class of *guarded morphisms*, governed by simple laws. Proper partiality of the guardedness predicate is relevant for various reasons, such as the following.

- Guarded fixpoints often *uniquely* satisfy the corresponding fixpoint equation [Uus03, Mil05, GSRP19], which greatly facilitates reasoning; this is extensively used in bisimulation-based process algebra [Mil89, Fok13].
- In a type-theoretic and constructive setting, guarded iteration can often be defined natively and more generally, e.g. the "simplest" guarded Elgot monad is Capretta's *delay monad* (initially called "partiality monad") [Cap05], rendered by final coalgebras $D = \nu\gamma.\,(- + \gamma)$, which yields an intensional counterpart of the maybe-monad; guardedness then means *productivity*, i.e., that the computation signals that it evolves if it does. Contrastingly, the "simplest" Elgot monad is much harder to construct and arguably requires additional principles to be available in the underlying metatheory [CUV17, ADK17, EK17, Gon21].
- Guardedness is a compositional type discipline, and hence it potentially helps to encapsulate additional information about the productivity of programs in types, like monads encapsulate the information about potential side-effects.

By allowing the iteration operator to be properly partial, we can accommodate a range of new examples of iterative behaviour. A notion of guardedness thus often plays an auxiliary role of determining, in a compositional way, which morphisms can be iterated.

As indicated above, strong monads can be regarded as structures, in a canonical way arising from FGCBV by adding the requirement of representability of certain function spaces in the category of values. This is behind the mechanism of representing computational effects via monads in type systems (e.g. in Fω, by quantification over higher kinds) and hence in programming languages (e.g. in Haskell). Our goal is to provide an analogous mechanism for guardedness and for its combinations with computational effects and strength. That is, (strong) monads are an answer to the question:

> *What is the categorical/type-theoretic structure that faithfully represents computational effects within a higher-order universe?*

Here, we are answering the question:

> *What is the categorical/type-theoretic structure that faithfully represents guarded computational effects within a higher-order universe?*

In other words, we seek to formulate guardedness as an intrinsic structural property of morphisms, rather than as additional data that (anonymously) identifies guarded morphisms among others. In doing so, we are inspired by the view of monads as structures for representing effects, as summarized above. In fact, we show that strength, representability and guardedness can be naturally arranged within FGCBV as three orthogonal dimensions, as shown in Figure 1 (the arrows point from more general concepts to more specific ones). The bottom face of the cube features the above-mentioned connection between Freyd categories and strong monads, and a corresponding connection between identity-on-object functors and (not necessarily strong) monads. We contribute with the top face, where guardedness is combined with other dimensions. The key point is the combination of guardedness with representability, which produces a certain class of *parameterized monads* [Uus03] that we dub *guarded parameterized monads*.

Related work We benefit from the analysis of Power and Robinson [PR97], who introduced *premonoidal categories* as an abstraction of Kleisli categories. Freyd categories were subsequently defined by Power and Thielecke [PT99] as premonoidal categories with

[1]More precisely, representability yields parameterized guarded monads, subject to an additional monicity condition. This is treated in detail in Section 7.

additional structure and also connected to strong monads. Levy [Lev04] came up with an equivalent definition, which we use throughout. In the previous characterization [PT99, LPT02], strong monads were shown to arise jointly with Kleisli exponentials from *closed Freyd categories*. We refine this characterization (Corollary 4.6) by showing that strong monads in fact arise independently of exponentials (Proposition 4.5). *Distributive Freyd categories* were defined by Staton [Sta14] – here we use them to extend the FGCBV language with coproducts and, subsequently, with guardedness predicates. Previous approaches to identifying structures for ensuring guardedness on monads involved *monad modules* [PG14, AMV02] – we make do with guarded parameterized monads instead, which combine monads with modules over them and arise universally.

Plan of the paper After short technical preliminaries, we start off by introducing a restricted version of FGCBV in Section 3 and extensively discuss motivating examples, which (with a little effort) can already be encoded despite restrictions. We establish a very simple form of the representability scenario, producing monads, and meant to serve as a model for subsequent sections. In Section 4 we deal with full FGCBV, Freyd categories, modelling them and strong monads, representing Freyd categories. The guardedness dimension is introduced in Section 5 where we define *guarded Freyd categories*, and in Section 6 we analyse the representability issue for them. Finally, in Section 7 we introduce an equational axiomatization of a categorical structure for representing guardedness, called *guarded parameterized monads*. As a crucial technical step, we establish a coherence property in the style of Mac Lane's coherence theorem for monoidal categories [ML71].

The present paper is an extended version of the conference paper [Gon23]. We added the proofs and more details to the examples and the general discussion. The original definition [Gon23] of the guarded parameterized monad was missing two coherence conditions, which are now added (Definition 7.1).

2. Preliminaries

We assume familiarity with the basics of category theory [ML71, Awo10]. For a category $\mathbf{V}$, $|\mathbf{V}|$ will denote the class of objects, and $\mathbf{V}(X,Y)$ will denote morphisms from X to Y. We tend to omit indices at natural transformations for readability. A category with finite (co-)products is called (co-)Cartesian. In a co-Cartesian category with selected coproducts, we write $!\colon 0 \to A$ for the initial morphism, and $\mathit{inl}\colon A \to A+B$ and $\mathit{inr}\colon B \to A+B$ for the left and right coproduct injections, respectively. A *distributive category* [Coc93] is a Cartesian and co-Cartesian category, in which the natural transformation

$$X \times Y + X \times Z \xrightarrow{[\mathit{id} \times \mathit{inl},\ \mathit{id} \times \mathit{inr}]} X \times (Y + Z)$$

is an isomorphism, whose inverse we denote $\mathit{dist}_{X,Y,Z}$ (a co-Cartesian and Cartesian closed category is always distributive). Let $\Delta = \langle \mathit{id}, \mathit{id} \rangle\colon X \to X \times X$ and $\nabla = [\mathit{id}, \mathit{id}]\colon X + X \to X$.

A monad $\mathbf{T}$ on $\mathbf{V}$ is determined by a *Kleisli triple* $(T, \eta, (-)^\star)$, consisting of a map $T\colon |\mathbf{V}| \to |\mathbf{V}|$, a family of morphisms $(\eta_X\colon X \to TX)_{X \in |\mathbf{V}|}$ and *Kleisli lifting* sending each $f\colon X \to TY$ to $f^\star\colon TX \to TY$ and obeying *monad laws*:

$$\eta^\star = \mathit{id}, \qquad f^\star \circ \eta = f, \qquad (f^\star \circ g)^\star = f^\star \circ g^\star.$$

It follows that T extends to a functor, η extends to a natural transformation – *unit*, $\mu = \mathit{id}^\star\colon TTX \to TX$ extends to a natural transformation – *multiplication*, and that (T, η, μ) is a monad in the standard sense [ML71]. We will generally use blackboard capitals

$$\frac{}{x\colon A \vdash_{\mathsf{v}} x\colon A} \qquad \frac{f\colon A \to B \in \Sigma_{\mathsf{v}} \quad \Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{v}} f(v)\colon B} \qquad \frac{f\colon A \to B \in \Sigma_{\mathsf{c}} \quad \Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{c}} f(v)\colon B}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v\colon A} \qquad \frac{\Gamma \vdash_{\mathsf{c}} p\colon A \quad x\colon A \vdash_{\mathsf{c}} q\colon B}{\Gamma \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow p;\, q\colon B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon 0}{\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v\colon A}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v\colon A + B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon B}{\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v\colon A + B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon A + B \quad x\colon A \vdash_{\mathsf{c}} p\colon C \quad y\colon B \vdash_{\mathsf{c}} q\colon C}{\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, y \mapsto q\colon C}$$

Figure 2: Simple FGCBV with coproducts.

(such as $\mathbf{T}$) to refer to monads and the corresponding Roman letters (such as T) to refer to their functor parts. Morphisms of the form $f\colon X \to TY$ are called *Kleisli morphisms* and form the *Kleisli category* $\mathbf{V}_{\mathbf{T}}$ of $\mathbf{T}$ under *Kleisli composition* $f, g \mapsto f^{\star} \circ g$ with identity η.

An endofunctor F is *strong* if it is equipped with a natural transformation *strength* $\tau\colon X \times FY \to F(X \times Y)$, such that the diagrams

$$\begin{array}{ccc} 1 \times FX & \xrightarrow{snd} & FX \\ {\scriptstyle\tau}\downarrow & \nearrow_{F\,snd} & \\ F(1 \times X) & & \end{array} \qquad \begin{array}{ccc} (X \times Y) \times FZ & \xrightarrow{\qquad\tau\qquad} & F((X \times Y) \times Z) \\ \downarrow\cong & & \downarrow\cong \\ X \times (Y \times FY) & \xrightarrow{id \times \tau} X \times F(Y \times Z) \xrightarrow{\tau} & F(X \times (Y \times Z)) \end{array}$$

commute. A natural transformation between two strong functors is strong if it preserves strength in the obvious sense, and a monad $\mathbf{T}$ is strong if T is strong with some strength $\tau\colon X \times TY \to T(X \times Y)$ and η and μ are strong with id being the strength of Id and $T\tau \circ \tau\colon X \times TTY \to TT(X \times Y)$ being the strength of TT.

3. Simple FGCBV with Coproducts

We start off with a restricted – single-variable – fragment of FGCBV, but extended with coproduct types. Since we will not deal with operational semantics, we simplify the language slightly (e.g. we do not include let-expressions for values). We also stick to a Haskell-style syntax with do-notation and case expressions. We fix a collection of sorts $S_1, S_2, \ldots$, a signature Σ_{v} of pure programs $f\colon A \to B$, and a signature Σ_{c} of effectful programs $f\colon A \to B$ (also called *generic effects* [PP01]) where A and B are types, generated with the grammar

$$A, B ::= S_1, S_2, \ldots \mid 0 \mid A + B. \tag{3.1}$$

We then define terms in context of the form $x\colon A \vdash_{\mathsf{v}} v\colon B$ and $x\colon A \vdash_{\mathsf{c}} p\colon B$ for value terms and computation terms inductively by the rules given in Figure 2. (where we chose to stick to the syntax of the familiar Haskell's do-notation): This language is essentially a refinement of Moggi's *simple (!) computational metalanguage* [Mog91], which has only one-variable contexts (i.e. Γ is of the form $x\colon A$ throughout), rather than fully fledged multi-variable contexts. In terms of monads, the present language corresponds to not necessarily strong ones. In terms of monads, the present language corresponds to not necessarily strong ones. Such monads are not very useful in traditional programming languages semantics; however we dwell on this case for several reasons. We aim to explore the interaction between guardedness and monads from a foundational perspective, while remaining as general as

possible to cover cases where strength does not exist or is not relevant. We would also like to identify the basic representation scenario, to be extended later to more sophisticated cases.

An obvious extension of the presented language would be the iteration operator:

$$\frac{\Gamma \vdash_{\mathsf{c}} p\colon A \qquad x\colon A \vdash_{\mathsf{c}} q\colon B + A}{\Gamma \vdash_{\mathsf{c}} \mathsf{iter}\, x \leftarrow p;\, q\colon B} \tag{3.2}$$

meant to satisfy the fixpoint equality

$$\mathsf{iter}\, x \leftarrow p;\, q = \mathsf{iter}\, x \leftarrow (\mathsf{do}\, x \leftarrow p;\, q);\, q$$

Presently, we focus on representing guardedness as such and do not deal with (3.2)

We present three examples that can be interpreted w.r.t. the single-variable fragment to demonstrate the unifying power of FGCBV and illustrate various flavours of guardedness.

Example 3.1 (Basic Process Algebra [BPS01])**.** *Basic process algebra (BPA)* over a set of actions $\mathcal{A}$ is defined by the grammar:

$$P, Q ::= (a \in \mathcal{A}) \mid P + Q \mid P \cdot Q.$$

One typically considers BPA-terms over free variables (seen as process names) to solve systems of recursive process equations w.r.t. these variables. E.g. we can specify a 2-bit FIFO buffer as a solution to

$$\begin{aligned} B_0 &= \mathsf{in}_0 \cdot B_1^0 + \mathsf{in}_1 \cdot B_1^1 \\ B_1^i &= \mathsf{in}_0 \cdot B_2^{0,i} + \mathsf{in}_1 \cdot B_2^{1,i} + \mathsf{out}_i \cdot B_0 && (i \in \{0,1\}) \\ B_2^{i,j} &= \mathsf{out}_j \cdot B_1^i && (i,j \in \{0,1\}) \end{aligned} \tag{3.3}$$

with $\mathcal{A} = \{\mathsf{in}_0, \mathsf{in}_1, \mathsf{out}_0, \mathsf{out}_1\}$. We view B_0 as an empty FIFO, B_1^i as a FIFO carrying only i and $B_2^{i,j}$ as a FIFO carrying i and j. For example, the trace

$$B_0 \xrightarrow{\mathsf{in}_0} B_1^0 \xrightarrow{\mathsf{in}_1} B_2^{1,0} \xrightarrow{\mathsf{out}_0} B_1^1 \xrightarrow{\mathsf{out}_1} B_0$$

is valid and represents the following course of action: push 0, push 1, pop 1 and then pop 0. We can model such systems of equations in FGCBV as follows. Let us fix a single sort 1 and identify an n-fold sum $(\ldots(1 + \ldots)\ldots) + 1$ with the natural number n. The injections $\mathsf{inj}_i\colon 1 \to n$ are defined inductively in the obvious way. Let $\Sigma_{\mathsf{v}} = \emptyset$ and $\Sigma_{\mathsf{c}} = \{a\colon 1 \to 1 \mid a \in \mathcal{A}\} \cup \{\mathsf{toss}\colon 1 \to 2\}$. A BPA-term over process names $\{N_1, \ldots, N_n\}$ can be translated to FGCBV recursively, with the following rules where $\rightsquigarrow$ reads as "translates":

$$\frac{}{N_i \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} \mathsf{return}(\mathsf{inr}(\mathsf{inj}_i\, x))\colon 1 + n} \qquad \frac{}{a \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow a(x);\, \mathsf{return}(\mathsf{inl}\, x)\colon 1 + n}$$

$$\frac{P \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} p\colon 1 + n \qquad Q \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} q\colon 1 + n}{P + Q \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow \mathsf{toss}(x);\, \mathsf{case}\, x\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, x \mapsto q\colon 1 + n}$$

$$\frac{P \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} p\colon 1 + n \qquad Q \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} q\colon 1 + n}{P \cdot Q \rightsquigarrow x\colon 1 \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow p;\, \mathsf{case}\, x\, \mathsf{of}\, \mathsf{inl}\, x \mapsto q(x);\, \mathsf{inr}\, x \mapsto \mathsf{return}(\mathsf{inr}\, x)\colon 1 + n}$$

Intuitively, the terms $x\colon 1 \vdash_{\mathsf{c}} p\colon 1 + n$ represent processes with $1 + n$ exit points: every process name N_i identifies an exit i, in addition to the global anonymous exit. The latter is associated with actions, which are not postcomposed with any other commands. The

generic effect toss induces binary nondeterminism as a coin-tossing act. For example, the result of translating the right-hand sides of (3.3) (after minor simplifications) is

$$\begin{aligned}
&\mathsf{do}\, x \leftarrow \mathsf{toss}(x);\ \mathsf{case}\, x\, \mathsf{of}\\
&\qquad \mathsf{inl}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{in}_0(x);\ \mathsf{return}(\mathsf{inr}(\mathsf{inj}_1^0\, x));\\
&\qquad \mathsf{inr}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{in}_1(x);\ \mathsf{return}(\mathsf{inr}(\mathsf{inj}_1^1\, x)),\\
&\mathsf{do}\, x \leftarrow \mathsf{toss}(x);\ \mathsf{case}\, x\, \mathsf{of}\\
&\qquad \mathsf{inl}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{in}_0(x);\ \mathsf{return}(\mathsf{inr}(\mathsf{inj}_2^{0,i}\, x));\\
&\qquad \mathsf{inr}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{toss}(x);\ \mathsf{case}\, x\, \mathsf{of}\\
&\qquad\qquad \mathsf{inl}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{in}_1(x);\ \mathsf{return}(\mathsf{inr}(\mathsf{inj}_2^{1,i}\, x));\\
&\qquad\qquad \mathsf{inr}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{out}_i(x);\ \mathsf{return}(\mathsf{inr}(\mathsf{inj}_0\, x)) \qquad (i \in \{0,1\})\\
&\mathsf{do}\, x \leftarrow \mathsf{out}_j(x);\ \mathsf{return}(\mathsf{inr}(\mathsf{inj}_1^i\, x)) \qquad (i,j \in \{0,1\})
\end{aligned}$$

where $n = 1+2+4 = 7$, $\mathsf{inj}_0 \colon 1 \to 7$, the $\mathsf{inj}_1^i \colon 1 \to 7$ and the $\mathsf{inj}_1^{i,j} \colon 1 \to 7$ are the injections, selecting the indices that address B_0, B_1^i and $B_2^{i,j}$ correspondingly. Every list of terms in the context $(x \colon 1 \vdash_{\mathsf{c}} p_0 \colon m), \ldots, (x \colon 1 \vdash_{\mathsf{c}} p_{n-1} \colon m)$ can be converted to a single term $x \colon n \vdash_{\mathsf{c}} \hat{p}_n \colon m$ recursively as follows:

$$\hat{p}_0 = \mathsf{init}\, x, \qquad \hat{p}_{n+1} = \mathsf{case}\, x\, \mathsf{of}\, \mathsf{inl}\, x \mapsto \hat{p}_n;\ \mathsf{inr}\, x \mapsto p_{n+1}.$$

Every system of n equations over $m+n$ process names in BPA is thus represented by a term $x \colon n \vdash_{\mathsf{c}} p \colon (1+m)+n$ in simple FGCBV. Now, an iteration operator (3.2) applied to the latter term "solves" the corresponding system of equations w.r.t. to n names, and keeping the remaining m names free, resulting in a term of the form $x \colon n \vdash_{\mathsf{c}} \mathsf{iter}\, x \leftarrow \mathsf{return}\, x;\ p \colon 1+m$. In our example (3.3), $n = m = 7$.

Guarded systems are those in which recursive calls are preceded by actions; (3.3) is an example. Such systems have a unique solution (under bisimilarity) [BW90, Fok13]. The simplest unguarded example $P = P$ has arbitrary solutions and translates to $x \colon 1 \vdash_{\mathsf{c}} \mathsf{iter}\, x \leftarrow \mathsf{return}\, x;\ \mathsf{return}(\mathsf{inr}\, x) \colon 1$.

Example 3.2 (Imperative Traces)**.** We adapt the semantic framework of Nakata and Uustalu [NU15] for imperative coinductive traces to our setting. Let us fix a set P of *predicates*, a set T of *state transformers*, and let the corresponding pure and effectful signatures be $\Sigma_{\mathsf{v}} = \{p \colon S \to S+S \mid p \in P\} \cup \{t \colon S \to S \mid t \in T\}$ and $\Sigma_{\mathsf{c}} = \{\mathsf{put} \colon S \to 1,\ \mathsf{get} \colon 1 \to S\}$ over the set of sorts $\{S, 1\}$. The intended interpretation of this data is as follows:

- S is a set of memory states, e.g. the set of finitely supported partial functions $\mathbb{N} \hookrightarrow 2$;
- T are state transformers, e.g. functions, updating precisely one specified memory bit;
- $p \in P$ encode predicates: $p(s) = \mathit{inl}(s)$ if the predicate is satisfied and $p(s) = \mathit{inr}(s)$ otherwise, e.g. p can capture functions that give a Boolean answer to the questions "is the specified bit 0?" and "is the specified bit 1?".

For example, the following program negates the i-th memory bit (if it is present)

$$x \colon 1 \vdash_{\mathsf{c}} \mathsf{do}\, s \leftarrow \mathsf{get}(x);\ \mathsf{case}\, (s[i] = 0)\, \mathsf{of}\, \mathsf{inl}\, s \mapsto \mathsf{put}(s[i := 1]);\ \mathsf{inr}\, s \mapsto \mathsf{put}(s[i := 0]) \colon 1,$$

where $(\text{–}[i] = 0)$, $(\text{–}[i := 0])$ and $(\text{–}[i := 1])$ are the obvious predicate and state transformers. Nakata and Uustalu [NU15] argued in favour of (infinite) traces as a particularly suitable semantics for reasoning about imperative programs. This means that store updates must contribute to the semantics, which can be ensured by a judicious choice of syntax, e.g., by using $\mathsf{skip} = \mathsf{do}\, s \leftarrow \mathsf{get}(x);\, \mathsf{put}(s)$, but not return. In FGCBV, however, iterating $x\colon 1 \vdash_{\mathsf{c}} \mathsf{return}(\mathsf{inr}\, x)\colon 1$ would not yield any trace. By restricting to guarded iteration, with guardedness meaning writing to the store, we can indeed prevent such programs from iterating by defining guardedness so that at least one put is executed before the body of the loop is repeated.

Example 3.3 (Hybrid Programs)**.** Hybrid programs combine discrete and continuous capabilities and can thus be used to describe the behaviours of cyber-physical systems. For simplicity, we consider time delays as the only hybrid facility – more sophisticated scenarios are treated elsewhere [GNP20]. Let $\mathbb{R}_{\geqslant 0}$ be the sort of non-negative real numbers and let Σ_{v} contain all unary operations on non-negative reals and additionally $\mathsf{is}_0\colon \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0} + \mathbb{R}_{\geqslant 0}$, which sends $n = 0$ to $\mathit{inl}(n)$ and $n > 0$ to $\mathit{inr}(n)$. Let $\Sigma_{\mathsf{c}} = \{\mathsf{wait}\colon \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}\}$. With $\mathsf{wait}(r)$ we can introduce a time delay of length r and return r. With iteration we can write programs like

$$
\begin{aligned}
x\colon \mathbb{R}_{\geqslant 0} \vdash_{\mathsf{c}} {} & \mathsf{iter}\, x \leftarrow \mathsf{return}\, x; \\
& \mathsf{case}\ \mathsf{is}_0(x)\ \mathsf{of}\ \mathsf{inl}\, x \mapsto \mathsf{return}(\mathsf{inl}\, x); \\
& \qquad\qquad \mathsf{inr}\, x \mapsto \mathsf{do}\, x \leftarrow \mathsf{wait}(x);\, \mathsf{return}(\mathsf{inr}\, f(x))\colon \mathbb{R}_{\geqslant 0},
\end{aligned}
$$

which terminate successfully in finite time ($f(x) = x \dot{-} 1$[2]), run infinitely ($f(x) = 1$), or exhibit *Zeno behaviour* ($f(x) = x/2$), i.e. consume finite time, but never terminate. In all these examples, every iteration consumes non-zero time. This is also often considered a well-behavedness condition, which we can naturally interpret as guardedness.

To interpret the language from Figure 2, let us fix two co-Cartesian categories $\mathbf{V}$ and $\mathbf{C}$, and an identity-on-objects functor $J\colon \mathbf{V} \to \mathbf{C}$ (hence $|\mathbf{V}| = |\mathbf{C}|$) that strictly preserves coproducts. A semantics of $(\Sigma_{\mathsf{v}}, \Sigma_{\mathsf{c}})$ over J assigns

- an object $[\![A]\!] \in |\mathbf{V}|$ to each sort A;
- a morphism $[\![f]\!] \in \mathbf{V}([\![A]\!], [\![B]\!])$ to each $f\colon A \to B \in \Sigma_{\mathsf{v}}$;
- a morphism $[\![f]\!] \in \mathbf{C}([\![A]\!], [\![B]\!])$ to each $f\colon A \to B \in \Sigma_{\mathsf{c}}$,

which extends to types as follows: $[\![0]\!] = 0$, $[\![A + B]\!] = [\![A]\!] + [\![B]\!]$. The semantics of terms are given in Figure 3. As observed by Power and Robinson [PR97] (cf. [Sch69, 0.1]), monads arise from the requirement that J is a left adjoint, thus simple FGCBV can be interpreted w.r.t. a monad on $\mathbf{V}$. A direct simple proof is given below for the sake of completeness.

Proposition 3.4. *Let $J\colon \mathbf{V} \to \mathbf{C}$ be an identity-on-objects functor. Then J is a left adjoint iff $\mathbf{C}$ is isomorphic to a Kleisli category of some monad $\mathbf{T}$ on $\mathbf{V}$ and $J = H \circ J_{\mathbf{T}}$ where $H\colon \mathbf{V}_{\mathbf{T}} \cong \mathbf{C}$ is the relevant isomorphism, which is necessarily identity-on-objects, and $J_{\mathbf{T}}\colon \mathbf{V} \to \mathbf{V}_{\mathbf{T}}$ is the canonical left adjoint sending every $f \in \mathbf{V}(X, Y)$ to $\eta \circ f \in \mathbf{V}(X, TY)$.*

Moreover, in this situation, finite coproducts in $\mathbf{C}$ are inherited from $\mathbf{V}$, i.e. $J\,!$ is the initial morphism in $\mathbf{C}$ and the triples $(X + Y, J\,\mathit{inl}, J\,\mathit{inr})$ are binary coproducts in $\mathbf{C}$.

[2] $\dot{-}$ refers to truncated subtraction: $x \dot{-} y = x - y$ if $x \geqslant y$, and $x \dot{-} y = 0$ otherwise.

$$\frac{}{[\![x\colon A \vdash_{\mathsf{v}} x\colon A]\!] = \mathit{id}} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} f(v)\colon B]\!] = [\![f]\!] \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} f(v)\colon B]\!] = [\![f]\!] \circ Jh}$$

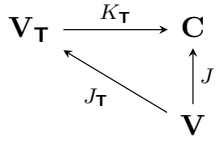
$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v\colon A]\!] = Jh} \qquad \frac{h_1 = [\![x\colon A \vdash_{\mathsf{c}} q\colon B]\!] \qquad h_2 = [\![\Gamma \vdash_{\mathsf{c}} p\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow p;\, q\colon B]\!] = h_1 \circ h_2}$$

$$\frac{}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v\colon A]\!] = !} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v\colon A + B]\!] = \mathit{inl} \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v\colon A + B]\!] = \mathit{inr} \circ h}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A + B]\!] \qquad h_1 = [\![x\colon A \vdash_{\mathsf{c}} p\colon C]\!] \qquad h_2 = [\![y\colon B \vdash_{\mathsf{c}} q\colon C]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, y \mapsto q\colon C]\!] = [h_1, h_2] \circ Jh}$$

Figure 3: Denotational semantics of simple FGCBV with coproducts.

Proof. Suppose that $J \dashv U$ and consider the diagram



where $K_{\mathbf{T}}$ is the comparison functor from the Kleisli category of $\mathbf{T}$ to $\mathbf{C}$. Note that $K_{\mathbf{T}}$ is generally full and faithful, because $\mathbf{V_T}(X,Y) = \mathbf{V}(X, UJY) \cong \mathbf{C}(JX, JY) = \mathbf{C}(K_{\mathbf{T}}X, K_{\mathbf{T}}Y)$. Moreover, $K_{\mathbf{T}}$ is identity-on-objects, for so is J by assumption. Thus, $K_{\mathbf{T}}$ is an isomorphism and $Jf = (K_{\mathbf{T}} \circ J_{\mathbf{T}})(f) = K_{\mathbf{T}}(\eta \circ f)$ for any $f \in \mathbf{V}(X,Y)$.

Now, suppose that for a suitable monad $\mathbf{T}$, $H\colon \mathbf{V_T} \cong \mathbf{C}$ and $Jf = H(\eta \circ f)$ for any $f \in \mathbf{V}(X,Y)$. Let $U \vdash J$ be the adjunction between $\mathbf{V}$ and $\mathbf{V_T}$, and show that $UH^{-1} \vdash J$. Note that $H^{-1} \vdash H$, and hence, by composing adjunctions $UH^{-1} \vdash HJ$. For every $f \in \mathbf{V}(X,Y)$, $HJf = H(\eta \circ f) = Jf$, i.e. indeed, $UH^{-1} \vdash J$.

That finite coproducts in $\mathbf{C}$ are inherited from $\mathbf{V}$ is easy to see. □

Example 3.5 (Monads)**.** Let us recall relevant monads on $\mathbf{V} = \mathbf{Set}$ for further reference.

(1) $TX = \nu\gamma.\, \mathcal{P}_\omega((X+1) + A \times \gamma)$ where $\mathcal{P}_\omega$ is the finite powerset functor and $\nu\gamma.\, F\gamma$ denotes a final F-coalgebra. This monad provides a standard strong bisimulation semantics for BPA (Example 3.1). The denotations in TX are finitely branching trees with edges labelled by actions and with terminal nodes labelled in X (free variables) or in 1 (successful termination). This monad is an instance of the *coinductive resumption monad* [PG14], and the inhabitants of TX are often called *synchronization trees* (e.g. [ACÉI12]).

(2) $TX = \mathcal{P}(A^\star \times (X+1) + A^\star)$ is the monad of finite traces (terminating successfully $A^\star \times (X+1)$ and divergent $A^\star$), which can again be used as a semantics of Example 3.1.

(3) $TX = \mathcal{P}(A^\star \times (X+1) + (A^\star + A^\omega))$ is a refinement of (2), collecting not only finite, but also infinite traces. If we extend BPA with countable non-determinism, we obtain a semantics properly between strong bisimilarity and finite trace equivalence. For example, the equation $P = a \cdot P$ produces the infinite trace a^ω and $P' = \sum_{i \in \mathbb{N}} P_i$ with $P_0 = a$ and $P_{i+1} = a \cdot P_i$ do not. Therefore, P is not infinite trace equivalent to P', while P and P' are finite trace equivalent.

(4) $TX = (\nu\gamma.\, X \times S + \gamma \times S)^S$ can be used for Example 3.2. In $\mathbf{Set}$, $TX \cong (X \times S^+ + S^\omega)^S$, i.e. an element TX is isomorphic to a function that takes an initial state in S and returns

$$\frac{x\colon A \ \text{ in } \ \Gamma}{\Gamma \vdash_{\mathsf{v}} x\colon A} \qquad \frac{f\colon A \to B \in \Sigma_{\mathsf{v}} \quad \Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{v}} f(v)\colon B} \qquad \frac{f\colon A \to B \in \Sigma_{\mathsf{c}} \quad \Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{c}} f(v)\colon B}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v\colon A} \qquad \frac{\Gamma \vdash_{\mathsf{c}} p\colon A \quad \Gamma, x\colon A \vdash_{\mathsf{c}} q\colon B}{\Gamma \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow p;\, q\colon B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon 0}{\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v\colon A}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v\colon A + B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon B}{\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v\colon A + B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon A + B \quad x\colon A \vdash_{\mathsf{c}} p\colon C \quad y\colon B \vdash_{\mathsf{c}} q\colon C}{\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, y \mapsto q\colon C}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A \qquad \Gamma \vdash_{\mathsf{v}} w\colon B}{\Gamma \vdash_{\mathsf{v}} \langle v, w\rangle\colon A \times B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon A \times B \qquad \Gamma, x\colon A, y\colon B \vdash_{\mathsf{c}} q\colon C}{\Gamma \vdash_{\mathsf{c}} \mathsf{case}\ v\ \mathsf{of}\ \langle x, y\rangle \mapsto q\colon C}$$

Figure 4: FGCBV with coproducts.

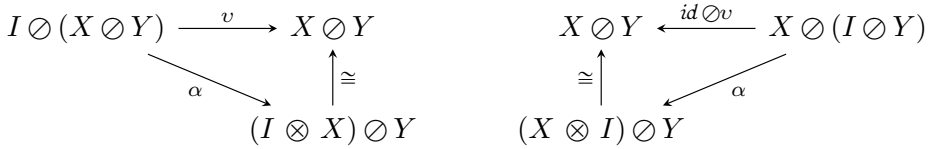
either a finite trace in $X \times S^+$ or an infinite trace in S^ω. We can use Proposition 3.4 to argue that T indeed extends to a monad. Indeed, let $\mathbf{C}$ be the category with $\mathbf{C}(X, Y) = \mathbf{Set}(X \times S, \nu\gamma.\, Y \times S + \gamma \times S)$, which is a full subcategory of the Kleisli category of the coinductive resumption monad $\nu\gamma.\, (\text{–} + \gamma \times S)$. Now, by definition, the obvious identity-on-objects functor $J\colon \mathbf{Set} \to \mathbf{C}$ is a left adjoint, yielding the original T.

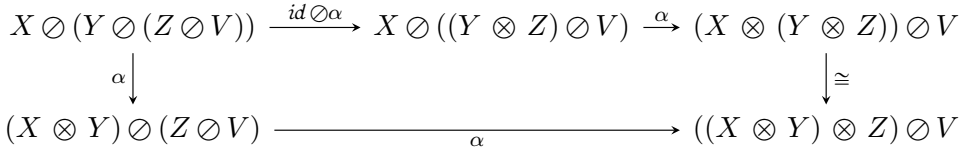
(5) $TX = \mathbb{R}_{\geqslant 0} \times X + \bar{\mathbb{R}}_{\geqslant 0}$ is a monad, which can be used for Example 3.3. Here, $\mathbb{R}_{\geqslant 0} \times X$ refers to terminating behaviours and $\bar{\mathbb{R}}_{\geqslant 0} = \mathbb{R}_{\geqslant 0} \cup \{\infty\}$ to Zeno and infinite behaviours.

4. Freyd Categories and Strong Monads

The full FGCBV (with coproducts) is obtained by extending the type syntax (3.1) with binary products $A \times B$, and by replacing the rules in Figure 2 with the rules in Figure 4. We now assume that variable contexts Γ are (possibly empty) lists $(x_1\colon A_1, \ldots, x_n\colon A_n)$ with non-repetitive $x_1, \ldots, x_n$. To interpret the resulting language, again, we need an identity-on-objects functor $J\colon \mathbf{V} \to \mathbf{C}$, an action of $\mathbf{V}$ on $\mathbf{C}$, and J to preserve this action.

Definition 4.1 (Actegory [JK01])**.** Let $(\mathbf{V}, \otimes, I)$ be a monoidal category. Then an *action* of $\mathbf{V}$ on a category $\mathbf{C}$ is a bifunctor $\oslash\colon \mathbf{V} \times \mathbf{C} \to \mathbf{C}$ together with the *unitor* and the *actor* natural isomorphisms $\upsilon\colon I \oslash X \cong X$, $\alpha\colon X \oslash (Y \oslash Z) \cong (X \otimes Y) \oslash Z$, satisfying the following coherence conditions





(eliding the names of canonical isomorphisms). Then $\mathbf{C}$ is called an *($\mathbf{V}$-)actegory*.

Note that every monoidal category trivially acts on itself via $\oslash = \otimes$. In the sequel, we will only consider *Cartesian* categories, i.e. actegories w.r.t. $(\mathbf{V}, \times, 1)$.

Definition 4.2 (Freyd Category [Lev04])**.** A *Freyd category* $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ consists of the following data:

(1) a Cartesian category $\mathbf{V}$;
(2) a category $\mathbf{C}$ with $|\mathbf{V}| = |\mathbf{C}|$;
(3) an identity-on-objects functor $J\colon \mathbf{V} \to \mathbf{C}$;
(4) an action of $\mathbf{V}$ on $\mathbf{C}$, such that J preserves the $\mathbf{V}$-action, i.e. $J(f \times g) = f \oslash Jg$ for all $f \in \mathbf{V}(X, X')$, $g \in \mathbf{V}(Y, Y')$ (entailing $X \times Y = X \oslash Y$ for all $X, Y \in |\mathbf{V}|$), $\upsilon = J\,\mathit{snd}$ and $\alpha = J\langle \mathit{id} \times \mathit{fst}, \mathit{snd} \circ \mathit{snd}\rangle$.

Let us reformulate this definition slightly more explicitly.

Lemma 4.3. *A tuple* $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ *is a Freyd category iff*

(1) $\mathbf{V}$ *is a Cartesian category;*
(2) $\mathbf{C}$ *is a category, such that* $|\mathbf{V}| = |\mathbf{C}|$*;*
(3) J *is an identity-on-objects functor* $\mathbf{V} \to \mathbf{C}$*;*
(4) $\oslash$ *is a bifunctor* $\mathbf{V} \times \mathbf{C} \to \mathbf{C}$*, such that*
   (a) $X \oslash Y = X \times Y$ *for all* $X, Y \in |\mathbf{V}|$*,*
   (b) $J(f \times g) = f \oslash Jg$ *for all* $f \in \mathbf{V}(X, X')$*,* $g \in \mathbf{V}(Y, Y')$*, and*
   (c) *every* $J\,\mathit{snd}\colon 1 \times X \to X$ *is natural in* X *(w.r.t.* $\mathbf{C}$*-morphisms) and every* $J\langle \mathit{id} \times \mathit{fst}, \mathit{snd} \circ \mathit{snd}\rangle\colon X \times (Y \times Z) \to (X \times Y) \times Z$ *is natural in* X, Y *(w.r.t.* $\mathbf{V}$*-morphisms) and* Z *(w.r.t.* $\mathbf{C}$*-morphisms).*

Proof. The claim follows from the observation that defining υ and α as $J\,\mathit{snd}$ and $J\langle \mathit{id} \times \mathit{fst}, \mathit{snd} \circ \mathit{snd}\rangle$ correspondingly, yields an action of $\mathbf{V}$ on $\mathbf{C}$ iff υ and α are natural – the coherence conditions from Definition 4.1 hold automatically. □

Definition 4.4 (Distributive Freyd Category [Sta14])**.** A Freyd category $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ is *distributive* if $\mathbf{V}$ is distributive, $\mathbf{C}$ is co-Cartesian, and J strictly preserves coproducts.

Note that it follows from the above definition that the action $\oslash$ preserves coproducts in the second argument. Indeed, by applying J to the isomorphism $X \times (Y + Z) \cong X \times Y + X \times Z$ in $\mathbf{V}$, we obtain that $X \oslash (Y + Z) \cong X \oslash Y + X \oslash Z$ is in $\mathbf{C}$.

Given a distributive Freyd category $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$, we update the semantics from Section 3 by extending the semantics of types with the clauses $[\![A \times B]\!] = [\![A]\!] \times [\![B]\!]$, $[\![x_1\colon A_1, \ldots, x_n\colon A_n]\!] = [\![A_1]\!] \times \ldots \times [\![A_n]\!]$, and by defining the semantics of terms as in Figure 5, where $\mathit{proj}_i\colon X_1 \times \ldots \times X_n \to X_i$ denotes the i-th projection.

Freyd categories are to strong monads as identity-on-objects functors to monads.

Proposition 4.5. *Let* $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ *be a Freyd category. Then* J *is a left adjoint iff* $\mathbf{C}$ *is isomorphic to a Kleisli category of some strong monad* $\mathbf{T}$ *on* $\mathbf{V}$ *and* $Jf = H(\eta \circ f)$ *for all* $f \in \mathbf{V}(X, Y)$ *where* $H\colon \mathbf{V}_{\mathbf{T}} \cong \mathbf{C}$ *is the relevant isomorphism.*

Proof. Freyd categories are initially designed to generalize Kleisli categories of strong monads [PT99], in particular, we obtain the *'If'* direction of the claim.

For the *'Only if'* direction, suppose that J is a left adjoint, and show that the requested strong monad exists. Indeed, we obtain a monad $\mathbf{T}$ by Proposition 3.4. W.l.o.g. suppose that $\mathbf{C} = \mathbf{V}_{\mathbf{T}}$. Let $Jf = \eta \circ f$ for every $f\colon X \to Y$ from $\mathbf{V}$. Let us define strength τ as

$$\frac{}{[\![x_1 \colon A_1, \ldots, x_n \colon A_n \vdash_{\mathsf{v}} x_i \colon A_i]\!] = \mathit{proj}_i}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} f(v) \colon B]\!] = [\![f]\!] \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} f(v) \colon B]\!] = [\![f]\!] \circ Jh}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v \colon A]\!] = Jh} \qquad \frac{h_1 = [\![\Gamma, x \colon A \vdash_{\mathsf{c}} q \colon B]\!] \qquad h_2 = [\![\Gamma \vdash_{\mathsf{c}} p \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{do}\, x \leftarrow p;\, q \colon B]\!] = h_1 \circ (\mathit{id} \oslash h_2) \circ J\Delta}$$

$$\frac{}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v \colon A]\!] = !} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v \colon A + B]\!] = \mathit{inl} \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v \colon A + B]\!] = \mathit{inr} \circ h}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A + B]\!] \qquad h_1 = [\![\Gamma, x \colon A \vdash_{\mathsf{c}} p \colon C]\!] \qquad h_2 = [\![\Gamma, y \colon B \vdash_{\mathsf{c}} q \colon C]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, y \mapsto q \colon C]\!] = [h_1, h_2] \circ J\, \mathit{dist} \circ (\mathit{id} \oslash Jh) \circ J\Delta}$$

$$\frac{h_1 = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!] \qquad h_2 = [\![\Gamma \vdash_{\mathsf{v}} w \colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \langle v, w \rangle \colon A \times B]\!] = \langle h_1, h_2 \rangle}$$

$$\frac{h_1 = [\![\Gamma \vdash_{\mathsf{v}} p \colon A \times B]\!] \qquad h_2 = [\![\Gamma, x \colon A, y \colon B \vdash_{\mathsf{c}} q \colon C]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{case}\ p\ \mathsf{of}\ \langle x, y \rangle \mapsto q \colon C]\!] = h_2 \circ (\mathit{id} \oslash Jh_1) \circ J\Delta}$$

Figure 5: Denotational semantics of FGCBV with coproducts.

$\mathit{id}_X \oslash \mathit{id}_{TY} \colon X \times TY \to T(X \times Y)$, which is clearly natural in X and Y. If follows that $f \oslash g = \tau \circ (f \times g)$. Indeed,

$$\begin{aligned} f \oslash g &= f \oslash (\mathit{id}^\star \circ \eta \circ g) \\ &= (\mathit{id} \oslash \mathit{id})^\star \circ (f \oslash \eta \circ g) \\ &= (\mathit{id} \oslash \mathit{id})^\star \circ (f \oslash Jg) \\ &= (\mathit{id} \oslash \mathit{id})^\star \circ J(f \times g) \\ &= \tau^\star \circ \eta \circ (f \times g) \\ &= \tau \circ (f \times g). \end{aligned}$$

Let $\gamma = \langle \mathit{id} \times \mathit{fst}, \mathit{snd} \circ \mathit{snd} \rangle \colon X \times (Y \times Z) \cong (X \times Y) \times Z$. The axioms of strength are verified as follows.

(1) Using Lemma 4.3 (4.c): $(T\,\mathit{snd}) \circ \tau = (J\,\mathit{snd})^\star \circ (\mathit{id} \oslash \mathit{id}) = \mathit{id}^\star \circ J\,\mathit{snd} = \mathit{id}^\star \circ \eta \circ \mathit{snd} = \mathit{snd}$.
(2) Using Lemma 4.3 (4.c): $T\gamma \circ \tau \circ (\mathit{id} \times \tau) = (J\gamma)^\star \circ (\mathit{id} \oslash (\mathit{id} \oslash \mathit{id})) = ((\mathit{id} \times \mathit{id}) \oslash \mathit{id})^\star \circ J\gamma = \tau^\star \circ \eta \circ \gamma = \tau \circ \gamma$.
(3) $\tau \circ (\mathit{id} \times \eta) = \tau^\star \circ \eta \circ (\mathit{id} \times \eta) = (\mathit{id} \oslash \mathit{id})^\star \circ J(\mathit{id} \times \eta) = (\mathit{id} \oslash \mathit{id})^\star \circ (\mathit{id} \oslash J\eta) = \mathit{id} \oslash (\mathit{id}^\star \circ J\eta) = \mathit{id} \oslash (\mathit{id}^\star \circ \eta \circ \eta) = \mathit{id} \oslash \eta = \mathit{id} \oslash J\,\mathit{id} = J(\mathit{id} \times \mathit{id}) = \eta$.
(4) $(\tau \circ (f \times g))^\star \circ \tau = (f \oslash g)^\star \circ (\mathit{id} \oslash \mathit{id}) = f \oslash g^\star = \tau \circ (f \times g^\star)$. □

Proposition 4.5 allows us to refactor the existing characterization of *closed Freyd categories* [LPT02, Theorem 7.3] along the following lines. In order to include higher-order types in the language, we would need to add $A \to B$ as a new type former and the following

term formation rules:

$$\frac{\Gamma, x\colon A \vdash_{\mathsf{c}} p\colon B}{\Gamma \vdash_{\mathsf{v}} \lambda x.\, p\colon A \to B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} w\colon A \qquad \Gamma \vdash_{\mathsf{v}} v\colon A \to B}{\Gamma \vdash_{\mathsf{c}} vw\colon B}$$

We would then need to provide the following additional semantic clauses:

$$\frac{h = [\![\Gamma, x\colon A \vdash_{\mathsf{c}} p\colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \lambda x.\, p\colon A \to B]\!] = \mathsf{curry}\, h} \qquad \frac{h_1 = [\![\Gamma \vdash_{\mathsf{v}} v\colon A \to B]\!] \qquad h_2 = [\![\Gamma \vdash_{\mathsf{v}} w\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} vw\colon B]\!] = (\mathsf{curry}^{-1}\, h_1) \circ (\mathit{id} \oslash J h_2) \circ J\Delta}$$

where $[\![A \to B]\!] = [\![A]\!] \multimap [\![B]\!]$, $\multimap\colon |\mathbf{V}| \times |\mathbf{C}| \to |\mathbf{C}|$, and curry is an isomorphism

$$\mathsf{curry}\colon \mathbf{C}(J(X \times A), B) \cong \mathbf{V}(X, A \multimap B) \tag{4.1}$$

natural in X. In particular, this says that J is left adjoint to $1 \multimap (\text{–})$, which, as we have seen in Proposition 3.4, means that $\mathbf{C}$ is isomorphic to the Kleisli category of a strong monad $\mathbf{T}$, and hence (4.1) amounts to $\mathbf{V}(X \times A, TB) \cong \mathbf{V}(X, A \multimap B)$, i.e. to the existence of *Kleisli exponentials*, which are exponentials of the form $(TB)^A$. We thus obtain the following

Corollary 4.6. *Let $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ be a Freyd category. The following are equivalent:*

- *an isomorphism* (4.1) *natural in X exists;*
- *for all $A \in |\mathbf{V}|$, $J(\text{–} \times A)\colon \mathbf{V} \to \mathbf{C}$ is a left adjoint;*
- *$\mathbf{C}$ is isomorphic to a Kleisli category of a strong monad, and Kleisli exponentials exist.*

A yet another way to express (4.1) is to state that the presheaves

$$\mathbf{C}(J(\text{–} \times A), B)\colon \mathbf{V}^{\mathsf{op}} \to \mathbf{Set}$$

are representable. We will use this formulation in our subsequent analysis of guardedness.

5. Guarded Freyd Categories

We proceed to recall the formal notion of guardedness [GSRP17, LG19].

Definition 5.1 (Guardedness)**.** A *guardedness predicate* on a co-Cartesian category $\mathbf{C}$ provides for all X, Y, $Z \in |\mathbf{C}|$ a subset $\mathbf{C}_\bullet(X, Y, Z) \subseteq \mathbf{C}(X, Y + Z)$, whose elements we write as $f\colon X \to Y \rangle Z$ and call guarded (in Z), such that

$$\textbf{(trv}_\bullet\textbf{)} \quad \frac{f\colon X \to Y}{\mathit{inl} \circ f\colon X \to Y \rangle Z} \qquad \textbf{(par}_\bullet\textbf{)} \quad \frac{f\colon X \to V \rangle W \qquad g\colon Y \to V \rangle W}{[f, g]\colon X + Y \to V \rangle W}$$

$$\textbf{(cmp}_\bullet\textbf{)} \quad \frac{f\colon X \to Y \rangle Z \qquad g\colon Y \to V \rangle W \qquad h\colon Z \to V + W}{[g, h] \circ f\colon X \to V \rangle W}$$

A *guarded (co-Cartesian) category* is a category equipped with a guardedness predicate. A *guarded functor* between two guarded categories is a functor $F\colon \mathbf{C} \to \mathbf{D}$ that strictly preserves coproducts, and preserves guardedness in the following sense: $f \in \mathbf{C}_\bullet(X, Y, Z)$ entails $f \in \mathbf{D}_\bullet(FX, FY, FZ)$.

It follows from the axioms of guardedness that $\mathbf{C}_\bullet$ is a functorial operator.

Proposition 5.2. *$\mathbf{C}_\bullet$ extends to a functor $\mathbf{C}^{\mathsf{op}} \times \mathbf{C} \times \mathbf{C} \to \mathbf{Set}$.*

Proof. The map $X, Y, Z \mapsto \mathbf{C}(X, Y + Z)$ is obviously functorial. We are left to check that given $f\colon X \to Y \rangle Z$, $g\colon X' \to X$, $h\colon Y \to Y'$ and $u\colon Z \to Z'$, $(h + u) \circ f \circ g\colon X' \to Y' \rangle Z'$. Observe first that $f \circ g\colon X' \to Y \rangle Z$. Indeed, $f \circ g = [f, f] \circ \mathit{inl} \circ g \to Y \rangle Z$ using **(trv$_\bullet$)** and **(cmp$_\bullet$)**. Next, again by **(cmp$_\bullet$)**, $[\mathit{inr} \circ h, \mathit{inl} \circ u] \circ f \circ g = (h + u) \circ f \circ g\colon X' \to Y' \rangle Z'$. □

In the sequel, we regard $\rangle$ as an operator that binds the weakest. Intuitively, $\mathbf{C}_\bullet(X,Y,Z)$ axiomatically distinguishes those morphisms $X \to Y + Z$ for which the program flow from X to Z is guarded, in particular, if $X = Z$ then the corresponding guarded loop can be safely closed. Note that the standard (totally defined) iteration is an instance with $\mathbf{C}_\bullet(X,Y,Z) = \mathbf{C}(X, Y+Z)$. Consider other instances.

Example 5.3 (Vacuous Guardedness [GS18])**.** The least guardedness predicate is as follows: $\mathbf{C}_\bullet(X,Y,Z) = \{\mathit{inl} \circ f \colon X \to Y + Z \mid f \in \mathbf{C}(X,Y)\}$. Such $\mathbf{C}$ is called *vacuously guarded*.

The following class of examples abstracts the monad of synchronization trees from Example 5.4: T can capture arbitrary "branching" computational effects besides $T = \mathcal{P}_\omega$ for nondeterminism, and H can capture arbitrary "action" functors besides $HX = A \times X$ for standard process algebra actions.

Example 5.4 (Coalgebraic Resumptions)**.** Let T be a monad on a co-Cartesian category $\mathbf{V}$, and let $H \colon \mathbf{V} \to \mathbf{V}$ be an endofunctor such that all fixpoints $T_H X = \nu\gamma.\, T(X + H\gamma)$ exist. These jointly yield a monad T_H, called the *(generalized) coalgebraic resumption monad (transform of* T*)* [PG14, GSRP17]. Then the Kleisli category of T_H is guarded with $f \colon X \to Y \rangle Z$ if

$$\begin{array}{ccc} X & \xrightarrow{\quad g \quad} & T(Y + HT_H(Y+Z)) \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle T(\mathit{inl}+\mathit{id})} \\ T_H(Y+Z) & \xrightarrow{\quad \mathsf{out} \quad} & T((Y+Z) + HT_H(Y+Z)) \end{array} \tag{5.1}$$

for some $g \colon X \to T(Y + HT_H(Y+Z))$. Guarded iteration operators canonically extend from T to T_H [LG19].

The next example is interesting in that the notion of guardedness is defined essentially the same way, but fixpoints of guarded morphisms need not exist.

Example 5.5 (Algebraic Resumptions)**.** A simple variation of the previous example involves least fixpoints $T^H X = \mu\gamma.\, T(X + H\gamma)$ instead of the greatest ones, and in^{-1} instead of out, where $\mathsf{in} \colon T(X + HT^H X) \to T^H X$ is the initial algebra structure of $T^H X$, which is an isomorphism by Lambek's lemma. However, we can no longer generally induce non-trivial (guarded) iteration operators for T^H.

Example 5.6. Let us describe natural guardedness predicates on the Kleisli categories of monads from Example 3.5.

(1) $TX = \nu\gamma.\, \mathcal{P}_\omega((X+1) + A \times \gamma)$ is a special case of Example 5.4. The guardedness condition (5.1) instantiates as follows: $f \colon X \to \nu\gamma.\, \mathcal{P}_\omega((Y+Z+1) + A \times \gamma)$ is guarded if $\mathsf{out} \circ f \colon X \to \mathcal{P}_\omega((Y+Z+1) + A \times T(Y+Z))$ factors through $\mathcal{P}_\omega((Y+1) + A \times T(Y+Z))$, i.e. the only allowed way to terminate through Z is that which is preceded by an action from A.

(2) For $TX = \mathcal{P}(A^\star \times (X+1) + A^\star)$, let $f \colon X \to Y \rangle Z$ if for every $x \in X$, $\mathit{inl}(w, \mathit{inl}(\mathit{inr}\, y)) \in f(x)$ entails $w \neq \epsilon$.

(3) For $TX = \mathcal{P}(A^\star \times (X+1) + (A^\star + A^\omega))$ guardedness is defined as in clause (2).

(4) For $TX = (\nu\gamma.\, X \times S + \gamma \times S)^S$, recall that $\mathbf{Set}_\mathsf{T}$ is isomorphic to a full subcategory of the Kleisli category of $\nu\gamma.\, (- + \gamma \times S)$, which is again an instance of Example 5.4 with $TX = X$ and $HX = X \times S$. The guardedness predicate for T thus restricts accordingly.

(5) For $TX = (\mathbb{R}_{\geqslant 0} \times X) + \bar{\mathbb{R}}_{\geqslant 0}$ let $f \colon X \to Y \rangle Z$ if $f(x) = \mathit{inl}\,(r, \mathit{inr}\, z)$ implies $r > 0$.

$$\frac{x\colon A \ \text{ in } \ \Gamma}{\Gamma \vdash_{\mathsf{v}} x\colon A} \qquad \frac{f\colon A \to B \in \Sigma_{\mathsf{v}} \quad \Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{v}} f(v)\colon B} \qquad \frac{f\colon A \to B \rangle C \in \Sigma_{\mathsf{c}} \qquad \Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{c}} f(v)\colon B \rangle C}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v\colon A \rangle B} \qquad \frac{\Gamma \vdash_{\mathsf{c}} p\colon A \rangle B \quad \Gamma, x\colon A \vdash_{\mathsf{c}} q\colon C \rangle D \quad \Gamma, y\colon B \vdash_{\mathsf{c}} r\colon C + D \rangle 0}{\Gamma \vdash_{\mathsf{c}} \mathsf{docase}\, p\, \mathsf{of}\, \mathsf{inl}\, x \mapsto q;\, \mathsf{inr}\, y \mapsto r\colon C \rangle D}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon 0}{\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v\colon A} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon A}{\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v\colon A + B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} v\colon B}{\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v\colon A + B}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A + B \qquad \Gamma, x\colon A \vdash_{\mathsf{c}} p\colon C \rangle D \qquad \Gamma, y\colon B \vdash_{\mathsf{c}} q\colon C \rangle D}{\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, y \mapsto q\colon C \rangle D}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} v\colon A \qquad \Gamma \vdash_{\mathsf{v}} w\colon B}{\Gamma \vdash_{\mathsf{v}} \langle v, w\rangle\colon A \times B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} p\colon A \times B \qquad \Gamma, x\colon A, y\colon B \vdash_{\mathsf{c}} q\colon C \rangle D}{\Gamma \vdash_{\mathsf{c}} \mathsf{case}\ p\ \mathsf{of}\ \langle x, y\rangle \mapsto q\colon C \rangle D}$$

Figure 6: Term formation rules of guarded FGCBV.

We proceed to extend the language in Figure 4 with guardedness data. As before, Σ_{v} consists of constructs of the form $f\colon A \to B$, while Σ_{c} consists of constructs of the form $f\colon A \to B \rangle C$, indicating guardedness in C. The new formation rules are then given in Figure 6. The rule for return now introduces a coproduct summand B with respect to which the computation is vacuously guarded, thus adhering to $\mathbf{(trv_\bullet)}$. The rule for binding now must incorporate $\mathbf{(cmp_\bullet)}$, which requires the following modification of the syntax:

$$\mathsf{docase}\, p\, \mathsf{of}\, \mathsf{inl}\, x \mapsto q;\, \mathsf{inr}\, y \mapsto r$$

The latter construct is meant to be equivalent to $\mathsf{do}\, z \leftarrow p;\, \mathsf{case}\, z\, \mathsf{of}\, \mathsf{inl}\, x \mapsto q;\, \mathsf{inr}\, y \mapsto r.$ modulo guardedness information. Finally, $\mathbf{(par_\bullet)}$ is captured by the formation rule for case, which is essentially unchanged w.r.t. Figure 4. An analogue of the iteration operator (3.2) in the new setting would be the rule:

$$\frac{\Gamma \vdash_{\mathsf{c}} p\colon A \rangle 0 \qquad \Gamma, x\colon A \vdash_{\mathsf{c}} q\colon B \rangle C + A}{\Gamma \vdash_{\mathsf{c}} \mathsf{iter}\, x \leftarrow p;\, q\colon B \rangle C}$$

Example 5.7 (Weakening)**.** One can expect that the judgement $f\colon X \to Y \rangle Z + W$ entails $f\colon X \to Y + Z \rangle W$, meaning that if a morphism is guarded w.r.t. an object $Z + W$, then it is guarded w.r.t. to its part W. The corresponding *weakening* principle

$$\mathbf{(wkn_\bullet)} \quad \frac{f\colon X \to Y \rangle Z + W}{f\colon X \to Y + Z \rangle W}$$

is indeed derivable from $\mathbf{(trv_\bullet)}$, $\mathbf{(par_\bullet)}$ and $\mathbf{(cmp_\bullet)}$. In terms of guarded FGCBV, this corresponds to constructing the following term from a given $\Gamma \vdash_{\mathsf{c}} p\colon A \rangle B + C$:

$$\begin{aligned}\Gamma \vdash_{\mathsf{c}} \mathsf{docase}\, p\, \mathsf{of}\, \mathsf{inl}\, x \mapsto\ & \mathsf{return}(\mathsf{inl}\, x);\\ & \mathsf{inr}\, z \mapsto\ \mathsf{case}\, z\, \mathsf{of}\, \mathsf{inl}\, x \mapsto \mathsf{return}(\mathsf{inl}(\mathsf{inr}\, x));\\ & \qquad\qquad \mathsf{inr}\, y \mapsto \mathsf{return}(\mathsf{inr}\, y)\colon A + B \rangle C.\end{aligned}$$

Example 5.8. The updated effectful signature of Example 3.1 now involves $a\colon 1 \to 0 \rangle 1$ and $\mathsf{toss}\colon 1 \to 2 \rangle 0$, indicating that actions guard everything, while nondeterminism guards

$$\frac{}{[\![x_1\colon A_1,\dots,x_n\colon A_n \vdash_{\mathsf{v}} x_i\colon A_i]\!] = \mathit{proj}_i}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} f(v)\colon B]\!] = [\![f]\!] \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} f(v)\colon B \between C]\!] = [\![f]\!] \circ Jh}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v\colon A \between B]\!] = J\,\mathit{inl} \circ Jh}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{c}} p\colon A \between B]\!] \quad h_1 = [\![\Gamma, x\colon A \vdash_{\mathsf{c}} q\colon C \between D]\!] \quad h_2 = [\![\Gamma, y\colon B \vdash_{\mathsf{c}} r\colon C + D \between 0]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{docase}\, p\, \mathsf{of}\, \mathsf{inl}\, x \mapsto q;\ \mathsf{inr}\, y \mapsto r\colon C \between D]\!] = [h_1, [\mathit{id}, !] \circ h_2] \circ J\,\mathit{dist} \circ (\mathit{id} \oslash h) \circ J\Delta}$$

$$\frac{}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v\colon A]\!] = !} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v\colon A + B]\!] = \mathit{inl} \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v\colon A + B]\!] = \mathit{inr} \circ h}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v\colon A + B]\!] \quad h_1 = [\![\Gamma, x\colon A \vdash_{\mathsf{c}} p\colon C \between D]\!] \quad h_2 = [\![\Gamma, y\colon B \vdash_{\mathsf{c}} q\colon C \between D]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\ \mathsf{inr}\, y \mapsto q\colon C \between D]\!] = [h_1, h_2] \circ J\,\mathit{dist} \circ (\mathit{id} \oslash Jh) \circ J\Delta}$$

$$\frac{h_1 = [\![\Gamma \vdash_{\mathsf{v}} v\colon A]\!] \qquad h_2 = [\![\Gamma \vdash_{\mathsf{v}} w\colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \langle v, w\rangle\colon A \times B]\!] = \langle h_1, h_2\rangle}$$

Figure 7: Denotational semantics of guarded FGCBV over guarded Freyd categories.

nothing. The signature Σ_{c} from Example 3.2 can be refined to $\{\mathsf{put}\colon S \to 0 \between 1, \mathsf{get}\colon 1 \to S \between 0\}$, meaning again that put guards everything and get guards nothing. Example 3.3 is more subtle since $\mathsf{wait}\colon \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ is meant to be guarded only for non-zero inputs. We thus can embed the involved case distinction into wait by redefining it as $\mathsf{wait}\colon \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0} \between \mathbb{R}_{\geqslant 0}$.

Definition 5.9 (Guarded Freyd Category)**.** A distributive Freyd category $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ is guarded if $\mathbf{C}$ is guarded and the action of $\mathbf{V}$ on $\mathbf{C}$ preserves guardedness in the following sense: Given $f \in \mathbf{V}(A, B)$, $g \in \mathbf{C}_\bullet(X, Y, Z)$, $J\,\mathit{dist} \circ (f \oslash g) \in \mathbf{C}_\bullet(A \times X, B \times Y, B \times Z)$.

The semantics of $(\Sigma_{\mathsf{v}}, \Sigma_{\mathsf{c}})$ over a guarded Freyd category $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ interprets types and operations from Σ_{v} as before and sends each $f\colon A \to B \between C \in \Sigma_{\mathsf{c}}$ to $[\![f]\!] \in \mathbf{C}_\bullet([\![A]\!], [\![B]\!], [\![C]\!])$. Terms in context are now interpreted as $[\![\Gamma \vdash_{\mathsf{v}} v\colon B]\!] \in \mathbf{V}([\![\Gamma]\!], [\![B]\!])$ and $[\![\Gamma \vdash_{\mathsf{c}} p\colon B \between C]\!] \in \mathbf{C}([\![\Gamma]\!], [\![B]\!] + [\![C]\!])$, according to the rules in Figure 7. This is well-defined, which can be easily shown by structural induction:

Proposition 5.10. *For any derivable* $\Gamma \vdash_{\mathsf{c}} p\colon A \between B$, $[\![\Gamma \vdash_{\mathsf{c}} p\colon A \between B]\!] \in \mathbf{C}_\bullet([\![\Gamma]\!], [\![A]\!], [\![B]\!])$.

6. Representing Guardedness

In Section 4 we explored the combination of strength (i.e., multivariable contexts) and the representability of presheaves $\mathbf{C}(J(\text{–}), X)\colon \mathbf{V}^{\mathsf{op}} \to \mathbf{Set}$, sticking to the bottom face of the cube in Figure 1. Our plan now is to obtain additional concepts by examining the representability of $\mathbf{C}_\bullet(J(\text{–}), X, Y)\colon \mathbf{V}^{\mathsf{op}} \to \mathbf{Set}$. Note that representability of guardedness together with function spaces amounts to representability of $\mathbf{C}_\bullet(J(\text{–} \times X), Y, Z)\colon \mathbf{V}^{\mathsf{op}} \to \mathbf{Set}$, i.e. to the existence of an endofunctor $\multimap\colon \mathbf{V}^{\mathsf{op}} \times \mathbf{C} \times \mathbf{C} \to \mathbf{C}$, such that $\mathbf{C}_\bullet(J(\text{–} \times X), Y, Z) \cong \mathbf{V}(\text{–}, X \multimap_Z$

Y). This is exactly the structure one would need to extend Figure 6 with function spaces as follows:

$$\frac{\Gamma, x\colon A \vdash_{\mathsf{c}} p\colon B \between C}{\Gamma \vdash_{\mathsf{v}} \lambda x.\, p\colon A \to_C B} \qquad \frac{\Gamma \vdash_{\mathsf{v}} w\colon A \qquad \Gamma \vdash_{\mathsf{v}} v\colon A \to_C B}{\Gamma \vdash_{\mathsf{c}} vw\colon B \between C}$$

The decorated function spaces $A \to_C B$ can then be interpreted as $[\![A]\!] \multimap_{[\![C]\!]} [\![B]\!]$, which is a subobject of the Kleisli exponential $[\![A]\!] \to T([\![B]\!] + [\![C]\!])$, consisting of guarded morphisms.

Definition 6.1. Given $J\colon \mathbf{V} \to \mathbf{C}$, where $\mathbf{C}$ is guarded, we call the guardedness predicate $\mathbf{C}_\bullet$ J-representable if for all $X, Y \in |\mathbf{C}|$ the presheaf $\mathbf{C}_\bullet(J(\text{–}), X, Y)\colon \mathbf{V}^{\mathsf{op}} \to \mathbf{Set}$ is representable, i.e. for all $X, Y \in |\mathbf{C}|$ there is $U(X,Y) \in |\mathbf{V}|$ such that

$$\mathbf{C}_\bullet(JZ, X, Y) \cong \mathbf{V}(Z, U(X,Y)) \tag{6.1}$$

naturally in Z. A guardedness predicate $\mathbf{C}_\bullet$ is called J-guarded if it is equipped with a J-representable guardedness predicate.

Lemma 6.2. *Given an identity-on-objects functor $J\colon \mathbf{V} \to \mathbf{C}$, $\mathbf{C}_\bullet$ is J-representable iff*

- *there is a family of objects $(U(X,Y) \in |\mathbf{V}|)_{X,Y \in |\mathbf{C}|}$;*
- *there is a family of guarded morphisms $(\epsilon_{X,Y}\colon U(X,Y) \to X \between Y)_{X,Y \in |\mathbf{C}|}$;*
- *there is an operator $(\text{–})^\sharp\colon \mathbf{C}_\bullet(Z, X, Y) \to \mathbf{V}(Z, U(X,Y))$ sending each $f\colon Z \to X \between Y$ to the unique morphism $f^\sharp$ for which the diagram*

$$\begin{array}{ccc} & & U(X,Y) \\ & {\scriptstyle Jf^\sharp}\nearrow & \downarrow{\scriptstyle \epsilon_{X,Y}} \\ Z & \xrightarrow[f]{} & X + Y \end{array}$$

commutes.

These conditions entail that U is a bifunctor and that $\epsilon_{X,Y}$ is natural in X and Y.

Proof. First, we argue that the declared characterization entails that U is a bifunctor and $\epsilon_{X,Y}$ is natural in X and Y. Let $g\colon X \to X'$, $h\colon Y \to Y'$, and note that $(g+h) \circ \epsilon_{X,Y}\colon JU(X,Y) \to X' \between Y'$. Then the diagram

$$\begin{array}{ccc} JU(X,Y) & \xrightarrow{JU(g,h)} & JU(X',Y') \\ {\scriptstyle \epsilon_{X,Y}}\downarrow & & \downarrow{\scriptstyle \epsilon_{X',Y'}} \\ X + Y & \xrightarrow{g+h} & X' + Y' \end{array}$$

commutes for some $U(g,h)\colon U(X,Y) \to U(X',Y')$, uniquely determined by g and h. The fact that thus defined $U(\text{–},\text{–})$ is functorial is obvious by definition. Moreover, the above diagram establishes the naturality of $\epsilon_{X,Y}$ in X and Y.

Observe that, by Yoneda lemma, for any bifunctor U, a natural transformation $\xi\colon \mathbf{V}(\text{–}, U(X,Y)) \to \mathbf{C}_\bullet(J(\text{–}), X, Y)$ is uniquely determined by a morphism $\epsilon_{X,Y}\colon UJ(X,Y) \to X \between Y$. We proceed to show that componentwise isomorphic ξ correspond to those $\epsilon_{X,Y}$ for which the above-described maps $(\text{–})^\sharp$ exist. Note that ξ and ϵ are connected as follows:

$$\begin{aligned} \epsilon_{X,Y} &= \xi_{U(X,Y)}(\mathit{id}\colon U(X,Y) \to U(X,Y)), \\ \xi_X(f\colon Z \to U(X,Y)) &= \epsilon_{X,Y} \circ Jf. \end{aligned}$$

The map $\xi_X\colon \mathbf{V}(Z, U(X,Y)) \to \mathbf{C}_\bullet(JZ, X, Y)$ is a bijection iff every $f\colon JZ \to JU(X,Y)$ is of the form $\epsilon_{X,Y} \circ Jg$ for some $g\colon Z \to U(X,Y)$, which is uniquely identified by f, in

other words, for every $f\colon JZ \to JU(X,Y)$ there is a unique $f^\natural\colon Z \to U(X,Y)$, such that $f = \epsilon_{X,Y} \circ Jf^\natural$. □

Lemma 6.3. *If* $\mathbf{C}$ *is* J*-guarded, then* $J \dashv U(\text{–}, 0)$ *with* U *as in Lemma 6.2.*

Proof. Suppose that $\mathbf{C}$ is J-representable with $J\colon \mathbf{V} \to \mathbf{C}$. Observe that $\mathbf{C}_\bullet(JX, A, 0)$ is isomorphic to $\mathbf{C}(JX, A)$ naturally in X: the components of the isomorphism are the maps $f \mapsto \mathit{inl} \circ f$ for $f \in \mathbf{C}(JX, A)$ and $g \mapsto [\mathit{id}, !] \circ g$ for $g \in \mathbf{C}_\bullet(JX, A, 0)$. Using (6.1), we thus arrive at

$$\mathbf{V}(X, U(A,0)) \cong \mathbf{C}_\bullet(JX, A, 0) \cong \mathbf{C}(JX, A),$$

i.e. $J \dashv U(\text{–}, 0)$. □

By Lemma 6.3, representability fails already if J has no right adjoint. Instructive examples of non-representability are thus only those where J *does* have a right adjoint.

Proposition 6.4. *Let* $\mathbf{T}$ *be a monad over the category of sets* **Set** *with the axiom of choice. If* $\mathbf{Set}_\mathbf{T}$ *is guarded, the guardedness predicate is representable iff every* $f\colon X \to T(Y + Z)$ *is guarded whenever all the compositions* $1 \hookrightarrow X \xrightarrow{f} T(Y + Z)$ *are guarded.*

Proof. It follows from previous results [GRS21, Proposition 12] that in any category, where every morphism admits an image factorization (specifically in **Set**), representability of guardedness in the Kleisli category of a monad $\mathbf{T}$ is equivalent to the following conditions.

(1) for all sets X and Y, there is a greatest subobject $Z \hookrightarrow T(X + Y)$, which is guarded as a morphism;

(2) for every regular epic $e\colon X' \to X$ and every morphism $f\colon X \to T(Y + Z)$, $f \circ e\colon X' \to Y \between Z$ implies $f\colon X \to Y \between Z$.

The second clause follows for **Set**: by the axiom of choice, every $e\colon X' \twoheadrightarrow X$ has a section, say m, and then $f \circ e\colon X' \to Y \between Z$ implies $f = f \circ e \circ m\colon X' \to Y \between Z$. The second clause is equivalent to the property that the injection $\bigcup_{Z \hookrightarrow X \between Y} Z \hookrightarrow T(X + Y)$ is guarded (and hence is the largest guarded subobject by construction). Precomposing this map with any map whose source is 1 yields a guarded map by definition, hence the condition from the proposition's statement is sufficient. Let us show that it is necessary. Let $f\colon X \to T(Y + Z)$, and suppose that all the compositions $1 \hookrightarrow X \xrightarrow{f} T(Y + Z)$ are guarded. If the largest guarded subobject exists, it must be the union of all such maps. Since this union is precisely the original map f, it is guarded. □

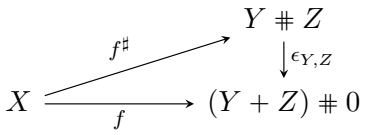
Example 6.5 (Failure of Representability)**.** In **Set**, let $f\colon X \to Y + Z$ be guarded in Z if $\{z \in Z \mid f^{-1}(\mathit{inr}\, z) \neq \emptyset\}$ is finite. The axioms of guardedness are easy to verify. By Proposition 6.4, this predicate is not Id-representable, as any $1 \hookrightarrow X \xrightarrow{\mathit{inr}} 0 + X$ is guarded, but inr is not, unless X is finite.

In what follows, we will use $\#$ as a binary operation that binds stronger than monoidal products ($\otimes$, $+$, …), so, e.g. $X \otimes Y \# Z$ will read as $X \otimes (Y \# Z)$.

Theorem 6.6. *Given an identity-on-objects guarded* $J\colon \mathbf{V} \to \mathbf{C}$, $\mathbf{C}_\bullet$ *is* J*-representable iff*

- *there is a bifunctor* $\#\colon \mathbf{V} \times \mathbf{V} \to \mathbf{V}$*, such that* $\text{–}\#0$ *is a monad and* $\mathbf{C} \cong \mathbf{V}_{\text{–}\#0}$*;*
- *there is a family of guarded morphisms (w.r.t. the guardedness predicate, induced by* $\mathbf{C} \cong \mathbf{V}_{\text{–}\#0}$*)* $(\epsilon_{X,Y}\colon X \# Y \to X \between Y)_{X,Y \in |\mathbf{V}|}$*, natural in* X *and* Y*;*

- *for every guarded $f\colon X \to Y \rangle Z$, there is unique $f^\sharp\colon X \to Y \mathbin{\#} Z$, such that the diagram*



commutes.

Proof. ($\Rightarrow$) By Lemma 6.3 and Proposition 3.4, assume w.l.o.g. that $\mathbf{C} = \mathbf{V}_\mathbf{T}$ for $T = U(J(\text{–}), 0)$. Let $f \mathbin{\#} g = U(Jf, Jg)$. Lemma 6.2 yields the desired guarded morphisms $\epsilon_{X,Y}\colon X \mathbin{\#} Y \to (X + Y) \mathbin{\#} 0$, which satisfy the requisite universal property, obtained by interpreting the corresponding property of Lemma 6.2 in $\mathbf{C} = \mathbf{V}_\mathbf{T}$.

($\Leftarrow$) Conversely, given a bifunctor $\#\colon \mathbf{V} \times \mathbf{V} \to \mathbf{V}$ with the described properties, let $\mathbf{T}$ be the monad on $\text{–} \mathbin{\#} 0$ and apply Lemma 6.2 with $\mathbf{C} = \mathbf{V}_\mathbf{T}$, J being the free functor $J\colon \mathbf{V} \to \mathbf{V}_\mathbf{T}$ and $U(X, Y) = X \mathbin{\#} Y$. □

Theorem 6.6 provides a bijective correspondence between morphisms $f\colon X \to Y \rangle Z$ in $\mathbf{C}$ and the morphisms $f^\sharp\colon X \to Y \mathbin{\#} Z$ in $\mathbf{V}$, representing them. Uniqueness of the $f^\sharp$ is easily seen to be equivalent to the monicity of the $\epsilon_{X,Z}$.

7. Guarded Parameterized Monads

Theorem 6.6 describes guardedness as a certain bifunctor $\#\colon \mathbf{V} \times \mathbf{V} \to \mathbf{V}$ and a family of morphisms $(\epsilon_{X,Y}\colon X \mathbin{\#} Y \to X \rangle Y)_{X,Y \in |\mathbf{V}|}$, so that the guardedness predicate is derivable. However, the guardedness laws are still formulated in terms of this predicate, and not in terms of $\#$ and ϵ. To make the new definition of guardedness self-contained, we must identify a collection of canonical morphisms and a complete set of equations relating them, in the sense that the guardedness laws for all derived guarded morphisms follow. For example, by applying $(\text{–})^\sharp$ to the composition

$$X \mathbin{\#} (Y + Z) \xrightarrow{\epsilon_{X,Y+Z}} (X + (Y + Z)) \mathbin{\#} 0 \cong ((X + Y) + Z)) \mathbin{\#} 0$$

we obtain a morphism $\upsilon_{X,Y,Z}\colon X \mathbin{\#} (Y + Z) \to (X + Y) \mathbin{\#} Z$, which represents weakening of the guardedness guarantee: in $X \mathbin{\#} (Y + Z)$ the guarded part is $Y + Z$, while in $(X + Y) \mathbin{\#} Z$ the guarded part is only Z. It should not make a difference though if starting from $X \mathbin{\#} (Y + (Z + V))$ we apply υ twice or rearrange $Y + (Z + V)$ by associativity and subsequently apply υ only once – the results must be canonically isomorphic, which is indeed provable. Similarly to this case we introduce further morphisms and derive laws relating them. We then prove that the resulting axiomatization enjoys a coherence property (Theorem 7.3) in the style of Mac Lane's coherence theorem for (symmetric) monoidal categories [ML71]. In what follows, we switch from coproducts to general symmetric tensor products, as coherence can only hold if the corresponding structure is not involved.

Definition 7.1 (Guarded Parameterized Monad)**.** A *guarded parameterized monad* on a symmetric monoidal category $(\mathbf{V}, \otimes, I)$ consists of a bifunctor $\#\colon \mathbf{V} \times \mathbf{V} \to \mathbf{V}$ and natural transformations

$$\eta\colon A \to A \mathbin{\#} I,$$
$$\xi\colon (A \mathbin{\#} B) \mathbin{\#} C \to A \mathbin{\#} (B \otimes C), \qquad \upsilon\colon A \mathbin{\#} (B \otimes C) \to (A \otimes B) \mathbin{\#} C,$$
$$\zeta\colon A \mathbin{\#} (B \mathbin{\#} C) \to A \mathbin{\#} (B \otimes C), \qquad \chi\colon A \mathbin{\#} B \otimes C \mathbin{\#} D \to (A \otimes C) \mathbin{\#} (B \otimes D).$$

such that the following diagrams commute, where $\cong$ refers to the obvious canonical isomorphisms

$$\begin{array}{ccc} (A \# I) \# B & \xrightarrow{\xi} & A \# (I \otimes B) \\ & {\scriptstyle \eta \# id} \nwarrow \quad \nearrow {\scriptstyle \cong} & \\ & A \# B & \end{array} \qquad \begin{array}{ccc} (A \# B) \# I & \xrightarrow{\xi} & A \# (B \otimes I) \\ & {\scriptstyle \eta} \nwarrow \quad \nearrow {\scriptstyle \cong} & \\ & A \# B & \end{array}$$

$$\begin{array}{ccc} ((A \# B) \# C) \# D & \xrightarrow{\xi} & (A \# B) \# (C \otimes D) \\ {\scriptstyle \xi \# id}\downarrow & & \\ (A \# (B \otimes C)) \# D & & \downarrow {\scriptstyle \xi} \\ {\scriptstyle \xi}\downarrow & & \\ A \# ((B \otimes C) \otimes D) & \xrightarrow{\cong} & A \# (B \otimes (C \otimes D)) \end{array}$$

$$\begin{array}{ccc} A \# (I \otimes B) & \xrightarrow{\upsilon} & (A \otimes I) \# B \\ & {\scriptstyle \cong} \nwarrow \quad \nearrow {\scriptstyle \cong} & \\ & A \# B & \end{array}$$

$$\begin{array}{ccc} A \# (B \otimes (C \otimes D)) & \xrightarrow{\cong} & A \# ((B \otimes C) \otimes D) \\ {\scriptstyle \upsilon}\downarrow & & \\ (A \otimes B) \# (C \otimes D) & & \downarrow {\scriptstyle \upsilon} \\ {\scriptstyle \upsilon}\downarrow & & \\ ((A \otimes B) \otimes C) \# D & \xrightarrow{\cong} & (A \otimes (B \otimes C)) \# D \end{array}$$

$$\begin{array}{ccc} A \otimes B & \xrightarrow{\eta \otimes \eta} & A \# I \otimes B \# I \\ {\scriptstyle \eta}\downarrow & & \downarrow {\scriptstyle \chi} \\ (A \otimes B) \# I & \xrightarrow{\cong} & (A \otimes B) \# (I \otimes I) \end{array} \qquad \begin{array}{ccc} A \# B \otimes C \# D & \xrightarrow{\cong} & C \# D \otimes A \# B \\ {\scriptstyle \chi}\downarrow & & \downarrow {\scriptstyle \chi} \\ (A \otimes C) \# (B \otimes D) & \xrightarrow{\cong} & (C \otimes A) \# (D \otimes B) \end{array}$$

$$\begin{array}{ccc} A \# B \otimes (C \# D \otimes E \# F) & \xrightarrow{\cong} & (A \# B \otimes C \# D) \otimes E \# F \\ {\scriptstyle id \# \chi}\downarrow & & \downarrow {\scriptstyle \chi \# id} \\ A \# B \otimes (C \otimes E) \# (D \otimes F) & & (A \otimes C) \# (B \otimes D) \otimes E \# F \\ {\scriptstyle \chi}\downarrow & & \downarrow {\scriptstyle \chi} \\ (A \otimes (C \otimes E)) \# (B \otimes (D \otimes F)) & \xrightarrow{\cong} & ((A \otimes C) \otimes E) \# ((B \otimes D) \otimes F) \end{array}$$

$$\begin{array}{ccc} (A \# B) \# (I \# I) & \xrightarrow{\chi} & (A \otimes I) \# (B \otimes I) \\ & {\scriptstyle id \otimes \eta} \nwarrow \quad \approx & \\ & (A \# B) \otimes I & \end{array} \qquad \begin{array}{ccc} A \# (B \# I) & \xrightarrow{\zeta} & A \# (B \otimes I) \\ & {\scriptstyle id \# \eta} \nwarrow \quad \approx & \\ & A \# B & \end{array}$$

$$\begin{array}{ccccc} & & A \# ((B \# C) \# (D \# E)) & & \\ & {\scriptstyle \zeta} \swarrow & & \searrow {\scriptstyle id \# \zeta} & \\ A \# ((B \# C) \otimes (D \# E)) & & & & A \# ((B \# C) \# (D \otimes E)) \\ {\scriptstyle id \# \chi}\downarrow & & & & \downarrow {\scriptstyle id \# \xi} \\ A \# ((B \otimes D) \# (C \otimes E)) & & & & A \# (B \# (C \otimes (D \otimes E))) \\ {\scriptstyle \zeta}\downarrow & & & & \downarrow {\scriptstyle \zeta} \\ A \# ((B \otimes D) \otimes (C \otimes E)) & & \xrightarrow{\cong} & & A \# (B \otimes (C \otimes (D \otimes E))) \end{array}$$

$$\begin{array}{ccc}
 & (A \# (B \# C)) \# (D \# E) & \\
{\scriptstyle \xi}\swarrow & & \searrow{\scriptstyle \zeta \# id} \\
A \# ((B \# C) \otimes (D \# E)) & & (A \# (B \otimes C)) \# (D \# E) \\
{\scriptstyle id \# \chi}\downarrow & & \downarrow{\scriptstyle \zeta} \\
A \# ((B \otimes D) \# (C \otimes E)) & & (A \# (B \otimes C)) \# (D \otimes E) \\
{\scriptstyle \zeta}\downarrow & & \downarrow{\scriptstyle \xi} \\
A \# ((B \otimes D) \otimes (C \otimes E)) & \xrightarrow{\cong} & A \# ((B \otimes C) \otimes (D \otimes E))
\end{array}$$

$$\begin{array}{ccc}
((A \# B) \# C) \otimes ((D \# E) \# F) & \xrightarrow{\chi} & (A \# B \otimes D \# E) \# (C \otimes F) \\
{\scriptstyle \xi \otimes \xi}\downarrow & & \downarrow{\scriptstyle \chi \# id} \\
A \# (B \otimes C) \otimes D \# (E \otimes F) & & ((A \otimes D) \# (B \otimes E)) \# (C \otimes F) \\
{\scriptstyle \chi}\downarrow & & \downarrow{\scriptstyle \xi} \\
(A \otimes D) \# ((B \otimes C) \otimes (E \otimes F)) & \xrightarrow{\cong} & (A \otimes D) \# ((B \otimes E) \otimes (C \otimes F))
\end{array}$$

$$\begin{array}{ccc}
(A \# (B \# C)) \otimes (D \# (E \# F)) & \xrightarrow{\chi} & (A \otimes D) \# (B \# C \otimes E \otimes F) \\
{\scriptstyle \zeta \otimes \zeta}\downarrow & & \downarrow{\scriptstyle id \# \chi} \\
A \# (B \otimes C) \otimes D \# (E \otimes F) & & (A \otimes D) \# ((B \otimes E) \# (C \otimes F)) \\
{\scriptstyle \chi}\downarrow & & \downarrow{\scriptstyle \zeta} \\
(A \otimes D) \# ((B \otimes C) \otimes (E \otimes F)) & \xrightarrow{\cong} & (A \otimes D) \# ((B \otimes E) \otimes (C \otimes F))
\end{array}$$

$$\begin{array}{ccc}
(A \# B) \# (C \# D \otimes E \# F) & \xrightarrow{\upsilon} & (A \# B \otimes C \# D) \# (E \# F) \\
{\scriptstyle id \# \chi}\downarrow & & \downarrow{\scriptstyle \chi \# id} \\
(A \# B) \# ((C \otimes E) \# (D \otimes F)) & & ((A \otimes C) \# (B \otimes D)) \# (E \# F) \\
{\scriptstyle \zeta}\downarrow & & \downarrow{\scriptstyle \zeta} \\
(A \# B) \# ((C \otimes E) \otimes (D \otimes F)) & & ((A \otimes C) \# (B \otimes D)) \otimes (E \# F) \\
{\scriptstyle \xi}\downarrow & & \\
A \# (B \otimes ((C \otimes D) \otimes (E \otimes F))) & & \downarrow{\scriptstyle \xi} \\
{\scriptstyle \cong}\downarrow & & \\
A \# (C \otimes ((B \otimes D) \otimes (E \otimes F))) & \xrightarrow{\upsilon} & (A \otimes C) \# ((B \otimes D) \otimes (E \otimes F))
\end{array}$$

$$\begin{array}{ccc}
A \# B \otimes C \# (D \otimes E) & \xrightarrow{id \otimes \upsilon} & A \# B \otimes (C \otimes D) \# E \\
\chi \downarrow & & \downarrow \chi \\
(A \otimes C) \# (B \otimes (D \otimes E)) & & (A \otimes (C \otimes D)) \# (B \otimes E) \\
\cong \downarrow & & \downarrow \cong \\
(A \otimes C) \# (D \otimes (B \otimes E)) & \xrightarrow{\upsilon} & ((A \otimes C) \otimes D)) \# (B \otimes E)
\end{array}$$

$$\begin{array}{ccc}
(A \# (B \otimes C)) \# D & \xrightarrow{\upsilon \# id} & ((A \otimes B) \# C) \# D \\
\xi \downarrow & & \\
A \# ((B \otimes C) \otimes D) & & \downarrow \xi \\
\cong \downarrow & & \\
A \# (B \otimes (C \otimes D)) & \xrightarrow{\upsilon} & (A \otimes B) \# (C \otimes D)
\end{array}$$

$$\begin{array}{ccc}
A \# (B \# (C \otimes D)) & \xrightarrow{id \# \upsilon} & A \# ((B \otimes C) \# D) \\
\zeta \downarrow & & \downarrow \zeta \\
A \# (B \otimes (C \otimes D)) & \xrightarrow{\cong} & A \# ((B \otimes C) \otimes D)
\end{array}$$

Remark 7.2. The first three laws (relating η and ξ) identify guarded parameterized monads as *parametric monads* in the sense of Melliés [Mel17], and subsequently renamed to *graded monads* [FKM16]. In our case, more specifically, $\#$ is a $\mathbf{V}$-graded monad on $\mathbf{V}$.

The relevance of the presented axiomatization is certified by the following

Theorem 7.3 (Coherence)**.** *Let $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_2'$ be expressions, built from $\otimes$, $\#$ and I over a set of letters, in such a way that $\mathcal{E}_1$ and $\mathcal{E}_2 \# \mathcal{E}_2'$ contain every letter at most once and neither $\mathcal{E}_2$ nor $\mathcal{E}_2'$ contain $\#$. Let f and g be two expressions built with $\otimes$ and $\#$ over identities, η, υ, ξ, ζ, χ, associators, unitors, braidings and inverses of associators and unitors, in such a way that the judgements $f \colon \mathcal{E}_1 \to \mathcal{E}_2 \# \mathcal{E}_2'$ and $g \colon \mathcal{E}_1 \to \mathcal{E}_2 \# \mathcal{E}_2'$ are formally valid. Then $f = g$ follows from the axioms of guarded parameterized monads.*

Proof. For the sake of the present proof, let us introduce some nomenclature. We will use the following names correspondingly for associators, right unitors and braidings:

$$\alpha \colon A \otimes (B \otimes C) \to (A \otimes B) \otimes C, \qquad \rho \colon A \otimes I \to A, \qquad \gamma \colon A \otimes B \to B \otimes A.$$

We dispense with the left unitor, since our monoidal structure is symmetric.

Let us refer to the expressions built from $\otimes$, $\#$ and I over some alphabet of *object names*, fixed globally from now on, in such a way that every object name occurs at most once, as *object expressions*. We refer to the expressions built with $\otimes$ and $\#$ over id, η, υ, ξ, ζ, χ, α, α^{-1}, ρ, ρ^{-1}, γ as *morphism expressions*. An *isomorphism expression* is then a morphism expression that does not involve η, υ, ξ, ζ, χ. Every morphism expression f unambiguously identifies object expressions $\mathcal{E}_1$ and $\mathcal{E}_2$ for which the judgement $f \colon \mathcal{E}_1 \to \mathcal{E}_2$ is formally valid (that is, in any category, where we can interpret f, $\mathcal{E}_1$ and $\mathcal{E}_2$, f is a morphism from $\mathcal{E}_1$ to $\mathcal{E}_2$). For two morphism expressions $f, g \colon \mathcal{E}_1 \to \mathcal{E}_2$, let $f \equiv g$ denote *'$f = g$ follows from the axioms of guarded parameterized monads'*. An object expression is *normal* if it is of

the form $\mathcal{E} \# \mathcal{E}'$ and $\mathcal{E}$ and $\mathcal{E}'$ do not contain $\#$. A morphism expression is *simple* if it is a composition of isomorphism expressions between normal form expressions and instances of υ.

For an object expression $\mathcal{E}$, we define object expressions $\mathsf{nf}_1(\mathcal{E})$ and $\mathsf{nf}_2(\mathcal{E})$ recursively with the clauses:

- $\mathsf{nf}_1(\mathcal{E}) = \mathcal{E}$, $\mathsf{nf}_2(\mathcal{E}) = I$ if $\mathcal{E} = I$ or $\mathcal{E}$ is an object name;
- $\mathsf{nf}_1(\mathcal{E} \otimes \mathcal{E}') = \mathsf{nf}_1(\mathcal{E}) \otimes \mathsf{nf}_1(\mathcal{E}')$, $\mathsf{nf}_2(\mathcal{E} \otimes \mathcal{E}') = \mathsf{nf}_2(\mathcal{E}) \otimes \mathsf{nf}_2(\mathcal{E}')$;
- $\mathsf{nf}_1(\mathcal{E} \# \mathcal{E}') = \mathsf{nf}_1(\mathcal{E})$, $\mathsf{nf}_2(\mathcal{E} \# \mathcal{E}') = \mathsf{nf}_2(\mathcal{E}) \otimes (\mathsf{nf}_1(\mathcal{E}') \otimes \mathsf{nf}_2(\mathcal{E}'))$.

Let $\mathsf{nf}(\mathcal{E}) = \mathsf{nf}_1(\mathcal{E}) \# \mathsf{nf}_2(\mathcal{E})$, so $\mathsf{nf}(\mathcal{E})$ is normal. For any object expression $\mathcal{E}$ we also define a *normalization morphism expression* $\mathsf{nm}(\mathcal{E}) \colon \mathcal{E} \to \mathsf{nf}(\mathcal{E})$, by induction as follows:

- $\mathsf{nm}(\mathcal{E}) = \eta$ if $\mathcal{E} = I$ or $\mathcal{E}$ is an object name;
- $\mathsf{nm}(\mathcal{E} \otimes \mathcal{E}') = \chi \circ (\mathsf{nm}(\mathcal{E}) \otimes \mathsf{nm}(\mathcal{E}'))$;
- $\mathsf{nm}(\mathcal{E} \# \mathcal{E}') = \xi \circ \zeta \circ (\mathsf{nm}(\mathcal{E}) \# \mathsf{nm}(\mathcal{E}'))$.

The statement of the theorem will follow from the following subgoals.

(1) If a morphism expression $f \colon \mathcal{E} \to \mathcal{E}'$ does not contain υ then $\mathsf{nm}(\mathcal{E}') \circ f \equiv \mathsf{nm}(\mathcal{E}) \circ g$ for some isomorphism expression g.

(2) If a morphism expression $f \colon \mathcal{E} \to \mathcal{E}'$ does not contain η, ξ, ζ and χ, then there exists a simple morphism expression $g \colon \mathsf{nf}(\mathcal{E}) \to \mathsf{nf}(\mathcal{E}')$, such that $\mathsf{nm}(\mathcal{E}') \circ f \equiv g \circ \mathsf{nm}(\mathcal{E})$.

(3) For any two simple morphism expressions $f \colon \mathcal{E} \to \mathcal{E}'$ and $g \colon \mathcal{E} \to \mathcal{E}'$, $f \equiv g$.

(4) For every normal object expression $\mathcal{E}$, $\mathsf{nm}(\mathcal{E}) \colon \mathcal{E} \to \mathsf{nf}(\mathcal{E})$ is an isomorphism expression.

Indeed, given $f, g \colon \mathcal{E} \to \mathcal{E}'$ with normal $\mathcal{E}'$, to prove $f \equiv g$, it suffices to prove that f is equal to $\mathcal{E} \xrightarrow{\mathsf{nm}(\mathcal{E})} \mathsf{nf}(\mathcal{E}) \xrightarrow{f'} \mathcal{E}'$ for some simple f' – the analogous statement would be true for g, and we would be done by (3). In order to construct f', let us represent f as a composition $f_n \circ \ldots \circ f_1$ where every f_i with even i contains precisely one occurrence of υ and every f_i with odd i contains no occurrences of υ. We obtain

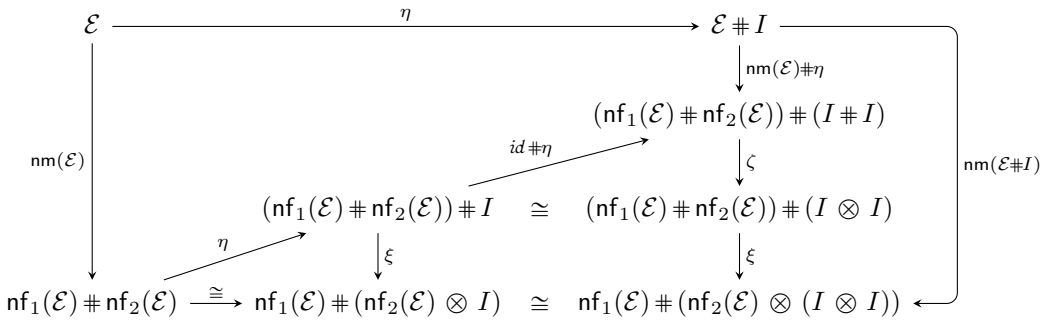
$$\begin{array}{ccccccc}
\mathcal{E} & \xrightarrow{f_1} & \mathcal{E}_1 & \xrightarrow{f_2} & \mathcal{E}_2 & \cdots & \mathcal{E}_n \\
{\scriptstyle \mathsf{nm}(\mathcal{E})}\downarrow & & {\scriptstyle \mathsf{nm}(\mathcal{E}_1)}\downarrow & & {\scriptstyle \mathsf{nm}(\mathcal{E}_2)}\downarrow & & \downarrow{\scriptstyle \mathsf{nm}(\mathcal{E}_n)} \\
\mathsf{nf}(\mathcal{E}) & \xrightarrow{\cong} & \mathsf{nf}(\mathcal{E}_1) & \xrightarrow{f_2'} & \mathsf{nf}(\mathcal{E}_2) & \cdots & \mathsf{nf}(\mathcal{E}_n)
\end{array}$$

where $\mathcal{E} = \mathcal{E}_n$, every odd diagram commutes by (1) and every even diagram commutes by (2). Note that $\mathsf{nm}(\mathcal{E}_n)$ is an isomorphism expression by (1), and therefore we obtain the desired presentation for f, by composing the left vertical arrow, the bottom horizontal sequence of arrows and the inverse of the right vertical arrow. It remains to show the subgoals (1)–(4).

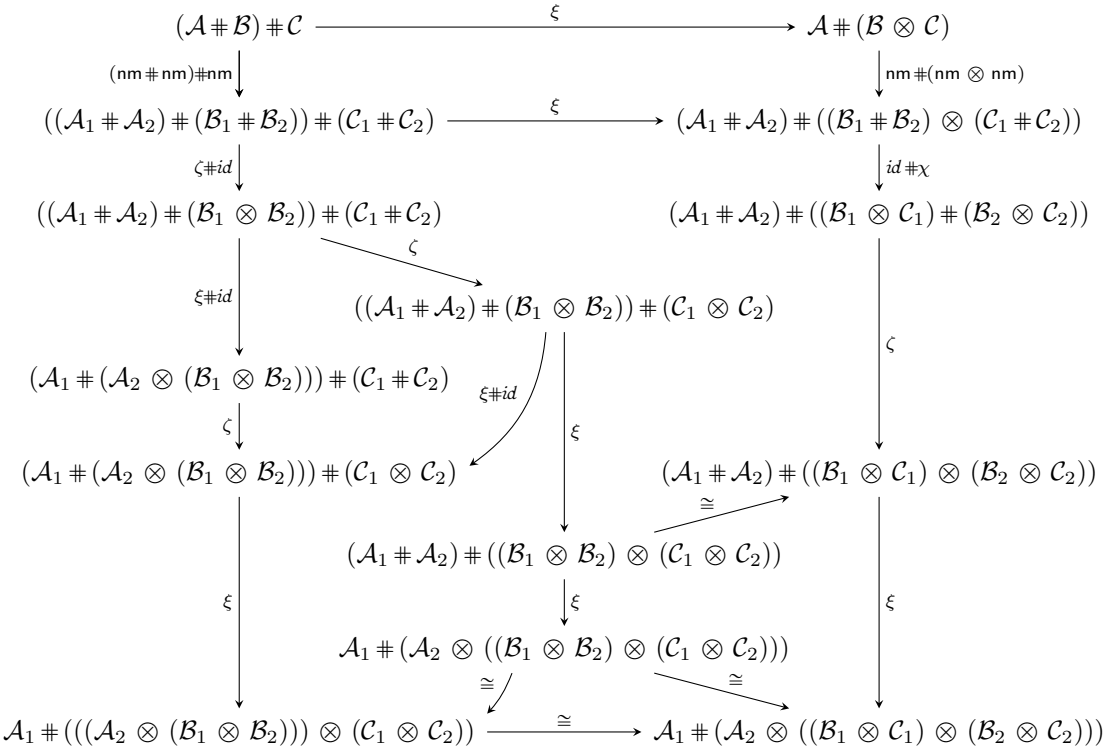
(1) We strengthen the claim by demanding the requisite isomorphism g to be of the form $g_1 \# g_2$ and proceed by structural induction on f.

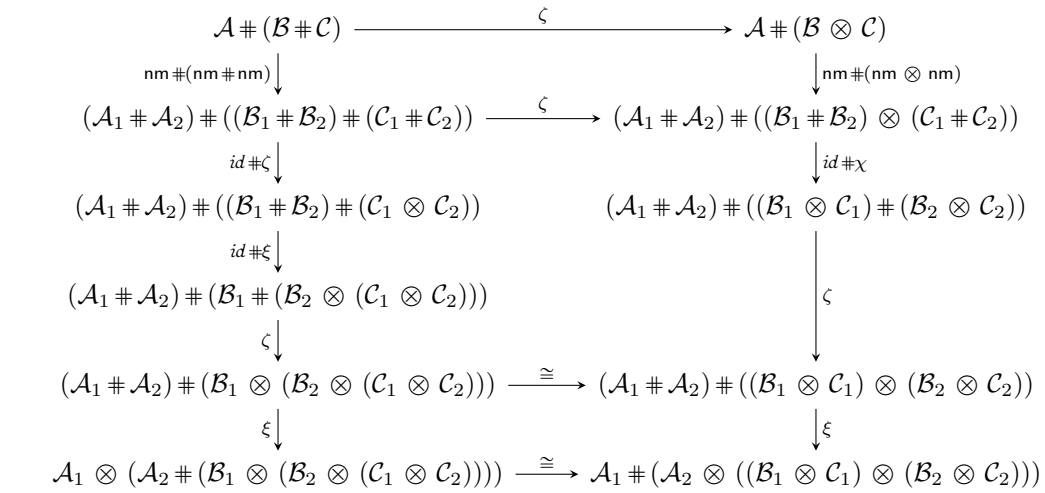
Induction Base: $f \in \{\mathit{id}, \eta, \xi, \zeta, \chi, \rho, \rho^{-1}, \alpha, \alpha^{-1}, \gamma\}$. If $f = \mathit{id}$, we are done trivially by taking $g = f$. Consider $f = \eta \colon \mathcal{E} \to \mathcal{E} \# I$. Then the following diagram commutes, and we

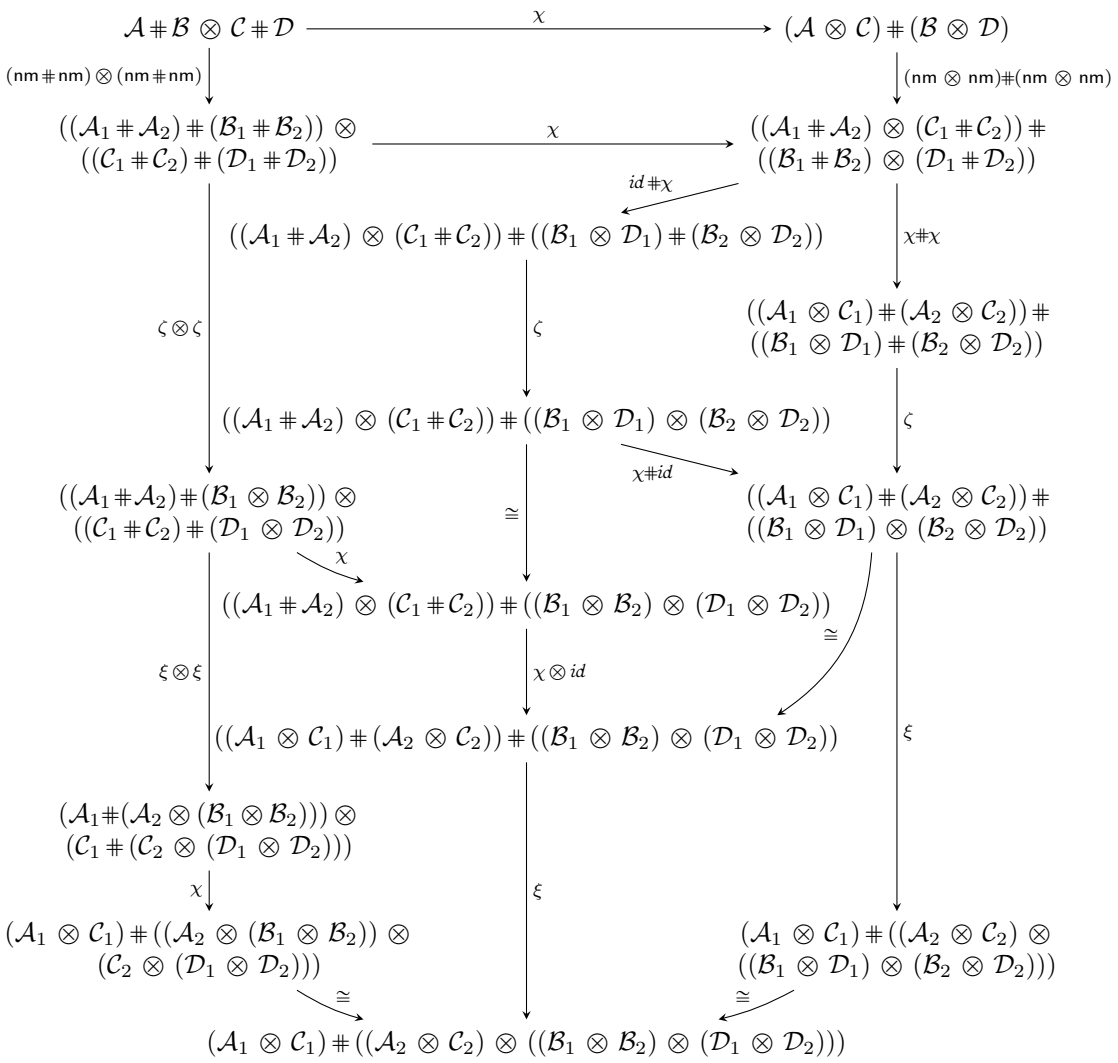
obtain the requisite isomorphism g as the bottom horizontal morphism:



The remaining cases are handled by producing analogous commutative diagrams, which are given below. We do not treat $f = \rho^{-1}$ and $f = \alpha^{-1}$, as these cases are obtained from the corresponding cases $f = \rho$ and $f = \alpha$ by flipping the corresponding diagrams. To save space and maintain readability, we write $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}$ for object expressions, $\mathcal{A}_1, \mathcal{B}_1, \mathcal{C}_1, \mathcal{D}_1$ for $\mathsf{nf}_1(\mathcal{A}), \mathsf{nf}_1(\mathcal{B}), \mathsf{nf}_1(\mathcal{C}), \mathsf{nf}_1(\mathcal{D})$ and $\mathcal{A}_2, \mathcal{B}_2, \mathcal{C}_2, \mathcal{D}_2$ for $\mathsf{nf}_2(\mathcal{A}), \mathsf{nf}_2(\mathcal{B}), \mathsf{nf}_2(\mathcal{C}), \mathsf{nf}_2(\mathcal{D})$ respectively.




$\mathcal{A} \# (\mathcal{B} \# \mathcal{C})$
ζ
$\mathcal{A} \# (\mathcal{B} \otimes \mathcal{C})$
$\mathsf{nm} \# (\mathsf{nm} \# \mathsf{nm})$
$\mathsf{nm} \# (\mathsf{nm} \otimes \mathsf{nm})$
$(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \# \mathcal{B}_2) \# (\mathcal{C}_1 \# \mathcal{C}_2))$
ζ
$(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \# \mathcal{B}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2))$
$id \# \zeta$
$id \# \chi$
$(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \# \mathcal{B}_2) \# (\mathcal{C}_1 \otimes \mathcal{C}_2))$
$(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \otimes \mathcal{C}_1) \# (\mathcal{B}_2 \otimes \mathcal{C}_2))$
$id \# \xi$
ζ
$(\mathcal{A}_1 \# \mathcal{A}_2) \# (\mathcal{B}_1 \# (\mathcal{B}_2 \otimes (\mathcal{C}_1 \otimes \mathcal{C}_2)))$
ζ
$(\mathcal{A}_1 \# \mathcal{A}_2) \# (\mathcal{B}_1 \otimes (\mathcal{B}_2 \otimes (\mathcal{C}_1 \otimes \mathcal{C}_2)))$
$\cong$
$(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \otimes \mathcal{C}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{C}_2))$
ξ
ξ
$\mathcal{A}_1 \otimes (\mathcal{A}_2 \# (\mathcal{B}_1 \otimes (\mathcal{B}_2 \otimes (\mathcal{C}_1 \otimes \mathcal{C}_2))))$
$\cong$
$\mathcal{A}_1 \# (\mathcal{A}_2 \otimes ((\mathcal{B}_1 \otimes \mathcal{C}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{C}_2)))$


$\mathcal{A} \# \mathcal{B} \otimes \mathcal{C} \# \mathcal{D}$
χ
$(\mathcal{A} \otimes \mathcal{C}) \# (\mathcal{B} \otimes \mathcal{D})$
$(\mathsf{nm} \# \mathsf{nm}) \otimes (\mathsf{nm} \# \mathsf{nm})$
$(\mathsf{nm} \otimes \mathsf{nm}) \# (\mathsf{nm} \otimes \mathsf{nm})$
$((\mathcal{A}_1 \# \mathcal{A}_2) \# (\mathcal{B}_1 \# \mathcal{B}_2)) \otimes ((\mathcal{C}_1 \# \mathcal{C}_2) \# (\mathcal{D}_1 \# \mathcal{D}_2))$
χ
$((\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2)) \# ((\mathcal{B}_1 \# \mathcal{B}_2) \otimes (\mathcal{D}_1 \# \mathcal{D}_2))$
$id \# \chi$
$((\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2)) \# ((\mathcal{B}_1 \otimes \mathcal{D}_1) \# (\mathcal{B}_2 \otimes \mathcal{D}_2))$
$\chi \# \chi$
$\zeta \otimes \zeta$
ζ
$((\mathcal{A}_1 \otimes \mathcal{C}_1) \# (\mathcal{A}_2 \otimes \mathcal{C}_2)) \# ((\mathcal{B}_1 \otimes \mathcal{D}_1) \# (\mathcal{B}_2 \otimes \mathcal{D}_2))$
$((\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2)) \# ((\mathcal{B}_1 \otimes \mathcal{D}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{D}_2))$
ζ
$\chi \# id$
$((\mathcal{A}_1 \# \mathcal{A}_2) \# (\mathcal{B}_1 \otimes \mathcal{B}_2)) \otimes ((\mathcal{C}_1 \# \mathcal{C}_2) \# (\mathcal{D}_1 \otimes \mathcal{D}_2))$
$\cong$
$((\mathcal{A}_1 \otimes \mathcal{C}_1) \# (\mathcal{A}_2 \otimes \mathcal{C}_2)) \# ((\mathcal{B}_1 \otimes \mathcal{D}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{D}_2))$
χ
$((\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2)) \# ((\mathcal{B}_1 \otimes \mathcal{B}_2) \otimes (\mathcal{D}_1 \otimes \mathcal{D}_2))$
$\cong$
$\xi \otimes \xi$
$\chi \otimes id$
$((\mathcal{A}_1 \otimes \mathcal{C}_1) \# (\mathcal{A}_2 \otimes \mathcal{C}_2)) \# ((\mathcal{B}_1 \otimes \mathcal{B}_2) \otimes (\mathcal{D}_1 \otimes \mathcal{D}_2))$
ξ
$(\mathcal{A}_1 \# (\mathcal{A}_2 \otimes (\mathcal{B}_1 \otimes \mathcal{B}_2))) \otimes (\mathcal{C}_1 \# (\mathcal{C}_2 \otimes (\mathcal{D}_1 \otimes \mathcal{D}_2)))$
χ
ξ
$(\mathcal{A}_1 \otimes \mathcal{C}_1) \# ((\mathcal{A}_2 \otimes (\mathcal{B}_1 \otimes \mathcal{B}_2)) \otimes (\mathcal{C}_2 \otimes (\mathcal{D}_1 \otimes \mathcal{D}_2)))$
$(\mathcal{A}_1 \otimes \mathcal{C}_1) \# ((\mathcal{A}_2 \otimes \mathcal{C}_2) \otimes ((\mathcal{B}_1 \otimes \mathcal{D}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{D}_2)))$
$\cong$
$\cong$
$(\mathcal{A}_1 \otimes \mathcal{C}_1) \# ((\mathcal{A}_2 \otimes \mathcal{C}_2) \otimes ((\mathcal{B}_1 \otimes \mathcal{B}_2) \otimes (\mathcal{D}_1 \otimes \mathcal{D}_2)))$

$$\begin{array}{ccc}
\mathcal{A} \otimes I & \xrightarrow{\rho} & \mathcal{A} \\
{\scriptstyle \mathsf{nm} \otimes id}\downarrow & & \downarrow{\scriptstyle \mathsf{nm}} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \otimes I & \xrightarrow{\rho} & \mathcal{A}_1 \# \mathcal{A}_2 \\
{\scriptstyle id \otimes \eta}\downarrow & & \| \\
(\mathcal{A}_1 \# \mathcal{A}_2) \otimes (I \# I) & & \| \\
{\scriptstyle \chi}\downarrow & & \| \\
(\mathcal{A}_1 \otimes I) \# (\mathcal{A}_2 \otimes I) & \xrightarrow{\cong} & \mathcal{A}_1 \# \mathcal{A}_2
\end{array}$$

$$\begin{array}{ccc}
\mathcal{A} \otimes (\mathcal{B} \otimes \mathcal{C}) & \xrightarrow{\alpha} & (\mathcal{A} \otimes \mathcal{B}) \otimes \mathcal{C} \\
{\scriptstyle \mathsf{nm} \otimes (\mathsf{nm} \otimes \mathsf{nm})}\downarrow & & \downarrow{\scriptstyle (\mathsf{nm} \otimes \mathsf{nm}) \otimes \mathsf{nm}} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \otimes ((\mathcal{B}_1 \# \mathcal{B}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2)) & \xrightarrow{\alpha} & ((\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{B}_1 \# \mathcal{B}_2)) \otimes (\mathcal{C}_1 \# \mathcal{C}_2) \\
{\scriptstyle id \otimes \chi}\downarrow & & \downarrow{\scriptstyle \chi \otimes id} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \otimes ((\mathcal{B}_1 \otimes \mathcal{C}_1) \# (\mathcal{B}_2 \otimes \mathcal{C}_2)) & & ((\mathcal{A}_1 \otimes \mathcal{B}_1) \# (\mathcal{A}_2 \otimes \mathcal{B}_2)) \otimes (\mathcal{C}_1 \# \mathcal{C}_2) \\
{\scriptstyle \chi}\downarrow & & \downarrow{\scriptstyle \chi} \\
(\mathcal{A}_1 \otimes (\mathcal{B}_1 \otimes \mathcal{C}_1)) \# (\mathcal{A}_2 \otimes (\mathcal{B}_2 \otimes \mathcal{C}_2)) & \xrightarrow{\cong} & ((\mathcal{A}_1 \otimes \mathcal{B}_1) \otimes \mathcal{C}_1) \# ((\mathcal{A}_2 \otimes \mathcal{B}_2) \otimes \mathcal{C}_2)
\end{array}$$

$$\begin{array}{ccc}
\mathcal{A} \otimes \mathcal{B} & \xrightarrow{\gamma} & \mathcal{B} \otimes \mathcal{A} \\
{\scriptstyle \mathsf{nm} \otimes \mathsf{nm}}\downarrow & & \downarrow{\scriptstyle \mathsf{nm} \otimes \mathsf{nm}} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{B}_1 \# \mathcal{B}_2) & \xrightarrow{\gamma} & (\mathcal{B}_1 \# \mathcal{B}_2) \otimes (\mathcal{A}_1 \# \mathcal{A}_2) \\
{\scriptstyle \chi}\downarrow & & \downarrow{\scriptstyle \chi} \\
(\mathcal{A}_1 \otimes \mathcal{B}_1) \otimes (\mathcal{A}_2 \otimes \mathcal{B}_2) & \xrightarrow{\cong} & (\mathcal{B}_1 \otimes \mathcal{A}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{A}_2)
\end{array}$$

Induction Step. Consider $f = f_1 \# f_2 \colon \mathcal{E}_1 \# \mathcal{E}_2 \to \mathcal{E}_1' \# \mathcal{E}_2'$. We obtain the requisite isomorphism from the diagram:

$$\begin{array}{ccc}
\mathcal{E}_1 \# \mathcal{E}_2 & \xrightarrow{f_1 \# f_2} & \mathcal{E}_1' \# \mathcal{E}_2' \\
{\scriptstyle \mathsf{nm}(\mathcal{E}_1) \# \mathsf{nm}(\mathcal{E}_2)}\downarrow & & \downarrow{\scriptstyle \mathsf{nm}(\mathcal{E}_1') \# \mathsf{nm}(\mathcal{E}_2')} \\
(\mathsf{nf}_1(\mathcal{E}_1) \# \mathsf{nf}_2(\mathcal{E}_1)) \# (\mathsf{nf}_1(\mathcal{E}_2) \# \mathsf{nf}_2(\mathcal{E}_2)) & \xrightarrow{(e_1 \# u_1) \# (e_2 \# u_2)} & (\mathsf{nf}_1(\mathcal{E}_1') \# \mathsf{nf}_2(\mathcal{E}_1')) \# (\mathsf{nf}_1(\mathcal{E}_2') \# \mathsf{nf}_2(\mathcal{E}_2')) \\
{\scriptstyle \zeta}\downarrow & & \downarrow{\scriptstyle \zeta} \\
(\mathsf{nf}_1(\mathcal{E}_1) \# \mathsf{nf}_2(\mathcal{E}_1)) \# (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2)) & & (\mathsf{nf}_1(\mathcal{E}_1') \# \mathsf{nf}_2(\mathcal{E}_1')) \# (\mathsf{nf}_1(\mathcal{E}_2') \otimes \mathsf{nf}_2(\mathcal{E}_2')) \\
{\scriptstyle \xi}\downarrow & & \downarrow{\scriptstyle \xi} \\
\mathsf{nf}_1(\mathcal{E}_1) \otimes (\mathsf{nf}_2(\mathcal{E}_1) \# (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2))) & \xrightarrow{\cong} & \mathsf{nf}_1(\mathcal{E}_1') \otimes (\mathsf{nf}_2(\mathcal{E}_1') \# (\mathsf{nf}_1(\mathcal{E}_2') \otimes \mathsf{nf}_2(\mathcal{E}_2')))
\end{array}$$

Here, the top cell commutes by induction hypothesis, with some isomorphisms e_1, e_2, u_1 and u_2, and the bottom cell commutes by naturality of ζ and ξ. The case $f = f_1 \otimes f_2$ is handled analogously.

(2) The given morphism expression f can be decomposed as $f_n \circ \ldots \circ f_1$ in such a way that every f_i contains at most one morphism name (not including id). It thus suffices to

establish the claim for every i. If the involved name is not υ, we are done by the previous clause (1). We thus continue with the proof of (2) w.l.o.g. assuming that f is built from id, υ, $\otimes$ and $\#$, and υ occurs in f precisely once.

Consider $f = \upsilon$. Using the naming conventions from clause (1), we obtain the following commutative diagram:

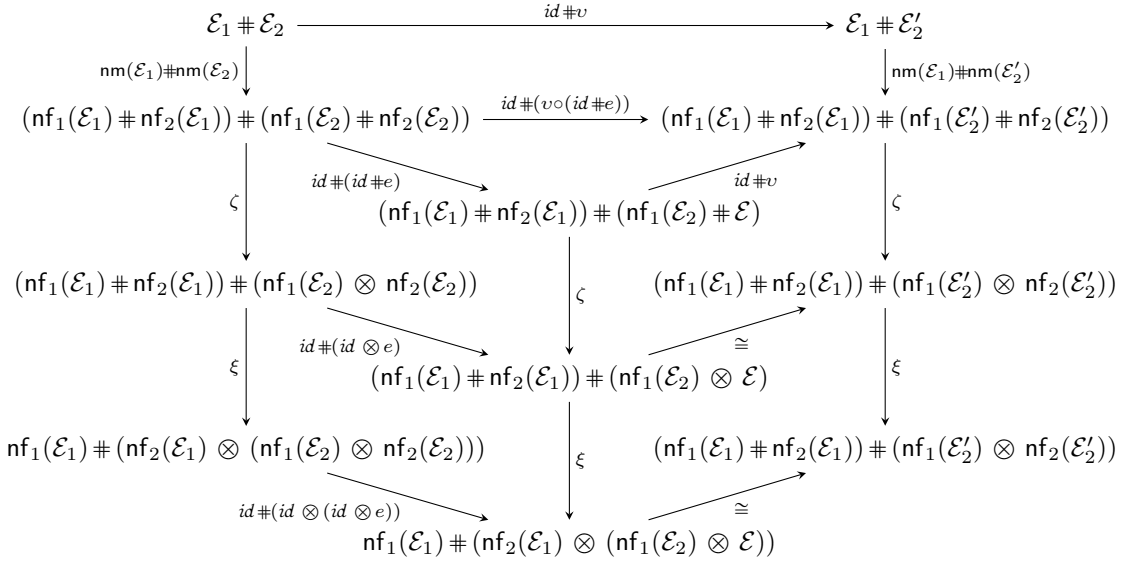
$$\begin{array}{ccc}
\mathcal{A} \# (\mathcal{B} \otimes \mathcal{C}) & \xrightarrow{\upsilon} & (\mathcal{A} \otimes \mathcal{B}) \# \mathcal{C} \\
{\scriptstyle \mathsf{nm} \# (\mathsf{nm} \otimes \mathsf{nm})}\downarrow & & \downarrow{\scriptstyle (\mathsf{nm} \otimes \mathsf{nm}) \# \mathsf{nm}} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \# \mathcal{B}_2) \otimes (\mathcal{C}_1 \# \mathcal{C}_2)) & \xrightarrow{\upsilon} & ((\mathcal{A}_1 \# \mathcal{A}_2) \otimes (\mathcal{B}_1 \# \mathcal{B}_2)) \# (\mathcal{C}_1 \# \mathcal{C}_2) \\
{\scriptstyle \mathit{id} \# \chi}\downarrow & & \downarrow{\scriptstyle \chi \# \mathit{id}} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \otimes \mathcal{C}_1) \# (\mathcal{B}_2 \otimes \mathcal{C}_2)) & & ((\mathcal{A}_1 \otimes \mathcal{B}_1) \# (\mathcal{A}_2 \otimes \mathcal{B}_2)) \# (\mathcal{C}_1 \# \mathcal{C}_2) \\
{\scriptstyle \zeta}\downarrow & & \downarrow{\scriptstyle \zeta} \\
(\mathcal{A}_1 \# \mathcal{A}_2) \# ((\mathcal{B}_1 \otimes \mathcal{C}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{C}_2)) & & ((\mathcal{A}_1 \otimes \mathcal{B}_1) \# (\mathcal{A}_2 \otimes \mathcal{B}_2)) \# (\mathcal{C}_1 \otimes \mathcal{C}_2) \\
{\scriptstyle \xi}\downarrow & & \downarrow{\scriptstyle \xi} \\
\mathcal{A}_1 \# (\mathcal{A}_2 \otimes ((\mathcal{B}_1 \otimes \mathcal{C}_1) \otimes (\mathcal{B}_2 \otimes \mathcal{C}_2))) & \xrightarrow{\upsilon \circ (\mathit{id} \# e)} & (\mathcal{A}_1 \# \mathcal{B}_1) \# ((\mathcal{A}_2 \otimes \mathcal{B}_2) \otimes (\mathcal{C}_1 \otimes \mathcal{C}_1))
\end{array}$$

which yields $g = \upsilon \circ (\mathit{id} \# e)$. Notably, e is an isomorphism. Next, we handle the cases $f = \upsilon \# \mathit{id}$, $f = \mathit{id} \# \upsilon$, $f = \upsilon \otimes \mathit{id}$ and $f = \mathit{id} \otimes \upsilon$. For $f = \upsilon \# \mathit{id}$, we have

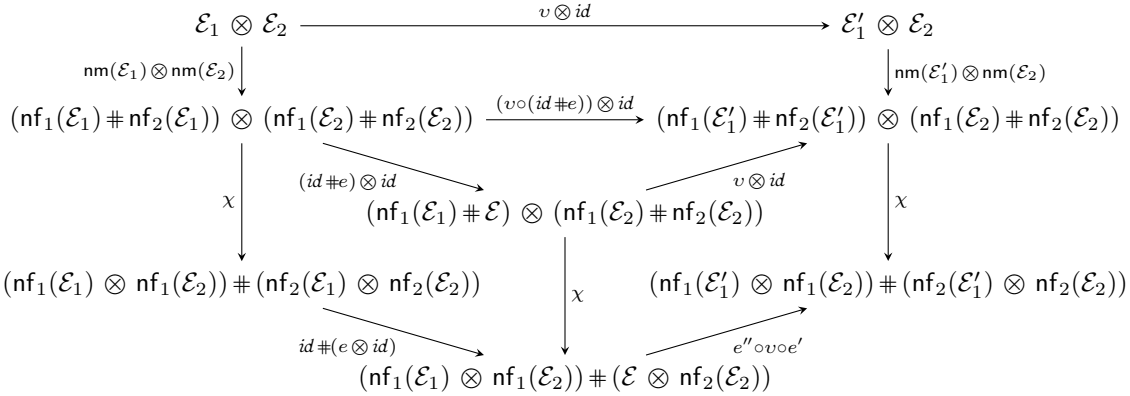
$$\begin{array}{ccccc}
\mathcal{E}_1 \# \mathcal{E}_2 & & \xrightarrow{\upsilon \# \mathit{id}} & & \mathcal{E}_1' \# \mathcal{E}_2 \\
{\scriptstyle \mathsf{nm}(\mathcal{E}_1) \# \mathsf{nm}(\mathcal{E}_2)}\downarrow & & & & \downarrow{\scriptstyle \mathsf{nm}(\mathcal{E}_1') \# \mathsf{nm}(\mathcal{E}_2)} \\
(\mathsf{nf}_1(\mathcal{E}_1) \# \mathsf{nf}_2(\mathcal{E}_1)) \# (\mathsf{nf}_1(\mathcal{E}_2) \# \mathsf{nf}_2(\mathcal{E}_2)) & & \xrightarrow{(\upsilon \circ (\mathit{id} \# e)) \# \mathit{id}} & & (\mathsf{nf}_1(\mathcal{E}_1') \# \mathsf{nf}_2(\mathcal{E}_1')) \# (\mathsf{nf}_1(\mathcal{E}_2) \# \mathsf{nf}_2(\mathcal{E}_2)) \\
{\scriptstyle \zeta}\downarrow & & & & \downarrow{\scriptstyle \zeta} \\
(\mathsf{nf}_1(\mathcal{E}_1) \# \mathsf{nf}_2(\mathcal{E}_1)) \# (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2)) & & \xrightarrow{(\upsilon \circ (\mathit{id} \# e)) \# \mathit{id}} & & (\mathsf{nf}_1(\mathcal{E}_1') \# \mathsf{nf}_2(\mathcal{E}_1')) \# (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2)) \\
{\scriptstyle \xi}\downarrow & \searrow^{(\mathit{id} \# e) \# \mathit{id}} & & \nearrow_{\upsilon \# \mathit{id}} & \downarrow{\scriptstyle \xi} \\
\mathsf{nf}_1(\mathcal{E}_1) \# (\mathsf{nf}_2(\mathcal{E}_1) \otimes (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2))) & & (\mathsf{nf}_1(\mathcal{E}_1) \# \mathcal{E}) \# (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2)) & & \mathsf{nf}_1(\mathcal{E}_1') \# (\mathsf{nf}_2(\mathcal{E}_1') \otimes (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2))) \\
 & \searrow_{\mathit{id} \# (e \otimes \mathit{id})} & \downarrow{\scriptstyle \xi} & \nearrow_{\upsilon \circ e'} & \\
 & & \mathsf{nf}_1(\mathcal{E}_1) \# (\mathcal{E} \otimes (\mathsf{nf}_1(\mathcal{E}_2) \otimes \mathsf{nf}_2(\mathcal{E}_2))) & &
\end{array}$$

where e and e' are isomorphisms, and e is the one that we inherit from the case $f = \upsilon$.

For $f = id \# \upsilon$, we have construct the requisite g analogously:



For $f = \upsilon \otimes id$, we have

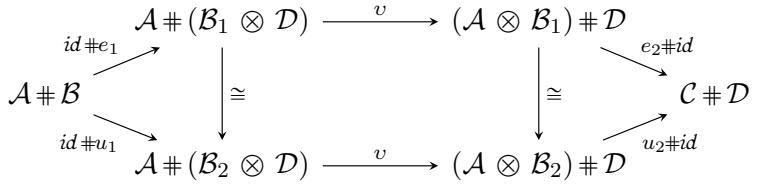


where, again, e is the one that we inherit from the case $f = \upsilon$, and e' and e'' are isomorphisms. The argument for $f = id \otimes \upsilon$ is symmetric.

The general case now follows by structural induction on f with the above cases serving as the induction base.

(3) Let us first argue that w.l.o.g., f and g are of the form $(e_2 \# id) \circ \upsilon \circ (id \# e_1)$ and $(u_2 \# id) \circ \upsilon \circ (id \# u_1)$ with some isomorphism expressions e_1, e_2, u_1, u_2. Indeed, by assumption, f is a composition of morphism expressions of the form: υ, $id \# e$ and $e \# id$ where e ranges over isomorphism expressions. We ensure that υ occurs at least once by using the fact that $id_{\mathcal{A} \# \mathcal{B}} \equiv (\rho_{\mathcal{A}} \# id_{\mathcal{B}}) \circ \upsilon_{\mathcal{A},I,\mathcal{B}} \circ (id_{\mathcal{A}} \# \gamma_{\mathcal{B},I}) \circ (id_{\mathcal{A}} \# \rho_{\mathcal{B}}^{-1})$. Since $(id \# e) \circ \upsilon \equiv \upsilon \circ (id \# (id \otimes e))$ and $\upsilon \circ (e \# id) \equiv ((id \otimes e) \# id) \circ \upsilon$, we can rearrange f equivalently in such a way that all components of the form $id \# e$ are gathered on the right of the expression and all components of the form $e \# id$ are gathered on the left of the expression. Using the axioms of guarded parameterized monads, we can subsequently replace compositions $\upsilon \circ \upsilon$ with a single υ, and thus arrive at the requisite form. The same reasoning applies to g.

Now the equality $f \equiv g$ follows from the diagram:



using the fact that $\mathcal{B}_1$ and $\mathcal{B}_2$ are necessarily isomorphic. That the triangles commute follows from the original Mac Lane's coherence theorem for symmetric monoidal categories [ML71].

(4) Since $\mathcal{E}$ is assumed to be normal, $\mathcal{E} = \mathcal{E}_1 \# \mathcal{E}_2$, where $\mathcal{E}_1$ and $\mathcal{E}_2$ do not contain $\#$. Let us show first, by induction over $\mathcal{E}_1$, that $\mathsf{nm}(\mathcal{E}_1) \equiv (id \# e_1) \circ \eta$ for some isomorphism e_1 between I and a tensor product of some number of copies of I. The induction base is trivial. For the induction step, let $\mathcal{E}_1 = \mathcal{E}_1' \otimes \mathcal{E}_1''$. Then, using the induction hypothesis, $\mathsf{nm}(\mathcal{E}_1) = \chi \circ (\mathsf{nm}(\mathcal{E}_1') \otimes \mathsf{nm}(\mathcal{E}_1'')) \equiv \chi \circ ((id \# e_1') \circ \eta \otimes (id \# e_1'') \circ \eta) \equiv (id \# (e_1' \otimes e_1'')) \circ \chi \circ (\eta \otimes \eta) \equiv (id \# (e_1' \otimes e_1'') \circ e) \circ \eta$ where $e \colon I \cong I \otimes I$. Analogously, we obtain $\mathsf{nm}(\mathcal{E}_2) \equiv (id \# e_2) \circ \eta$.

Now, $\mathsf{nm}(\mathcal{E}) = \mathsf{nm}(\mathcal{E}_1 \# \mathcal{E}_1) = \xi \circ \zeta \circ (\mathsf{nm}(\mathcal{E}_1) \# \mathsf{nm}(\mathcal{E}_2)) \equiv \xi \circ \zeta \circ ((id \# e_1) \circ \eta \# (id \# e_2) \circ \eta) \equiv (id \otimes (e_1 \otimes (id \otimes e_2))) \circ \xi \circ \zeta \circ (\eta \# \eta)$. Using the axioms of guarded parameterized monads, observe that $\xi \circ \zeta \circ (\eta \# \eta)$ is an isomorphism expression, which finishes the proof. □

It is an open question if a stronger version of the above coherence theorem with general $f, g \colon \mathcal{E}_1 \to \mathcal{E}_2$ can be proven. In the sequel, we will only deal with guarded parameterized monads over $(\mathbf{V}, +, 0)$. Recall that a parameterized monad (in the sense of Uustalu [Uus03]) is a bifunctor $T \colon \mathbf{V} \times \mathbf{V} \to \mathbf{V}$, such that each $T(\text{–}, X)$ is a monad and each $T(\text{–}, f)$ is a monad morphism. Of course, a guarded parameterized monad is meant to be a parameterized monad in this sense. This follows from Remark 7.2 and the following general fact.

Proposition 7.4. *Every* $\mathbf{V}$*-graded monad on* $(\mathbf{V}, +, 0)$ *is a parameterized monad.*

Proof. A $\mathbf{V}$-graded monad on $\mathbf{V}$ is equivalently a lax monoidal functor from $\mathbf{V}$ to the monoidal category of endofunctors $([\mathbf{V}, \mathbf{V}], \circ, \mathrm{Id})$. Let this lax monoidal functor send each $X \in |\mathbf{V}|$ to $T(\text{–}, X) \colon \mathbf{V} \to \mathbf{V}$. Lax monoidal functors preserve monoids, which unravels as follows. Every object X in $\mathbf{V}$ is a monoid under $! \colon 0 \to X$ and $\nabla \colon X + X \to X$, and monoids in $([\mathbf{V}, \mathbf{V}], \circ, \mathrm{Id})$ are precisely monads. Therefore, $T(\text{–}, X)$ is a monad for every X. Likewise, every morphism $f \colon X \to Y$ in $\mathbf{V}$ is a monoid morphism, and hence induces a monoid morphism from $T(\text{–}, X)$ to $T(\text{–}, Y)$, i.e. a monad morphism. □

Explicitly, for a guarded parameterized monad $\#$ we obtain parameterized multiplication transformation:

$$\mu_{X,Y} = \big((X \# Y) \# Y \xrightarrow{\xi} X \# (Y + Y) \xrightarrow{id \# \nabla} X \# Y\big)$$

Theorem 7.5. *Given co-Cartesian* $\mathbf{V}$ *and an identity-on-object functor* $J \colon \mathbf{V} \to \mathbf{C}$ *strictly preserving coproducts,* $\mathbf{C}$ *is guarded and* $\mathbf{C}_\bullet$ *is representable iff* $\mathbf{C} \cong \mathbf{V}_{\text{–}\#0}$ *for a guarded parameterized monad* $(\#, \eta, \upsilon, \chi, \xi, \zeta)$*, the compositions* $\upsilon_{X,Y,0} \circ (id \# inl)$ *are all monic and* $f \colon X \to Y \rangle Z$ *iff* f *factors through* $Y \# (Z + 0) \xrightarrow{\upsilon} (Y + Z) \# 0$*.*

Proof. ($\Rightarrow$) Let $\mathbf{C}_\bullet$ be J-representable. By Theorem 6.6, assume w.l.o.g. that $\mathbf{C} = \mathbf{V}_{\mathbf{T}}$ where $T = \text{–}\#0$, and let ϵ and $(\text{–})^\sharp$ be the corresponding structure, belonging to $\#$. Let η be the unit of $\mathbf{T}$. We obtain the remaining transformations υ, ξ, ζ and χ by universality as follows:

$$\upsilon = \big(X \# (Y + Z) \xrightarrow{\epsilon} (X + (Y + Z)) \# 0 \cong ((X + Y) + Z)) \# 0\big)^\sharp,$$

$$\xi = \big((X \# Y) \# Z \xrightarrow{\epsilon} (X \# Y + Z) \# 0 \xrightarrow{[T(id + inl) \circ \epsilon, \eta \circ inr \circ inr]^\star} (X + (Y + Z)) \# 0\big)^\sharp,$$
$$\zeta = \big(X \# (Y \# Z) \xrightarrow{\epsilon} (X + Y \# Z) \# 0 \xrightarrow{[\eta \circ inl, T\, inr \circ \epsilon]^\star} (X + (Y + Z)) \# 0\big)^\sharp,$$
$$\chi = \big(X \# Y + Z \# V \xrightarrow{\epsilon + \epsilon} (X + Y) \# 0 + (Z + V) \# 0 \xrightarrow{T[inl + inl, inr + inr]} ((X + Y) + (Z + V)) \# 0\big)^\sharp.$$

It is clear by definition that $f^\sharp$ is mono as long as f is mono, hence υ is mono. The characterization of the guardedness predicate follows from Theorem 6.6. The laws of guarded parameterized monad all follow by postcomposition with ϵ and using the fact that it is mono.

($\Leftarrow$) Let $\mathbf{C} = \mathbf{V}_{- \# 0}$ for a guarded parameterized monad $(\#, \eta, \upsilon, \chi, \xi, \zeta)$. We define $\epsilon \colon X \# Y \to (X + Y) \# 0$ as $X \# Y \cong X \# (Y + 0) \xrightarrow{\upsilon} (X + Y) \# 0$, which is monic, since υ is so by assumption. This yields a unique $f^\sharp$ for every guarded f, by definition of the guardedness predicate. The only non-trivial condition of Theorem 6.6, which is left to verify, is that the guardedness predicate is well-defined.

- **(trv$_\bullet$)** Given $f \colon X \to Y$, $\eta \circ inl \circ f = \epsilon \circ (id \# !) \circ \eta \circ f$ is thus guarded.
- **(par$_\bullet$)** Given $f \colon X \to V \# W$ and $g \colon Y \to V \# W$, $[\epsilon \circ f, \epsilon \circ g] = \epsilon \circ [f, g]$ is guarded.
- **(cmp$_\bullet$)** Given $f \colon X \to Y \# Z$, $g \colon Y \to V \# W$ and $h \colon Z \to (V + W) \# 0$, observe first that

$$\begin{aligned} [\epsilon \circ g, h]^\star \circ \epsilon \circ f &= \mu \circ ([\epsilon \circ g, h] \# id) \circ \epsilon \circ f \\ &= (id \# \nabla) \circ \xi \circ ([\epsilon \circ g, h] \# id) \circ \epsilon \circ f \\ &= (id \# \nabla) \circ \xi \circ ([\epsilon, id] \# id) \circ \epsilon \circ (g \# h) \circ f. \end{aligned}$$

That is, we are left to show that $(id \# \nabla) \circ \xi \circ ([\epsilon, id] \# id) \circ \epsilon$ factors through ϵ. Observe that $(\nabla \# \nabla) \circ \chi = \nabla$. Therefore,

$$\begin{aligned} &(id \# \nabla) \circ \xi \circ (\nabla \# id) \circ ((\epsilon + id) \# id) \circ \epsilon \\ &\quad = (id \# \nabla) \circ \xi \circ ((\nabla \# \nabla) \circ \chi \# id) \circ ((\epsilon + id) \# id) \circ \epsilon \\ &\quad = (id \# \nabla) \circ (\nabla \# (\nabla + id)) \circ \xi \circ (\chi \# id) \circ ((\epsilon + id) \# id) \circ \epsilon \\ &\quad = (\nabla \# \nabla \circ (\nabla + id)) \circ \xi \circ (\chi \# id) \circ ((\epsilon + id) \# id) \circ \epsilon \\ &\quad = ((\nabla + \nabla) \# \nabla \circ (\nabla + id)) \circ ([inl + inl, inr + inr] \# id) \circ \xi \circ (\chi \circ (\epsilon + id) \# id) \circ \epsilon. \end{aligned}$$

Using coherence, $([inl + inl, inr + inr] \# id) \circ \xi \circ (\chi \circ (\epsilon + id) \# id) \circ \epsilon \colon (V \# W) \# ((V + W) \# 0) \to ((V + V) + (W + W)) \# ((0 + 0) + 0)$ can be factored through υ, and hence the entire expression factors through ϵ. □

Vacuous guardedness is clearly representable and by Theorem 7.5 corresponds to those guarded parameterized monads $\#$, which do not depend on the parameter, i.e. to monads.

Example 7.6. Let us revisit Example 5.4. Let $X \# Y = T(X + HT_H(X + Y))$, and note that $- \# 0$ is isomorphic to T_H. Assuming the existence of some morphism $p \colon 1 \to H1$, for every X, we obtain the final map $\hat{p} \colon 1 \to T_H X$, induced by the coalgebra map $1 \xrightarrow{\eta \circ inr \circ p} T(X + H1)$. Now, $T(inl + id)$ is a section, since $T[id + H\hat{p} \circ p \circ !, inr] \circ T(inl + id)$ is the identity. By Theorem 6.6, $\#$ is a guarded parameterized monad.

Example 7.7. Let us revisit Example 5.6. Let $X \# Y = \mathbb{R}_{\geqslant 0} \times X + \mathbb{R}_{>0} \times Y + \bar{\mathbb{R}}_{\geqslant 0}$. Then $X \# 0 \cong \mathbb{R}_{\geqslant 0} \times X + \bar{\mathbb{R}}_{\geqslant 0}$ and there is an obvious injection $\epsilon_{X,Y}$ from $X \# Y$ to $(X + Y) \# 0$. By definition, every guarded $f \colon X \to Y \# Z$ uniquely factors through $\epsilon_{Y,Z}$, and hence $\#$ is a guarded parameterized monad.

Definition 7.8 (Strong Guarded Parameterized Monad)**.** A guarded parameterized monad $(\#, \eta, \upsilon, \chi, \xi, \zeta)$ is strong, if $\#$ is strong as a monad in the first argument and as a functor in the second argument, and the diagram

$$\begin{array}{ccccccc} X \times (Y \# Z) & \xrightarrow{id \times \epsilon} & X \times (Y + Z) \# 0 & \xrightarrow{\tau} & (X \times (Y + Z)) \# 0 & \xrightarrow{dist \# 0} & (X \times Y + X \times Z) \# 0 \\ \downarrow{\scriptstyle \tau} & & & & & & \downarrow{\scriptstyle (id + snd) \# 0} \\ (X \times Y) \# Z & & \xrightarrow{\qquad\qquad\qquad \epsilon \qquad\qquad\qquad} & & & & (X \times Y + Z) \# 0 \end{array}$$

commutes, where $\epsilon_{X,Y} = \upsilon_{X,Y,0} \circ (id \# inl)$ and τ is the monadic strength of $\#$.

Remark 7.9. Strength is commonly referred to as a "technical condition". This is justified by the fact that in self-enriched categories, strength is equivalent to enrichment of the corresponding functor or a monad [Koc72], and in foundational categories, like **Set**, every functor and every natural transformation are canonically enriched w.r.t. Cartesian closeness as the self-enrichment structure. Then the canonical strength $\rho_{X,Y} \colon X \times FY \to F(X \times Y)$ for a functor F is defined by the expression $\rho_{X,Y} = \lambda(x, z).\, F(\lambda y.\, (x, y))(z)$. We conjecture that the strengths involved in Definition 7.8 are technical in the same sense, in particular, the requested commutative diagram is entailed by enrichment of ϵ.

Finally, let us establish the analogue of Theorem 7.5 for Freyd categories.

Theorem 7.10. *A Freyd category $(\mathbf{V}, \mathbf{C}, J(\text{–}), \oslash)$ is guarded and $\mathbf{C}_\bullet$ is representable iff $\mathbf{C} \cong \mathbf{V}_{\text{–}\#0}$ for a strong guarded parameterized monad $(\#, \eta, \upsilon, \chi, \xi, \zeta)$, the compositions $\upsilon_{X,Y,0} \circ (id \# inl)$ are all monic and $f \colon X \to Y \rangle Z$ iff f factors through $Y \# (Z + 0) \xrightarrow{\upsilon} (Y + Z) \# 0$.*

Proof. Theorem 7.5 and Proposition 4.5 jointly imply that $\mathbf{C}$ is a representable guarded Freyd category iff

- $\mathbf{C} \cong \mathbf{V}_{\text{–}\#0}$ for a guarded parameterized monad $(\#, \eta, \upsilon, \chi, \xi, \zeta)$,
- υ is componentwise monic,
- $f \colon X \to Y \rangle Z$ iff f factors through $Y \# (Z + 0) \xrightarrow{\upsilon} (Y + Z) \# 0$,
- $T = (\text{–}) \# 0$ is strong and $(T\, dist) \circ \tau \circ (id \times \epsilon) \colon X \times (Y \# Z) \to T(X \times Y + X \times Z)$ uniquely factors through $\epsilon \colon X \times Y \# X \times Z \to T(X \times Y + X \times Z)$ where τ is the strength of $\mathbf{T}$.

This yield strength for both sides of $\#$ by composition:

$$X \times (Y \# Z) \to (X \times Y) \# (X \times Z) \xrightarrow{id \# snd} (X \times Y) \# Z,$$
$$X \times (Y \# Z) \to (X \times Y) \# (X \times Z) \xrightarrow{snd \# id} Y \# (X \times Z).$$

The diagram in Definition 7.8 is thus satisfied by definition. The axioms of strength are checked routinely. □

For a strong guarded parameterized monad $\#$, let $\tilde{\tau}$ be the composition

$$\begin{aligned} X \times (Y \# Z) &\xrightarrow{\Delta \times id} (X \times X) \times (Y \# Z) \cong X \times (X \times (Y \# Z)) \\ &\xrightarrow{id \times \rho} X \times (Y \# (X \times Z)) \xrightarrow{\tau} (X \times Y) \# (X \times Z) \end{aligned}$$

where τ is the monadic strength of $\#$ and ρ is the functorial strength of $\#$. It is easy to check that τ and ρ are derivable from $\tilde{\tau}$, and in the sequel, we will include it as the last element in a tuple $(\#, \eta, \upsilon, \chi, \xi, \zeta, \tilde{\tau})$, defining a strong guarded parameterized monad.

$$\frac{}{[\![x_1 \colon A_1, \ldots, x_n \colon A_n \vdash_{\mathsf{v}} x_i \colon A_i]\!] = \mathit{proj}_i}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} f(v) \colon B]\!] = [\![f]\!] \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} f(v) \colon B]\!] = [\![f]\!] \circ h}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{return}\, v \colon A \between B]\!] = \eta \circ h}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{c}} p \colon A \between B]\!] \qquad h_1 = [\![\Gamma, x \colon A \vdash_{\mathsf{c}} q \colon C \between D]\!] \qquad h_2 = [\![\Gamma, y \colon B \vdash_{\mathsf{c}} r \colon (C + D) \between 0]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{docase}\, p\, \mathsf{of}\, \mathsf{inl}\, x \mapsto q;\, \mathsf{inr}\, y \mapsto r \colon C \between D]\!] = (\nabla \mathbin{\#} \mathit{id}) \circ \upsilon \circ \mu \circ ((\mathit{id} \mathbin{\#} \mathit{inr}) \mathbin{\#} [\mathit{id}, !]) \circ \zeta \circ (h_1 \mathbin{\#} h_2) \circ \tilde{\tau} \circ \langle \mathit{id}, h \rangle}$$

$$\frac{}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{init}\, v \colon A]\!] = !} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inl}\, v \colon A + B]\!] = \mathit{inl} \circ h} \qquad \frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \mathsf{inr}\, v \colon A + B]\!] = \mathit{inr} \circ h}$$

$$\frac{h = [\![\Gamma \vdash_{\mathsf{v}} v \colon A + B]\!] \qquad h_1 = [\![\Gamma, x \colon A \vdash_{\mathsf{c}} p \colon C \between D]\!] \qquad h_2 = [\![\Gamma, y \colon B \vdash_{\mathsf{c}} q \colon C \between D]\!]}{[\![\Gamma \vdash_{\mathsf{c}} \mathsf{case}\, v\, \mathsf{of}\, \mathsf{inl}\, x \mapsto p;\, \mathsf{inr}\, y \mapsto q \colon C \between D]\!] = (\nabla \mathbin{\#} \nabla) \circ \chi \circ (h_1 + h_2) \circ \mathit{dist} \circ \langle \mathit{id}, h \rangle}$$

$$\frac{h_1 = [\![\Gamma \vdash_{\mathsf{v}} v \colon A]\!] \qquad h_2 = [\![\Gamma \vdash_{\mathsf{v}} w \colon B]\!]}{[\![\Gamma \vdash_{\mathsf{v}} \langle v, w \rangle \colon A \times B]\!] = \langle h_1, h_2 \rangle}$$

Figure 8: Denotational semantics of guarded FGCBV over guarded parameterized monads.

Finally, we can interpret the guarded version of FGCBV over a strong guarded parameterized monad $(\#, \eta, \upsilon, \chi, \xi, \zeta, \tilde{\tau})$ on $\mathbf{V}$. A semantics of $(\Sigma_{\mathsf{v}}, \Sigma_{\mathsf{c}})$ then assigns

- an object $[\![A]\!] \in |\mathbf{V}|$ to each sort A;
- a morphism $[\![f]\!] \in \mathbf{V}([\![A]\!], [\![B]\!])$ to each $f \colon A \to B \in \Sigma_{\mathsf{v}}$;
- a morphism $[\![f]\!] \in \mathbf{V}([\![A]\!], [\![B]\!] \mathbin{\#} [\![C]\!])$ to each $f \colon A \to B \between C \in \Sigma_{\mathsf{c}}$;

This semantics extends to types as before and to terms in context with the assignments in Figure 8. Let us spell out the most sophisticated morphism corresponding to the rule for docase:

$$\begin{aligned}
&[\![\Gamma]\!] \xrightarrow{\langle id, h \rangle} [\![\Gamma]\!] \times ([\![A]\!] \mathbin{\#} [\![B]\!]) \xrightarrow{\tilde{\tau}_{[\![\Gamma]\!],[\![A]\!],[\![B]\!]}} ([\![\Gamma]\!] \times [\![A]\!]) \mathbin{\#} ([\![\Gamma]\!] \times [\![B]\!]) \\
&\xrightarrow{h_1 \# h_2} ([\![C]\!] \mathbin{\#} [\![D]\!]) \mathbin{\#} (([\![C]\!] + [\![D]\!]) \mathbin{\#} 0) \xrightarrow{\zeta_{[\![C]\!]\#[\![D]\!],[\![C]\!]+[\![D]\!],0}} ([\![C]\!] \mathbin{\#} [\![D]\!]) \mathbin{\#} (([\![C]\!] + [\![D]\!]) + 0) \\
&\xrightarrow{(id \# inr) \# [id, !]} ([\![C]\!] \mathbin{\#} ([\![C]\!] + [\![D]\!])) \mathbin{\#} ([\![C]\!] + [\![D]\!]) \xrightarrow{\mu_{[\![C]\!],[\![C]\!]+[\![D]\!]}} [\![C]\!] \mathbin{\#} ([\![C]\!] + [\![D]\!]) \\
&\xrightarrow{\upsilon_{[\![C]\!],[\![C]\!],[\![D]\!]}} ([\![C]\!] + [\![C]\!]) \mathbin{\#} [\![D]\!] \xrightarrow{\nabla \# id} [\![C]\!] \mathbin{\#} [\![D]\!]
\end{aligned}$$

Note that what allows us to sidestep the monicity condition of the representability criterion (Theorem 7.5) is that we gave up on the assumption that the space of guarded morphisms $X \to Y \# Z$ injectively embeds into the space of all morphisms $X \to (Y + Z) \# 0$, in particular, the entire notion of guardedness predicate is eliminated.

8. Conclusions and Further Work

We investigated a combination of FGCBV and guardedness, drawing inspiration from previous work relating Freyd categories to strong monads via a natural representability condition for certain presheaves. An abstract notion of guardedness naturally fits into the FGCBV paradigm and gives rise to more general formats of presheaves, which must be representable, e.g., to interpret higher-order (guarded) functions. In our case, the representability requirement gave rise to a novel categorical structure — we dub it a (strong) guarded parameterized monad — that encapsulates the computational effects under consideration while providing guardedness guarantees.

We regard our present results as a prerequisite step for implementing guarded programs in existing higher-order languages, such as Haskell, and in proof assistants with strict support of the propositions-as-types discipline, such as Coq and Agda, where unproductive recursive definitions cannot be implemented directly, and thus guarded iteration is particularly significant. It would be interesting to further refine guarded parameterized monads to include quantitative information about how productive a computation is, or how unproductive it is, so that this relative unproductivity could possibly be cancelled out by composition with something very productive. Another strand for future work arises from the observation that guarded iteration is a formal dual of guarded recursion [GS18].

A good deal of the present theory can be easily dualized, which will presumably lead to guarded parameterized comonads and comonadic recursion – we are planning to investigate these structures from the perspective of comonadic notion of computation [UV08]. In terms of syntax, a natural extension of fine-gain call-by-value is call-by-push-value [Lev99]. We expect it to be a natural environment for analyzing the above-mentioned aspects in the style of the presented approach.

As we demonstrated, guarded parameterized monads emerge as an answer to a very natural representability question, but the resulting notion, i.e. Definition 7.1, admittedly appears to be rather unwieldy. It involves five natural transformations, two of which (η and ξ) render guarded parameterized monads as graded [FKM16] or parametric monads [Mel17]. Each of the remaining transformations has its specific role in governing guardedness guarantees. They ensure that a guardedness guarantee can be weakened (υ), that independent guardedness guarantees can be merged (χ), and that nested guardedness guarantees can be flattened (ζ). Numerous coherence conditions between these transformations are vital for the coherence theorem, and it seems that not much can be done to simplify them significantly. One seemingly natural idea is to replace the natural transformation $\chi\colon A \# B \otimes C \# D \to (A \otimes C) \# (B \otimes D)$ with a more elementary transformation $\kappa\colon A \otimes B \# C \to (A \otimes B) \# C$ from which χ can be derived. This, however, does not have a straightforward simplifying effect on the coherence conditions, in particular on the condition describing the interaction of χ and associativity transformations. Outside the context of coherence, the only useful example of $\otimes$ known presently is the binary coproduct functor $+$, and the only useful candidate for χ in this case is $[\mathit{inl} \# \mathit{inl}, \mathit{inr} \# \mathit{inr}]$. In this special case Definition 7.1 might be possible to simplify.

Acknowledgement

The author would like to thank anonymous reviewers of the present and previous editions of the paper for their diligence in their effort to improve it.

References

[ACÉI12] Luca Aceto, Arnaud Carayol, Zoltán Ésik, and Anna Ingólfsdóttir. Algebraic synchronization trees and processes. In Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer, editors, *Proc. of 39th Int. Coll on Automata, Languages, and Programming (ICALP 2012), Part 2*, volume 7392 of *LNCS*, pages 30–41. Springer, 2012. doi:10.1007/978-3-642-31585-5_7.

[ADK17] Thorsten Altenkirch, Nils Danielsson, and Nicolai Kraus. Partiality, revisited — the partiality monad as a quotient inductive-inductive type. In Javier Esparza and Andrzej Murawski, editors, *Foundations of Software Science and Computation Structures (FOSSACS 2017)*, volume 10203 of *LNCS*, pages 534–549, 2017. doi:10.1007/978-3-662-54458-7.

[AMV02] Jiří Adámek, Stefan Milius, and Jiří Velebil. On rational monads and free iterative theories. In *Proc. Category Theory and Computer Science (CTCS 2002)*, volume 69 of *Electron. Notes Theor. Comput. Sci.*, pages 23–46, 2002. doi:10.1016/s1571-0661(04)80557-7.

[AMV10] Jiří Adámek, Stefan Milius, and Jiří Velebil. Equational properties of iterative monads. *Inf. Comput.*, 208(12):1306–1348, 2010. doi:10.1016/j.ic.2009.10.006.

[Awo10] Steve Awodey. *Category Theory (Oxford Logic Guides)*. Oxford University Press, USA, 2 edition, 2010. doi:10.1093/acprof:oso/9780198568612.001.0001.

[BE93] Stephen Bloom and Zoltán Ésik. *Iteration theories: The equational logic of iterative processes*. Springer, 1993. doi:10.1007/978-3-642-78034-9.

[BPS01] J. Bergstra, A. Ponse, and Scott Smolka, editors. *Handbook of Process Algebra*. Elsevier, 2001. doi:10.1016/b978-0-444-82830-9.x5017-6.

[BW90] Jos C. M. Baeten and W. P. Weijland. *Process algebra*, volume 18 of *Cambridge tracts in theoretical computer science*. Cambridge University Press, 1990. doi:10.1017/cbo9780511624193.

[Cap05] Venanzio Capretta. General recursion via coinductive types. *Log. Meth. Comput. Sci.*, 1(2), 2005. doi:10.2168/lmcs-1(2:1)2005.

[Coc93] J. Robin B. Cockett. Introduction to distributive categories. *Mathematical Structures in Computer Science*, 3(3):277–307, 1993. doi:10.1017/s0960129500000232.

[CUV17] James Chapman, Tarmo Uustalu, and Niccoló Veltri. Quotienting the delay monad by weak bisimilarity. volume 29, page 67–92. Cambridge University Press (CUP), October 2017. doi:10.1017/s0960129517000184.

[EK17] Martín H. Escardó and Cory M. Knapp. Partial Elements and Recursion via Dominances in Univalent Type Theory. In Valentin Goranko and Mads Dam, editors, *26th EACSL Annual Conference on Computer Science Logic (CSL 2017)*, volume 82 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 21:1–21:16, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CSL.2017.21.

[Fio04] M.P. Fiore. *Axiomatic Domain Theory in Categories of Partial Maps*. Distinguished Dissertations in Computer Science. Cambridge University Press, 2004. doi:10.1017/CBO9780511526565.

[FKM16] Soichiro Fujii, Shin-ya Katsumata, and Paul-André Melliès. Towards a formal theory of graded monads. In Bart Jacobs and Christof Löding, editors, *Proc. 19th International Conference on Foundations of Software Science and Computation Structures (FOSSACS 2016)*, volume 9634 of *Lecture Notes in Computer Science*, pages 513–530. Springer, 2016. doi:10.1007/978-3-662-49630-5_30.

[Fok13] W. Fokkink. *Introduction to Process Algebra*. Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-662-04293-9.

[GNP20] Sergey Goncharov, Renato Neves, and José Proença. Implementing hybrid semantics: From functional to imperative. In Volker Stolz Violet Ka I Pun, Adenilso da Silva Simão, editor, *17th International Colloquium on Theoretical Aspects of Computing (ICTAC 2020)*, 2020. doi:10.1007/978-3-030-64276-1_14.

[Gon21] Sergey Goncharov. Uniform Elgot Iteration in Foundations. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *LIPIcs*, pages 131:1–131:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.131.

[Gon23] Sergey Goncharov. Representing Guardedness in Call-By-Value. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023)*, volume 260 of *Leibniz International Proceedings in Informatics (LIPIcs)*,

pages 34:1–34:21, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSCD.2023.34.

[GRS15] Sergey Goncharov, Christoph Rauch, and Lutz Schröder. Unguarded recursion on coinductive resumptions. In *Proc. Mathematical Foundations of Programming Semantics (MFPS 2015)*, volume 319 of *ENTCS*, pages 183–198. Elsevier, 2015. doi:10.23638/lmcs-14(3:10)2018.

[GRS21] Sergey Goncharov, Christoph Rauch, and Lutz Schröder. A metalanguage for guarded iteration. *Theoretical Computer Science*, 880:111–137, 2021. doi:10.1016/j.tcs.2021.04.005.

[GS18] Sergey Goncharov and Lutz Schröder. Guarded traced categories. In Christel Baier and Ugo Dal Lago, editors, *Proc. 21th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2018)*, LNCS. Springer, 2018. doi:10.1007/978-3-319-89366-2_17.

[GSRP17] Sergey Goncharov, Lutz Schröder, Christoph Rauch, and Maciej Piróg. Unifying guarded and unguarded iteration. In Javier Esparza and Andrzej Murawski, editors, *Proc. 20th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2017)*, volume 10203 of *LNCS*, pages 517–533. Springer, 2017. doi:10.1007/978-3-662-54458-7_30.

[GSRP19] Sergey Goncharov, Lutz Schröder, Christoph Rauch, and Maciej Piróg. Guarded and unguarded iteration for generalized processes. *Logical Methods in Computer Science*, 15(3), 2019. doi:10.23638/LMCS-15(3:1)2019.

[JK01] George Janelidze and Gregory M Kelly. A note on actions of a monoidal category. *Theory Appl. Categ*, 9(61-91):02, 2001.

[Koc72] Anders Kock. Strong functors and monoidal monads. *Archiv der Mathematik*, 23(1):113–120, 1972. doi:10.1007/bf01304852.

[Lev99] Paul Blain Levy. Call-by-push-value: A subsuming paradigm. In Jean-Yves Girard, editor, *TLCA*, volume 1581 of *Lecture Notes in Computer Science*, pages 228–242. Springer, 1999. doi:10.1007/978-94-007-0954-6_2.

[Lev04] Paul Blain Levy. *Call-By-Push-Value: A Functional/Imperative Synthesis (Semantics Structures in Computation, V. 2)*. Kluwer Academic Publishers, USA, 2004. doi:10.1007/978-94-007-0954-6.

[LG19] Paul Blain Levy and Sergey Goncharov. Coinductive resumption monads: Guarded iterative and guarded elgot. In *Proc. 8rd international conference on Algebra and coalgebra in computer science (CALCO 2019)*, LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CALCO.2019.13.

[LPT02] Paul Blain Levy, John Power, and Hayo Thielecke. Modelling environments in call-by-value programming languages. *Inf. & Comp*, 185:2003, 2002. doi:10.1016/s0890-5401(03)00088-9.

[Mel17] Paul-André Melliés. The parametric continuation monad. *Mathematical Structures in Computer Science*, 27(5):651–680, 2017. doi:10.1017/s0960129515000328.

[Mil89] R. Milner. *Communication and concurrency*. Prentice-Hall, 1989.

[Mil05] Stefan Milius. Completely iterative algebras and completely iterative monads. *Inf. Comput.*, 196(1):1–41, 2005. doi:10.1016/j.ic.2004.05.003.

[ML71] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer, 1971. doi:10.1007/978-1-4612-9839-7.

[Mog91] Eugenio Moggi. Notions of computation and monads. *Inf. Comput.*, 93:55–92, 1991. doi:10.1016/0890-5401(91)90052-4.

[NU15] Keiko Nakata and Tarmo Uustalu. A Hoare logic for the coinductive trace-based big-step semantics of While. *Log. Meth. Comput. Sci.*, 11(1), 2015. doi:10.2168/lmcs-11(1:1)2015.

[PG14] Maciej Piróg and Jeremy Gibbons. The coinductive resumption monad. In *Mathematical Foundations of Programming Semantics, MFPS 2014*, volume 308 of *ENTCS*, pages 273–288, 2014. doi:10.1016/j.entcs.2014.10.015.

[PP01] Gordon Plotkin and John Power. Adequacy for algebraic effects. In *Proc. 4th International Conference in Foundations of Software Science and Computation Structures (FOSSACS 2001)*, volume 2030 of *LNCS*, pages 1–24, 2001. doi:10.1007/3-540-45315-6_1.

[PR97] A. J. Power and E. P. Robinson. Premonoidal categories and notions of computation. *Mathematical Structures in Computer Science*, 7(5):453–468, October 1997. doi:10.1017/s0960129597002375.

[PT99] A. John Power and Hayo Thielecke. Closed Freyd- and kappa-categories. In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming (ICALP 1999)*, Lecture notes in Computer Science, page 625–634, Berlin, Heidelberg, 1999. Springer-Verlag. doi:10.1007/3-540-48523-6_59.

[Sch69] Dietmar Schumacher. Minimale und maximale tripelerzeugende und eine bemerkung zur tripelbarkeit. *Archiv der Mathematik*, 20(4):356–364, Sep 1969. doi:10.1007/BF01899590.

[SP00] Alex Simpson and Gordon Plotkin. Complete axioms for categorical fixed-point operators. In *Proc. 15th Annual IEEE Symposium on Logic in Computer Science (LICS 2000)*, pages 30–41. IEEE Comput. Soc, 2000. doi:10.1109/lics.2000.855753.

[Sta14] Sam Staton. Freyd categories are enriched Lawvere theories. *Electron. Notes Theor. Comput. Sci.*, 303:197–206, mar 2014. doi:10.1016/j.entcs.2014.02.010.

[Uus03] Tarmo Uustalu. Generalizing substitution. *ITA*, 37(4):315–336, 2003. doi:10.1051/ita:2003022.

[UV08] Tarmo Uustalu and Varmo Vene. Comonadic notions of computation. *Electron. Notes Theor. Comput. Sci.*, 203(5):263–284, 2008. doi:10.1016/j.entcs.2008.05.029.