Computads and 2 dimensional pasting diagrams

by M Makkai

Introduction

1. This paper is the second installment of a series whose first item is the paper [M]. In [M], a paper was promised, [M4] in the references section there, with the tentative title "A 2-categorical pasting theorem: revisiting John Power's paper [P1] of the same title". The present paper is what [M4] has become.

The introduction of [M] should serve as a general introduction to the present paper as well.

The notions of " ω-category" and "computad" come from the work of Ross Street.

The basic notions of and around " ω-category" and "computad" will not be recalled here. By now, these concepts belong (or should belong ...) to the common knowledge in category theory. For instance, the reader is not far wrong if he/she takes "computad" to mean "free ω-category". However, the ways these concepts are formulated in this paper, and the special notations used when dealing with them, will have to be gleaned from [M], which is intended as a "foundational" paper for these concepts.

In the introductory first two sections, two things are done. First, we recall the necessary background material on computads, mainly by citing definitions and results from [M], but also by introducing new terms and statements which are in [M] only implicitly. The definitions and results cited are relevant or valid in arbitrary dimensions. The results cited from [M] are marked by the symbol [M], and numbered in the style [M](i), [M](ii), ...

Secondly, in sections 1 and 2, we also state some new results. The theorems and propositions in sections 1 and 2 marked in the style 1.1, 2.1, 2.2, ... will be proved only later in the paper. On the other hand, similarly numbered corollaries of the above are proved on the spot.

There is one constraint observed in sections 1 and 2: only such new results are stated which have straightforward *conjectured* higher-dimensional generalizations, although the results themselves are claimed and stated only for dimension 2, and occasionally 3.

In section 2, among others, an analog of John Power's theorem, 3.3 Theorem in [P1], "Every labelling of a pasting scheme has a unique composite", is stated (2.12 Theorem).

In the second part, from section 3 on, the concepts and results of a new "geometric theory" of computads, presently established only in dimension 2, are presented. After the purely combinatorial and elementary section 3, concerning what we call "planar arrangements", the first of two forms of the main result of the paper, Theorem 4.2, is formulated in Section 4. With the exception of those in section 9, all results of the paper, including the ones stated in sections earlier than the fourth, are essentially (that is, modulo the basics in [M]) corollaries of the main result 4.2.

Theorem 4.2 is a reconstruction of the "geometry" of a 2-dimensional pasting diagram (2-pd), valid for the class of 2-pd's called *anchored* (for the definition, see below; the terminology has been suggested by Andre Joyal). The geometry in question is given by *postulation* in [P1];

The author's research is supported by NSERC Canada and OTKA Hungary

here the "geometry" is computed from the algebraic expression of the pd.

2. There are two concerns in the paper, one explicit, the other somewhat implicit: the interest in general laws on the one hand, and computational procedures on the other.

The "geometry" of a pasting diagram is what we display when we *draw* the diagram. This is *the primary* aspect of the subject: it is what *we are given*, informally of course, when we start the investigation (witness the first few sentences of John Power's paper). It is a compelling idea to follow the hunch that there are general laws and procedures behind the drawing of categorical diagrams.

The theorems of the first four sections state the *laws*, proved for a small beginning range of cases and conjectured for others, of pasting diagrams. The *computational procedures* of the subject are shown only later; nevertheless, they are the first motivation for the paper.

For example, 2.2 Theorem, part (c), says that there is a so-called *planar arrangement* of the occurrences of the indeterminate 2-cells of a 2-dimensional pasting diagram (pd), under a mild, but important, restriction on the pd itself. This is our way of stating that a 2-pd *can be drawn in the plane*. But in fact, the complete point is not just that this "drawing" *exists*, but also that it can be *computed*. Namely, given a symbolic representation for the 2-pd, in the form that we call a molecule -- which is just a somewhat normalized syntactical term in the language of operations for the notion of 2-category -- we can effectively and "naturally" calculate said planar arrangement.

This concern for calculation explains a certain repetitiveness in the paper. The calculation just alluded to leads naturally to a *tree*, depending on the given molecule, that represents the steps in the calculation. The given molecule stands for a 2-pd that can be defined by numerous other molecules -- in fact, these latter molecules will all appear at one stage or another in the construction of the tree attached to the given molecule. The trees induced by these variant molecules are all different from one another, but they are all, essentially, spanning trees of a certain *graph* which is an invariant object attached to the 2-pd itself.

The graph is mentioned early on; 2.4 Corollary is a result, in the "anchored 2D" case, that gives an abstract description of it. On the other hand, the trees appear only in section 5. They are used to prove all the results stated in the earlier sections. The trees would have been easy to avoid altogether, by somewhat reformulating the proofs, if we had been only interested in the abstract/invariant laws without the calculations. As things are now, in the preparatory stages of dealing with the computational trees, we are compelled to state variants of a number of constructs that had been mentioned in the context of the graphs.

Computads and pasting diagrams serve as the basic carriers of the syntax of higher dimensional categories, weak and strict, as explained, for instance, in the introduction of [M]. This explains the interest in computational aspects of computads: following the lead of Gottlob Frege and David Hilbert, we adhere to the doctrine that all aspects of pure syntax have to be calculable and/or decidable.

3. I will now comment on the two main new concepts of the paper, that of *planar pasting prescheme*, related to dimension 2, and the more general *pasting prescheme* relevant in arbitrary dimensions.

"Pℓanar Pasting PreSchemes", PℓPPS's for short, are introduced in section 4. John Power has "pasting schemes" in both [P1] and [P2]. PℓPPS's are different from Power's concept for

dimension 2 in [P1] (and of course, different from that in [P2] too), but serve in similar roles. The prefix "pre" is there because the term "planar pasting scheme", PℓPS, is reserved for a PℓPPS which is "complete" in a suitable sense. PℓPPS's have unique composites, by design (that is, the proof that they do is more direct than in Power's case). The composite is a general 2-cell, also called 2-pasting diagram (2-pd), in (the underlying 2-category of) the underlying computad.

The main result, 4.2 Theorem, says, in essence, that every 2-pd satisfying a smallish but essential restriction ("anchored") *has* a complete PℓPPS displaying it (the composite of the PℓPPS is the given 2-pd). The uniqueness of the displaying PℓPPS is essentially obvious; but it is returned to in section 10.

Notice the opposite natures of the general outline here and of that in [P1]. In [P1], the 2D diagrams are *defined* as those given by a pasting scheme, and the work to be done is in showing that they make sense as 2-categorical composites. Here, the 2D diagrams are given in advance algebraically as 2-categorical composites of indeterminate cells in a computad ("free 2-category"); the work is to show that there are pasting schemes in the new sense that display them.

The general notion of "Pasting PreScheme", PPS, is introduced in section 9. It is formulated in arbitrary dimensions. The main result of the paper concerning this concept is that any PPS has a unique composite (9.3 Proposition). There is no analog in the paper of the hard work done for the planar pasting schemes, the construction of them for a large class of 2-pd's; this analog is planned for the future installments of the series.

Although it is not true that every 2D pasting prescheme is planar, the truth is not far from saying that. The main result of the paper, expressed in terms of the general notion of pasting prescheme, is that any PPS of an anchored 2-pd has a planar extension, and (therefore, by 4.2 Theorem) there is a unique pasting scheme (complete pasting prescheme) displaying any given anchored 2-pd, which is in fact planar (see 10.4 Theorem).

§1 Types, shapes and occurrences

Pasting diagrams

Let us codify the concept of "pasting diagram". A *pasting diagram* (*Pd* for short) is a pair $(\boldsymbol{X}, \Gamma)$ where $\boldsymbol{X}$ is a computad, Γ is a cell of the ω-category $\boldsymbol{X}$, $\Gamma \in \|\boldsymbol{X}\|$, and

$$\boldsymbol{X} = \mathrm{Supp}_{\boldsymbol{X}}(\Gamma) \ . \qquad (1)$$

The idea is that $\boldsymbol{X}$ is the diagram itself, which pastes (composes) into the composite Γ. So, in fact, the expression "pasted diagram" would be more suitable. Fortunately, "Pd" is neutral with respect to the two readings.

The capitalized version "Pd" is used for the concept that contains it's own "context" as $(\boldsymbol{X}, \Gamma)$ contains a reference to $\boldsymbol{X}$. A "pd" uncapitalized is an element of $\|\boldsymbol{X}\|$, with $\boldsymbol{X}$ given in a larger context.

The equality (1) means that *all* the indeterminates in $\boldsymbol{X}$ are *used* in writing Γ. This way of saying the matter is a correct definition if the cells in a computad are taken to be equivalence

classes of terms formed from the indeterminates as "variables" (see [Pe], or [M] where Jacques Penon's [Pe] definition of computad ("polygraph" in French) via terms is re-stated). On the other hand, one can define, for any computad $\boldsymbol{X}$ and any $\Gamma \in \|\boldsymbol{X}\|$, $\mathtt{Supp}_{\boldsymbol{X}}(\boldsymbol{X})$, a subcomputad of $\boldsymbol{X}$, whose indeterminates are the ones "used" in Γ, in a purely algebraic manner too; see [M].

The datum Γ, the composite itself, is not a superfluous item. With $\boldsymbol{X}^*$ generated by the single 0-cell X, and the single 1-cell $f: X \to X$, we have infinitely many Γ for which $(\boldsymbol{X}^*, \Gamma)$ is a Pd: all the composites f^m (m=1, 2, 3, ...). The reader will be right if he/she thinks that we should be interested in when a computad $\boldsymbol{X}$ *has a unique composite*, meaning there is a unique Γ for which $(\boldsymbol{X}, \Gamma)$ is a Pd.

The notations $(\boldsymbol{X}, \Gamma)$, $(\boldsymbol{Y}, \Lambda)$ will always mean Pd's in the sense just codified. We also write $\underline{\Gamma}$ for $(\boldsymbol{X}, \Gamma)$, $\underline{\Lambda}$ for $(\boldsymbol{Y}, \Lambda)$.

Let's define the *category of pasting diagrams,* $\mathtt{Pd}$, to have objects the Pd's, and arrows $(\boldsymbol{X}, \Gamma) \xrightarrow{f} (\boldsymbol{Y}, \Lambda)$ those $\boldsymbol{X} \xrightarrow{f} \boldsymbol{Y}$ in $\mathtt{Comp}$ for which $f(\Gamma)=\Lambda$. $\mathtt{Pd}$ has a forgetful functor $\mathtt{Pd} \longrightarrow \mathtt{Comp}$. ($\mathtt{Comp}$ is the category of all (small) computads: see [M]).

The *dimension* of the Pd $(\boldsymbol{X}, \Gamma)$ is the dimension of Γ (as a cell of the ω-category $\boldsymbol{X}$). We have $\mathtt{dim}(\boldsymbol{X},\Gamma)=\mathtt{max}\{\mathtt{dim}(x): x \in |\boldsymbol{X}|\}$.

$\mathtt{Pd}_n$ is the full subcategory of $\mathtt{Pd}$ whose objects are the Pd's of dimension n. The notation $\mathtt{Pd}_{\leq n}$ is analogous.

An *Indeterminate* (Indet for short) will be a Pd $\underline{x}=(\boldsymbol{X}, x)$ where $x \in |\boldsymbol{X}|$, that is, x is an indeterminate (=free generator; see [M]) in $\boldsymbol{X}$. $\mathtt{Indet}$ is the full subcategory of $\mathtt{Pd}$ consisting of the Indets.

If $(\boldsymbol{X}, x)$, $(\boldsymbol{X}, y)$ are Indets with the same underlying computad $\boldsymbol{X}$, then they are the same: $x=y$: this is obvious, since x is the unique maximal-dimensional indet in $\boldsymbol{X}$. On the other hand, two different Pd's may have the same underlying computad.

$\mathtt{Indet}$ is not only a full subcategory of $\mathtt{Pd}$, but it is a sieve in $\mathtt{Pd}$: if $\underline{\Gamma}=(\boldsymbol{X}, \Gamma) \longrightarrow (\boldsymbol{Y}, y)$, and (Y, y) is an indeterminate, then $\underline{\Gamma}$ is itself an Indet; $f(\Gamma)=y$ implies that $\Gamma \in |\boldsymbol{X}|$: this was proved in [M].

$\mathtt{Indet}_{\mathtt{m/1}}$ is defined as the full subcategory of $\mathtt{Indet}$ whose objects $\underline{x}=(\boldsymbol{X}, x)$ are the many-to-one Indets, that is, are such that $\mathtt{c}x$ is an indet too. $\mathtt{Indet}_{\mathtt{m/1}}$ is also a sieve in $\mathtt{Pd}$.

The Indets play the central role among the Pd's; in fact, in a sense, every Pd can be "replaced" by an Indeterminate, albeit by one of one-higher dimension. For the Pd $\underline{\Gamma}=(\boldsymbol{X}, \Gamma)$, consider the many-to-one Indet $\underline{\tilde{\Gamma}}=(\boldsymbol{X}[x][y], y)$ defined, in two steps, by first adjoining to $\boldsymbol{X}$ the *new* indet x of the dimension of Γ with the specification $\mathtt{d}x=\mathtt{d}\Gamma$, $\mathtt{c}x=\mathtt{c}\Gamma$, and then adjoining y of one higher dimension with $\mathtt{d}y=\Gamma$, $\mathtt{c}y=x$.

$\underline{\tilde{\Gamma}}$ is, of course, defined up to isomorphism only, although, as usual, we pretend that it is strictly specified.

There is an obvious bijection between $\mathtt{hom}(\underline{\Gamma},\underline{\Lambda})$ and $\mathtt{hom}(\underline{\tilde{\Gamma}},\underline{\tilde{\Lambda}})$. In fact, we have an equivalence of categories

$$(\underline{\Gamma}\mapsto\underline{\tilde{\Gamma}}):\mathtt{Pd}\xrightarrow{\simeq}\mathtt{Indet}_{\mathtt{m/1}}$$

Typing and occurrence

We will call the Pd $\underline{\Gamma}=(\boldsymbol{X},\Gamma)$ *separated* if for all $\underline{\Lambda}=(\boldsymbol{Y},\Lambda)\in\mathtt{Pd}$ and all $\underline{\Lambda}\xrightarrow{f}\underline{\Gamma}$ in Pd, f is necessarily an isomorphism.

A *computope* (see [M]) is an Indet $(\boldsymbol{X},x)$ such that for all Indets $(\boldsymbol{Y},y)$ and arrows $(\boldsymbol{Y},y)\xrightarrow{f}(\boldsymbol{X},x)$, f is necessarily an isomorphism.

We say that the computad $\boldsymbol{X}$ is a *computope* if there is a, necessarily unique, computope $(\boldsymbol{X},x)$ with underlying computad $\boldsymbol{X}$.

From the fact that Indet is a sieve in Pd, we immediately see that an Indet is a computope iff it is a separated Pd, and the Pd $\underline{\Gamma}$ is separated if and only if $\underline{\tilde{\Gamma}}$ is a computope.

The *category of all computopes*, Ctp, is defined as the skeletal full subcategory of Comp itself, whose objects form a full set of representatives of isomorphism types of all the computopes. (Thus, we allow all computad morphisms $f:A\to B$ for computopes (A,x) and (B,y), not just the ones in Indet.)

It is an important fact (see [M]) that Ctp is a *finitary one-way* category. A category $\boldsymbol{D}$ is *finitary* if for all objects X in $\boldsymbol{D}$, the set $\{f\in\mathtt{Arr}(\boldsymbol{D}):c(f)=X\}$ is finite. $\boldsymbol{D}$ is *one-way* if there is no infinite descending chain

$$X_0\xleftarrow{f_0}X_1\xleftarrow{f_1}\ldots X_n\xleftarrow{f_0}X_{n+1}\xleftarrow{\quad}\ldots$$

of non-identity arrows in it. (The finitary-ness of Ctp is not immediate; the one-way quality is.)

In [M], the following are proved.

Theorem [M]

(i) For every Indet $\underline{x}$, there is a computope $\underline{y}$ with a morphism $\underline{y}\to\underline{x}$.

(ii) For every Indet $\underline{x}$, there are only finitely many non-isomorphic

Indets $\underline{y}$ having an arrow $\underline{y} \to \underline{x}$ to $\underline{x}$.

((i) is 11.(4) in [M]; (ii) is stated in the proof of the same 11.(4) as "the isomorphism types of resolvents of B form a non-empty finite set".)

Corollary For every Pd $\underline{\Gamma}$, there is a separated Pd $\underline{\Lambda}$ with a morphism $\underline{\Lambda} \longrightarrow \underline{\Gamma}$; up to isomorphism, there are only finitely many such $\underline{\Lambda}$.

To get the Corollary, apply the Theorem to $\hat{\underline{\Gamma}}$ as $\underline{x}$.

Referring to the Corollary, $\underline{\Lambda}$ is called a *type* for $\underline{\Gamma}$; a morphism $\underline{\Lambda} \longrightarrow \underline{\Gamma}$ a *specializing morphism* for $\underline{\Gamma}$.

We say that the Pd $\underline{\Gamma}$ is *uniquely typed* if

1) the specializing morphism for $\underline{\Gamma}$ from any type of $\underline{\Gamma}$ to $\underline{\Gamma}$ is unique: for $\underline{\Lambda}$ separated, if $\underline{\Lambda} \overset{f}{\underset{g}{\rightrightarrows}} \underline{\Gamma}$, then $f=g$;

and

2) the type of $\underline{\Gamma}$ is unique up to isomorphism: if $\underline{\Lambda}$, $\underline{\Xi}$ are separated, and $\underline{\Lambda} \longrightarrow \underline{\Gamma} \longleftarrow \underline{\Xi}$, we have $\underline{\Lambda} \cong \underline{\Xi}$.

Note that 1) is equivalent to the seemingly stronger condition

1*) for any $\underline{\Lambda}$, if $\underline{\Lambda} \overset{f}{\underset{g}{\rightrightarrows}} \underline{\Gamma}$, then $f=g$.

The reason is that, given $\underline{\Lambda}$, by the previous Corollary, there are separated $\hat{\underline{\Lambda}}$ and $\hat{\underline{\Lambda}} \xrightarrow{h} \underline{\Lambda}$; by 1), $f \circ h = g \circ h$; but h, as any map of Pd's, is surjective on indeterminates; it follows that $f=g$.

The main motivation for the foregoing notions is the desire to understand the idea of an *occurrence of* a generator $x \in |\boldsymbol{X}|$ in a Pd $(\boldsymbol{X}, \Gamma)$.

In the example $\underline{\Gamma}_m = (\boldsymbol{X}^*, f^m)$ (m=1, 2, 3, ...) above, it is natural to say that the 0-cell X "occurs $m+1$ times", and f "occurs m times", because this way of talking will match the *drawing* of the arrow f^m as the composite of a diagram:

$$X \xrightarrow{f} X \xrightarrow{f} X \xrightarrow{f} \ldots \xrightarrow{f} X \ . \qquad (2)$$

Let $\underline{\Lambda}_m=(\boldsymbol{Y}^*, \Lambda_m)$ be such that $\boldsymbol{Y}^*$ is generated by the distinct 0-cells X_i (i=1, ..., m, m+1) and the 1-cells $f_i:X_i \to X_{i+1}$. Let $\Lambda_m=f_1 \cdot \ldots \cdot f_m$. $\underline{\Lambda}_m$ is separated. The drawing of $\underline{\Lambda}_m$ is

$$X_1 \xrightarrow{f_1} X_2 \xrightarrow{f_2} X_3 \xrightarrow{f_3} \ldots \xrightarrow{f_{m-1}} X_m \ .$$

There is a unique map $\underline{\Lambda}_m \longrightarrow \underline{\Gamma}_m$; and, up to isomorphism, $\underline{\Lambda}_m$ is the only separated Pd $\underline{\Lambda}$ with a map $\underline{\Lambda} \to \underline{\Gamma}_m$. These facts allow us to say that *the* ith occurrence of X in (2) *is* X_i, and *the* ith occurrence of f in (2) *is* f_i. We have not only accounted for the number of occurrences of each generator, but have succeeded in defining *what* an occurrence is.

We may conclude that if the Pd $\underline{\Gamma}$ is uniquely typed, by $f:\underline{\Lambda} \longrightarrow \underline{\Gamma}$ say, the notion of an occurrence of any given indet $x \in |\underline{\Gamma}|$, as well as the number of distinct occurrences of x, are clarified: an occurrence of x is any element of the set $f^{-1}(x)$; the number of occurrences of x is the cardinality of the set $f^{-1}(x)$. The fact that the typing $(\underline{\Lambda}, f)$ is defined from $\underline{\Gamma}$ uniquely up to a unique isomorphism tells us that we will have a sound notion of occurrence.

Let us review the low dimensions as to unique typing.

In dimension 0, everything is trivial.

Next, one sees easily that every 1-Pd is uniquely typed.

However, in dimension 2 , it is not difficult to find a Pd that is not uniquely typed. In [M], the following example is given.

We let $\boldsymbol{X}$ be generated by the indets X, u and v, where $\dim(X)=0$, $\dim(u)=\dim(v)=2$, and $1_X \underset{v}{\overset{u}{\rightrightarrows}} 1_X$. We let $\Gamma=u \cdot v$. Since $u \cdot v=v \cdot u$ (Eckmann-Hilton), we have the automorphism $h:(\boldsymbol{X}, \Gamma) \xrightarrow{\cong} (\boldsymbol{X}, \Gamma)$ that flips u and v. Since $(\boldsymbol{X}, \Gamma)$ is separated, $(\boldsymbol{X}, \Gamma)$ is its own type, and 1) fails.

Thorsten Palm showed me an example for which 2) fails -- but, unfortunately, I do not understand it.

On the other hand, every 2-Indet (Indet of dimension 2) is uniquely typed. In fact, if $\underline{x}=(\boldsymbol{X}, x)$ is a 2-Indet, then $(\boldsymbol{Y}, y) \xrightarrow{f} (\boldsymbol{X}, x)$ is a typing for $\underline{x}$ iff, with the definitions $\boldsymbol{Y}_1=\mathrm{Supp}_{\boldsymbol{Y}}(dy)$, $f_1=f \restriction \boldsymbol{Y}_1$, $\underline{dy} = (\boldsymbol{Y}_1, dy)$, etc, we have that

(a) $\mathtt{dd}y \neq \mathtt{cc}y$ unless $\mathtt{d}x$ or $\mathtt{c}x$, hence $\mathtt{d}y$ or $\mathtt{c}y$, is an identity; and

(b) $\underline{\mathtt{d}y} \xrightarrow{f_1} \underline{\mathtt{d}x}$ and $\underline{\mathtt{c}y} \xrightarrow{f_2} \underline{\mathtt{c}x}$ are typings for $\underline{\mathtt{d}x}$ and $\underline{\mathtt{c}x}$, respectively,

and $\boldsymbol{Y}$ is the pushout of $\boldsymbol{Y}_1 \xleftarrow{\mathtt{incl}} \boldsymbol{Y}_3 \xrightarrow{\mathtt{incl}} \boldsymbol{Y}_2$ where $\boldsymbol{Y}_3 = \mathtt{Supp}_{\boldsymbol{Y}}(\{\mathtt{dd}y, \mathtt{cc}y\})$, with f defined compatibly with the pushout diagram.

Since $\underline{\Gamma} = (\boldsymbol{X}, \Gamma)$ is uniquely typed iff the Indet $\tilde{\underline{\Gamma}}$ is, we have that not all 3-Indets are uniquely typed.

A large class of 2-Pd's, and the corresponding class of 3-Indets, the so-called 2-*anchored* ones, are uniquely typed. We call an indeterminate x *anchored* if x is of dimension ≤ 1, or, if $\mathtt{dim}(x) \geq 2$, $\mathtt{c}x$ is a non-identity cell, $\mathtt{c}x \neq 1_{\mathtt{cc}x}$. A computad $\boldsymbol{X}$ is *anchored* if all indets in $\boldsymbol{X}$ are anchored; a Pd $(\boldsymbol{X}, \Gamma)$ is *k-anchored* if all indets of dimension k in $\boldsymbol{X}$ are anchored.

Of course, the dual notion when we disallow identities as domains, rather than codomains, of indeterminates gives rise to similar conclusions. The "Eckmann-Hilton" example above shows that bad effect of allowing indeterminates whose domain and codomain are both identities. In section 4, there will be a (simple) example showing that allowing two indeterminates, one with an identity domain, the other an identity codomain, is also bad. In other words, one has to globally exclude either identity domains, or identity codomains, for indets.

One of the main results of the present paper is

1.1 Theorem All 2-anchored 3-Indets, and as a particular case, all anchored 2-Pd's are uniquely typed.

Shape

The word "shape" instead of that of "type" is appropriate here too.

Let us say that Pd's $\underline{\Gamma}$ and $\underline{\Lambda}$ *have the same shape* if they belong to the same connected component of the category $\mathtt{Pd}$; that is, if there is a zig-zag

$$\underline{\Gamma} = \underline{\Gamma}_0 \longrightarrow \underline{\Gamma}_1 \longleftarrow \underline{\Gamma}_2 \longrightarrow \ldots \longleftarrow \underline{\Gamma}_k = \Lambda$$

of morphisms in $\mathtt{Pd}$.

Let me remind the reader of the fact that $\mathtt{Comp}$, the category of computads, is a locally finitely presentable category, in particular, it is both complete and cocomplete; see [M]. In $\mathtt{Comp}$, the colimits are "easy"; but the limits are only inferred from the "aleph-zero accessibility" of $\mathtt{Comp}$ (which is also "easy") plus the existence of the colimits. In particular, $\mathtt{Comp}$ has a terminal object $\boldsymbol{T}$, the terminal computad, but $\boldsymbol{T}$ is very far from being a trivial object. For more, see (also) [M].

Using a fixed copy of the terminal computad $\boldsymbol{T}$, and the morphism $!_{\boldsymbol{X}} : \boldsymbol{X} \longrightarrow \boldsymbol{T}$, every Pd

$\underline{\Gamma}=(\boldsymbol{X}, \Gamma)$ has a unique morphism $\underline{\Gamma} \xrightarrow{!} \underline{\Sigma}$ to a Pd $\underline{\Sigma}$ where $\underline{\Sigma}=(\boldsymbol{Z}, \Sigma)$ has its underlying computad $\boldsymbol{Z}$ a subcomputad of $\boldsymbol{T}$. Following Ross Street, we call this $\underline{\Sigma}$ the *shape* of $\underline{\Gamma}$. We mean by *a shape*, in general, a Pd whose underlying computad is a subcomputad of $\boldsymbol{T}$.

Note that this fits the previous terminology: the two meanings of "having the same shape" coincide -- and in the zig-zag of the first definition, we may always take $k=2$.

A type of a Pd is also a type of the shape of the Pd.

If two Pd's have the same type (the same separated Pd is a type of both), then they also have the same shape. I do not know if the converse holds.

The concepts of "type" and "shape" are, in a sense, dual to each other. Obviously, the "type" works less smoothly than the "shape". However, this is not simply a drawback of the notion of "type". The non-uniqueness related to "types" is a real difficulty with the idea of occurrence that cannot be ignored.

Concrete presheaf categories of computads

The question whether or not various categories of computads are *presheaf categories*, a question that has been investigated in the literature, is closely related to the question which Pd's are uniquely typed. I introduce this subject with some new terminology.

A class $\boldsymbol{C}$ of computads is said to be *standard* if

1) it is a sieve in $\mathtt{Comp}$: whenever $\boldsymbol{X} \to \boldsymbol{Y}$ is an arrow in $\mathtt{Comp}$, and $\boldsymbol{Y} \in \boldsymbol{C}$, then $\boldsymbol{X} \in \boldsymbol{C}$; and

2) it is closed under coverings in $\mathtt{Comp}$: whenever $(\boldsymbol{X}_i \xrightarrow{f_i} \boldsymbol{Y})_{i \in I}$ is a family of arrows in $\mathtt{Comp}$, $\boldsymbol{X}_i \in \boldsymbol{C}$ for all $i \in I$, and the derived family of the arrows $(|\boldsymbol{X}_i| \xrightarrow{|f_i|} |\boldsymbol{Y}|)_{i \in I}$ on the sets of indeterminates is a surjective family, then $\boldsymbol{Y} \in \boldsymbol{C}$.

There are many important examples of standard classes. The total class is an example. So is the class of *anchored* computads; for the term, see above (it is an easy fact seen in [M] that if $f:\boldsymbol{X} \to \boldsymbol{Y}$ is a morphism of computads, and $a \in ||\boldsymbol{X}||$ is not an identity, then $f(a)$ is not an identity either). The class of *positive computads*, in which there are no indeterminates with codomain or domain equal to an identity, is a natural standard class; in fact, it seems that the pasting schemes of [P] or [S] are meant to be positive.

An important example is the class of *many-to-one computads*: the class of computads $\boldsymbol{X}$ for which for every $x \in |\boldsymbol{X}|$, $\dim(x) \geq 1$, we have that cx is an indeterminate itself. (See, e.g., [M] for why many-to-one computads are important.) For any fixed $n \in \mathbb{N}$, the computads of dimension at most n is another example.

Given any class $\boldsymbol{C}$ of computads, the Pd's *associated with* $\boldsymbol{C}$ are those Pd's $(\boldsymbol{X}, \Gamma)$ for which $\boldsymbol{X} \in \boldsymbol{C}$. The class of Pd's associated with $\boldsymbol{C}$ is written as $\mathtt{Pd}(\boldsymbol{C})$. Similarly, we have

$\mathtt{Indet}(\boldsymbol{C})$, the class of Indets associated with $\boldsymbol{C}$.

If $\boldsymbol{C}$ is standard, then each of the classes $\mathtt{Indet}(\boldsymbol{C})$ and $\mathtt{Pd}(\boldsymbol{C})$ uniquely determines $\boldsymbol{C}$. Namely, $\boldsymbol{X}\in\boldsymbol{C}$ iff for all $\Gamma\in\|\boldsymbol{X}\|$, $(\mathrm{Supp}_{\boldsymbol{X}}(\Gamma),\Gamma)\in\mathtt{Pd}(\boldsymbol{X})$ iff for all $x\in|\boldsymbol{X}|$, $(\mathrm{Supp}_{\boldsymbol{X}}(x),x)\in\mathtt{Indet}(\boldsymbol{C})$.

For a class $\boldsymbol{I}$ of Indets, there is a standard class $\boldsymbol{C}$ with $\mathtt{Indet}(\boldsymbol{C})=\boldsymbol{I}$, if and only if the following both hold:

1) whenever $\underline{x}=(\boldsymbol{X},x)\in\boldsymbol{I}$, and $y\in|\boldsymbol{X}|$, then $\underline{y}=(\mathrm{Supp}_{\boldsymbol{X}}(y),y)\in\boldsymbol{I}$;

2) whenever $\underline{x}\in\boldsymbol{I}$, and $\underline{y}\longrightarrow\underline{x}\longrightarrow\underline{z}$ are arrows in $\mathtt{Indet}$, then both $\underline{y}$, $\underline{z}$ belong to $\boldsymbol{I}$.

Note that 2) can be said equivalently in this way: $\boldsymbol{I}$ is *shape-determined*: if two indets have the same shape, and one of them is in $\boldsymbol{I}$, then so is the other.

1) is a natural "reasonability condition": if we "accept" an indeterminate, we should also "accept" all indets involved in it.

A *standard class of Indets* is one that satisfies the last-listed conditions 1) and 2).

We may say that the standard classes of computads, and the standard classes of Pd's are the ones that are selected by the shapes, or equivalently, the types of indets involved in them.

A *concrete category* is a pair $(\boldsymbol{A},|\text{-}|)$ where $\boldsymbol{A}$ is a category, $|\text{-}|$ is a functor $|\text{-}|:\boldsymbol{A}\longrightarrow\mathtt{Set}$ to the category of sets. The concrete categories $(\boldsymbol{A},|\text{-}|_{\boldsymbol{A}})$, $(\boldsymbol{B},|\text{-}|_{\boldsymbol{B}})$ are said to be *equivalent* if there exists an equivalence of categories $E:\boldsymbol{A}\xrightarrow{\simeq}\boldsymbol{B}$ that is compatibly with the underlying-set functors: $|\text{-}|_{\boldsymbol{B}}\circ E\cong|\text{-}|_{\boldsymbol{A}}$.

Any category of the form $\hat{\boldsymbol{D}}\overset{\mathrm{def}}{=}\mathtt{Set}^{\boldsymbol{D}^{\mathrm{op}}}$, with $\boldsymbol{D}$ a small category, is regarded as a *concrete category* with the underlying-set functor $(F\in\boldsymbol{D}^{\mathrm{op}})\mapsto\coprod_{X\in\mathtt{Ob}(\boldsymbol{D})}F(X)$.

Every subcategory of $\mathtt{Comp}$ is regarded as a *concrete category* with the underlying-set functor defined as $\boldsymbol{X}\mapsto|\boldsymbol{X}|$ =the set of indets in $\boldsymbol{X}$.

We say of a concrete category that it is a *concrete presheaf category* if it is equivalent to the concrete category $\hat{\boldsymbol{D}}$ for some small category $\boldsymbol{D}$.

Any class of computads determines a full subcategory of $\mathtt{Comp}$, and thus a concrete category; if the class is standard, we call the resulting concrete category a *standard category of computads*.

The following is stated with a different wording, and proved, in [M].

Proposition [M] (iii) A standard category $\boldsymbol{C}$ of computads is a concrete presheaf category if and only if every Indet in $\mathtt{Indet}(\boldsymbol{C})$ is uniquely typed.

Remarks **1** The phrase "$(\boldsymbol{X}, x)$ is uniquely typed" is meant here in the exact sense stated above, without relativization to the subcategory $\boldsymbol{C}$ -- although such relativization would result in a correct statement too.

2 Modulo Theorem [M] (i), Prop (iii) is elementary category theory, involving the Yoneda functor and the like. On the other hand, I consider the Theorem [M] (i) on the existence of typing, quoted above from [M], to be a real theorem, requiring for its proof more than a superficial look at what it says -- at least until I am shown that I am wrong.

3 Note that we have that $\mathtt{Comp}$ itself is *not* a concrete presheaf category since there are Pd's that are not uniquely typed. In fact, $\mathtt{Comp}$ is not a presheaf category in the usual more general, "non-concrete", sense either: see [M]. Although I do not know, it may be true that a standard category of computads that is a presheaf category is necessarily a concrete presheaf category.

4 The most important example of a standard category of computads which is a concrete presheaf category is the category of many-to-one computads: the class of computads $\boldsymbol{X}$ for which for every $x \in |\boldsymbol{X}|$, $\dim(x) \geq 1$, we have that cx is an indeterminate itself.

5 Since every 2-Indet is uniquely typed, $\mathtt{Comp}_{\leq 2}$, the category of computads of dimension at most 2 is a concrete presheaf category. This is an old observation of Steve Schanuel's.

In case $(\boldsymbol{A}, |\text{-}|)$ is a concrete category which is equivalent to a concrete presheaf category $(\mathtt{Set}^{\boldsymbol{D}^{\mathrm{op}}}, |\text{-}|)$, the category $\boldsymbol{D}$ involved is determined *up to isomorphism* by $(\boldsymbol{A}, |\text{-}|)$ itself. This contrasts with the "non-concrete" case when the exponent category $\boldsymbol{D}$ is not determined even up to equivalence (although its idempotent-splitting completion is).

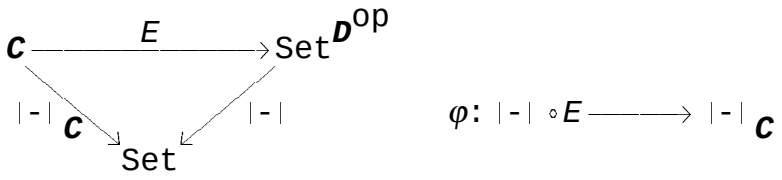
When $(\boldsymbol{A}, |\text{-}|)$ is equivalent to $(\mathtt{Set}^{\boldsymbol{D}^{\mathrm{op}}}, |\text{-}|)$, we call $\boldsymbol{D}$ the *type-category* for $(\boldsymbol{A}, |\text{-}|)$.

We would like to call it, rather, the "shape-category"; but it is related to the "types" rather than the "shapes".

In fact, we can identify what $\boldsymbol{D}$ should be even before we know that $(\boldsymbol{A}, |\text{-}|)$ is a concrete presheaf category. In particular, we define the *type category* of any standard category $\boldsymbol{C}$ of computads as $\boldsymbol{C} \cap \mathtt{Ctp}$, i.e. the full subcategory of $\boldsymbol{C}$ whose objects consist of exactly one isomorphic copy for each computope that belongs to $\boldsymbol{C}$. The type-category, as a subcategory of $\mathtt{Ctp}$, is always a one-way and finitary category.

(I emphasize again that the notion of computope is an *absolute* notion: whether or not something is a computope is decided in $\mathtt{Comp}$, rather than some subcategory of it -- although if you relativize the definition to the standard subcategory, you still have a correct definition.)

We observe that, if $\boldsymbol{D}$ is the type category of $\boldsymbol{C}$, then we have a canonical functor E and a natural transformation φ as in



defined by $E(\boldsymbol{X})=\boldsymbol{D}(i(-), \boldsymbol{X})$, where $i:\boldsymbol{D}\to\boldsymbol{C}$ is the inclusion, and

$$\varphi_{\boldsymbol{X}}(A): \quad \boldsymbol{D}(A, \boldsymbol{X}) \longrightarrow |\boldsymbol{X}| \qquad (\underline{x}=(A, x) \text{ a computope in } \boldsymbol{C}).$$
$$(f:A\to X) \longmapsto f(x)$$

We have (exercise!) that $(\boldsymbol{C}, |-|_{\boldsymbol{C}})$ is a concrete presheaf category if an only if E is an equivalence of categories and φ is an isomorphism of functors.

What we have said here about concrete presheaf categories and their type categories is general and simple category theory. On the other hand, the theoretical simplicity of the definition should not mislead one into believing that it is easy to get a concrete, workable description of the type-category, or that it is easy to see whether or not the standard category in question is a concrete presheaf category. For instance, $\mathtt{Comp}_{\mathtt{m/1}}$, the category of many-to-one computads is a concrete presheaf category; but the "concrete" description of its type-category, the category of *multitopes*, whose theoretical definition we now have as $\mathtt{Comp}_{\mathtt{m/1}}\cap\mathtt{Ctp}$, and the proof that it works as $\boldsymbol{D}$ in the last "exercise", are far from obvious; see [M] and the references there.

We write $\mathtt{Comp}_{\leq 3}^{\mathtt{2\text{-}anch}}$ for the full subcategory of $\mathtt{Comp}$ consisting of the computads of dimension at most 3 all whose 2-indets are anchored. The following is a consequence of 1.1 and (iii).

1.2 Corollary $\mathtt{Comp}_{\leq 3}^{\mathtt{2\text{-}anch}}$ is a concrete presheaf category.