Monoidal weak ω-categories as models of a type theory

Thibaut Benjamin

Abstract

Weak ω-categories are notoriously difficult to define because of the very intricate nature of their axioms. Various approaches have been explored, based on different shapes given to the cells. Interestingly, homotopy type theory encompasses a definition of weak ω-groupoid in a globular setting, since every type carries such a structure. Starting from this remark, Brunerie could extract this definition of globular weak ω-groupoids, formulated as a type theory. By refining its rules, Finster and Mimram have then defined a type theory called CaTT, whose models are weak ω-categories. Here, we generalize this approach to *monoidal* weak ω-categories. Based on the principle that they should be equivalent to weak ω-categories with only one 0-cell, we are able to derive a type theory MCaTT whose models are monoidal categories. This requires generalizing rules from contexts to lists of contexts, in order to encode the information carried by the unique 0-cell. The correctness of the resulting type theory is shown by defining a pair of translations between our type theory MCaTT and the type theory CaTT. Our main contribution is to show that these translations relate the models of our type theory to the models of the type theory CaTT consisting of ω-categories with only one 0-cell, by analyzing in details how they interact with the structural rules of both type theories.

Contents

Weak ω-categories are algebraic structures occurring naturally in modern algebraic topology and type theory. They consist of collections of cells in every dimension, which can be composed in various ways. The main difficulty in properly establishing a definition for those is due to the fact that the usual coherence axioms imposed on composition, such as associativity, are here relaxed and only supposed up to *invertible* higher cells, that we call *witnesses* for these axioms. Moreover, these witnesses themselves admit compositions, which satisfy axioms up to new witnesses, and so on, making the compositions and their coherence axioms intricate. There have been different approaches to propose a definition of weak ω-categories, which are summed up in a couple of surveys [15, 10]. These approaches are based on various shapes, such as simplicial sets or opetopic sets. In this article, we are interested in approaches based on globular sets. Examples of such approaches can be found in the work of Batanin [5] and Leinster [16], relying on the structure of globular operad. Independently Maltsioniotis proposed an alternative approach [19], inspired by a definition of weak ω-groupoid (i.e., weak ω-categories whose all cells are invertible) proposed by Grothendieck [14], and which is based on presheaves preserving some structure on a well-chosen category. The two approaches have been proved equivalent by Ara [2].

Type theoretical approach. An important observation which came along with the development of homotopy type theory [20] is the fact that the types in Martin-Löf type theory, and in homotopy type theory carry a structure of weak ω-groupoid, where the higher cells are given by identity types [17, 21, 1]. This allowed Brunerie to extract a minimal set of rules from homotopy type theory for generating the weak ω-categories [9], that he could prove to be equivalent to the definition of Grothendieck. Recently, Finster and Mimram proposed a generalization of Brunerie's type theory to weak ω-categories [13], parallel to the generalization Maltsiniotis proposed from Grothendieck definition. The type theory they introduced is called CaTT, and has been proved to be equivalent to the definition of Maltsiniotis [7].

Monoidal categories. In this article, we are interested in monoidal weak ω-categories. These are categories equipped with a tensor product allowing new ways to compose cells of every dimension. In particular, one cannot compose the cells of dimension 0 in weak ω-categories, but one can compose them in monoidal weak ω-categories. Moreover these new compositions are required to satisfy axioms like associativity, but since it happens in a weak setup, these axioms are again relaxed versions with witnesses in higher dimensions. This can be seen as a categorification of the notion of monoid. Our goal here is to provide a variant of the CaTT type theory in order to describe monidal weak ω-categories.

As noticed by Baez and Dolan [4, 3], monoidal weak ω-categories should be equivalent to weak ω-categories with only one 0-cell. In this correspondence, there is a shift in dimension: the 0-cells of the monoidal ω-category are the 1-cells of the ω-category with one 0-cell, and so on. The monoidal tensor product becomes the composition of arrows of the ω-category, and all its coherences are exactly those satisfied by these arrows in the ω-category.

Taking this correspondence as a starting point, we work out an explicit type theory MCaTT whose models are monoidal categories, by describing CaTT with the extra restriction that there should be only one 0-cells. This requires the introduction of new kinds of contexts, which are supported by lists of lists, along with types, terms and substitutions associated to those. We then define a pair of translations back and forth between CaTT and MCaTT, and use the interaction of these translations with the structure of the type theory to show that our proposed definition satisfies the correspondence we started with.

Plan of the paper. In section 1, we introduce the type theory CaTT along with the general tools we use to study type theory. Then in section 2 we define the type theory MCaTT and give an informal justification of the rules. Section 3 is devoted to formally justifying our choice of rules for MCaTT by first defining a pair of translations between CaTT and MCaTT, and then showing how these

translations lift to the models of these type theories to provide an equivalence between the models of MCaTT and the models of CaTT that are categories with only one 0-cell.

The author would like to thank Samuel Mimram for his valuable discussions regarding the work presented in this article.

1 Type theory for weak ω-categories

Before working on monoidal weak ω-categories, we recall in this section the type theory CaTT whose models are weak ω-categories. We refer the reader to [13, 7] for a more detailed presentation of it. We begin by defining what we mean here by a type theory, and construct, in this setting, a type theory describing globular sets. Then, by adding extra rules, we show how to extend it in order to obtain a type theory describing weak ω-categories.

1.1 Type theoretical notations and conventions

We first recall some basic definitions in type theory in order to establish the notations, terminology and conventions that are used throughout this article. Note that our method is to formulate a theory by defining a type theory, as opposed to a developing structures internally to Martin-Löf type theory or homotopy type theory.

Terms. A type theory manipulates various kind of objects, that we present here along with the convention we use for naming them.

- *variables*, which are elements of a given infinite countable set of variables, and that we denote $x, y, z, \ldots$,
- *terms*, which are built out of variables and constructors that will be introduced later on, they are denoted $t, u, \ldots$,
- *types*, which are built out of terms and constructors that will be introduced later on, they are denoted $A, B, \ldots$,

- *contexts*, which are supported by lists of associations of the form $x : A$, they are denoted $\Gamma, \Delta, \ldots$,
- *substitutions*, which are supported by lists of terms, denoted $\sigma, \tau, \ldots$.

All of these notions come with an associated set of variables, that we denote Var and that is the set of variables needed to build it out. Moreover, we also denote $\mathrm{Var}(t : A)$ for the union of $\mathrm{Var}(t)$ and $\mathrm{Var}(A)$.

Judgments. There are four kinds of judgments that are commons to all of our type theories, and that we refer to as *structural judgments*, they express the well-definedness of the previously introduced objects:

$$\begin{array}{lcl} \Gamma \vdash & : & \Gamma \text{ is a valid context} \\ \Gamma \vdash A & : & A \text{ is a valid type in } \Gamma \\ \Gamma \vdash t : A & : & t \text{ is a valid term of type } A \text{ in } \Gamma \\ \Gamma \vdash \sigma : \Delta & : & \sigma \text{ is a valid substitution from } \Gamma \text{ to } \Delta \end{array}$$

Structural rules. These judgments are always subject to the same structural rules. The only difference between the various type theories that we introduce is that they have different type constructors term constructors, with different introduction rules. The structural rules that are common to all type theories are the following

$$\frac{}{\varnothing \vdash} \qquad \frac{\Gamma \vdash A}{\Gamma, x : A \vdash} \; (x \notin \mathrm{Var}(\Gamma))$$

$$\frac{\Gamma, x : A \vdash}{\Gamma, x : A \vdash x : A} \qquad \frac{\Gamma \vdash t : A}{\Gamma, x : B \vdash t : A}$$

$$\frac{\Gamma \vdash}{\Gamma \vdash \langle\rangle : \varnothing} \qquad \frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash A \qquad \Gamma \vdash t : A[\sigma]}{\Gamma \vdash \langle \sigma, t \rangle : \Delta, x : A}$$

Note that we do not suppose any term or type constructors at first, because we want to study different theories, and the constructors vary from one to the

other. In particular, our type theories do not support Σ-types, Π-types, nor any construction of inductive types such as W-types or identity types.

We write $A[\sigma]$ for the application of a substitution σ on the type A, and it will be defined inductively later on, together with $t[\sigma]$, which is the application of σ on the term t, for each type and term constructor that we introduce. These definitions always make the following rules admissible

$$\frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash A}{\Gamma \vdash A[\sigma]} \qquad\qquad \frac{\Gamma \vdash \sigma : \Delta \qquad \Delta \vdash t : A}{\Gamma \vdash t[\sigma] : A[\sigma]}$$

These actions also define a composition of substitutions given inductively by

$$\langle\rangle \circ \tau = \langle\rangle \qquad\qquad \langle \sigma, t\rangle \circ \tau = \langle \sigma \circ \tau, t[\tau]\rangle$$

It is always be possible to check by induction that the action of substitutions is compatible with the composition, and thus making the composition associative

$$t[\sigma][\tau] = t[\sigma \circ \tau]$$
$$\sigma \circ (\tau \circ \delta) = (\sigma \circ \tau) \circ \delta$$

Since there is also always an identity substitution, this shows that for a type theory $\mathfrak{T}$, one can construct a *syntactic category* $\mathcal{S}_{\mathfrak{T}}$ associated to $\mathfrak{T}$, whose objects are the contexts of the type theory, and whose morphisms are the substitutions.

Category with families. We use the formalism of *categories with families* as our categorical axiomatization of the models of such a theory, and we refer to [11] for a detailed presentation of those. Intuitively, they correspond to a categorical reformulation of the structural rules we just introduced. In particular, the syntactic category can be equipped with a structure of category with families, given by the sets of types and terms of the theory. We do not introduce fully this concept, since we only work on a syntactical level: we define translations between

type theories, which act simultaneously on contexts, substitutions, types and terms. Such a translation induces a functor on the syntactic categories, and the only characterization we need is that this functor defines a morphism of category with families when the translation preserves all the structure we have introduced so far (i.e., the structural judgments, the structural rules and the applications of substitutions).

Models of a type theory. The category **Set** comes equipped with a structure of (large) category with families: technically, one should in fact consider a category with proper classes of types and terms associated to be the structure associated to **Set**, but we ignore this size issue in this article. The category of *models* of the type theory $\mathfrak{T}$ is defined to be the category of morphisms of category with families from $\mathcal{S}_{\mathfrak{T}}$ to **Set**, and is denoted $\mathbf{Mod}\,(\mathcal{S}_{\mathfrak{T}})$.

1.2 Globular sets

We study globular weak ω-categories, that is weak ω-categories whose underlying structure is a globular set, and hence we first define a type theory whose models are globular sets, following [13]. The *category of globes* $\mathbf{G}$ is the category generated by

$$[0] \underset{t}{\overset{s}{\rightrightarrows}} [1] \underset{t}{\overset{s}{\rightrightarrows}} [2] \underset{t}{\overset{s}{\rightrightarrows}} \cdots$$

with the relations $ts = ss$ and $st = tt$. The category of *globular sets* $\mathbf{GSet}$ is the category of presehaves over $\mathbf{G}$. It comes equipped with the *Yoneda embedding* $\mathrm{Y} : \mathbf{G} \to \mathbf{GSet}$. A globular set which is in the image of this functor is called *representable* or a *disk* and we denote it $\mathcal{D}^n = \mathrm{Y}([n])$. Every globular set is a colimit of representables in a canonical way [18].

Type theory for globular sets. In order to manipulate globular sets, we introduce two type constructors $\star$,which is the type of the 0-cells, and $\to$, which constructs the types of higher cells between two given cells. These constructors

are subject to the following introduction rules

$$\frac{\Gamma \vdash}{\Gamma \vdash \star} \qquad\qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : A}{\Gamma \vdash t \underset{A}{\longrightarrow} u}$$

In the other type theories, we study structures that are supported by a globular sets, thus we keep the same type constructors, the extra structure being added via new term constructors. For now there are no term constructors, so the only terms in this theory are variables, we call this theory $\mathfrak{G}$.

Syntactic category and models of $\mathfrak{G}$. Here, we present basic results about the theory $\mathfrak{G}$ without proofs, but by illustrating them with examples; a detailed presentation can be found in [7].

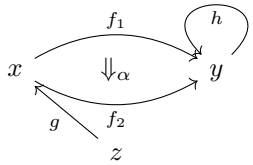
Proposition 1. *The syntactic category of the theory $\mathfrak{G}$ is equivalent to the opposite of the category of finite globular sets.*

Example 2. To understand this correspondance, one can simply read a globular set on a context, and conversely. For instance, the context on the left below corresponds to the globular set depicted on the right:

$x : \star, y : \star, z : \star$

$f_1 : x \to y, f_2 : x \to y, g : x \to z, h : y \to y$

$\alpha : f_1 \to f_2$



Proposition 3. *The category of models of $\mathfrak{G}$ is equivalent to the category of globular sets.*

Note that it respects the usual correspondence where the theory is the opposite of the finitely generated objects, and it inludes in its models via a Yoneda embedding.

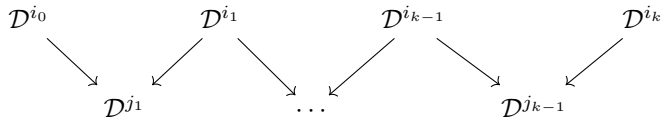
Disk contexts. Note that the category of finite globular sets contains in particular the representable objects $\mathcal{D}^n$, thus the equivalence provides corresponding objects in the syntactic category, that we again denote $\mathcal{D}^n$ and that we call *disk contexts*. In low dimensions, these contexts are given (up to renaming of their variables), by

$$\begin{aligned}
\mathcal{D}^0 &: (x : \star) \\
\mathcal{D}^1 &: (x : \star, y : \star, f : x \underset{\star}{\to} y) \\
\mathcal{D}^2 &: (x : \star, y : \star, f : x \underset{\star}{\to} y, g : x \underset{\star}{\to} y, \alpha : f \xrightarrow[x \underset{\star}{\to} y]{} g)
\end{aligned}$$

1.3 Pasting schemes and ps-contexts

In order to indtroduce new terms in our type theory, we need special kind of contexts, that we describe using the correspondence with finite globular sets.

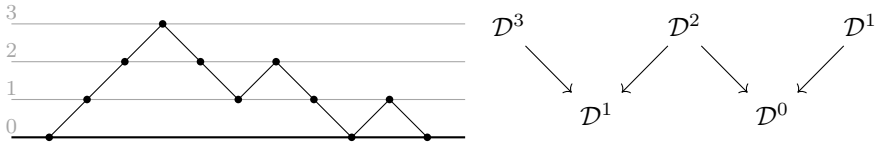
Globular sums. In the category of globular sets, we call a *globular sum* a colimit of the form



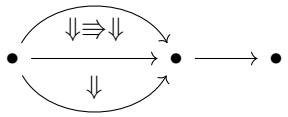
Where all the arrows pointing to the right are iterated sources and all the arrows pointing to the left are iterated targets. The *pasting schemes* are the globular sets that are obtained as a globular sum, note that they necessarily are finite.

Combinatorial description. We represent the combinatorial data provided by a pasting scheme with a diagram of the following form, where we determine

the corresponding globular sum by reading the heights of the peaks.



These two representation are both description of the pasting scheme which, as a globular set, is the following



More formally, this combinatorial data can be encoded as non-decreasing parking functions, that is non-decreasing functions $f : \{0, \cdots, n\} \to \{0, \cdots, n\}$ such that $f(0) = 0$ and $f(n) = n$, where we plot the function $f(n) - n$. There are various other combinatorial descriptions of pasting schemes, like Batanin trees [5], or Dyck words.

Ps-contexts. We define *ps-contexts* to be the contexts of the type theory $\mathfrak{G}$ which correspond to pasting schemes. Those can be characterized by a simple algorithm, which can be itself described using rules of a type-theoretical flavor: a ps-context Γ is precisely a context such that the judgment $\Gamma \vdash_{\mathsf{ps}}$ described below is derivable. We thus introduce two new forms of judgments

$\Gamma \vdash_{\mathsf{ps}}$	:	the context Γ is a ps-context
$\Gamma \vdash_{\mathsf{ps}} x : A$	:	Γ is a partial ps-context with dangling variable x

The second judgment should be understood as an auxiliary function. These two judgments are subject to the follwing rules

$$\frac{}{(x : \star) \vdash_{\mathsf{ps}} x : \star}\,(\text{PSS}) \qquad \frac{\Gamma \vdash_{\mathsf{ps}} f : x \underset{A}{\longrightarrow} y}{\Gamma \vdash_{\mathsf{ps}} y : A}\,(\text{PSD})$$

$$\frac{\Gamma \vdash_{\mathsf{ps}} x : A}{\Gamma, y : A, f : x \underset{A}{\longrightarrow} y \vdash_{\mathsf{ps}} f : x \underset{A}{\longrightarrow} y}\,(\text{PSE}) \qquad \frac{\Gamma \vdash_{\mathsf{ps}} x : \star}{\Gamma \vdash_{\mathsf{ps}}}\,(\text{PS})$$

Source and target. Given an integer $i \in \mathbb{N}$, a ps-context Γ comes equipped with an *i-source* $\partial_i^- \Gamma$, defined inductively on the structure by $\partial_i^- (x : \star) = (x : \star)$ and

$$\partial_i^- (\Gamma, y : A, f : x \to y) = \begin{cases} \partial_i^- \Gamma & \text{if } \dim A \geq i \\ \partial_i^- \Gamma, y : A, f : x \to y & \text{otherwise} \end{cases}$$

Similarly, a ps-context Γ also has an *i-target* $\partial_i^+ \Gamma$, defined inductively on its structure by $\partial_i^+ (x : \star) = (x : \star)$ and

$$\partial_i^+ (\Gamma, y : A, f : x \to y) = \begin{cases} \partial_i^+ \Gamma & \text{if } \dim A > i \\ \mathrm{drop}(\partial_i^+ \Gamma), y : A & \text{if } \dim A = i \\ \partial_i^+ \Gamma, y : A, f : x \to y & \text{otherwise} \end{cases}$$

where $\mathrm{drop}(\Gamma)$ is the context Γ with its last variable removed, i.e.,

$$\mathrm{drop}(\Gamma, x : A) = \Gamma$$

Moreover, we write

$$\partial^- \Gamma = \partial^-_{\dim \Gamma - 1} \qquad \partial^+ \Gamma = \partial^+_{\dim \Gamma - 1}$$

Note that with these conventions, $\partial^- (x : \star)$ and $\partial^+ (x : \star)$ are not defined.

1.4 The type theory CaTT

In order to axiomatize the weak ω-category structure, we add new terms, that we call coherences. This is done in our theory by adding a term constructor coh, together with introducion rules. The work we have done so far with ps-contexts shows relevance here, since those index the introduction of coherences, dually to how pasting schemes index the operations in other definitions of weak ω-categories [5, 16, 19].

Side conditions. The introduction rules for the term constructor coh have to verify some side conditions regarding the variables that are used in the type that we derive. There are two such rules, intuitively one of them creates witnesses for *operations*, for instance the composition, or whiskering, which a priori have no reason to be invertible, whereas the other creates witnesses for *equalities*, as for instance the associators, which are always weakly invertible. In order to simplify the notations, we encompass the requirements for these rules along with their side conditions in new judgments

$$\Gamma \vdash_{\mathsf{op}} A \quad : \quad \text{The type } A \text{ defines an admissible operation in } \Gamma$$

$$\Gamma \vdash_{\mathsf{eq}} A \quad : \quad \text{The type } A \text{ defines an admissible equality in } \Gamma$$

These two judgments are subject to the following derivation rules, which express all the requirements for the introduction rules of the constructor coh

$$\frac{\Gamma \vdash_{\mathsf{ps}} \qquad \partial^-(\Gamma) \vdash t : A \qquad \partial^+(\Gamma) \vdash u : A}{\overline{\Gamma} \vdash_{\mathsf{op}} t \underset{A}{\longrightarrow} u} \quad \begin{cases} \mathrm{Var}(t : A) = \mathrm{Var}(\partial^-(\Gamma)) \\ \mathrm{Var}(u : A) = \mathrm{Var}(\partial^+(\Gamma)) \end{cases}$$

$$\frac{\Gamma \vdash_{\mathsf{ps}} \qquad \Gamma \vdash A}{\Gamma \vdash_{\mathsf{eq}} A} \quad \mathrm{Var}(A) = \mathrm{Var}(\Gamma)$$

Coherences. We can now give the introduction rules for the new term constructor coh.

$$\frac{\Gamma \vdash_{\mathsf{op}} A \qquad \Delta \vdash \sigma : \Gamma}{\Delta \vdash \mathsf{coh}_{\Gamma,A}(\sigma) : A[\sigma]} \qquad\qquad \frac{\Gamma \vdash_{\mathsf{eq}} A \qquad \Delta \vdash \sigma : \Gamma}{\Delta \vdash \mathsf{coh}_{\Gamma,A}(\sigma) : A[\sigma]}$$

The resulting type theory obtained by adding these rules is called CaTT.

Examples. We give a few examples of derivations in this system illustrating how it describes weak ω-categories. This shows the introduction of new coherences and emphasizes the role of the substitutions taken as arguments of these coherences.

- Composition: In CaTT one can use a coherence to derive a witness for composition of 1-cells. Start by considering the context

  $$\Gamma_{\mathtt{comp}} = (x : \star, y : \star, f : x \to y, z : \star, g : y \to z)$$

  One can check that $\Gamma_{\mathtt{comp}} \vdash_{\mathsf{ps}}$, and compute its source $\partial^-(\Gamma_{\mathtt{comp}}) = (x : \star)$ and target $\partial^+(\Gamma_{\mathtt{comp}}) = (z : \star)$. Thus, the judgment $\Gamma_{\mathtt{comp}} \vdash_{\mathsf{op}} x \to z$ is derivable. Now considering any context Γ in CaTT, with two terms v, w, such that $\Gamma \vdash v : u \to u'$ and $\Gamma \vdash w : u' \to u''$ (i.e., two composable 1-cells u and v), there is a substitution $\sigma = \langle u, u', v, u'', w\rangle$ defined by this data, such that

  $$\Gamma \vdash \sigma : \Gamma_{\mathtt{comp}}$$

  So the introduction rule for coherences applies and builds a witness of the composition of v and w

  $$\Gamma \vdash \mathsf{coh}_{\Gamma_{\mathtt{comp}}, x \to z}(\sigma) : u \to u''$$

We denote this term with the simpler and more usual notation

$$\Gamma \vdash \mathtt{comp}\ v\ w : u \to u''$$

- Associativity: Similarly, one can pose the context

$$\Gamma_{\mathtt{assoc}} = (x : \star, y : \star, f : x \to y, z : \star, g : y \to z, w : \star, h : z \to w)$$

and one can check $\Gamma_{\mathtt{assoc}} \vdash_{\mathsf{ps}}$, and

$$\Gamma_{\mathtt{assoc}} \vdash_{\mathsf{eq}} \mathtt{comp}\ (\mathtt{comp}\ f\ g)\ h \to \mathtt{comp}\ f\ (\mathtt{comp}\ g\ h)$$

Whenever a context Γ defines three terms u, v, w that are composable 1-cells, this provides a witness for the associativity of their compositions, denoted

$$\Gamma \vdash \mathtt{assoc}\ u\ v\ w : \mathtt{comp}\ (\mathtt{comp}\ u\ v)\ w \to \mathtt{comp}\ u\ (\mathtt{comp}\ v\ w)$$

Models. The models of of the type theory CaTT have been studied in details in [7] and it is not the purpose of this article to study them in depth. However, we rely on them as our definition of weak ω-categories. This is justified by the following theorem, which we do not prove here, establishing the equivalence between the models of CaTT and the definition of weak ω-categories proposed by Maltsiniotis [19]:

Theorem 4. *The category* $\mathbf{Mod}\,(\mathcal{S}_{\mathsf{CaTT}})$ *of models of the type theory CaTT is equivalent to the category of weak* ω*-categories.*

In this article, for all intents and purposes, weak ω-categories are defined as models of the type theory CaTT, so from now on, the category of weak ω-categories is denoted $\mathbf{Mod}\,(\mathcal{S}_{\mathsf{CaTT}})$.

Syntactic category. Note that the syntactic category $\mathcal{S}_{\mathsf{CaTT}}$ can be concieved as the opposite category of the full subcategory of $\mathbf{Mod}\,(\mathcal{S}_{\mathsf{CaTT}})$ whose objects are freely finitely generated.

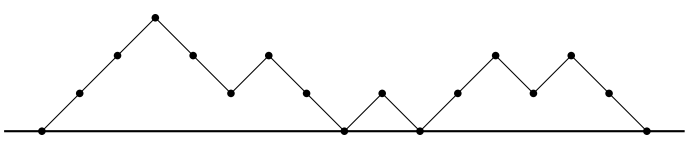
2 Type theory for monoidal weak ω-category

We now focus on monoidal categories. The idea here is to adapt the type theory for weak ω-categories, in order to enforce the constraint that our categories should always have exactly one 0-cell. To this end, we simulate the existence of "a virtual object of dimension -1" in our monoidal categories. There is no formal way in the theory for considering this object, but all the rules act as if it existed.

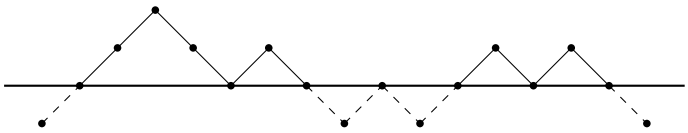
The category of monoidal weak ω-categories. Given a weak ω-category F in $\mathbf{Mod}\,(\mathcal{S}_{\mathsf{CaTT}})$, we call its set of 0-cells the image of the 0-th dimensional disk $F(\mathcal{D}^0)$. We thus denote $\mathbf{Mod}_{\bullet}\,(\mathcal{S}_{\mathsf{CaTT}})$ the full subcategory of $\mathbf{Mod}\,(\mathcal{S}_{\mathsf{CaTT}})$ whose objects are exactly the models F of CaTT, such that $F(\mathcal{D}^0) = \{\bullet\}$.

2.1 Lists of contexts

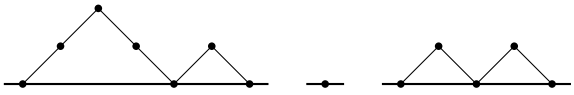
In the theory CaTT, the introduction rules for coherences relies on the structure of ps-contexts. In the case of monoidal categories, we have to adapt this structure, to what we call a *monoidal ps-context*. Such a context is defined as a list of ps-contexts, the intuition being that it is combinatorially equivalent to a single ps-context, but with a shift of dimension. Let us illustrate this correspondence with an example: Considering the following ps-context,



we shift its dimension by -1



and remove the (-1)-cells in order to finally obtain the following list of ps-contexts



Conversely, given a monoidal ps-context, there is a unique way to assemble together all the elements of the list to produce a ps-context. This is the intuition behind the type theory we introduce for monoidal categories, and this intuition will be made formal via a pair of translations between this theory and CaTT in Section 3. Note that monoidal ps-contexts simulate many virtual objects of dimensions -1, and not a single one, this will be corrected later on, by only considering judgments in a regular context.

Notations. For manipulating the lists, we always use overlined characters to denote lists of the same type than the character, $[]$ to denote the empty list, and $[\overline{a}, b]$ to denote the list whose tail is $\overline{a}$ and whose head is b. Moreover, given two lists $\overline{a}$ and $\overline{b}$, we denote $\overline{a}@\overline{b}$ their concatenation. Finally, for a list $\overline{a}$, we denote $\ell(\overline{a})$ its length.

Monoidal ps-contexts. We introduce a new judgment to express the fact that a data is a monoidal ps-context.

$$\overline{\Gamma} \vdash_{\mathsf{ps}} \quad : \quad \overline{\Gamma} \text{ is a well-formed monoidal ps-context}$$

The derivation rule for this judgment expresses that monoidal ps-contexts are nothing more that a list of ps-contexts, with non clashing variable names

$$\frac{}{[] \vdash_{\mathsf{ps}}} \qquad \frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \Delta \vdash_{\mathsf{ps}}}{\left[\overline{\Gamma}; \Delta\right] \vdash_{\mathsf{ps}}} \quad \left(\mathrm{Var}(\Delta) \cap \mathrm{Var}(\overline{\Gamma}) = \emptyset\right)$$

The monoidal ps-contexts come equipped with a notions of i-source, by applying successively the i-source of ps-contexts to all its elements.

$$\partial_i^-([]) = [] \qquad\qquad \partial_i^-(\left[\overline{\Gamma}; \Gamma'\right]) = \left[\partial_i^- \overline{\Gamma}; \partial_i^- \Gamma'\right]$$

This lets us define the general notion of source for monoidal ps-contexts, which are useful for defining the introduction rules for coherences

$$\partial^- \overline{\Gamma} = \partial^-_{\dim \overline{\Gamma} - 1} \overline{\Gamma}$$

Note that in the case of a monoidal ps-context of dimension 0, (i.e., a list of ps-contexts all isomorphic to $\mathcal{D}^0$), this definition does not make sense and we then use the convention that $\partial^- \overline{\Gamma} = []$. In general the case of the monoidal ps-context of dimension 0 is a limit case, and it will appear in various places. Similarly, monoidal ps-contexts come equipped with a notion of target $\partial^+ \overline{\Gamma}$, defined the same way as the source from the corresponding notion on ps-contexts. The following lemma is immediate by definition of the source and the target,

Lemma 5. *Any monoidal ps-context $\overline{\Gamma}$ of dimension non 0 has the same length than its source and its target $\ell(\overline{\Gamma}) = \ell(\partial^-(\overline{\Gamma})) = \ell(\partial^+(\overline{\Gamma}))$.*

Judgments for monoidal ps-contexts. Unlike the theory CaTT, in this theory the monoidal ps-contexts are not particular contexts (they are *lists* of contexts) and therefore we have to define a dedicated notion of type, term and substitution in a ps-context. For this reason, we introduce new dedicated

judgments:

$$\begin{array}{rcl}
\overline{\Gamma} \vdash A & : & A \text{ is a valid type in } \overline{\Gamma} \\
\overline{\Gamma} \vdash t : A & : & t \text{ is a valid term of type } A \text{ in } \overline{\Gamma} \\
\overline{\Gamma} \vdash \sigma : \Delta & : & \sigma \text{ is a valid substitution from } \overline{\Gamma} \text{ to } \Delta \\
\overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta} & : & \text{The list } \overline{\sigma} \text{ is a valid substitution from } \overline{\Gamma} \text{ to } \overline{\Delta}
\end{array}$$

Note that even if the notation are similar, these judgments are specific to the case where the left hand side is a monoidal ps-contexts and should not be confused with the regular judgments of our type theory. In particular, they are subject to the following derivation rules, which provide a guarantee regarding the order in which the ps-contexts of the list $\overline{\Gamma}$ are used.

$$\frac{}{\overline{\Gamma} \vdash \star} \qquad \frac{\overline{\Gamma} \vdash t : A \qquad \overline{\Gamma} \vdash u : A}{\overline{\Gamma} \vdash t \underset{A}{\longrightarrow} u}$$

$$\frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \Gamma' \vdash t : A}{[\overline{\Gamma}; \Gamma'] \vdash t : A} \qquad \frac{\overline{\Gamma} \vdash t : A \qquad \Gamma' \vdash_{\mathsf{ps}}}{\left[\overline{\Gamma}; \Gamma'\right] \vdash t : A}$$

$$\frac{\overline{\Gamma} \vdash_{\mathsf{ps}}}{\overline{\Gamma} \vdash \langle\rangle : \varnothing} \qquad \frac{\overline{\Gamma} \vdash \sigma : \Delta \qquad \Delta \vdash A \qquad \overline{\Gamma} \vdash t : A[\overline{\sigma}]}{\overline{\Gamma} \vdash \langle \sigma, t \rangle : \Delta, x : A}$$

$$\frac{}{[] \vdash [] : []} \qquad \frac{\overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta} \qquad \overline{\Gamma}' \vdash \sigma' : \Delta' \qquad \Delta' \vdash_{\mathsf{ps}}}{\overline{\Gamma} @ \overline{\Gamma}' \vdash [\overline{\sigma}; \sigma'] : \left[\overline{\Delta}; \Delta'\right]}$$

Imagining that the 0-cells we are describing are in fact arrows over objects of dimension -1, they come with an order and all derivable terms, types and substitutions should respect this order. The rules we just introduce are designed to enforce this order condition while still never mentionning any dimension -1.

Categorical structure. There is a compoisition for substitutions between two monoidal ps-context, as well as composition of such a substitution with a

substitution from a monoidal ps-context to a regular context. These are defined mutually inductively as follows.

$$\langle\rangle \circ \overline{\sigma} = \langle\rangle \qquad \langle \tau, t\rangle \circ \overline{\sigma} = \langle \tau \circ \overline{\sigma}, t[\overline{\sigma}]\rangle$$

$$[] \circ \overline{\sigma} = [] \qquad [\overline{\tau}, \tau'] \circ \overline{\sigma} = [\overline{\tau} \circ \overline{\sigma}; \tau' \circ \overline{\sigma}]$$

Moreover, we also define the identity substitution between on monoidal ps-context and itselft as the list constituted of only identity substitutions. The composition previously defined is associative and has the identity as left and right neutral element, this makes the monoidal ps-contexts into a category, whith morphisms being the monoidal substitutions.

Side conditions. Like in CaTT, in order to derive coherences, we need to check side conditions, and like in that case we encompass them into new judgments.

$\overline{\Gamma} \vdash_{\mathsf{op}} A$: The type A defines an admissible operation in $\overline{\Gamma}$

$\overline{\Gamma} \vdash_{\mathsf{eq}} A$: The type A defines an admissible equality in $\overline{\Gamma}$

These judgments are very similar to those of the theory CaTT except that they use our newly introduced judgments for monoidal ps-contexts as premises, instead of regular judgments of the theory. Besides, we have to treat separately the case of the type $\star$, which has to be made into an admissible operation, in order to describe the monoidal product

$$\frac{}{\left[\mathcal{D}^0\right] \vdash_{\mathsf{op}} \star} \qquad \frac{\overline{\Gamma} \vdash_{\mathsf{op}} \star}{\left[\overline{\Gamma}; \mathcal{D}^0\right] \vdash_{\mathsf{op}} \star}$$

$$\frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \partial^-(\overline{\Gamma}) \vdash t : A \qquad \partial^+(\overline{\Gamma}) \vdash u : A}{\overline{\Gamma} \vdash_{\mathsf{op}} t \underset{A}{\longrightarrow} u} \quad \begin{cases} \mathrm{Var}(t : A) = \mathrm{Var}(\partial^-(\overline{\Gamma})) \\ \mathrm{Var}(u : A) = \mathrm{Var}(\partial^+(\overline{\Gamma})) \end{cases}$$

$$\frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \overline{\Gamma} \vdash A}{\overline{\Gamma} \vdash_{\mathsf{eq}} A} \quad \mathrm{Var}(A) = \mathrm{Var}(\overline{\Gamma})$$

Coherences. We now introduce a term constructor coh, whose arguments are a monoidal ps-context, a type and a substitution between moniodal ps-contexts. We denote the term $\mathsf{coh}_{\overline{\Delta},A}[\overline{\sigma}]$. The introduction rules for such terms are the two following ones

$$\frac{\overline{\Delta} \vdash_{\mathsf{op}} A \qquad \overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta}}{\overline{\Gamma} \vdash \mathsf{coh}_{\overline{\Delta},A}(\overline{\sigma}) : A[\overline{\sigma}]} \qquad \frac{\overline{\Delta} \vdash_{\mathsf{eq}} A \qquad \overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta}}{\overline{\Gamma} \vdash \mathsf{coh}_{\overline{\Delta},A}(\overline{\sigma}) : A[\overline{\sigma}]}$$

Where the notation $A[\overline{\sigma}]$ (resp. $t[\overline{\sigma}]$) is the application of the monoidal substiution $\overline{\sigma}$ to the type A (resp. to the term t), defined by induction a follows

$$\begin{array}{rcl} \star[\overline{\sigma}] & \equiv & \star \\ t \underset{A}{\longrightarrow} u[\overline{\sigma}] & \equiv & t[\overline{\sigma}] \underset{A[\overline{\sigma}]}{\longrightarrow} u[\overline{\sigma}] \end{array} \qquad \mathsf{coh}_{\overline{\Gamma},A}(\overline{\gamma})[\overline{\sigma}] \equiv \mathsf{coh}_{\overline{\Gamma},A}(\overline{\gamma} \circ \overline{\sigma})$$

Note that with these definitions for coherences and for applying monoidal substitutions, the following rules are admissible. These rule express a compatibility between application of the substitution and the typing rules.

$$\frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \overline{\Gamma} \vdash A \qquad \overline{\Delta} \vdash \overline{\sigma} : \overline{\Gamma}}{\overline{\Delta} \vdash A[\overline{\sigma}]} \qquad \frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \overline{\Gamma} \vdash t : A \qquad \overline{\Delta} \vdash \overline{\sigma} : \overline{\Gamma}}{\overline{\Delta} \vdash t[\overline{\sigma}] : A[\overline{\sigma}]}$$

2.2 The type theory MCaTT

Up until now, the system that we have presented is incomplete, as it only adds judgments whose left hand side is a monoidal ps-contexts. Those judgments are not part of the structure of type theory, hence they do not change the models. Moreover, we had announced that we want to simulate a single object in dimension -1, and we presented a system that simulates many of them. In order to fix this, we present a way to derive regular judgments from these new judgments we have introduced. This requires the use of a new kind of

substitution going from a regular context to a monoidal ps-context, and which plays two roles. Firstly it allows for judgments in a monoidal ps-context to generate judgments in regular contexts, which has an effect on the models of the theory, and secondly it ensures that all the virtual objects of dimension -1 that were simulated in monoidal ps-contexts get (virtually) sent onto a unique virtual object of dimension -1 simulated in a regular context.

Substitutions to a monoidal ps-context. In order to build a bridge between the world of monoidal ps-context and our type theory, and transfer our newly introduced terms built out of coherences to a regular context, we need to introduce a new sort of substitution, that goes from a regular context to a monoidal ps-context. These substitution are characterized by the following judgment:

$$\Delta \vdash \overline{\sigma} : \overline{\Gamma} \quad : \quad \overline{\sigma} \text{ is a well-formed substitution from } \Delta \text{ to } \overline{\Gamma}$$

As the notation suggests, these substitutions are supported by lists of susbtitutions, and the derivation rules to which they are subject are the following

$$\frac{\Delta \vdash}{\Delta \vdash [] : []} \qquad \frac{\Delta \vdash \overline{\sigma} : \overline{\Gamma} \qquad \Delta \vdash \sigma' : \Gamma'}{\Delta \vdash [\overline{\sigma}; \sigma'] : \left[\overline{\Gamma}; \Gamma'\right]}$$

Note that such subsitution compose on the right with the usual subsitutions, and on the left with the substitutions from a monoidal ps-context. These compositions are associative and given by the following formulas

$$[] \circ \gamma \equiv [] \qquad [\overline{\sigma}; \sigma'] \circ \gamma \equiv [\overline{\sigma} \circ \gamma; \sigma' \circ \gamma]$$

$$\langle\rangle \circ \overline{\sigma} \equiv \langle\rangle \qquad \langle \gamma, t \rangle \circ \overline{\sigma} \equiv \langle \gamma \circ \overline{\sigma}, t[\overline{\sigma}] \rangle$$

$$[] \circ \overline{\sigma} \equiv [] \qquad [\overline{\gamma}; \gamma'] \circ \overline{\sigma} \equiv [\overline{\gamma} \circ \overline{\sigma}; \gamma' \circ \overline{\sigma}]$$

By orienting these relations left to right, it provides a confluent rewriting system for such substitutions, which gives an algorithm deciding whether two such substitutions are equal or not.

Defining terms in general contexts. We now use this new kind of substitutions to define terms and types in regular contexts from terms and types derived in monoidal ps-contexts. This is achieved by two new type constructors $t[\overline{\sigma}]$ and $A[\overline{\sigma}]$, along with the following introduction rules

$$\frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \overline{\Gamma} \vdash A \qquad \Delta \vdash \overline{\sigma} : \overline{\Gamma}}{\Delta \vdash A[\overline{\sigma}]} \qquad \frac{\overline{\Gamma} \vdash_{\mathsf{ps}} \qquad \overline{\Gamma} \vdash t : A \qquad \Delta \vdash \overline{\sigma} : \overline{\Gamma}}{\Delta \vdash t[\overline{\sigma}] : A[\overline{\sigma}]}$$

This introduces too many terms that representing the same operation, so in order to reduce the number of derivable terms to the minimum, we need to quotient by some relations, expressing the compatibility of type formers and term formers with the substitution operations.

$$\begin{array}{rcl@{\qquad}rcl} \star[\overline{\sigma}] & \equiv & \star & \mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})[\gamma] & \equiv & \mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma} \circ \gamma) \\ t \underset{A}{\longrightarrow} u[\overline{\sigma}] & \equiv & t[\overline{\sigma}] \underset{A[\overline{\sigma}]}{\longrightarrow} u[\overline{\sigma}] & \mathsf{coh}_{\overline{\Gamma},A}(\overline{\gamma})[\overline{\sigma}] & \equiv & \mathsf{coh}_{\overline{\Gamma},A}(\overline{\gamma} \circ \overline{\sigma}) \end{array}$$

By orienting these relations left to right as they are written, they define a confluent rewriting system, thus providing with an algorithm for type checking.

Coherences. The above rewriting rules give normal forms for terms which are not variables, of the form

$$\Delta \vdash \mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma}) : A[\overline{\sigma}]$$

for a substitution $\overline{\sigma}$ going from a context to a monoidal ps-context. In order to simplify our syntax, we introduce *coherences*, that are terms of the form

(implicitly taking as argument an identity substitution)

$$\overline{\Gamma} \vdash \mathsf{coh}_{\overline{\Gamma},A} : A$$

Then our rewriting system shows that all terms are obtained by applying a substitution to such a coherence.

2.3 Examples of derivations

Here are few examples and counter-exmaple of derivable terms in MCaTT, to illustrate how they describe monoidal weak ω-categories.

- Monoidal product: we can define the monoidal tensor product in MCaTT as the coherence

  $$\mathtt{prod} := \mathsf{coh}_{[(x:\star);(y:\star)],\star}$$

  By composing with a substitution, for every objects t and u (of type $\star$) in a given context Γ, we can form their tensor product as

  $$\Gamma \vdash \mathtt{prod}\ t\ u : \star$$

- Associativity of monoidal product: similarly, we can define the valid coherence

  $$\mathtt{assoc} := \mathsf{coh}_{[(x:\star);(y:\star);(z:\star)],\mathtt{prod}\ x\ (\mathtt{prod}\ y\ z) \to \mathtt{prod}\ (\mathtt{prod}\ x\ y)\ z}$$

  Using this coherence, for every object t, u and v (of type $\star$) in a given context Γ, we can form a witness of associativity

  $$\Gamma \vdash \mathtt{assoc}\ t\ u\ v : \mathtt{prod}\ t\ (\mathtt{prod}\ u\ v) \underset{\star}{\to} \mathtt{prod}\ (\mathtt{prod}\ t\ u)\ v$$

- Neutral element: it is also possible to derive the neutral element for the monoidal product, which can be viewed as a nullary monoidal product. It

is defined as a coherence

$$\mathsf{e} := \mathsf{coh}_{[],\star}$$

In any context, one compose with the empty substitution, in order to get the term witnessing the neutral element.

$$\Gamma \vdash \mathsf{e}\,[] : \star$$

In order to simplify the notations, we may simply denote e in any context, and omit the empty substitutions [] which carries no information.

- Cancellation witness: one can also prove in MCaTT that the term e is indeed a neutral element for the monoidal product, here for instance on the left. This is done by the coherence

$$\texttt{l-unit} := \mathsf{coh}_{[(x:\star)],(\texttt{prod e } x)\to x}$$

This coherence can be used to derive, for eny object u in a context Γ, the following witness of left unitality of e

$$\Gamma \vdash \texttt{l-unit}\ u : (\texttt{prod e}\ u) \to u$$

- Functoriality of monoidal product: The following coherence defines the functoirality for monoidal product

$$\texttt{funl} := \mathsf{coh}_{[(x:\star,y:\star,f:x\to y);(z:\star)],(\texttt{prod } x\ z)\to(\texttt{prod } y\ z)}$$

Given any three objects u, u', v together with a term t of type $u \to u'$ in a context Γ, this coherence can be used to derive a witness for the fonctoriality of the monoidal tensor product on the left

$$\Gamma \vdash \texttt{funl}\ t\ v : (\texttt{prod}\ u\ v) \to (\texttt{prod}\ u'\ v)$$

- Symmetry of the monoidal product: note that the same idea does not apply to derive the symmetry of the monoidal product. Indeed, if we try to build a witness for symmetry, it would be a term

$$\mathtt{sym} := \mathsf{coh}_{[(x:\star);(y:\star)],\mathtt{prod}\ x\ y \to \mathtt{prod}\ y\ x}$$

  It turns out that the list $[(x : \star); (y : \star)] \vdash [\langle y \rangle; \langle x \rangle] : [(x : \star); (y : \star)]$ is not a valid substitution, which makes the judgment $[(x : \star); (y; \star)] \vdash \mathtt{prod}\ y\ x$ not derivable. So the term `sym` is not derivable in MCaTT. This is a safety check, since the internal language for monoidal categories is supposed to be unable to derive a witness for commutativity.

3 Correctness

We now prove that MCaTT is the internal language for monoidal weak ω-categories. Our strategy relies on the fact that CaTT is the internal language for weak ω-categories, establishes a translation between MCaTT and a restriction of the theory CaTT such that all contexts contain exactly 0-cell. This is made formal by constructing a pair of functors from the syntactic category of MCaTT to a subcategory of the syntactic category of CaTT. Although these translations require some care in order to ensure that they are properly defined, they merely formalize our initial intuition behind MCaTT and most of the proofs are technical but simple mutual inductions on the structure of the type theory. Our main contribution is to show how these translations give the expected equivalence, by analyzing their interaction with the structure of category with families of the syntactic categories on each side.

3.1 Translation for pasting schemes

We now define the operation that formalizes the combinatorial equivalence between monoidal ps-contexts and ps-contexts shifted in dimension.

Flattening of monoidal ps-contexts. Intuitively, given a monoidal ps-context $\overline{\Gamma}$ the operation we define generates a ps-context, by connecting all the components of $\overline{\Gamma}$ through a new variable. Later on, the full translation will identify all the newly introduced variables to end up in a context with only one object. In order to perform this translation, we need first to pick an inifnite sequence of fresh variable names $x_0, x_1, \ldots$ to be used (which can always be done since proofs are finite and the set of variables is supposed to be infinite countable). We then define the operation of *flattening* by mutual induction on monoidal ps-contexts and types containing only variables:

$$[]^\flat = (x_0 : \star)$$

$$\left[\overline{\Gamma}; \varnothing\right]^\flat = (\overline{\Gamma}^\flat, x_{k+1} : \star) \qquad \left[\overline{\Gamma}; (\Delta, y : A)\right]^\flat = \left(\left[\overline{\Gamma}; \Delta\right]^\flat, y : \Sigma_{k,k+1} A\right)$$

for contexts, with $k = \ell(\overline{\Gamma})$, and for types

$$\Sigma_{n,m}(\star) = x_n \underset{\star}{\to} x_m \qquad \Sigma_{n,m}\left(x \underset{A}{\to} y\right) = x \xrightarrow[\Sigma_{n,m} A]{} y$$

In order to prove that this operation respects ps-contexts, we show

Lemma 6. *The two following statements are verified*

(1) if $\overline{\Gamma} \vdash_{\mathsf{ps}}$ *then* $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}} x_{\ell(\overline{\Gamma})} : \star$

(2) if $\overline{\Gamma} \vdash_{\mathsf{ps}}$ *and* $\Delta \vdash_{\mathsf{ps}} y : A$ *then* $\left[\overline{\Gamma}; \Delta\right]^\flat \vdash_{\mathsf{ps}} y : \Sigma_{\ell(\overline{\Gamma}),\ell(\overline{\Gamma})+1} A$

Proof. These two statements are proved by mutual induction on monoidal ps-contexts and types containing only variables.

(1) We prove this statement by induction over the list structure of monoidal ps-contexts.

- First note that for the monoidal ps-context $[]$, we have $[]^\flat = (x_0 : \star)$ which satisfies indeed$(x_0 : \star) \vdash_{\mathsf{ps}} x_0 : \star$.

- Suppose that a monoidal ps-context is of the form $\left[\overline{\Gamma}; \Delta\right]$. Then necessarily, this implies $\overline{\Gamma} \vdash_{\mathsf{ps}}$ and $\Delta \vdash_{\mathsf{ps}}$, and thus there is a judgment of the form $\Delta \vdash_{\mathsf{ps}} y : \star$. By induction, this implies that $\left[\overline{\Gamma}; \Delta\right]^\flat \vdash_{\mathsf{ps}} x_{\ell(\overline{\Gamma})} \underset{\star}{\to} x_{\ell(\overline{\Gamma})+1}$. By the rule (PSD), this proves the derivability of $\left[\overline{\Gamma}; \Delta\right]^\flat \vdash_{\mathsf{ps}} x_{\ell(\overline{\Gamma})+1} : \star$.

(2) Fix a monoidal ps-context $\overline{\Gamma}$ and denote k its length, we prove this statement by induction over the structure of ps-context.

- For a ps-context of the form $(y : \star) \vdash_{\mathsf{ps}} y : \star$, we have that

$$\left[\overline{\Gamma}; (y : \star)\right]^\flat = (\overline{\Gamma}, x_{k+1} : \star, y : x_k \underset{\star}{\to} x_{k+1})$$

  and since by induction we have that $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}} x_k : \star$, this proves by application of the rule (PSE) that

$$\left[\overline{\Gamma}; (y : \star)\right]^\flat \vdash_{\mathsf{ps}} y : \Sigma_{k,k+1}\,(\star)$$

- For a ps-context of the form $(\Delta, z : A, f : y \underset{A}{\to} z) \vdash_{\mathsf{ps}} z : y \underset{A}{\to} z$ assuming that we have $\Delta \vdash y : A$, we have that

$$\left[\overline{\Gamma}; (\Delta, z : A, f : y \underset{A}{\to} z)\right]^\flat = \left(\left[\overline{\Gamma}; \Delta\right]^\flat, z : \Sigma_{k,k+1} A, f : y \xrightarrow[\Sigma_{k,k+1} A]{} z\right)$$

  and since by induction, we have $\left[\overline{\Gamma}; \Delta\right]^\flat \vdash_{\mathsf{ps}} y : \Sigma_{k,k+1} A$, it follows by application of the rule (PSE) that

$$\left[\overline{\Gamma}; (\Delta, z : A, f : y \underset{A}{\to} z)\right]^\flat \vdash_{\mathsf{ps}} f : \Sigma_{k,k+1}\left(y \underset{A}{\to} z\right)$$

- For a ps-context of the form $\Delta \vdash_{\mathsf{ps}} z : A$, assuming that we have $\Delta \vdash_{\mathsf{ps}} f : y \underset{A}{\to} z$, by induction we have that

$$\left[\overline{\Gamma}; \Delta\right]^\flat \vdash_{\mathsf{ps}} f : y \xrightarrow[\Sigma_{k,k+1} A]{} z$$

and thus by application of the rule (PSD), this proves that

$$\left[\overline{\Gamma};\Delta\right]^\flat \vdash_{\mathsf{ps}} z : \Sigma_{k,k+1}A \qquad \square$$

Proposition 7. *The falttening operation sends monoidal ps-contexts in* MCaTT *onto ps-contexts in* CaTT*:* $\overline{\Gamma} \vdash_{\mathsf{ps}}$ *implies* $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}}$.

Proof. By the previous lemma, if we have $\overline{\Gamma} \vdash_{\mathsf{ps}}$, then we also have $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}} x_{\ell(\overline{\Gamma})} : \star$. So, by application of the rule (PS), this proves that $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}}$. $\square$

Folding of a ps-context. We now define the operation that goes the opposite way: Given a ps-context in CaTT, we shift its dimension by -1, and remove all objects of dimension -1 to define a monoidal ps-context. In order to properly define this operation we introduce the notation

$$(\overline{\Gamma},(x:A)) \quad \text{is the list } \overline{\Gamma} \text{ with its last context extended with } (x:A)$$

We call this operation *folding*, and denote it with $\_^\sharp$. It is defined on contexts as follows

$$\varnothing = [] \qquad (\Gamma, x:\star) = \left[\Gamma^\sharp;\varnothing\right] \qquad (\Gamma, x:A) = (\Gamma^\sharp, x:A^-)$$
$$\left(x \underset{\star}{\to} y\right)^- = \star \qquad \left(x \underset{A}{\to} y\right)^- = x \underset{A^-}{\longrightarrow} y$$

Note that there is no case for the type $\star$, and this is intentional since it becomes in MCaTT the virtual object, that should never be referred to. This lack of translation for the type $\star$ is very important and manifests itslef throughout our comparison between the two theories.

Proposition 8. *The folding operation sends ps-contexts onto monoidal ps contexts. More precisely, we have the two following statements*

(1) If $\Gamma \vdash_{\mathsf{ps}} x : A$ *with* A *distinct from* $\star$*, then necessarily* $\Gamma^\sharp$ *is of the form* $\left[\overline{\Gamma};\Delta\right]$*, with* $\overline{\Gamma} \vdash_{\mathsf{ps}}$ *and* $\Delta \vdash_{\mathsf{ps}} x : A^-$.

(2) If $\Gamma \vdash_{\mathsf{ps}}$ *then* $\Gamma^\sharp \vdash_{\mathsf{ps}}$.

Proof. We prove these two statements by induction. The second one is a consequence of the first one and one of the induction cases of the first one uses the second one.

(1) We prove this result by induction on the structure of ps-context.

- Here we prevent the type A to be $\star$, so the base case is actually the case of the context of a derivation of the form

$$(\Gamma, y : \star, f : x \underset{\star}{\to} y) \vdash_{\mathsf{ps}} f : x \underset{\star}{\to} y$$

  assuming that $\Gamma \vdash_{\mathsf{ps}} x : \star$. Then by the rule (PS), we also have $\Gamma \vdash_{\mathsf{ps}}$, and thus by induction $\Gamma^\sharp \vdash_{\mathsf{ps}}$. Moreover, we also have that $(f : \star) \vdash_{\mathsf{ps}} f : \star$, hence the expected result

$$(\Gamma, y : \star, f : x \underset{\star}{\to} y)^\sharp = \left[\Gamma^\sharp; (f : \star)\right] \quad \text{with} \begin{cases} \overline{\Gamma} \vdash_{\mathsf{ps}} \\ f : \star \vdash_{\mathsf{ps}} f : \star \end{cases}$$

- In the case of a derivation of the form

$$(\Gamma, y : A, f : x \underset{A}{\to} y) \vdash_{\mathsf{ps}} f : x \underset{A}{\to} y$$

  assuming $\Gamma \vdash_{\mathsf{ps}} x : A$ and A distinct from $\star$, we have by induction that $\Gamma^\sharp$ is of the form $\left[\overline{\Gamma}; \Delta\right]$ with $\overline{\Gamma} \vdash_{\mathsf{ps}}$ and $\Delta \vdash_{\mathsf{ps}} x : A^-$. By application of the rule (PSE) for Δ, we have that

$$(\Delta, y : A^-, f : x \underset{A^-}{\longrightarrow} y) \vdash_{\mathsf{ps}} f : x \underset{A^-}{\longrightarrow} y$$

This proves that

$$(\Gamma, y : A, f : x \underset{A}{\to} y)^\sharp = \left[\overline{\Gamma}; (\Delta, y : A^-, f : x \underset{A^-}{\longrightarrow} y)\right]$$
$$\text{with} \begin{cases} \overline{\Gamma} \vdash_{\mathsf{ps}} \\ (\Delta, y : A^-, f : x \underset{A^-}{\longrightarrow} y) \vdash_{\mathsf{ps}} f : x \underset{A^-}{\longrightarrow} y \end{cases}$$

- In the case of a derivation of the form $\Gamma \vdash_{\mathsf{ps}} y : A$, assuming $\Gamma \vdash_{\mathsf{ps}} f : x \underset{A}{\to} y$ with A distinct of the type $\star$, we have by induction that $\Gamma^\sharp$ is of the form $\left[\overline{\Gamma}, \Delta\right]$ with $\overline{\Gamma} \vdash_{\mathsf{ps}}$ and $\Delta \vdash_{\mathsf{ps}} f : x \underset{A^-}{\longrightarrow} y$. By applying the rule (PSD) to Δ, this proves that $\Delta \vdash_{\mathsf{ps}} y : A^-$, and thus

$$\Gamma^\sharp = \left[\overline{\Gamma}; \Delta\right] \qquad \text{with} \begin{cases} \overline{\Gamma} \vdash_{\mathsf{ps}} \\ \Delta \vdash_{\mathsf{ps}} y : A^- \end{cases}$$

(2) First note that if Γ is the ps-context obtained by a unique point: $\Gamma = (x : \star)$, then we have $\Gamma^\sharp = []$, and hence $\Gamma^\sharp \vdash_{\mathsf{ps}}$. So it suffices to check this property for a ps-context Γ which is not just one point. For such a ps-context the judgment $\Gamma \vdash_{\mathsf{ps}}$ necessarily comes from a judgment $\Gamma \vdash_{\mathsf{ps}} y : \star$, which in turns has to come from a judgment of the form $\Gamma \vdash_{\mathsf{ps}} f : y \underset{\star}{\to} z$. By the previous lemma, this proves that $\Gamma^\sharp$ is of the form $\left[\overline{\Gamma}; \Delta\right]$, with $\overline{\Gamma} \vdash_{\mathsf{ps}}$ and $\Delta \vdash_{\mathsf{ps}} f : \left(y \underset{\star}{\to} z\right)^- = \star$. By application of the rule (PS) on Δ, this proves that $\Delta \vdash_{\mathsf{ps}}$, and thus

$$\Gamma^\sharp = \left[\overline{\Gamma}; \Delta\right] \vdash_{\mathsf{ps}} \qquad \square$$

Note that in fact only the second statement is of interest for us, the first one should be seen as a auxialiary lemma that is mutually inductive with it and allows for a proof.

Interaction of flattening and folding. As the notations suggest, the flattening and folding operations are inverse to one another. Although this is not

true on the nose in one of the directions, due to the fact that the flattening introduces fresh variables, it is holds up to renaming of those variables. In order to prove this, we first prove the following lemma concerning the action on types with only variables.

Lemma 9. *The transformations that we have introduced on types cancel each other.*

(1) For any type A with only variables, we have that $(\Sigma A)^- = A$

(2) For any type A with only variables and distinct from the the type $\star$, $\Sigma\,(A^-) = A$ up to renaming of the variables of type $\star$

Proof. These statements are both proved by induction on the structure of the types

(1) For the type $\star$, we have that

$$(\Sigma\star)^- = \left(x_n \underset{\star}{\to} x_m\right)^- = \star$$

And for a type of the form $x \underset{A}{\to} y$, we have that

$$\left(\Sigma\left(x \underset{A}{\to} y\right)\right)^- = x \xrightarrow[(\Sigma A)^-]{} y$$

By induction this proves the equality.

(2) For a type of the form $x \underset{\star}{\to} y$, we have that

$$\Sigma\left(\left(x \underset{\star}{\to} y\right)^-\right) = \Sigma\star = x_n \underset{\star}{\to} x_m$$

And for a type of the form $x \underset{A}{\to} y$, we have that

$$\Sigma\left(\left(x \underset{A}{\to} y\right)\right)^- = x \xrightarrow[\Sigma(A^-)]{} y$$

By induction this proves the equality up to renaming of the variables of type $\star$.

□

Proposition 10. *The flattening and folding operations are inverse to one another*

(1) If $\overline{\Gamma} \vdash_{\mathsf{ps}}$, *then* $\left(\overline{\Gamma}^{\flat}\right)^{\sharp} = \overline{\Gamma}$

(2) If $\Gamma \vdash_{\mathsf{ps}}$, *then* $\left(\Gamma^{\sharp}\right)^{\flat} = \Gamma$ *up to renaming of the variables of type* $\star$

Proof. These two statements are both proved by induction

(1) We prove this by induction on the structure of lists of contexts

- For the monoidal ps-context $[]$, we have that

$$\left([]^{\flat}\right)^{\sharp} = (x_0 : \star)^{\sharp} = []$$

- For a list of contexts of the form $\left[\overline{\Gamma}; \varnothing\right]$, we have

$$\left(\left[\overline{\Gamma}; \varnothing\right]^{\flat}\right)^{\sharp} = \left(\overline{\Gamma}^{\flat}, x_{\ell(\overline{\Gamma})} : \star\right)^{\sharp} = \left(\overline{\Gamma}^{\flat}\right)^{\sharp} = \overline{\Gamma}$$

- For a list of context of the form $\left[\overline{\Gamma}; (\Delta, y : A)\right]$, we have

$$\begin{aligned}\left(\left[\overline{\Gamma}; (\Delta, y : A)\right]^{\flat}\right)^{\sharp} &= \left(\left[\overline{\Gamma}; \Delta\right]^{\flat}, y : \Sigma A\right)^{\sharp} \\ &= \left(\left(\left[\overline{\Gamma}; \Delta\right]^{\flat}\right)^{\sharp}, y : (\Sigma A)^{-}\right) \\ &= \left(\left[\overline{\Gamma}; \Delta\right], y : A\right) \\ &= \left[\overline{\Gamma}; (\Delta, y : A)\right]\end{aligned}$$

the penultimate step is obtained by induction and applying the previous lemma.

(2) We prove this statement by induction on the structure of context

- Since a ps-context is never empty, we do the induction case on the context $(y : \star)$. We then have

$$\left((y : \star)^\sharp\right)^\flat = []^\flat = (x_0 : \star)$$

So the two contexts are indeed equal up to the renaming of the variable.

- For a context of the form $(\Gamma, y : \star)$, we have that

$$\left((\Gamma, y : \star)^\sharp\right)^\flat = \left[\Gamma^\sharp; \varnothing\right]^\flat = \left(\left(\Gamma^\sharp\right)^\flat, x_{\ell(\Gamma^\sharp)+1} : \star\right)$$

By induction, this is equal to $(\Gamma, y : \star)$ up to renaming of the variables of type $\star$.

- For a context of the form $(\Gamma, y : A)$, we have that

$$\left((\Gamma, y : A)^\sharp\right)^\flat = \left[(\Gamma^\sharp, y : A^-)\right]^\flat = \left(\left(\Gamma^\sharp\right)^\flat, y : \Sigma\left(A^-\right)\right)$$

By induction and by the previous lemma, this is equal to $(\Gamma, y : A)$ up to renaming of the variables of type $\star$. □

3.2 Translations for the structure in pasting schemes

We have defined a pair of translations back and forth between the monoidal ps-contexts and the ps-contexts, proving that they are equivalent. We now show that these translations extend to the types, terms and substitutions associated to those.

Flattening. In order to be consistent with the shift in dimension occurring while flattening a monoidal ps-context, we define the flattening to shift dimensions of types, terms and substitutions as well. This is done by replacing the type $\star$ by an arrow and propagating this change by induction, to types, terms and substitutions. We again use a family of fresh variables $x_0, x_1, \ldots$, and al-

ways replace the type $\star$ by an arrow between two of these fresh variables. In order to know which of these variables should be used, we carry indices.

$$\star^\flat_{n,m} = x_n \to x_m \qquad \left(t \underset{A}{\longrightarrow} u\right)^\flat_{n,m} = t^\flat_{n,m} \underset{A^\flat_{n,m}}{\longrightarrow} u^\flat_{n,m}$$

$$(x)^\flat_{n,m} = x \qquad \left(\mathsf{coh}_{\overline{\Gamma},A}[\overline{\sigma}]\right)^\flat_{n,m} = \mathsf{coh}_{\overline{\Gamma}^\flat, A^\flat_{0,\ell(\overline{\Gamma})}}\left(\overline{\sigma}^\flat_{n,m}\right)$$

$$[]^\flat_{n,n} = \langle x_n : \star \rangle \qquad [\overline{\sigma}; \langle\rangle]^\flat_{n,m} = \langle \overline{\sigma}^\flat_{n,i}, x_m \rangle$$
$$[\overline{\sigma}; \langle \tau, t\rangle]^\flat_{n,m} = \langle [\overline{\sigma}; \tau]^\flat_{n,m}, t^\flat_{j,m} \rangle$$

Where in the last case, i is the minimal index such that no variable in $\overline{\sigma}$ is contained in a ps-context in position after i in $\overline{\Gamma}$, and j is the maximal index such that no variable of t is contained in a ps-context in position before $j-1$ in $\overline{\Gamma}$.

Proposition 11. *The flattening sends derivable judgments in* MCaTT *onto derivable judgments in* CaTT*, more precisely, chose n the maximal integer such that no variable in* $\mathrm{Var}(A)$ *(resp.* $\mathrm{Var}(t)$*,* $\mathrm{Var}(\sigma)$*) appears in a context in position before* $(n-1)$ *in the list* $\overline{\Gamma}$*, and similarly, chose m the minimal integer such that no variable in* $\mathrm{Var}(A)$ *(resp.* $\mathrm{Var}(t)$*,* $\mathrm{Var}(\sigma)$*) appears in a context in position after m before.*

- *If* $\overline{\Gamma} \vdash A$ *then* $\overline{\Gamma}^\flat \vdash A^\flat_{n,m}$*, (if* $A = \star$*, for all* $i < j \leq \ell(\overline{\Gamma})$*, we have* $\overline{\Gamma}^\flat \vdash \star^\flat_{i,j}$*)*
- *If* $\overline{\Gamma} \vdash t : A$ *then* $\overline{\Gamma}^\flat \vdash t^\flat_{n,m} : A^\flat_{n,m}$
- *If* $\overline{\Gamma} \vdash_{\mathsf{eq}} A$ *then* $\overline{\Gamma}^\flat \vdash_{\mathsf{eq}} A^\flat_{0,\ell(\overline{\Gamma})}$
- *If* $\overline{\Gamma} \vdash_{\mathsf{op}} A$ *then* $\overline{\Gamma}^\flat \vdash_{\mathsf{op}} A^\flat_{0,\ell(\overline{\Gamma})}$
- *If* $\overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta}$ *then* $\overline{\Gamma}^\flat \vdash \overline{\sigma}^\flat_{n,m} : \overline{\Delta}^\flat$ *(if* $\overline{\sigma}$ *is of the form* $[\overline{\sigma}, \langle\rangle]$*, then there is no condition on m)*

Proof. This proof is done on mutual induction on the strucutres that we study

- Given a type A:

– If A is the type $\star$, and for any indices $i < j < \ell(\overline{\Gamma})$, the associations $x_i : \star$ and $x_j : \star$ appear in $\overline{\Gamma}^\flat$. Since by proposition 7, $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}}$, this shows that $\overline{\Gamma} \vdash x_i : \star$ and $\overline{\Gamma} \vdash x_j : \star$, hence $\overline{\Gamma} \vdash x_i \to x_j$.

– If A is of the form $t \underset{A}{\longrightarrow} u$, then the indices n, m necessarily satisfy the conditions for applying induction for the type A, and for the terms u and v, thus by induction, we have that $\overline{\Gamma}^\flat \vdash A^\flat_{n,m}$, $\overline{\Gamma}^\flat \vdash t^\flat_{n,m} : A^\flat_{n,m}$ and $\overline{\Gamma}^\flat \vdash u^\flat_{n,m} : A^\flat_{n,m}$, so this proves that

$$\overline{\Gamma}^\flat \vdash t^\flat_{n,m} \underset{A^\flat_{n,m}}{\longrightarrow} u^\flat_{n,m}$$

- Given a term $\overline{\Gamma} \vdash t : A$:

  – Either $t = y$ is a variable, and in this case, A contains only variables, thus $A^\flat_{n,m} = \Sigma_{n,m} A$. Moreover, if the pairing $y : A$ appears in the ps-contexts in position k in $\overline{\Gamma}$, then $n = k, m = k+1$ and the pairing $y : \Sigma_{k,k+1} A$ appears in $\overline{\Gamma}^\flat$. Since by proposition 7, $\overline{\Gamma}^\flat \vdash_{\mathsf{ps}}$, it follows that

  $$\overline{\Gamma}^\flat \vdash y : A^\flat_{k,k+1}$$

  – Or t is of the form $\mathsf{coh}_{\overline{\Delta},B}(\overline{\sigma})$. In this case, by induction $\overline{\Gamma}^\flat \vdash \overline{\sigma}_{n,m} : \overline{\Delta}$, and moreover, either $\overline{\Delta} \vdash_{\mathsf{op}} B$, in which case we have $\overline{\Delta}^\flat \vdash_{\mathsf{op}} B^\flat_{0,\ell(\overline{\Delta})}$, or $\overline{\Delta} \vdash_{\mathsf{eq}} B$, in which case we have $\overline{\Delta}^\flat \vdash_{\mathsf{eq}} B^\flat_{0,\ell(\overline{\Delta})}$. In both cases, this shows that

  $$\overline{\Gamma}^\flat \vdash \mathsf{coh}_{\overline{\Delta}^\flat, B^\flat_{0,\ell(\overline{\Delta})}}(\overline{\sigma}^\flat_{n,m}) : B^\flat_{0,\ell(\overline{\Delta})}[\overline{\sigma}^\flat_{n,m}]$$

  Moreover, by lemma 12 $B^\flat_{0,\ell(\overline{\Delta})}[\overline{\sigma}^\flat_{n,m}] = (B[\overline{\sigma}])^\flat_{n,m}$. This proves that

  $$\overline{\Gamma}^\flat \vdash \mathsf{coh}_{\overline{\Delta}^\flat, B^\flat_{0,\ell(\overline{\Delta})}}(\overline{\sigma}^\flat_{n,m}) : A^\flat_{n,m}$$

- For a derivable judgment of the form $\overline{\Gamma} \vdash_{\mathsf{eq}} A$, necessarily, we have that $\overline{\Gamma} \vdash A$ and $\mathrm{Var}(\Gamma) = \mathrm{Var}(A)$, hence by induction $\overline{\Gamma}^\flat \vdash A^\flat_{0,\ell(\overline{\Gamma})}$. Moreover,

by lemma 12

$$\mathrm{Var}(\overline{\Gamma}^\flat) = \mathrm{Var}(A^\flat)$$

This proves the derivability of the following judgment

$$\overline{\Gamma}^\flat \vdash_{\mathsf{eq}} A^\flat_{0,\ell(\overline{\Gamma})}$$

- For a derivable judgment of the form $\overline{\Gamma} \vdash_{\mathsf{op}} A$, if $\overline{\Gamma}$ is not of dimension 0, we necessarily have $A = t \underset{B}{\longrightarrow} u$, with $\overline{\Gamma} \vdash_{\mathsf{ps}}$, $\partial^-(\overline{\Gamma}) \vdash t : B$ and $\mathrm{Var}(\partial^-(\overline{\Gamma})) = \mathrm{Var}(t : B)$. By induction, and by the variable conditions, we have that $\partial^-(\overline{\Gamma})^\flat \vdash t^\flat_{0,\ell(\partial^-(\overline{\Gamma}))} : A^\flat_{0,\ell(\partial^-(\overline{\Gamma}))}$. Moreover, by the lemma 12, $\partial^-(\overline{\Gamma})^\flat = \partial^-(\overline{\Gamma}^\flat)$, moreover by lemma 5, $\ell(\partial^-(\overline{\Gamma})) = \ell(\overline{\Gamma})$, thus $\partial^-(\overline{\Gamma}^\flat) \vdash t^\flat_{0,\ell(\overline{\Gamma})} : A^\flat_{0,\ell(\overline{\Gamma})}$, finally, again by lemma 12

$$\mathrm{Var}(\partial^-(\overline{\Gamma})^\flat) = \mathrm{Var}(t^\flat_{0,\ell(\overline{\Gamma})} : \mathrm{Var}(A^\flat_{0,\ell(\overline{\Gamma})}))$$

The same argument holds for the judgments regarding the target, thus

$$\overline{\Gamma}^\flat \vdash_{\mathsf{op}} t^\flat_{0,\ell(\overline{\Gamma})} : A^\flat_{0,\ell(\overline{\Gamma})}$$

In the case where $\overline{\Gamma}$ is of dimension 0, then it is of the form

$$\overline{\Gamma} = [(f_1 : \star); \cdots ; (f_k : \star)]$$

So $\overline{\Gamma}^\flat$ can be computed explicitly as

$$\overline{\Gamma}^\flat = (x_0 : \star, x_1 : \star, f_1 : x_0 \to x_1, x_2 : \star, \cdots , x_k : \star, f_k : x_{k-1} \to x_k)$$

And we have $\star^\flat_{0,k} = x_0 \to x_k$, so we can check explicitly that

$$\overline{\Gamma}^\flat \vdash_{\mathsf{op}} \star^\flat_{0,k}$$

- If $\overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta}$ is a substitution, then there are three cases to solve.

  - If $\overline{\sigma}$ is the empty substitution, for any index $n < \ell(\overline{\Gamma})$, we have that $\overline{\Gamma}^\flat \vdash x_n : \star$, and $[]^\flat_{n,n} = \langle x_n \rangle$ hence

$$\overline{\Gamma}^\flat \vdash []^\flat_{n,n} : []^\flat$$

  - If $\overline{\sigma}$ is of the form $\overline{\Gamma} \vdash [\overline{\sigma}; \langle\rangle] : \left[\overline{\Delta}; \varnothing\right]$, then the indices n, i satisfies the induction condition, and by induction we have that $\overline{\Gamma}^\flat \vdash \overline{\sigma}^\flat_{n,i} : \overline{\Delta}^\flat$, and $\overline{\Gamma}^\flat \vdash x_m : (y : \star)$, hence the derivation

$$\overline{\Gamma}^\flat \vdash \langle \overline{\sigma}^\flat_{n,i}, x_m \rangle : \overline{\Delta}^\flat$$

  - If $\overline{\sigma}$ is of the form $\overline{\Gamma} \vdash [\overline{\sigma}; \langle \sigma', t \rangle] : \left[\overline{\Delta}; (\Delta', y : A)\right]$, then

$$[\overline{\sigma}; \langle \sigma', t \rangle]^\flat_{n,m} = \langle [\overline{\sigma}; \sigma']^\flat_{n,m}, t^\flat_{j,m} \rangle$$

    all the indices satisfy the induction conditions, an by induction, we have the derivability of the two following judgments

$$\overline{\Gamma}^\flat \vdash [\overline{\sigma}; \tau]^\flat_{n,m} : \left[\overline{\Delta}; \Delta'\right]^\flat$$
$$\overline{\Gamma}^\flat \vdash t^\flat_{j,m} : (A\,[\overline{\sigma}; \sigma'])^\flat_{j,m}$$

    Since the lemma 12 also shows, $(A\,[\overline{\sigma}; \sigma'])^\flat_{j,m} = A^\flat_{\ell(\overline{\Delta})-1,\ell(\overline{\Delta})}\left[[\overline{\sigma}; \sigma']^\flat_{j,m}\right]$, this proves

$$\overline{\Gamma}^\flat \vdash [\overline{\sigma}; \langle \sigma', t \rangle]^\flat : \left[\overline{\Delta}; (\Delta', y : A)\right]^\flat \qquad \square$$

From now on, for the sake of readability, and since they usually can be inferred, we will often omit the indices while working with the flattening. Along the proof of previous proposition, we have used the following properties, which should be proved during the induction, and are simple enough to be left to the reader:

Lemma 12. *The flattening is compatible with substitution application, source*

and target in dimension non 0*, and respect the equality of set of variables.*

(1) If A *is a type and* σ *is a substitution, then* $(A[\sigma])^\flat = A^\flat[\sigma^\flat]$

(2) If $\overline{\Gamma}$ *is a ps-context of dimension non* 0*, then* $\left(\partial^-(\overline{\Gamma})\right)^\flat = \partial^-(\overline{\Gamma}^\flat)$ *and* $\left(\partial^+(\overline{\Gamma})\right)^\flat = \partial^+(\overline{\Gamma}^\flat)$

(3) If $\mathrm{Var}(\overline{\Gamma}) = \mathrm{Var}(A)$ *then* $\mathrm{Var}(\overline{\Gamma}^\flat) = \mathrm{Var}(A^\flat)$*, if* $\mathrm{Var}(\overline{\Gamma}) = \mathrm{Var}(t : A)$*, then* $\mathrm{Var}(\overline{\Gamma}^\flat) = \mathrm{Var}(t^\flat : A^\flat)$

Notations. In order to simplify the notation, we consider that the translation $\_^\flat$ acts directly on the judgment of the theory, and we write

$$\left(\overline{\Gamma} \vdash_{\mathsf{ps}}\right)^\flat = \overline{\Gamma}^\flat \vdash_{\mathsf{ps}}$$
$$\left(\overline{\Gamma} \vdash A\right)^\flat = \overline{\Gamma}^\flat \vdash A^\flat$$
$$\left(\overline{\Gamma} \vdash t : A\right)^\flat = \overline{\Gamma}^\flat \vdash t^\flat : A^\flat$$
$$\left(\overline{\Gamma} \vdash \overline{\sigma} : \overline{\Delta}\right)^\flat = \overline{\Gamma}^\flat \vdash \overline{\sigma}^\flat : \overline{\Delta}^\flat$$

where the indices that have been left implicit are the ones used in the proposition 11. We refer to these four judgments as *monoidal ps-judgments.*

Folding. Similarly, the folding operation that we have defined on ps-context extends to types and terms in ps-contexts, along with substitutions between ps-contexts. In order to treat the case for substitution, we need the following notation.

$\langle \overline{\sigma}, t \rangle$ is the list $\overline{\sigma}$ with its last substitution extended with t

This allows to define the folding as a partial translation by mutual induction as follows:

$$\left(t \underset{\star}{\to} u\right)^\sharp = \star \qquad \left(t \underset{\star}{\to} u\right)^\sharp = t^\sharp \underset{A^\sharp}{\longrightarrow} u^\sharp$$

$$x^\sharp = x \qquad \mathsf{coh}_{\Gamma,A}[\sigma]^\sharp = \mathsf{coh}_{\Gamma^\sharp,A^\sharp}[\sigma^\sharp]$$

$$\langle t\rangle^\sharp = [] \qquad \langle \sigma, t\rangle^\sharp = \sigma^\sharp \qquad \langle \sigma, t\rangle^\sharp = \langle \sigma^\sharp, t^\sharp\rangle$$

Note that $\star^\sharp$ is not defined.

Proposition 13. *The folding operation sends derivable judgments in* CaTT *onto derivable judgments in* MCaTT*, more precisely,*

- *If* $\Gamma \vdash A$ *and* A *is not the type* $\star$ *then* $\Gamma^\sharp \vdash A^\sharp$
- *If* $\Gamma \vdash t : A$ *then* $\Gamma^\sharp \vdash t^\sharp : A^\sharp$
- *If* $\Gamma \vdash_{\mathsf{eq}} A$ *then* $\Gamma^\sharp \vdash_{\mathsf{eq}} A^\sharp$
- *If* $\Gamma \vdash_{\mathsf{op}} A$ *then* $\Gamma^\sharp \vdash_{\mathsf{op}} A^\sharp$
- *If* $\Gamma \vdash \sigma : \Delta$ *then* $\Gamma^\sharp \vdash \sigma^\sharp : \Delta^\sharp$

Proof. Again, this is proved by mutual induction on the structure. The case for substitution being very involved because of a double layered dependancy, we just give a sketch of the proof.

- For a type $\Gamma \vdash A$, with A distinct from $\star$:
  - If A is of the form $t \underset{\star}{\to} u$, then $A^\sharp = \star$, and by proposition 8, we have that $\Gamma^\sharp \vdash_{\mathsf{ps}}$, hence $\Gamma^\sharp \vdash \star$.
  - If A is on the form $t \underset{B}{\longrightarrow} u$, with $B \neq \star$, then necessarily we have $\Gamma \vdash B$, $\Gamma \vdash t : B$ and $\Gamma \vdash u : B$, and thus by induction cases on types and terms, we have $\Gamma^\sharp \vdash B^\sharp$, $\Gamma^\sharp \vdash t^\sharp : B^\sharp$ and $\Gamma^\sharp \vdash u^\sharp : B^\sharp$. This proves $\Gamma^\sharp \vdash A^\sharp$.
- For a term $\Gamma \vdash t : A$, with $A \neq \star$:

– If $t = x$ is a variable, then since Γ is a ps-context, necessarily A contains only variables, and the pairing $x : A$ appears in Γ. By definition, the pairing $x : A^-$ appears in $\Gamma^\sharp$, and thus $\Gamma^\sharp \vdash x : A^-$. Since A contains only variables, we have that $A^- = A^\sharp$, and thus $\Gamma^\sharp \vdash A^\sharp$.

– If $t = \mathsf{coh}_{\Delta,B}(\sigma)$ is a coherence, then $t^\sharp = \mathsf{coh}_{\Delta^\sharp,B^\sharp}(\sigma^\sharp)$. We necessarily have $\Gamma \vdash \sigma : \Delta$, and by induction, $\Gamma^\sharp \vdash \sigma^\sharp : \Delta^\sharp$. Moreover, either $\Delta \vdash_{\mathsf{eq}} B$, in which case the induction gives $\Delta^\sharp \vdash_{\mathsf{eq}} B^\sharp$, or $\Delta \vdash_{\mathsf{op}} B$, in which case induction gives $\Delta^\sharp \vdash_{\mathsf{eq}} B^\sharp$. In both cases, this proves that $\Gamma^\sharp \vdash t^\sharp : B^\sharp[\sigma^\sharp]$, and since by lemma 14, $B^\sharp[\sigma^\sharp] = (B[\sigma])^\sharp$, this proves $\Gamma^\sharp \vdash t^\sharp : A^\sharp$.

- For a judgment of the form $\Gamma \vdash_{\mathsf{eq}} A$, we have $\Gamma \vdash A$ and $\mathrm{Var}(\Gamma) = \mathrm{Var}(A)$, so by induction, we also have $\Gamma^\sharp \vdash A^\sharp$. Moreover, we have by lemma 14, $\mathrm{Var}(\Gamma^\sharp) = \mathrm{Var}(A^\sharp)$ and thus $\Gamma^\sharp \vdash_{\mathsf{eq}} A^\sharp$.

- For a judgment of the form $\Gamma \vdash_{\mathsf{op}} A$, necessarily we have $A = t \underset{B}{\longrightarrow} u$, with $\partial^-(\Gamma) \vdash t : B$ and $\partial^+(\Gamma) \vdash u : B$, and $\mathrm{Var}(\partial^-(\Gamma)) = \mathrm{Var}(t : B)$, $\mathrm{Var}(\partial^+(\Gamma)) = \mathrm{Var}(u : B)$.

  – If B is the type $\star$, then $A^\sharp = \star$ and t and u are necessarily variables. The variable conditions then imply that $\partial^-(\Gamma)$ and $\partial^+(\Gamma)$ are just one point. This is only possible when Γ is of dimension 1, i.e., $\Gamma^\sharp$ is a list of containing all contexts isomorphic to $\mathcal{D}^0$, which proves that $\Gamma^\sharp \vdash_{\mathsf{op}} \star$.

  – If B is not the type $\star$, then $A^\sharp = t^\sharp \underset{B^\sharp}{\longrightarrow} u^\sharp$. By induction we have $(\partial^-(\Gamma))^\sharp \vdash t^\sharp : A^\sharp$, with, by lemma 14, $\mathrm{Var}(\partial^-(\Gamma)^\sharp) = \mathrm{Var}(t^\sharp : A^\sharp)$. Again by lemma 14, we have $(\partial^-(\Gamma))^\sharp = \partial^-(\Gamma^\sharp)$, hence,

$$\partial^-(\Gamma^\sharp) \vdash t^\sharp : A^\sharp$$

$$\mathrm{Var}(\partial^-(\Gamma^\sharp)) = \mathrm{Var}(t^\sharp : A^\sharp)$$

The demonstration is symmetric for the case of $\partial^+(\Gamma)$, hence this proves that $\Gamma^\sharp \vdash_{\mathsf{op}} A^\sharp$

- For substitutions :
  - In the case of the substitution $\Gamma \vdash \langle x \rangle : \mathcal{D}^0$, then $\sigma^\sharp = []$, and since $\Gamma^\sharp \vdash$, it follows that $\Gamma^\sharp \vdash [] : []$
  - For a substitution of the form $\Gamma \vdash \langle \sigma, t, u \rangle : (\Delta, y : \star, f : x \to y)$, we have $\langle \sigma, t, u \rangle^\sharp = \left[\sigma^\sharp; \langle u^\sharp \rangle\right]$. Necessarily, t is a variable in Γ, and thus this provides a split $\Gamma^\sharp = \overline{\Gamma}_1 @ \overline{\Gamma}_2$, with all variables in $\overline{\Gamma}_1$ appearing before u and Γ and all variables in $\overline{\Gamma}_2$ appearing after u in Γ. By induction, we have $\Gamma^\sharp \vdash \sigma^\sharp : \Delta^\sharp$. Moreover $\sigma^\sharp$ contains only variables in $\overline{\Gamma}_1$, thus in fact $\overline{\Gamma}_1 \vdash \sigma^\sharp : \Delta^\sharp$. Similarly, because of the way u uses the variables, $\overline{\Gamma}_2 \vdash u^\sharp : \star$. This justifies the derivability of the substitution $\overline{\Gamma}_2 \vdash \langle u \rangle : (f : \star)$, and hence, this proves that

$$\overline{\Gamma}_1 @ \overline{\Gamma}_2 \vdash \left[\sigma, \langle u^\sharp \rangle\right] : [\Delta; (f : \star)]$$

  - In the case of a substitution $\Gamma \vdash \langle \sigma, t, u \rangle : (\Delta, y : A, f : x \to y)$, we have that $\langle \sigma, t, u \rangle^\sharp = \langle \sigma^\sharp, t^\sharp, u^\sharp \rangle$. By induction, $\Delta^\sharp = \left[\overline{\Delta}_1; \Delta'\right]$, $\sigma^\sharp = [\overline{\sigma}; \sigma']$ and we have a split of the form $\Gamma^\sharp = \overline{\Gamma}_1 @ \overline{\Gamma}_2$, such that $\overline{\Gamma}_1 @ \overline{\Gamma}_2 \vdash \langle \overline{\sigma}_1; \sigma' \rangle : \left[\overline{\Delta}'_1; \Delta'\right]$, with $\overline{\Gamma}_1 \vdash \overline{\sigma}_1 : \overline{\Delta}_1$ and $\overline{\Gamma}_2 \vdash \sigma' : \Delta'$. Moreover, by applying the induction twice, we have $\overline{\Gamma}_2 \vdash t^\sharp : A^\sharp[\sigma']$ and $\overline{\Gamma}_2 \vdash u : A^\sharp[\langle \sigma', t^\sharp \rangle]$, thus $\overline{\Gamma}_2 \vdash \langle \sigma', t^\sharp, u^\sharp \rangle : (\Delta, y : A^\sharp, f : x \to y)$. This justifies that

$$\overline{\Gamma}_1 @ \overline{\Gamma}_2 \vdash \langle \overline{\sigma}_1; \langle \sigma', t^\sharp, u^\sharp \rangle \rangle : \left[\overline{\Delta}_1; (\Delta'; y : A^\sharp; f : x \to y)\right]$$

□

Similarly to the case of the flattening, we have used the following properties, which should be proved during the induction, but are simple enough to be left

out

Lemma 14. *The folding operation respects application of substitutions, source and target of ps-contexts and preserves equality of variables*

(1) If A is a type distinct from $\star$ and σ is a substitution, $(A[\sigma])^\sharp = A^\sharp[\sigma^\sharp]$

(2) If Γ is a ps-context, $(\partial^-(\Gamma))^\sharp = \partial^-(\Gamma^\sharp)$ and $(\partial^+(\Gamma))^\sharp = \partial^+(\Gamma^\sharp)$

(3) If $\mathrm{Var}(\Gamma) = \mathrm{Var}(A)$, then $\mathrm{Var}(\Gamma^\sharp) = \mathrm{Var}(A^\sharp)$, and if $\mathrm{Var}(\Gamma) = \mathrm{Var}(t : A)$, then $\mathrm{Var}(\Gamma^\sharp) = \mathrm{Var}(t^\sharp : A^\sharp)$

Notations. Similarly, we consider $\_^\sharp$ as acting on judgments on the theory and denote

$$(\Gamma \vdash_{\mathsf{ps}})^\sharp = \Gamma^\sharp \vdash_{\mathsf{ps}}$$
$$(\Gamma \vdash A)^\sharp = \Gamma^\sharp \vdash A^\sharp$$
$$(\Gamma \vdash t : A)^\sharp = \Gamma^\sharp \vdash t^\sharp : A^\sharp$$
$$(\Gamma \vdash \sigma : \Delta)^\sharp = \Gamma^\sharp \vdash \sigma^\sharp : \Delta^\sharp$$

These four judgments be called *ps-judgments* in the theory CaTT. In this case they are no different from usual judgments between ps-contexts, but they play an important role in their interaction with the theory MCaTT.

Interaction between flattening and folding. In proposition 10, we have proved that the flattening and the folding were inverse of each other on ps-contexts, we now show that this extends to all ps-judgments, i.e., these operations are inverse on terms, types and substitutions in ps-contexts.

Proposition 15. *For any monoidal ps-judgment J in MCaTT,*

$$(J^\flat)^\sharp = J$$

For any ps-judgment J in CaTT*,*

$$(J^\sharp)^\flat \simeq J \qquad \textit{up to renaming of the variables of type } \star.$$

Proof. Note that we have already proved the equality in the case of a judgment of the form $\Gamma \vdash_{\mathsf{ps}}$. The other cases are done by mutual induction on the structure.

- For types:
  - In the case of the type $\star$, we have that
  $$\left((\star)^\flat_{n,m}\right)^\sharp = \left(x_n \underset{\star}{\to} x_m\right)^\sharp = \star$$
  - In the case of the type $t \underset{A}{\longrightarrow} u$, we have
  $$\left(\left(\overline{\Gamma} \vdash t \underset{A}{\longrightarrow} u\right)^\flat\right)^\sharp = (t^\flat)^\sharp \xrightarrow[(A^\flat)^\sharp]{} (u^\flat)^\sharp$$
  and by induction $\left(A^\flat\right)^\sharp = A$, $\left(t^\flat\right)^\sharp = t$ and $\left(u^\flat\right)^\sharp = u$, which proves that
  $$\left(\left(t \underset{A}{\longrightarrow} u\right)^\flat\right)^\sharp = t \underset{A}{\longrightarrow} u$$
- For terms :
  - In the case of a variable, we have
  $$\left(x^\flat\right)^\sharp = x^\sharp = x$$
  - In the case of a coherence, we have
  $$\left(\left(\mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})\right)^\flat\right)^\sharp = \left(\mathsf{coh}_{\overline{\Gamma}^\flat,A^\flat}(\overline{\sigma}^\flat)\right)^\sharp = \mathsf{coh}_{\left(\overline{\Gamma}^\flat\right)^\sharp,\left(A^\flat\right)^\sharp}\left(\left(\overline{\sigma}^\flat\right)^\sharp\right)$$

By induction, $\left(\overline{\Gamma}^{\flat}\right)^{\sharp} = \overline{\Gamma}$, $\left(A^{\flat}\right)^{\sharp} = A$ and $\left(\overline{\sigma}^{\flat}\right)^{\sharp} = \overline{\sigma}$, hence

$$\left(\left(\mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})\right)^{\flat}\right)^{\sharp} = \mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})$$

- For substitutions :
  - In the case of the empty list, we have

$$\left(\left[\right]_{n,m}^{\flat}\right)^{\sharp} = \langle x_n \rangle^{\sharp} = \left[\right]$$

  - In the case of a non empty list, notice that the none of the component of the list can be empty since this is not allowed for substitutions in monoidal ps-contexts. In the case where the last substitution of the list is of lentgh 1, we have

$$\left([\overline{\sigma}; \langle t \rangle]^{\flat}\right)^{\sharp} = \langle \overline{\sigma}^{\flat}, x_{k+1}, t^{\flat} \rangle^{\sharp} = \left[\left(\overline{\sigma}^{\flat}\right)^{\sharp}, \langle \left(t^{\flat}\right)^{\sharp} \rangle\right]$$

    since x_{k+1} is necessarily of dimension 0. The induction gives the equalities $\left(\overline{\sigma}^{\flat}\right)^{\sharp} = \overline{\sigma}$ and $\left(t^{\flat}\right)^{\sharp} = t$, which proves that

$$\left([\overline{\sigma}; \langle t \rangle]^{\flat}\right)^{\sharp} = [\overline{\sigma}; \langle t \rangle]$$

  - In the case of a non emtpy list, with the last substitution of length $n > 1$, we have

$$\left([\overline{\sigma}; \langle \tau, t \rangle]^{\flat}\right)^{\sharp} = \langle [\overline{\sigma}; \tau]^{\flat}, t^{\flat} \rangle^{\sharp} = \langle \left([\overline{\sigma}; \tau]^{\flat}\right)^{\sharp}, \left(t^{\flat}\right)^{\sharp} \rangle$$

    The induction provides the equalities $\left([\overline{\sigma}; \tau]^{\flat}\right)^{\sharp} = [\overline{\sigma}; \tau]$ and $\left(t^{\flat}\right)^{\sharp} = t$, hence we have

$$\left([\overline{\sigma}; \langle \tau, t \rangle]^{\flat}\right)^{\sharp} = [\overline{\sigma}; \langle \tau, t \rangle]$$

For the converse equality, the same mutually inductive arguments applies, but

one has to carry all the renaming of variables of dimension 0 everywhere. We do not give a full proof here, since the equality in this direction will not be useful for us. □

3.3 Translations for the syntactic categories

We have defined two operations of flattening and folding that establish a correspondence between the judgments in a ps-context of CaTT and the judgments in a monoidal ps-context of MCaTT. We now extend these operations to all judgments on both type theories, defining a pair of functors between $\mathcal{S}_{\mathsf{CaTT}}$ and $\mathcal{S}_{\mathsf{MCaTT}}$. We also show that these functors respect a lot of the structure of category with families of both syntactic categories. We will see in the next section what is the exact structure that is missing, and how this reflects on the models.

Flattening. We extend the falttening operation to all judgments of the type theory, defining a translation that we call ι which associates to any judgment in MCaTT a judgment in CaTT. This is done by induction on the structure of the type theory.

$$\begin{aligned}
\iota\varnothing &= (x_0 : \star) & \iota(\Gamma, y : A) &= (\iota\Gamma, y : \iota A)\\
\iota\star &= x_0 \underset{\star}{\to} x_0 & \iota x \underset{A}{\longrightarrow} y &= \iota x \underset{\iota A}{\longrightarrow} \iota y\\
\iota x &= x & \iota\left(\mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})\right) &= \left(\mathsf{coh}_{\overline{\Gamma},A}\right)^{\flat}(\iota\overline{\sigma})\\
\iota\langle\rangle &= \langle x_0\rangle & \iota\langle\sigma, t\rangle &= \langle\iota\sigma, \iota t\rangle\\
\iota[] &= \langle x_0\rangle & \iota\left[\overline{\sigma}; \tau\right] &= \langle\iota\overline{\sigma}, \iota\tau\rangle
\end{aligned}$$

Proposition 16. *This translation sends derivable judgments in MCaTT onto derivable judgments in CaTT, more precisely*

- *If* $\Gamma \vdash$ *then* $\iota\Gamma \vdash$
- *If* $\Gamma \vdash A$ *then* $\iota\Gamma \vdash \iota A$ *and* $\iota(\Gamma, y : A) = (\iota\Gamma, y : \iota A)$
- *If* $\Gamma \vdash t : A$ *then* $\iota\Gamma \vdash \iota t : \iota A$

- *If* $\Gamma \vdash \sigma : \Delta$ *then* $\iota\Gamma \vdash \iota\sigma : \iota\Delta$ *and* $\iota\langle\sigma, t\rangle = \langle\iota\sigma, \iota t\rangle$
- *If* $\Gamma \vdash \overline{\sigma} : \overline{\Delta}$ *then* $\iota\Gamma \vdash \iota\overline{\sigma} : \overline{\Delta}^{\flat}$

Proof. All these statements are proved by mutual induction on the structure of a type theory. In the case of substitutions to a monoidal ps-context, we just give a sketch of proof, since it contains two layers of inductions, which makes the reasoning hard to track if we write it all completely.

- A context is either the empty context $\varnothing$ or of the form $(\Gamma, y : A)$.
  - In the case of the empty context, we have that $\iota\varnothing = (x_0 : \star)$, and thus $\iota\varnothing \vdash$.
  - In the case of a context of the form $(\Gamma, y : A)$, we necessarily have that $\Gamma \vdash A$, and thus by induction on the case for types, $\iota\Gamma \vdash \iota A$, which proves that $\iota(\Gamma, y : A) \vdash$.

  Note that that for any context Γ in MCaTT, we have $\iota\Gamma \vdash x_0 : \star$, since it is already true for the empty context.
- A type is either $\star$ or of the form $t \underset{A}{\longrightarrow} u$.
  - In the case of the type $\Gamma \vdash \star$, we have $\iota\star = x_0 \underset{\star}{\to} x_0$, and since we always have $\iota\Gamma \vdash x_0 : \star$, this proves that $\iota\Gamma \vdash \iota\star$.
  - In the case of a type of the form $\Gamma \vdash t \underset{A}{\longrightarrow} u$, we necessarily have $\Gamma \vdash t : A$ and $\Gamma \vdash u : A$. The induction case for the term on both judgments shows that $\iota\Gamma \vdash \iota u : \iota A$ and $\iota\Gamma \vdash \iota t : \iota A$. This proves $\iota\Gamma \vdash \iota\left(t \underset{A}{\longrightarrow} u\right)$
- A term $\Gamma \vdash t : A$ is either a variable or a coherence.
  - For a variable $\Gamma \vdash x : A$, we then have that the pairing $x : A$ appears in the list Γ, and thus by construction, the pairing $x : \iota A$ appears in the list $\iota\Gamma$. This proves that $\iota\Gamma \vdash x : \iota A$.

– For a coherence of the form $t = \mathsf{coh}_{\overline{\Delta},B}(\overline{\sigma})$, we have

$$\iota t = \left(\mathsf{coh}_{\overline{\Delta},B}(\mathrm{id}_{\overline{\Delta}})\right)^{\flat}[\overline{\sigma}]$$

By the induction case for substitution towards a monoidal ps-context, we have $\iota\Gamma \vdash \iota\sigma : \overline{\Delta}^{\flat}$, and by the proposition 11, we have that $\overline{\Delta}^{\flat} \vdash \left(\mathsf{coh}_{\overline{\Delta},B}(\mathrm{id}_{\overline{\Delta}})\right)^{\flat}$, so this proves $\iota\Gamma \vdash \iota t : B^{\flat}[\iota\overline{\sigma}]$, and since, by lemma 18, $B^{\flat}[\iota\overline{\sigma}] = \iota\,(B[\overline{\sigma}])$, this shows that $\iota\Gamma \vdash \iota t : \iota A$.

- A substitution is either $\langle\rangle$ or of the form $\langle\sigma, t\rangle$.
  - For the substitution $\Gamma \vdash \langle\rangle : \varnothing$, we have $\iota\langle\rangle = \langle x_0\rangle$, and since we necessarily have that $\iota\Gamma \vdash x_0 : \star$, this proves $\iota\Gamma \vdash \langle x_0 : \star\rangle : (x_0 : \star)$.
  - For a substitution of the form $\Gamma \vdash \langle\sigma, t\rangle : (\Delta, y : A)$, we have $\iota\langle\sigma, t\rangle = \langle\iota\sigma, \iota t\rangle$, and by induction cases on substitutions on types and on terms, we have $\iota\Gamma \vdash \iota\sigma : \iota\Delta$, $\iota\Delta \vdash \iota A$ and $\iota\Gamma \vdash \iota t : \iota\,(A[\sigma])$. Since, by lemma 18, $\iota\,(A[\sigma]) = \iota A[\iota\sigma]$, this proves that the substitution $\iota\Gamma \vdash \langle\iota\sigma, \iota t\rangle : (\iota\Delta, y : \iota A)$ is derivable.
- A substitution to a monoidal ps-context is either $[]$ or of the form $[\overline{\sigma}, \tau]$.
  - For the substitution $\Gamma \vdash [] : []$, we have $\iota\,[] = \langle x_0\rangle$ and since we always have $\iota\Gamma \vdash x_0 : \star$, this proves that $\iota\Gamma \vdash \langle x_0\rangle : (x_0 : \star)$.
  - For a substitution of the form $\Gamma \vdash [\overline{\sigma}; \langle\rangle] : \left[\overline{\Delta}; []\right]$, we have that $\iota\,[\overline{\sigma}; []] = [\iota\overline{\sigma}; \langle x_0\rangle]$, and by the induction, we have $\Gamma \vdash \overline{\sigma} : \overline{\Delta}$ and $\iota\Gamma \vdash x_0 : \star$. This proves that $\iota\Gamma \vdash \langle\iota\sigma, x_0\rangle : \left[\overline{\Delta}, []\right]^{\flat}$.
  - For a substitution of the form

$$\Gamma \vdash \langle\tau, t, u\rangle : \Delta', z : B, f : y \to z$$

then by induction we have that $\iota\Gamma \vdash \iota\,[\overline{\sigma}; \tau] : \left[\overline{\Delta}, \Delta'\right]^{\flat}$, and by two successive induction $\iota\Gamma \vdash \iota t : \iota\,(B[\overline{\sigma}; \tau])$ and $\iota\Gamma \vdash \iota u : \iota\,((y \to z)\,[\overline{\sigma}; \langle\tau, t\rangle])$.

Using the lemma 18 twice, we get the derivation of the judgment $\iota\Gamma \vdash \iota\langle\sigma, t, u\rangle : \iota A$

□

Note 17. This proposition may seem to indicate that ι defines a morphism of categories with families, but crucially this is not actually the case. In fact ι preserves all the structural rules of the type theory but one: it does not send the terminal object onto the terminal object. Indeed, the terminal object in $\mathcal{S}_{\mathsf{MCaTT}}$ is the context empty context $\varnothing$, and it is sent onto the context $(x_0 : \star)$ in $\mathcal{S}_{\mathsf{CaTT}}$, which is not terminal.

Along the proof of the proposition 16, we have used some properties of the translation ι, which should be proved by induction together with the statements, but which we have left out.

Lemma 18. *The translation ι respects the two types of substitutions.*

(1) If A is a type and $\overline{\sigma}$ is a substitution to a monoidal ps-context, then $\iota\,(A[\overline{\sigma}]) = A^\flat[\iota\overline{\sigma}]$

(2) If A is a type and σ is a substitution, then $\iota\,(A[\sigma]) = \iota A[\iota\sigma]$

Folding. Similarly, we extend the folding operation defined on judgments in a ps-context to any judgment, by a translation κ which sends judgments of CaTT

onto judgments of MCaTT.

$$\kappa\varnothing = \varnothing \qquad \kappa(\Gamma, x : A) = \begin{cases} \kappa\Gamma & \text{if } A = \star \\ (\kappa\Gamma, x : \kappa A) & \text{otherwise} \end{cases}$$

$$\kappa\left(t \underset{\star}{\to} u\right) = \star \qquad \kappa\left(t \underset{A}{\to} u\right) = \kappa t \underset{\kappa A}{\longrightarrow} \kappa u$$

$$\kappa x = x \qquad \kappa\left(\mathsf{coh}_{\Gamma,A}(\sigma)\right) = \left(\mathsf{coh}_{\Gamma,A}\right)^\sharp\left(\kappa^\sharp\sigma\right)$$

$$\kappa\langle\rangle = \langle\rangle \qquad \kappa\langle\sigma, t\rangle = \begin{cases} \kappa\sigma & \text{if } \dim t = 0 \\ \langle\kappa\sigma, \kappa t\rangle & \text{otherwise} \end{cases}$$

$$\kappa^\sharp\langle t\rangle = [] \qquad \kappa^\sharp\langle\sigma, t\rangle = \begin{cases} \left[\kappa^\sharp\sigma; \langle\rangle\right] & \text{if } \dim t = 0 \\ \langle\kappa^\sharp\sigma, \kappa t\rangle & \text{otherwise} \end{cases}$$

Note that again, the translation κ is not properly defined on the type $\star$

Proposition 19. *This translation sends derivable judgments in CaTT onto derivable judgments in MCaTT, more precisely*

- *If* $\Gamma \vdash$ *then* $\kappa\Gamma \vdash$
- *If* $\Gamma \vdash A$ *with* $A \neq \star$, *then* $\kappa\Gamma \vdash \kappa A$ *and* $\kappa(\Gamma, y : A) = (\iota\Gamma, y : \kappa A)$
- *If* $\Gamma \vdash t : A$ *with* $A \neq \star$, *then* $\kappa\Gamma \vdash \kappa t : \kappa A$
- *If* $\Gamma \vdash \sigma : \Delta$ *then* $\kappa\Gamma \vdash \kappa\sigma : \kappa\Delta$ *and when* $\dim t > 1$, $\kappa\langle\sigma, t\rangle = \langle\kappa\sigma, \kappa t\rangle$
- *If* $\Gamma \vdash \sigma : \Delta$ *then* $\kappa\Gamma \vdash \kappa^\sharp\sigma : \Delta^\sharp$

Proof. All these statements are proved by mutual induction on the structure that we study

- A context is either $\varnothing$ or of the form $(\Gamma, y : A)$.
  - In the case of $\varnothing$, we have that $\kappa\varnothing = \varnothing$ and thus $\kappa\varnothing \vdash$.
  - For a context of the form $(\Gamma, y : \star)$, we have that $\kappa(\Gamma, y : \star) = \kappa\Gamma$, and thus by induction $\kappa(\Gamma, y : \star) \vdash$.

– For a context of the form $(\Gamma, y : A)$, with $A \neq \star$, we have that $\kappa(\Gamma, y : A) = (\kappa\Gamma, y : \kappa A)$. By induction, we have $\kappa\Gamma \vdash$ and $\kappa\Gamma \vdash \kappa A$, thus this proves $\kappa(\Gamma, y : A) \vdash$

- A term distinct from $\star$ is necessarily of the form $\Gamma \vdash t \underset{A}{\longrightarrow} u$.
  - If $A = \star$, then we have $\kappa\left(t \underset{\star}{\to} u\right) = \star$. By induction we have that $\kappa\Gamma \vdash$, and thus $\kappa\Gamma \vdash \star$.
  - If A is not the type $\star$, we have $\kappa\left(t \underset{A}{\longrightarrow} u\right) = \kappa t \underset{\kappa A}{\longrightarrow} \kappa u$. Moreover, necessarily we have $\Gamma \vdash t : A$ and $\Gamma \vdash u : A$, thus by induction we also have, $\kappa\Gamma \vdash \kappa t : \kappa A$ and $\kappa\Gamma \vdash \kappa u : \kappa A$. This proves that $\kappa\Gamma \vdash \kappa\left(t \underset{A}{\longrightarrow} u\right)$
- A term $\Gamma \vdash t : A$ is either a variable or a coherence.
  - For a variable y of type A which is not $\star$, the pairing $y : A$ is contained in the context Γ, thus the pairing $(y : \kappa A)$ is contained in the context $\kappa\Gamma$, hence we have $\kappa\Gamma \vdash y : \kappa A$.
  - In the case of a coherence $t = \mathsf{coh}_{\Delta,B}(\sigma)$, we have that

$$\kappa\left(\mathsf{coh}_{\Delta,B}(\sigma)\right) = \left(\mathsf{coh}_{\Delta,B}\right)^\sharp\left(\kappa^\sharp\sigma\right)$$

    By the proposition 13, we have that $\Delta^\sharp \vdash \left(\mathsf{coh}_{\Delta,B}\right)\left(\mathrm{id}_\Delta\right)$, and by induction we have $\kappa\Gamma \vdash \kappa^\sharp\sigma : \Delta^\sharp$. This proves that

$$\kappa\Gamma \vdash \left(\mathsf{coh}_{\Delta,B}\right)^\sharp\left(\kappa^\sharp\sigma\right) : B^\sharp[\kappa^\sharp\sigma]$$

    Since by lemma 21, $B^\sharp[\kappa^\sharp\sigma] = \kappa\left(B[\sigma]\right)$, this proves that $\kappa\Gamma \vdash \kappa t : \kappa A$.
- A substitution is either $\langle\rangle$ or it is of the form $\langle\sigma, t\rangle$.
  - In the case of the substitution $\Gamma \vdash \langle\rangle : \varnothing$, we have $\kappa\langle\rangle = \langle\rangle$ and $\kappa\varnothing = \varnothing$. So we have $\kappa\Gamma \vdash \kappa\langle\rangle : \kappa\varnothing$.

– In the case of a substitution of the form $\Gamma \vdash \langle \sigma, t \rangle : (\Delta, y : \star)$, we necessarily have $\Gamma \vdash \sigma : \Delta$, thus by induction $\kappa\Gamma \vdash \kappa\sigma : \kappa\Delta$. Moreover, necessarily $\dim t = 1$, and thus $\kappa\langle \sigma, t \rangle = \kappa\sigma$. This proves that $\kappa\Gamma \vdash \kappa\langle \sigma, t \rangle : \kappa(\Delta, y : \star)$.

– In the case of a substitution of the form $\Gamma \vdash \langle \sigma, t \rangle : (\Delta, y : A)$, with A distinct from $\star$, then $\kappa\langle \sigma, t \rangle = \langle \kappa\sigma, \kappa t \rangle$ and we necessarily have that $\Gamma \vdash \sigma : \Delta$, $\Delta \vdash A$ and $\Gamma \vdash t : A[\sigma]$. So by induction, we have that $\kappa\Gamma \vdash \kappa\sigma : \kappa\Delta$, $\kappa\Delta \vdash \kappa A$ and also $\kappa\Gamma \vdash \kappa t : \kappa\left(A[\sigma]\right)$. Since, by lemma 21, $\kappa\left(A[\sigma]\right) = \kappa A[\kappa\sigma]$, this proves that

$$\kappa\Gamma \vdash \kappa\langle \sigma, t \rangle : \kappa(\Delta, y : A)$$

- A substitution is either $\langle\rangle$ or it is of the form $\langle \sigma, t \rangle$.

  – In the case of the substitution $\Gamma \vdash \langle\rangle : \varnothing$, we have $\kappa^\sharp\langle\rangle = []$. So we have $\kappa^\sharp\Gamma \vdash [] : []$.

  – In the case of a substitution of the form $\Gamma \vdash \langle \sigma, t \rangle : (\Delta, y : \star)$, we necessarily have $\Gamma \vdash \sigma : \Delta$, thus by induction $\kappa\Gamma \vdash \kappa^\sharp\sigma : \Delta^\sharp$. Moreover, necessarily $\dim t = 1$, and thus $\kappa^\sharp\langle \sigma, t \rangle = \left[\kappa^\sharp\sigma, \langle\rangle\right]$. This proves that $\kappa\Gamma \vdash \kappa^\sharp\langle \sigma, t \rangle : (\Delta, y : \star)^\sharp$.

  – In the case of a substitution of the form $\Gamma \vdash \langle \sigma, t \rangle : (\Delta, y : A)$, with A distinct from the type $\star$, then $\kappa^\sharp\langle \sigma, t \rangle = \langle \kappa^\sharp\sigma, \kappa t \rangle$ and we necessarily have that $\Gamma \vdash \sigma : \Delta$, $\Delta \vdash A$ and $\Gamma \vdash t : A[\sigma]$. So by induction, we have that $\kappa\Gamma \vdash \kappa^\sharp\sigma : \Delta^\sharp$, $\kappa\Delta \vdash \kappa A$ and also $\kappa\Gamma \vdash \kappa t : \kappa\left(A[\sigma]\right)$. Since by the lemma 21, $\kappa\left(A[\sigma]\right) = \kappa A[\kappa^\sharp\sigma]$, this proves that $\kappa\Gamma \vdash \kappa^\sharp\langle \sigma, t \rangle : (\Delta, y : A)^\sharp$.

□

Note 20. Again this proposition states that κ almost defines a morphism of categories with families from $\mathcal{S}_{\mathsf{CaTT}}$ to $\mathcal{S}_{\mathsf{MCaTT}}$. Here it fails to do so because it is partial, there is no image for the type $\star$, nor for any term of dimension 0.

Again, during the proof of the proposition 19, we have used some properties followed by κ and $\kappa^\sharp$, which should be proved by induction together with these statements, but that we omit

Lemma 21. *The translations κ and $\kappa^\sharp$ respect the application of substitutions.*

(1) If A is a type and σ is a substitution to a ps-context, $\kappa\left(A[\sigma]\right) = \kappa A[\kappa^\sharp\sigma]$

(2) If A is a type and σ is a substitution, $\kappa\left(A[\sigma]\right) = \kappa A[\kappa\sigma]$

Interaction between flattening and folding. The previous result exhibiting the flattening and folding as inverses for ps-judgments extends to the full fledged type theory only in one direction.

Proposition 22. *the functor κ is a retract of the functor ι*

$$\kappa \circ \iota \quad = \quad \mathrm{id}_{\mathcal{S}_{\mathbf{MCaTT}}}$$

Proof. This is proven by structural induction on the various objects of the theory

- Context are either of the form $\varnothing$, or of the form $(\Gamma, x : A)$.
  - For $\varnothing$, we have $\iota\varnothing = (x_0 : \star)$, and $\kappa(x_0 : \star) = \varnothing$, thus $\kappa(\iota\varnothing) = \varnothing$.
  - For a context of the form $(\Gamma, x : A)$, we have $\iota(\Gamma, x : A) = (\iota\Gamma, x : \iota A)$, and $\kappa(\iota(\Gamma, x : A)) = (\kappa(\iota\Gamma), x : \kappa(\iota A))$. By the induction case for context, we have $\kappa(\iota\Gamma) = \Gamma$, and by the induction case for type we have $\kappa(\iota A) = A$, hence $\kappa(\iota(\Gamma, x : A)) = (\Gamma, x : A)$.
- Types are either of the form $\star$ of or the form $t \to u$.
  - For the type $\star$, we have $\iota\star = x_0 \underset{\star}{\to} x_0$ and $\kappa\left(x_0 \underset{\star}{\to} x_0\right) = \star$, hence $\kappa(\iota\star) = \star$.
  - For a type of the form $t \underset{A}{\to} u$, we have $\iota\left(t \underset{A}{\to} u\right) = \iota t \underset{\iota A}{\longrightarrow} \iota u$, and $\kappa\left(\iota\left(t \underset{A}{\to} u\right)\right) = \kappa(\iota t) \underset{\kappa(\iota A)}{\longrightarrow} \kappa(\iota u)$. By the induction cases for types and terms, we have$\kappa\left(\iota\left(t \underset{A}{\to} u\right)\right) = t \underset{A}{\to} u$.

- Terms are either a variable or a coherence applied.
  - For a term of the form of a variable x, $\iota x = x$ and $\kappa x = x$, thus $\kappa(\iota x) = x$.
  - For a term which is an applied coherence $t = \mathsf{coh}_{\overline{\Gamma},A}[\overline{\sigma}]$, we have

$$\iota t = \left(\mathsf{coh}_{\overline{\Gamma},A}\right)^{\sharp}(\iota\overline{\sigma})$$
$$\kappa(\iota t) = \left(\left(\mathsf{coh}_{\overline{\Gamma},A}\right)^{\sharp}\right)^{\flat}(\kappa^{\sharp}(\iota\overline{\sigma})).$$

    By the proposition 15 and the induction case for the substitutions to a monoidal ps-context, this proves

$$\kappa\iota(\mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})) = \mathsf{coh}_{\overline{\Gamma},A}(\overline{\sigma})$$

- Substitutions are of the form $\langle\rangle$ or $\langle\sigma, t\rangle$.
  - For the subsitution $\langle\rangle$, we have $\iota\langle\rangle = \langle x_0\rangle$ and $\kappa\langle x_0\rangle = \langle\rangle$ since x_0 is of dimension 0. Hence $\kappa(\iota\langle\rangle) = \langle\rangle$.
  - For a substitution of the form $\langle\sigma, t\rangle$, we have

$$\iota\langle\sigma, t\rangle = \langle\iota\sigma, \iota t\rangle \qquad\qquad \kappa\langle\iota\sigma, \iota t\rangle = \langle\kappa\iota\sigma, \kappa\iota t\rangle$$

    since necessarily $\dim(\iota t) > 0$. Hence, by induction case for the subsitutions and for the terms, $\kappa(\iota\langle\sigma, t\rangle) = \langle\sigma, t\rangle$.

- A substitution to a monoidal ps-context is either $[]$ or of the form $[\overline{\sigma}, \tau]$.
  - For the substitution $[]$, we have $\iota([]) = \langle x_0\rangle$, and $\kappa^{\sharp}(\langle x_0\rangle) = []$, hence $\kappa^{\sharp}(\iota\,[]) = []$.
  - For a substitution of the form $[\overline{\sigma}, \tau]$, we have $\iota\,[\overline{\sigma}, \tau] = \langle\iota\overline{\sigma}, \iota\tau\rangle$. Since $\iota\tau$ has to start with the variable x_0 of dimension 0 and contains no other variable of dimension 0, it follows that $\kappa(\langle\iota\overline{\sigma}, \iota\tau\rangle) = [\kappa(\iota\overline{\sigma}); \kappa(\iota\tau)]$.

By the induction case on substitution ot a monoidal ps-context and on substitution, we deduce that

$$\kappa(\iota\overline{\sigma}) = \overline{\sigma} \qquad \square$$

In the converse direction, the result does not hold anymore, intuitively, given a context Γ in the theory CaTT, the context $\iota(\kappa\Gamma)$ is equal to the context Γ where all variables of dimension 0 have been identified. The correct way to formalize this intuition would be to use a most general unifier, which in the case of this theory is a particular case of equalizer[7], however we will not need this construction for proving our result of interest.

3.4 Correctness

Now that we have established translations between MCaTT and CaTT, we can prove the correctness of the type theory MCaTT, i.e., its models are equivalent to the models of CaTT with only one 0-cell. While the definition of the translations merely formalized the intuition behind MCaTT, this part really justifies what makes this intuition correct, by making these translations interact with the structures of category with families of the syntactic categories on each side.

Lemma 23. *Given a model $F : \mathcal{S}_{\mathsf{CaTT}} \to \mathbf{Set}$, the functor induced by the pre-composition with ι, $\iota^*(F) : \mathcal{S}_{\mathsf{MCaTT}} \to \mathbf{Set}$ defines a model of $\mathcal{S}_{\mathsf{MCaTT}}$ if and only if F is a ω-category with only one 0-cell (i.e., F is an object of $\mathbf{Mod}_{\bullet}(\mathcal{S}_{\mathsf{CaTT}})$)*

Proof. Consider a model of $F : \mathcal{S}_{\mathsf{CaTT}} \to \mathbf{Set}$ together with the induced functor $\iota^*(F) : \mathcal{S}_{\mathsf{MCaTT}} \to \mathbf{Set}$. By the proposition 16 and the note 17, $\iota^* F$ may fail to be a model of $\mathcal{S}_{\mathsf{MCaTT}}$ in only one way: it might not preserve the terminal object. All the rest of the structure of category with families is necessarily preserved by $\iota^*(F)$. This means that $\iota^*(F)$ is a model of $\mathcal{S}_{\mathsf{MCaTT}}$ if and only if $\iota^*(F)(\varnothing)$ is the terminal set $\{\bullet\}$. Note that $\iota^*(F)(\varnothing) = F(\iota\varnothing) = F(\mathcal{D}^0)$. So $\iota^*(F)$ is a model of $\mathcal{S}_{\mathsf{MCaTT}}$ if and only if $F(\iota\mathcal{D}^0) = \{\bullet\}$, that is, by definition, if and only if F is an object of $\mathbf{Mod}_{\bullet}(\mathcal{S}_{\mathsf{CaTT}})$. $\square$

In particular, this lemma implies that the functor ι induces a functor

$$\iota^* : \mathbf{Mod}_{\bullet}\left(\mathcal{S}_{\mathsf{CaTT}}\right) \to \mathbf{Mod}\left(\mathcal{S}_{\mathsf{MCaTT}}\right)$$

Lemma 24. *For any model of the theory* MCaTT, $F \in \mathbf{Mod}\left(\mathcal{S}_{\mathsf{MCaTT}}\right)$, *the induced functor* $\kappa^*(F) : \mathcal{S}_{\mathsf{CaTT}} \to \mathbf{Set}$ *defines essentially uniquely a model of* $\mathcal{S}_{\mathsf{CaTT}}$.

Proof. Since κ preserves all the structure of category with families where it is defined, it is also the case for $\kappa^*(F)$. So it suffices to check that $\kappa^*(F)$ can be extended to transport all judgments in a unique way. In particular the only two kinds of judgments that cannot be transported through κ are the ones of the form $\Gamma \vdash \star$ and $\Gamma \vdash t : \star$. Suppose that there is such a way to transport the type $\star$, then it has to satisfy

$$\kappa^*(F)(\Gamma, x : \star) = (\kappa^*(F)(\Gamma), \kappa^*(F)(\star))$$

Since $\kappa(\Gamma, x : \star) = \kappa\Gamma$, it follows that $\kappa^*(F)(\Gamma, x : \star) = \kappa^*(F)(\Gamma)$. On the other hand, $(\kappa^*(F)(\Gamma), \kappa^*(F)(\star))$ is obtained as the pullback

$$\begin{array}{ccc} (\kappa^*(F)(\Gamma), \kappa^*(F)(\star)) & \longrightarrow & (\kappa^*(F)(\star)) \\ \downarrow & \lrcorner & \downarrow \\ \kappa^*(F)(\Gamma) & \longrightarrow & \{\bullet\} \end{array}$$

so $\kappa^*(F)(\star)$ has to satisfy the equation, for all context Γ,

$$\kappa^*(F)(\Gamma) \times \kappa^*(F)(\star) \simeq \kappa^*(F)(\Gamma)$$

This implies that $\kappa^*(F)(\star)$ has to be isomorphic to the singleton $\{\bullet\}$. Now we have to define an image for the term judgments of the form $\Gamma \vdash t : \star$. The image of such a judgment has to be an element of the type image of the judgment $\Gamma \vdash \star$, which is a singleton set $\{\bullet\}$, so there is no other choice than taking the unique element of the singleton. Conversely, extending $\kappa^*(F)$ with these

images defines a morphism of category with families, as we designed them to be compatible with context extension and substitution extension. □

This lemma lets us consider κ^* as a functor $\kappa^* : \mathbf{Mod}\,(\mathcal{S}_{\mathsf{MCaTT}}) \to \mathbf{Mod}\,(\mathcal{S}_{\mathsf{CaTT}})$ and by definition of $\kappa^*(F)$, we have that $\kappa^*(F)(\mathcal{D}^0) = \{\bullet\}$, so in fact κ^* corestricts as

$$\kappa^* : \mathbf{Mod}\,(\mathcal{S}_{\mathsf{MCaTT}}) \to \mathbf{Mod}_\bullet\,(\mathcal{S}_{\mathsf{CaTT}})$$

Theorem 25. *The category of models of $\mathcal{S}_{MCaTT}$ is equivalent to the category $\mathbf{Mod}_\bullet\,(\mathcal{S}_{CaTT})$. More precisely the pair functors*

$$\mathbf{Mod}_\bullet\,(\mathcal{S}_{CaTT}) \underset{\kappa^*}{\overset{\iota^*}{\rightleftarrows}} \mathbf{Mod}\,(\mathcal{S}_{MCaTT})$$

is an equivalence of categories.

Proof. Since κ is a retract of ι, it follows that $\iota^* \circ \kappa^* = \mathrm{id}_{\mathbf{Mod}(\mathcal{S}_{\mathsf{MCaTT}})}$, so it suffices to show that conversely $\kappa^* \circ \iota^* = \mathrm{id}_{\mathbf{Mod}_\bullet(\mathcal{S}_{\mathsf{CaTT}})}$. Consider a category with only one object, $F \in \mathbf{Mod}_\bullet\,(\mathcal{S}_{\mathsf{CaTT}})$, we show by induction on the length of Γ that $\kappa^*(\iota^* F)(\Gamma)$ is isomorphic to $F(\Gamma)$.

- If $\Gamma = \varnothing$, and $\iota(\kappa\Gamma)$ is the context $(x_0 : \star)$ (disk of dimension 0). Since $F \in \mathbf{Mod}_\bullet\,(\mathcal{S}_{\mathsf{CaTT}})$, $F(\iota(\kappa\Gamma)) = \{\bullet\}$ is a singleton, and by definition of a model of $\mathcal{S}_{\mathsf{CaTT}}$, $F(\varnothing) = \{\bullet\}$ is also a singleton. Hence, $\kappa^*(\iota^* F)(\varnothing)$ is isomorphic to $F(\varnothing)$.

- If Γ is of the form $(\Gamma', x : A)$ with A distinct from the type $\star$, then $\kappa(\Gamma', x : A) = (\kappa\Gamma', x : \kappa A)$ and thus

  $$\iota(\kappa(\Gamma', x : A)) = (\iota(\kappa\Gamma'), \iota\kappa A)$$

  Hence, by induction hypothesis,

  $$F(\iota(\kappa(\Gamma', x : A))) = (F(\iota(\kappa\Gamma')), F(\iota(\kappa A))) \simeq (F(\Gamma'), F(A)) = F(\Gamma', x : A)$$

- If A is the type $\star$, then $\kappa(\Gamma', x : \star) = \kappa(\Gamma')$ and $\iota(\kappa(\Gamma', x : \star)) = \iota(\kappa\Gamma')$, hence $F(\iota(\kappa(\Gamma', x : \star))) = F(\iota(\kappa\Gamma'))$. But on the other hand, the image $F(\Gamma', x : \star) = (F(\Gamma'), F(\star))$ is obtained as the following pullback

$$\begin{array}{ccc} F(\Gamma) & \longrightarrow & F(\mathcal{D}^0) \\ \downarrow & \lrcorner & \downarrow \\ F(\Gamma') & \longrightarrow & F(\varnothing) \end{array}$$

  Since F is a category with only one object, this pullback rewrites as

$$\begin{array}{ccc} F(\Gamma) & \longrightarrow & \{\bullet\} \\ \downarrow & \lrcorner & \downarrow \\ F(\Gamma') & \longrightarrow & \{\bullet\} \end{array}$$

  which shows that $F(\Gamma)$ is isomorphic to $F(\Gamma')$. Hence we have proved that in both cases, $F(\Gamma)$ is isomorphic to $F(\iota(\kappa\Gamma))$. □

4 Conclusion

We have developed the type theory MCaTT whose models are monoidal weak ω-categories. This requires to shift perspective a little and define lists of contexts together with types, terms and substitutions associated. Although this is not a usual structure in type theory, it appears quite naturally here. We then have defined, using purely syntactic methods, translations between these two type theories. We showed how to prove a result of correctness of the models of this theory using these translations, thus establishing a close connection between the syntactic comparison of the two theories, and the comparison of their models. This gives a the sketch for a general method that could apply to study weak ω-categories with other extra structure.

A forgetful functor. There is another (more obvious) translation from $\mathcal{S}_{\mathsf{CaTT}}$ to $\mathcal{S}_{\mathsf{MCaTT}}$ that we have not presented here. Given a ps-context Γ in the CaTT theory, it simply gives a monoidal ps-context by considering the one-element list

[Γ]. This observation lifts on types, terms, regular contexts and substitutions, and gives a morphism of category with families

$$u : \mathcal{S}_{\mathsf{CaTT}} \to \mathcal{S}_{\mathsf{MCaTT}}$$

The induced functor on the categories of models

$$u^* : \mathbf{Mod}\left(\mathcal{S}_{\mathsf{MCaTT}}\right) \to \mathbf{Mod}\left(\mathcal{S}_{\mathsf{CaTT}}\right)$$

is the forgetful functor, which, given a monoidal weak ω-category, forgets the monoidal product and gives the underlying weak ω-category. Importantly, in this case there is no shift of dimension; the objects of the ω-category are the objects of the monoidal ω-category, but without the ability to be composed. The existence of this functor is closely related to the well definedness of an operation of *suspension* in the theory CaTT[8].

Multiply monoidal ω-categories A natural generalisation of this result would consist in giving a type-theoretical definition of k-tuply weak ω-categories [4, 3] for any $k \in \mathbb{N}$. These can be defined inductively by saying that $(k+1)$-tuply monoidal categories are equivalent to k-tuply monoidal categories with only one object. The theory CaTT solves the case $k = 0$, and the theory MCaTT that we present in this article solves the case $k = 1$. We believe that similar ideas will also work in all other cases. More precisely, we define inductively a $(k+1)$-*monoidal ps-context* to be a list of k-monoidal ps-contexts, and we endow them with types, terms and substitutions, satisfying rules that are akin to the ones of MCaTT. Let us denote the resulting theories k-MCaTT. We expect the models of $(k+1)$-MCaTT to be equivalent to the models of k-MCaTT with only one object, and the proof should work along the same line, by defining a pair of translations between the two theories. However, the translations shall be significantly more difficult to define, and let us consider the case $k = 1$ to understand why. The syntactic category of MCaTT has, as objects, the monoidal

weak ω-categories which are freely finitely generated, so intuitively, we want to define a functor relating contexts in 2-MCaTT with contexts of MCaTT which have only one object. However, no freely finitely generated monoidal weak ω-category has only one object, since as soon as it contains one of them, it also contains all its iterated products. Thus in order to translate all the terms, the translation will have to encompass a strictification procedure. We expect this to be a technical difficulty rather than a theoretical obstacle, since at the levels of categories, the strictification only cares for compositions of identities and it is a very minimal setting in which a strictification can be expected.

Symmetric monoidal ω-categories Notice that the argument giving the functor $u : \mathcal{S}_{\mathsf{CaTT}} \to \mathcal{S}_{\mathsf{MCaTT}}$ works in a similar way for defining

$$u : \mathcal{S}_{k\text{-}\mathsf{MCaTT}} \to \mathcal{S}_{(k+1)\text{-}\mathsf{MCaTT}}$$

A k-monoidal context is naturally a $(k+1)$-monoidal context, by considering the one-element list containing this k-monoidal context. This allows for a slight reformulation of the theory k-MCaTT, where instead of considering k-iterated lists, we consider n-iterated lists with $(n \leq k)$ and we identify an object with the list containing only this object. Due to this reformulation, one can transfer proofs from the theory k-MCaTT to the theory $(k+1)$-MCaTT, showing that any result derived in k-MCaTT still holds in $(k+1)$-MCaTT. This would allow to use developments carried in one theory, to be freely obtained in other theories, thus factorizing a lot of common proofs in monoidal categories with various multiplicity. On a theoretical level, introducing this change seems harmless, but it allows for a proposition of a theory ∞-MCaTT whose models are ∞-monoidal weak ω categories (aka. symmetric monoidal weak ω-categories), simply by dropping the condition $(n \leq k)$.

Monoidal closed higher categories. It would be valuable to connect the result we have presented with recent work of Finster for integrating the theory

of monoidal higher categories in the tool Opetopic [12], although it necessitates to establish a connection between globular shapes and opetopic shapes.

Implementation. An implementation of the type theory CaTT is available at [6]. Although it has not been done yet due to time constraints, another implementation for the type theory MCaTT is planned for future works, as well as eventually an implementation for k-MCaTT and ∞-MCaTT

References

[1] Thorsten Altenkirch and Ondrej Rypacek. A syntactical approach to weak omega-groupoids. In *Computer Science Logic (CSL'12)-26th International Workshop/21st Annual Conference of the EACSL*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

[2] Dimitri Ara. *Sur les ∞-groupoïdes de Grothendieck et une variante ∞-catégorique*. PhD thesis, Ph. D. thesis, Université Paris 7, 2010.

[3] John Baez. Lectures on n-categories and cohomology. Notes by M. Shulman.

[4] John C Baez and James Dolan. Higher-dimensional algebra and topological quantum field theory. *Journal of Mathematical Physics*, 36(11):6073–6105, 1995.

[5] Michael A Batanin. Monoidal globular categories as a natural environment for the theory of weakn-categories. *Advances in Mathematics*, 136(1):39–103, 1998.

[6] Thibaut Benjamin, Eric Finster, and Samuel Mimram. The CaTT proof assistant, 2018. `https://github.com/ThiBen/catt`.

[7] Thibaut Benjamin, Eric Finster, and Samuel Mimram. Globular weak ω-categories as models of a type theory. In preparation, 2019.

[8] Thibaut Benjamin and Samuel Mimram. Suspension et Fonctorialité: Deux Opérations Implicites Utiles en CaTT. In *Journées Francophones des Langages Applicatifs*, 2019.

[9] Guillaume Brunerie. On the homotopy groups of spheres in homotopy type theory. *arXiv preprint arXiv:1606.05916*, 2016.

[10] Eugenia Cheng and Aaron Lauda. Higher-dimensional categories: an illustrated guide book. *Preprint*, 2004.

[11] Peter Dybjer. Internal Type Theory. In *Types for Proofs and Programs. TYPES 1995*, pages 120–134. Springer, Berlin, Heidelberg, 1996.

[12] Eric Finster. Opetopic. `https://github.com/ericfinster/opetopic`.

[13] Eric Finster and Samuel Mimram. A Type-Theoretical Definition of Weak ω-Categories. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, 2017.

[14] Alexander Grothendieck. *Pursuing stacks*. unpublished manuscript, 1983.

[15] Tom Leinster. A survey of definitions of n-category. *Theory and applications of Categories*, 10(1):1–70, 2002.

[16] Tom Leinster. *Higher operads, higher categories*, volume 298. Cambridge University Press, 2004.

[17] Peter LeFanu Lumsdaine. Weak ω-categories from intensional type theory. In *International Conference on Typed Lambda Calculi and Applications*, pages 172–187. Springer, 2009.

[18] Saunders Mac Lane. *Categories for the working mathematician*, volume 5. Springer Science & Business Media, 2013.

[19] Georges Maltsiniotis. Grothendieck ∞-groupoids, and still another definition of ∞-categories. Preprint `arXiv:1009.2331`, 2010.

[20] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

[21] Benno Van Den Berg and Richard Garner. Types are weak ω-groupoids. *Proceedings of the London Mathematical Society*, 102(2):370–394, 2011.