The Tensor-Plus Calculus

ANONYMOUS

***Abstract*—We propose a graphical language that accommodates two monoidal structures: a multiplicative one for pairing and an additional one for branching. In this colored PROP, whether wires in parallel are linked through the multiplicative structure or the additive structure is implicit and determined contextually rather than explicitly through tapes, world annotations, or other techniques, as is usually the case in the literature. The diagrams are used as parameter elements of a commutative semiring, whose choice is determined by the kind of computation we want to model, such as non-deterministic, probabilistic, or quantum.**

Given such a semiring, we provide a categorical semantics of diagrams and show the language as universal for it. We also provide an equational theory to identify diagrams that share the same semantics and show that the theory is sound and complete and captures semantical equivalence.

In categorical terms, we design an internal language for semi-additive categories (C,+,0) with a symmetric monoidal structure (C,x,1) distributive over it, and such that the homset C(1,1) is isomorphic to a given commutative semiring, e.g., the semiring of non-negative real numbers for the probabilistic case.

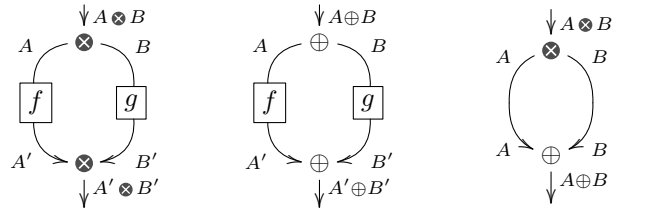
I. Introduction

From a computational perspective, the low-level control flow of a program is inherently linked to the information it acts upon. Besides composition, two program manipulations can be considered native: juxtaposition and branching.

The juxtaposition of programs can happen when they operate on distinct areas of the memory: a function f acting on A can be executed in parallel to a function g acting on B as long as they do not interact. Such operations can run asynchronously—they can be juxtaposed. The categorical interpretation of such an operation is a monoidal structure: the joint system of A and B is written $A \otimes B$,[1] and the combined action of f and g is $f \otimes g$. The operation is said multiplicative: $A \otimes B$ represents pairs of elements of A and B.

These multiplicative monoidal structures can be equipped with a natural graphical interpretation (a PROP): types are represented as wires and actions as boxes. The system's action $f \otimes g$ on the type $A \otimes B$ can be represented graphically as shown in Figure 1a. For this multiplicative construction, in the interpretation discussed above, the wire $A \otimes B$ is intuitively understood as a bundle of two wires of type A and B: the two boxes $\otimes$ represent the split and merge of these two bundled wires.

The other arguably native operation is branching: the possibility to choose a course of action based on the state of the memory. This choice operation becomes the if-then-else for Boolean values and the case distinction for more general data structures. From a categorical standpoint, the choice is typically modeled using a coproduct or a byproduct $\oplus$. A Boolean value lives in $1 \oplus 1$: it is either the left injection (say, False) or the right injection (say, True). In general, the type $A \oplus B$ stands for either something of type A or something of type B. The additive monoidal structure is graphically similar to the multiplicative one, shown in Figure 1b. The main difference lies in the interpretation of the wire of type $A \oplus B$: something of type A or type B will get in, and this piece of data will be routed to the correct wire A or B upon reaching the first split-node $\oplus$. The merge-node $\oplus$ will then secure the result again in the wire $A' \oplus B'$.



(a) Multiplicative (b) Additive (c) Incompatibility

Fig. 1: typical graphical structure

These multiplicative and additive structures are pervasive in all models of computation. They serve as a baseline for the multiplicative, additive fragment of linear logic [1], [2]. Dozens of categories feature these structures: relational models [3], [4], [5], [6], coherent spaces [7], vectorial models such as modules or vector spaces [8], [9], etc.

Despite this ubiquity, it is notoriously difficult to define a graphical language that handles both a multiplicative, monoidal structure and an additive structure, whether coproduct or byproduct. The main difficulty lies in keeping track of how wires are related: if the language is purely multiplicative, all wires are tensored; if the language is purely additive, all wires are in sum. However, if the language features both tensor and sum, three wires of type A, B, and C are ambiguous: they could, for instance, be seen as $A \otimes (B \oplus C)$, but also as the (incompatible) $(A \otimes B) \oplus C$ or even $A \oplus (B \otimes C)$. In the literature, this has been approached with the addition of external information on the graph representing the computation: worlds [10], sheets [11] or tapes [12]. Considering our three wires of types A, B, and C, additional information is added to know how to relate them: for instance, whether A and B should first be tensored before summing C.

Multiplicative and additive structures canonically support algebraic effects. Relational models can be weighted with non-deterministic [13] or probabilistic effects [4], and finite-

[1]The combinator $\otimes$ is usually written $\otimes$. We chose to modify its representation to make the distinction with $\oplus$ clearer, which is especially beneficial for the upcoming generators of the graphical language.

dimensional vector spaces is a model for (pure) quantum computation [14]. In this realm, the "choice" operation becomes effectful, and wires carry a weight: a probability, a complex number representing a quantum coefficient, etc. Such effectful computations take an additional toll on the design of a graphical language: one has to handle the multiplicative and additive aspects and the actions of the algebraic effects.

This paper is devoted to studying such a setting: We propose a graphical language unifying multiplicative and additive actions, supporting probabilistic, non-deterministic, and more exotic effects such as purely quantum effects. Contrary to the above-mentioned worlds, sheets, or tapes, the handling of wires does not require any additional structure.

A. Strategy followed in the paper.

The main difficulty consists in giving a sense to the juxtaposition of two wires: it cannot simply be a tensor or a sum; it needs to be able to be both and in a homogeneous manner.

Our proposal, therefore, defines a special monoidal structure capturing these two possibilities at once. Formally, we set ourselves in the context of a symmetric monoidal category (representing the multiplicative tensor) with an additive structure: the additive structure gives a biproduct, and when enriched this framework is expressive enough to represent the algebraic effects we care about. Wire juxtaposition is then represented with a sum of either a tensor or a sum: $(A \otimes B) \oplus (A \oplus B)$. If this is defined formally in Section III, let us simply mention here that this binary operation gives a well-defined monoidal structure and a PROP: we provide in the paper a graphical language for which this category is the target model.

Moreover, because these two wires might be used inconsistently, for instance, the diagram shown in Figure 1c: we need to be able to represent a notion of "error" state. We capitalize on the additive structure (and the enrichment) and represent it with the "zero" map inherited from the additive structure of the category.

B. Contributions and plan of the paper.

Formally, the contributions of the paper are as follows.

- A graphical language: the Tensor-Plus Calculus, unifying multiplicative and additive structures and capturing algebraic effects: non-determinism, probability, vectorial (and quantum). The language is described using the notion of PROP, and comes in two variants: $\mathbf{TP}^R$ and $\mathbf{FTP}^R$, with R a given commutative semiring.
- An interpretation based on symmetric, monoidal, semi-additive categories, with a universality result for both (Theorem III.2);
- An equational theory allowing us to rewrite diagrams, proven sound and complete (Proposition IV.1 and Theorem IV.2). We write $\mathbf{TP}^R_{\equiv}$ and $\mathbf{FTP}^R_{\equiv}$ for the categories $\mathbf{TP}^R$ and $\mathbf{FTP}^R$ quotiented by the corresponding equivalence (rewriting) relation. From the completeness result, we show that the Tensor-Plus Calculus can be regarded as an internal language for the corresponding category (Theorem IV.8).

The plan of the paper is as follows. We present the language and several examples in Section II. We then define the categorical semantics of our language and show its universality in Section III. Section IV is devoted to presenting a sound and complete equational theory. We then discuss extensions of these results

II. The Tensor-Plus Calculus

The aim of the paper is to define a graphical language that allows both for pairing and branching of data. In the latter case, the two branches may not be used at the same time (we use either one branch or the other, or none), while in the pairing case, the two pieces of data are either used together, or are both not used. We want all these possible interactions between wires to remain implicit, and not resort to explicit additional cues, like sheets, tapes, annotations, etc. Moreover, we want to be able to account for algebraic effects such as non-deterministic, probabilistic, and quantum effects (or, more generally, vectorial effects).

The language we propose, called the Tensor-Plus Calculus is parameterized by a commutative semiring $(R, +, 0, \times, 1)$ to account for the kind of algebraic effects we target. It can be instantiated by the complex numbers $(\mathbb{C}, +, 0, \times, 1)$ to represent pure quantum computations, the non-negative real numbers $(\mathbb{R}_{\geq 0}, +, 0, \times, 1)$ for probabilistic computations, or the Boolean values $(\{0, 1\}, \vee, 0, \wedge, 1)$ for non-deterministic computations. We come back to these semirings in Section II-C.

We define the Tensor-Plus Calculus within the paradigm of colored PROP [15], [16]: A morphism is a diagram composed of nodes, called *generators*, linked to each other through *colored wires*—where each *color* corresponds to a datatype such as "bit"—that are allowed to cross each other. This graphical representation allows the rewriting equations provided in Figure 2, where f and g stand for any diagrams with any numbers of inputs and outputs, and the wires could be of any colors.

The Tensor-Plus Calculus is equipped with a denotational semantics (Section III) and an equational theory (Section IV). In this section, we only focus on the syntax and the intuition through a series of examples.

A. The Wires: the Objects of our Category

In a colored PROP, the set of objects is freely generated by a set of *colors* and a strict[2] monoidal product $\parallel$ with $\varnothing$ as neutral element. In other words, an object corresponds to a collection of wires in parallel, written $A_1 \parallel \cdots \parallel A_n$, each wire being "typed" by a color A_i. We use calligraphic letters $\mathcal{X}, \mathcal{Y}, \ldots$ to denote the objects of our colored PROP, and non-calligraphic letters $A, B, \ldots$ for colors. The goal is for the colors to represent datatypes such as "bit", so the colors will themselves be generated by the syntax

$$A, B \quad ::= \quad \mathbb{0} \mid \mathbb{1} \mid (A \oplus B) \mid (A \otimes B),$$

[2]So $\parallel$ is strictly associative and $\varnothing$ is strictly its neutral element.

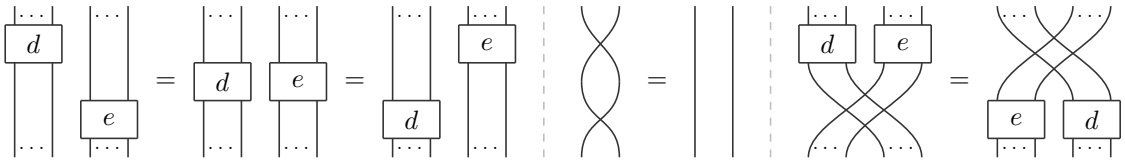


Fig. 2: Equations of a colored PROP. These are supposed true for any colors on the wires.

where $\oplus$ is used to build sum types such as "bit = $\mathbb{1} \oplus \mathbb{1}$", and $\otimes$ is used to build pairs. The symbol $\mathbb{0}$ represents the unit of the sum $\oplus$ while $\mathbb{1}$ is the unit of the tensor $\otimes$. For example,

$$((A \oplus B) \oplus \mathbb{0}) \parallel \mathbb{1} \parallel (A \otimes \mathbb{0})$$

is an object composed of three colors: $(A \oplus B) \oplus \mathbb{0}$, $\mathbb{1}$ and $A \otimes \mathbb{0}$. We can then represent them as three wires in parallel:

Note that while the monoidal product $\parallel$ is strictly associative, we do not consider $\oplus$ and $\otimes$ to be associative as they are purely syntactical. We instead have an equivalence relation $\approx_{\lambda\rho\alpha}$ on colors, generated by the rules:

$$A \oplus \mathbb{0} \approx_{\lambda\rho\alpha} A, \qquad A \otimes \mathbb{1} \approx_{\lambda\rho\alpha} A,$$
$$\mathbb{0} \oplus A \approx_{\lambda\rho\alpha} A, \qquad \mathbb{1} \otimes A \approx_{\lambda\rho\alpha} A,$$

for the units,

$$(A \oplus B) \oplus C \approx_{\lambda\rho\alpha} A \oplus (B \oplus C),$$
$$(A \otimes B) \otimes C \approx_{\lambda\rho\alpha} A \otimes (B \otimes C),$$

for the associativity of the binary operators, and

$$A \oplus C \approx_{\lambda\rho\alpha} B \oplus C, \qquad A \otimes C \approx_{\lambda\rho\alpha} B \otimes C,$$
$$C \oplus A \approx_{\lambda\rho\alpha} C \oplus B, \qquad C \otimes A \approx_{\lambda\rho\alpha} C \otimes B,$$

whenever $A \approx_{\lambda\rho\alpha} B$, for the congruence rules.

The choice of the notation $\parallel$ for wires in parallel is uncommon: the notation $\otimes$ is often preferred in the literature. We use it to put an emphasis on the fact that contrary to languages such as proof-nets [2], Boolean (or quantum) circuits, or the ZX-calculus [17], wires that are in parallel are not necessarily "in tensor with one another". In fact, $A \parallel B$ can be understood semantically as "either $A \otimes B$ or $A \oplus B$". This intepretation is formalized in the categorical semantics in Section Section III-A, and it will be the underlying meaning of the equation (mix) of Figure 7.

B. The Diagrams: the Morphisms of our Category

The generators of our language are described in Figure 3. On the top line, one can respectively find: the Identity $\mathbf{id}_A : A \to A$, the Swap $\sigma_{A,B} : A \parallel B \to B \parallel A$, the Tensor $\otimes_{A,B} : A \parallel B \to (A \otimes B)$, the Plus $\oplus_{A,B} : A \parallel B \to (A \oplus B)$, the Contraction $\bigtriangledown_A : A \parallel A \to A$, the Null $\mathbf{zero}_A : \varnothing \to A$ (represented with a triangle pointing down), the Unit $\otimes_{\mathbb{1}} : \varnothing \to \mathbb{1}$. On the bottom line, one can read the Adapter $\bullet_{A,A'} : A \to A'$ (whenever $A \approx_{\lambda\rho\alpha} A'$), the Scalar $[s]_A : A \to A$ for s ranging over the commutative semiring R, and the up-down mirrored version of all of the generators[3]. We write $f^{\mathbb{T}}$ for the up-down mirrored version of f. Diagrams are read top-to-bottom: the top-most wires are the *input* wires and the bottom-most wires are the *output* wires.

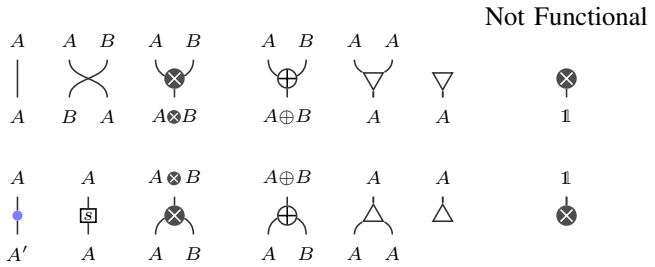


Fig. 3: Generators of our Language ($A \approx_{\lambda\rho\alpha} A'$, $s \in R$)

These generators encompass at once:

- the PROP structure of the Tensor-Plus Calculus (with the Identity and the Swap);
- the multiplicative, tensor structure (with the Tensor, the Unit and their duals);
- the additive, biproduct structure (with the Plus, the Contraction and their duals);
- the scalars (with the Scalar generator).

The associativity and unit of the multiplicative and additive structures is given by the Adapter (representing the equivalence $\approx_{\lambda\rho\alpha}$).

Equationally, the language as it stands does not contain more than the equational theory of PROP. In particular, with respect to the Sum, the Contraction corresponds to the codiagonal and its dual to the diagonal morphism: this is not yet enforced. In Section III we present a categorical semantics formally describing this intuition, and Section IV equips the Tensor-Plus Calculus with an equivalence relation $\equiv$ capturing the structure of the denotational semantics.

Diagrams are obtained by composing the generators (Figure 3) in parallel (written $\parallel$), or sequentially (written $\circ$), as follows:

[3]The Identity, the Swap, the Adapter and the Scalar are their own mirrored version.

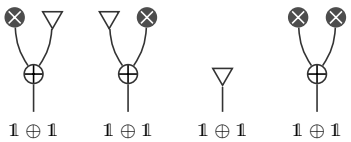
Sequential composition requires the color (and number) of wires to match. We write $\mathbf{TP}^R$ for the PROP category of such diagrams (with the equality given by the equational theory of PROP), and $\mathbf{TP}^R(\mathcal{X}, \mathcal{Y})$ for the set of diagrams that are morphisms from $\mathcal{X}$ to $\mathcal{Y}$.

The goal we announced in the introduction was to create an internal language for semiadditive categories with some additional properties. As such, one would expect $\varnothing$ to be a terminal and initial object of our category, and might question the existence of the Unit as a non-trivial morphism from $\varnothing$ to $\mathbb{1}$. The Unit and its mirrored version are considered "non-functional". Computationally, they allow the process to start generating non-zero outputs without having received any input. More generally, while useful for practical examples and when representing "values" (true, false) instead of "functions" (negation, etc), they come with some significant technical complexities, and they muddy the categorical properties of our language. As such, we define the sub-language of **Functional** Tensor-Plus Calculus as the same but without the Unit and its mirrored version. We write $\mathbf{FTP}^R$ for the corresponding category.

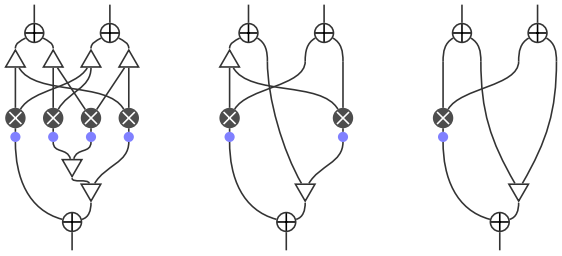
C. Examples

To illustrate the expressive power of our diagrams, we give one example for each of the commutative semirings that were announced at the beginning of the section.

Example II.1. We consider R to be the boolean ring, and look at the type $\mathbb{1} \oplus \mathbb{1}$. There are four different values over this type: True, False, $\bot$ (or Failure), $\top$ (or the Nondeterministic superposition of True and False). One can represent them as



We can define several versions of the logical OR: $(\mathbb{1} \oplus \mathbb{1}) \parallel (\mathbb{1} \oplus \mathbb{1}) \to \mathbb{1} \oplus \mathbb{1}$, respectively the strict OR, the lazy OR, and the parallel OR:



All these versions of the OR produce the same output when they receive well-defined Boolean inputs. Let us give some intuition to better understand those pictures:

- The left branch of a Plus corresponds to False, while its right branch corresponds to True.
- The Contractions should be seen as rail-switches where data must go left or right and cannot do both.
- The Tensor should be thought of as grouping data together.
- The Adapter can be ignored, its effect is purely administrative.

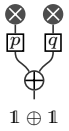
In all three diagrams, the right branches of the input Pluses are connected in many different ways to the right branch of the output Plus, and never to the left branch. As such, as soon as one input is True, the output cannot be False. The other way around, the left branch of the output Plus is only connected to the left branches of the input Pluses, so for the output to be False, both inputs must be False.

These three versions of the OR differ in the way they deal with failing inputs – represented by the Null $\mathbf{zero}_{\mathbb{1}\oplus\mathbb{1}}$. The strict OR expects a well-defined Boolean value for both its inputs and will return Null if any of its two input is Null, while the lazy OR can "short-circuit" the computation if the first input turns out to be True – in that case, we don't care about the second input, and directly output True even if that second output would be Null. The parallel OR symmetrizes the lazy OR, and produces True whenever one of the two inputs is True.

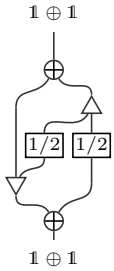
These behaviors are completely captured by the upcoming equational theory, and examples of how it can be used to verify said behaviors are provided in Section IV.

Example II.2. When considering $R = \mathbb{R}_{\geq 0}$, we can encode some basic probabilistic primitives in it and show how they operate. The most basic data we can represent is a probabilistic bit (pbit), seen as a vector $\begin{pmatrix} p \\ q \end{pmatrix}$ where p is the probability of False and q the probability of True – we do not enforce $q = 1 - p$ here, although it is required for this to be an actual pbit.

In $\mathbf{TP}^R$, the Boolean values are represented by the type $\mathbb{1} \oplus \mathbb{1}$. The above pbit is then represented by the following diagram:

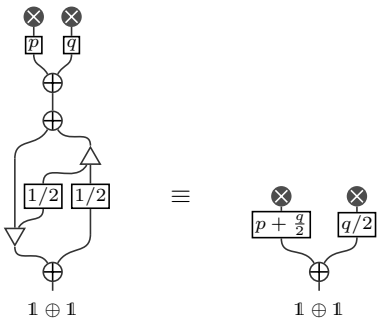
$$\begin{pmatrix} p \\ q \end{pmatrix} \rightsquigarrow$$  $$\in \mathbf{TP}^{\mathbb{R}_{\geq 0}}(\varnothing, \mathbb{1} \oplus \mathbb{1})$$

A program manipulating pbits can be understood as a non-negative-valued (usually stochastic) matrix. For example, the matrix $\begin{pmatrix} 1 & 1/2 \\ 0 & 1/2 \end{pmatrix}$ corresponds to the program "if x then coin() else False". This operation can be represented as follows:

$$\begin{pmatrix} 1 & 1/2 \\ 0 & 1/2 \end{pmatrix} \rightsquigarrow$$  $$\in \mathbf{TP}^{\mathbb{R}_{\geq 0}}(\mathbb{1} \oplus \mathbb{1}, \mathbb{1} \oplus \mathbb{1})$$

In that figure, the top-most $\oplus$ allows us to "open a vector" (the input) to recover its corresponding scalars, the contractions allow us to duplicate and sum scalars. Finally, the bottom-most $\oplus$ will build a new vector from two scalars.

For example, one can look at the composition of the above matrix and vector. Using the upcoming equational theory (Section IV), we can rewrite the composite diagram into a simpler diagram (done in Example IV.5):

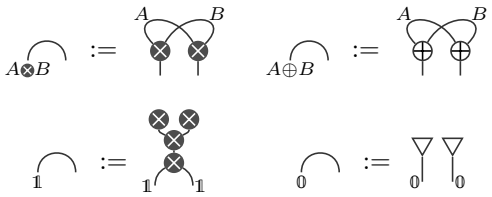


and this resulting diagram indeed corresponds to the result of the matrix product $\begin{pmatrix} p + \frac{q}{2} \\ q/2 \end{pmatrix}$.

Example II.3. We consider $R = \mathbb{C}$. The situation here is very similar to the probabilistic case, as our diagrams represent matrices. The main difference is that instead of expecting probabilistic bits to satisfy $p + q = 1$, we expect the quantum bits to satisfy $|p|^2 + |q|^2 = 1$, where p, q are now complex numbers. The presence of negative numbers means that interference is now a possibility: multiple executions of the programs can cancel each other. For example, applying the Hadamard unitary $\begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix}$ to $\begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$ returns False, because the two executions outputting True cancel each other. This interference will also be visible in the upcoming equational theory, where two opposite scalars will cancel each others with Equation (R_+).

D. The Compact Closure

While we do not have a Cup and a Cap as generators in order to "bend" wires, we can define them inductively as follows:



Since they rely on Unit, they are not part of $\mathbf{FTP}^R$. As proved in **??**, the equational theory ensures that they satisfy the snake equations. In other words, $\mathbf{TP}^R_{\equiv}$ is compact closed. Those bent wires can be used to represent relatively advanced control flows and higher order programs, as shown in the example below.

Example II.4. We take inspiration from the literature of quantum computation and study the "(quantum) switch" [18], [19], [20]. Note that despite being quantum-inspired, this example works for any commutative semiring R. It comes from a physical protocol allowing physicists to apply two operators U and V in an indefinite causal order, that is either $U \circ V$ or $V \circ U$, while relying on only one physical instance of U and one physical instance of V [21].

As such, representing it in our language is an interesting case study of both "higher order" protocols, as U and V are inputs of the programs, and of advanced control flow, as it is indefinite whether U is executed first or second. We temporarily set aside the "higher order" part, and focus on the latter. That is, we start by implementing $\texttt{switch}_{\texttt{U,V}} : (\mathbb{1} \oplus \mathbb{1}) \otimes A \to (\mathbb{1} \oplus \mathbb{1}) \otimes A$, assuming two given diagrams for $\texttt{U}, \texttt{V} : A \to A$. The expected operational behavior is the following:

```
switch_U,V b a = if b then (b, VUa)
                      else (b, UVa)
```

In $\mathbf{TP}^R$, we can easily represent the $\texttt{switch}_{\texttt{U,V}}$ using two instances of U and V, it is the left side of Figure 4. The diagram is a simple branch-out depending on the value of b, where we either apply $U \circ V$ or $V \circ U$. Finding a graphical representation using only on instance of U and V is harder – and in fact impossible in some graphical languages [18] – but still doable in our language, see the middle of Figure 4.

From this diagram with only a single instance of U and V, we can easily extract those out of the diagram by bending wires. This yields the diagram at the right of Figure 4, which represents

$$\texttt{switch} : (\mathbb{1} \oplus \mathbb{1}) \otimes A \parallel (A \parallel A) \parallel (A \parallel A) \to (\mathbb{1} \oplus \mathbb{1}) \otimes A$$

where the first $(A \parallel A)$ expects U as an input and the second $(A \parallel A)$ expects V.

Now that the expressive power of the graphical language is well illustrated, let's provide diagrams with a formal meaning.

III. Categorical Semantics

We provide a semantics based on category theory, although this semantics could also be stated in terms of matrices instead[4]. For example, in the case where the scalars of R are complex numbers, this is simply a semantics toward matrices with complex coefficients, so toward $(\mathbf{FdHilb}, \oplus, \{0\}, \otimes, \mathbb{C})$ where **FdHilb** is the category of finite dimensional Hilbert spaces, $\oplus$ is the direct sum (also called Cartesian product), $\{0\}$ is the trivial Hilbert space, $\otimes$ is the tensor product (also called Kronecker product), and $\mathbb{C}$ is the field of complex numbers. The categorical framework used here can be deduced from three structures:

- a semiadditive[5] category $(\mathbf{H}, \oplus, \mathbb{0})$
- a symmetric monoidal structure $(\mathbf{H}, \otimes, \mathbb{1})$ that is distributive over the semiadditive structure.
- a semiring isomorphism $\mathbf{scal} : R \to \mathbf{H}(\mathbb{1}, \mathbb{1})$.

As those notions are relatively standard, in the core of the paper, we simply present the parts that are of interest for the categorical semantics. We still provide the full definitions in the appendix (**??**), together with proofs of the properties we rely on.

[4]We provide such a matrix-based semantics in **????**.

[5]Semiadditive categories are also called "categories with finite biproducts".

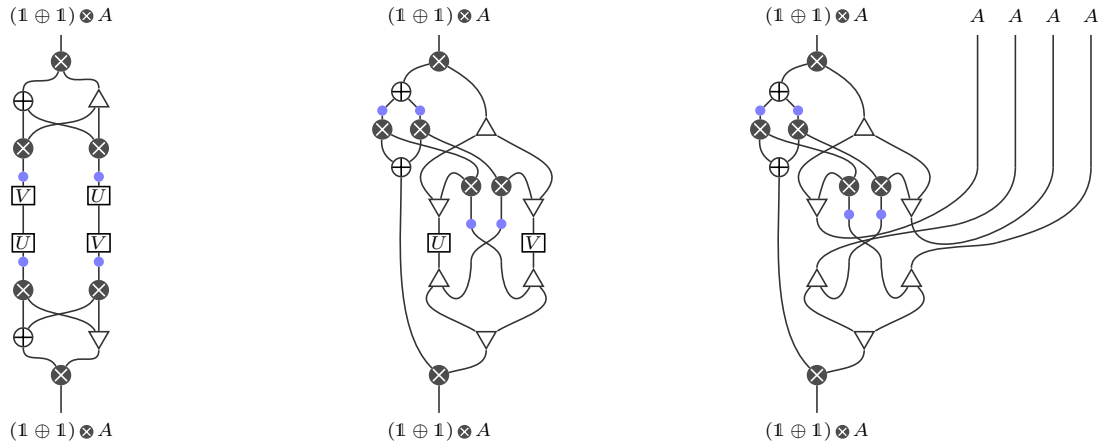


Fig. 4: The diagrams for $\texttt{switch}_{\texttt{U,V}}$ with duplicates, $\texttt{switch}_{\texttt{U,V}}$ with single instances, and $\texttt{switch}$ respectively.

A. The Categorical Framework

Let $\mathbf{H}$ be a category verifying the above three items. $\mathbf{H}$ is enriched over R-semimodules, in other words given two morphisms $f, g : H \to K$ and two scalars $s, t \in R$ we can build the weighted sum $s \cdot f + t \cdot g$ by relying on the isomorphism **scal**. In particular, whenever $R = \mathbb{R}_{\geq 0}$, this can be used to represent probabilistic distribution morphisms. We write **zero** : $H \to K$ for the null morphism, which is the unit of that sum.

Additionally, $\mathbf{H}$ comes with three different symmetric monoidal structures, $\otimes$ representing the "pairing" or data, $\oplus$ representing the "superposition" of multiple potential outcomes, and $|$ informally representing "either $\otimes$ or $\oplus$" and more formally defined as:

$$H \mid K = (H \otimes K) \oplus (H \oplus K)$$

Those operations are bifunctorial. So for $f : H \to K$ and $f' : H' \to K'$, we have:

$$\begin{array}{rcccc} f \otimes f' & : & H \otimes H' & \to & K \otimes K' \\ f \oplus f' & : & H \oplus H' & \to & K \oplus K' \\ f \mid f' & : & H \mid H' & \to & K \mid K' \end{array}$$

Those operations are, up to isomorphism, associative and respectively have $\mathbb{1}$, $\mathbb{0}$ and $\mathbb{0}$ as units. We denote by $m^{\otimes}$ all the isomorphisms obtained from the associators and unitors of $\otimes$ by composing them with $\circ$, $\otimes$, $\oplus$ and $|$. While this notation is ambiguous, Mac Lane's coherence theorem (**??** and [22], [23]) ensures that this ambiguity is never problematic. We define $m^{\oplus}$ and $m^{|}$ similarly.

Those operations are also symmetric, meaning that we have

$$\begin{array}{rcccc} \sigma^{\otimes} & : & H \otimes K & \to & K \otimes H \\ \sigma^{\oplus} & : & H \oplus K & \to & K \oplus H \\ \sigma^{|} & : & H \mid K & \to & K \mid H \end{array}$$

Lastly, $\oplus$ is a biproduct, meaning that we have injections ι, projections π, and diagonals $\Delta = \iota_\ell + \iota_r$ and codiagonals $\nabla = \pi_\ell + \pi_r$:

$$\begin{array}{rcccc} \pi_\ell & : & H \oplus K & \to & H \\ \pi_r & : & H \oplus K & \to & K \\ \nabla & : & H \oplus H & \to & H \end{array} \qquad \begin{array}{rcccc} \iota_\ell & : & H & \to & H \oplus K \\ \iota_r & : & K & \to & H \oplus K \\ \Delta & : & H & \to & H \oplus H \end{array}$$

B. Additional Structures for Non Functional Morphisms

By virtue of $(\mathbf{H}, \oplus, \mathbb{0})$ being a semiadditive category, $\mathbb{0}$ is an initial object. Said otherwise, for any object H, the only morphism $\mathbb{0} \to H$ is **zero**. This actually matches the functional fragment of our language, as the only generator with no input is the Null. However, in the full language, the Unit is a generator with no input that is distinct from the Null. In order to represent that unit, we need to go slightly beyond $\mathbf{H}$.

Definition III.1. *We define* $\mathbf{H}^{\oplus\mathbb{1}}$ *as the category with the same objects as* $\mathbf{H}$ *and for morphisms* $f \in \mathbf{H}^{\oplus\mathbb{1}}(H, K)$*, the morphisms of* $\mathbf{H}(H \oplus \mathbb{1}, K \oplus \mathbb{1})$ *of the form*

$$f = g + (\iota_r \circ \pi_r)$$

for some $g \in \mathbf{H}(H \oplus \mathbb{1}, K \oplus \mathbb{1})$*. In matrix terms, it is a matrix of* $\mathbf{H}(H \oplus \mathbb{1}, K \oplus \mathbb{1})$ *where the bottom-right coefficient is of the form* c**+id** *for* $c \in \mathbf{H}(\mathbb{1}, \mathbb{1})$*. The identity and composition are the same as in* $\mathbf{H}$*.*

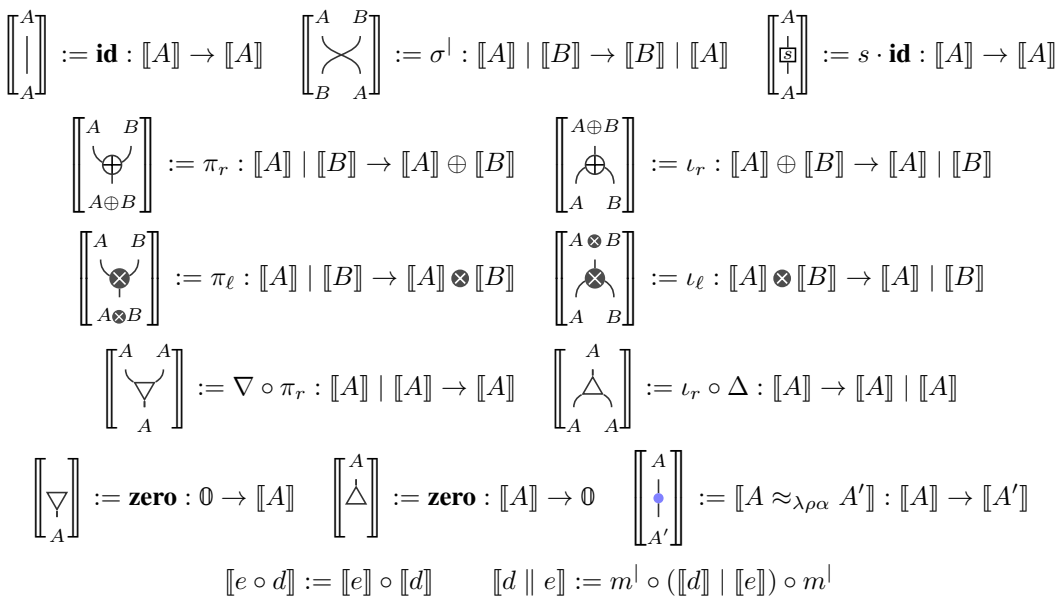
The restriction to morphisms of the shape $g + (\iota_r \circ \pi_r)$ is irrelevant if we have access to negative numbers, that is if our semiring R is actually a ring. However, in absence of negative numbers, Tensor-Plus Calculus has no diagram that actually behaves like the **zero** morphism[6], meaning that we need to exclude the **zero** morphism from $\mathbf{H}^{\oplus\mathbb{1}}$ in order to get universality.

In order to utilize this category, we rely on the following natural isomorphims:

$$\textbf{expand} : (H \oplus \mathbb{1}) \otimes (K \oplus \mathbb{1}) \to (H \mid K) \oplus \mathbb{1}$$

which follow from the distributivity of $\otimes$ over $\oplus$.

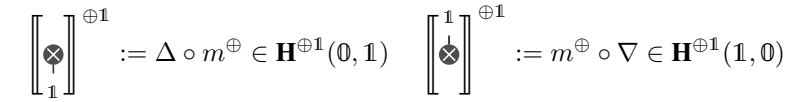
[6]For example, the empty diagram has for semantics that identity of $\mathbf{H}(1, 1)$, so to obtain **zero** one would need to be "less" than empty.



$$[\![e \circ d]\!] := [\![e]\!] \circ [\![d]\!] \qquad [\![d \parallel e]\!] := m^{|} \circ ([\![d]\!] \mid [\![e]\!]) \circ m^{|}$$

Fig. 5: Categorical Semantics for the Functional Tensor-Plus Calculus

$$[\![d]\!]^{\oplus\mathbb{1}} := [\![d]\!] \oplus \mathbf{id}_{\mathbb{1}} \text{ whenever } d \in \mathbf{FTP}^R$$



$$[\![e \circ d]\!]^{\oplus\mathbb{1}} := [\![e]\!]^{\oplus\mathbb{1}} \circ [\![d]\!]^{\oplus\mathbb{1}} \qquad [\![d \parallel e]\!]^{\oplus\mathbb{1}} := (m^{|} \oplus \mathbf{id}_{\mathbb{1}}) \circ \mathbf{expand} \circ ([\![d]\!]^{\oplus\mathbb{1}} \otimes [\![e]\!]^{\oplus\mathbb{1}}) \circ \mathbf{expand}^{-1} \circ (m^{|} \oplus \mathbf{id}_{\mathbb{1}})$$

Fig. 6: Categorical Semantics for the Tensor-Plus Calculus

C. The Semantics

We assume that we have $(\mathbf{H}, \oplus, \mathbb{0}, \otimes, \mathbb{1}, \mathbf{scal})$ as described above. For any color A of $\mathbf{TP}^R$ (or $\mathbf{FTP}^R$), we can give its semantics $[\![A]\!]$ in $\mathbf{H}$ as follows:

$$[\![A \oplus B]\!] := [\![A]\!] \oplus [\![B]\!] \quad [\![\mathbb{0}]\!] := \mathbb{0}$$
$$[\![A \otimes B]\!] := [\![A]\!] \otimes [\![B]\!] \quad [\![\mathbb{1}]\!] := \mathbb{1}$$

We can then extend this semantics to objects as follows:

$$[\![\mathcal{X} \parallel B]\!] := [\![\mathcal{X}]\!] \mid [\![B]\!] \qquad [\![\varnothing]\!] := \mathbb{0}$$
$$= ([\![\mathcal{X}]\!] \otimes [\![B]\!]) \oplus ([\![\mathcal{X}]\!] \oplus [\![B]\!])$$

In order to give a semantics to every diagram, we will start by defining a semantics for the functional fragment $[\![\_]\!] : d \in \mathbf{FTP}^R(\mathcal{X}, \mathcal{Y}) \mapsto [\![d]\!] \in \mathbf{H}([\![\mathcal{X}]\!], [\![\mathcal{Y}]\!])$ and then we generalize it to the whole calculus at the cost of a slightly different target category $[\![\_]\!]^{\oplus\mathbb{1}} : d \in \mathbf{TP}^R(\mathcal{X}, \mathcal{Y}) \mapsto [\![d]\!]^{\oplus\mathbb{1}} \in \mathbf{H}^{\oplus\mathbb{1}}([\![\mathcal{X}]\!], [\![\mathcal{Y}]\!])$.

The semantics for the functional fragment is given in Figure 5, where $m^{|}$ denote the unique morphisms given by Mac Lane's coherence theorem[7] on $(\mathbf{H}, |, \mathbb{0})$ that apply the sequence of associators for $|$ required for the definition to typecheck, and where $[\![A \approx_{\lambda\rho\alpha} A']\!]$ is the unique morphism given by our generalization of Mac Lane's coherence theorem to categories with two monoidal structures (see **??** in the appendices) that applies the sequence of associators for $\otimes$ and $\oplus$, unitors for $\otimes$ and $\oplus$ and their inverses required to obtain a morphism from $[\![A]\!]$ to $[\![A']\!]$. In the semantics, remember that $A \mid B$ stands for $(A \otimes B) \oplus (A \oplus B)$, therefore the morphisms $\pi_r, \pi_l, \iota_l, \iota_r$ act on this type and, for example, we can have $\pi_r : A \mid B \to A \oplus B$ and $\pi_\ell : A \mid B \to A \otimes B$. This semantics is universal:

Theorem III.2 (Universality)**.** *For every object $\mathcal{X}, \mathcal{Y}$, for all $f \in \mathbf{H}([\![\mathcal{X}]\!], [\![\mathcal{Y}]\!])$, there exists a diagram $d \in \mathbf{FTP}^R(\mathcal{X}, \mathcal{Y})$ such that $[\![d]\!] = f$.*

Then, the semantics for the whole calculus is given in Figure 6. For readers preferring matrix notation instead of categorical notations, we provide an equivalent semantics using matrices in the appendices, **????**. The universality result also holds in the general case, with $\mathbf{H}^{\oplus\mathbb{1}}$ instead of $\mathbf{H}$, although we remind the reader that the definition of $\mathbf{H}^{\oplus\mathbb{1}}$ explicitly excludes morphisms such as **zero** when R has no negative element.

[7] See [22], [23], or **??** from the appendices, using the fact that $|$ is a monoidal product as proven in **??** in the appendices.

IV. The Equational Theory

A. The Functional Case

The equivalence relation $\equiv$ is generated by the non-bracketed equations in Figures 7 to 11, which include the up-down mirrored version of each equation. The bracketed equations can be deduced from the non-bracketed ones (see **??**) but are left for convenience. The name of those deduced equations also use square brackets, unless they are simply the left-right mirror of an existing axioms in which case we keep the same name. The most important equations are found in Figure 7:

- ($\otimes$), ($\oplus$) and ($\bot$) show that the Tensor and the Plus act as projections/injections, where plugging one to its mirrored version gives the identity while plugging one to the other gives the Null.
- (0) shows that the color $\mathbb{0}$ is useless as no data can travel through wires of that color.
- ($\sigma\triangledown$) shows that the Contraction is a (co-)commutative operation.
- ($N\otimes$) means that Tensors have no computational effect on the state, other than enforcing that either both inputs are used at the same time, or they are both not used
- ($X\oplus$) means that two Pluses head-to-head have no computational effect on the tokens other than enforcing that at most one input is used.
- ($\oplus \to \triangledown$) shows that the Contraction and the Plus have very similar behaviors.
- (mix) means that $A \parallel B$ is "either $A \otimes B$ or $A \oplus B$".

Then:

- The equations provided in Figures 8 and 9 describe how both the Null and the Contraction can commute with most other constructors.
- The equations in Figure 10 explain how the definition of $\approx_{\lambda\rho\alpha}$ is expressed in the equational theory.
- The equations in Figure 11 explain how the semiring $(R, +, 0, \times, 1)$ is expressed in the equational theory.

Lastly, Figure 13 are a collection of various other equations that can be deduced from the others, included for convenience. Only $[X\oplus \to \triangledown]$ and $[X\triangledown \to \triangledown]$ rely on the Unit, so all the others are in $\mathbf{FTP}^R$.

All these equations were obviously chosen so that their application does not change the semantics of the diagrams:

Proposition IV.1 (Soundness)**.** *For $d, e \in \mathbf{FTP}^R(\mathcal{X}, \mathcal{Y})$, if $d \equiv e$ then $[\![d]\!] = [\![e]\!]$.*

The proof consists in checking that each individual equation is sound with respect to the categorical semantics, which is done in details in **??**.

Soundness tells us that the equational theory captures part of the semantics associated to the graphical language. We can go one step beyond and show that it captures *exactly* the semantics, i.e. the converse of soundness:

Theorem IV.2 (Completeness)**.** *For $d, e \in \mathbf{FTP}^R(\mathcal{X}, \mathcal{Y})$, if $[\![d]\!] = [\![e]\!]$ then $d \equiv e$.*

This is proven through a normal form. We refer to **??** in the appendix for the proof that every morphism can be put in normal form, and to **??** for the proof that if two morphisms share the same semantics, then they share the same normal form.

B. The General Case

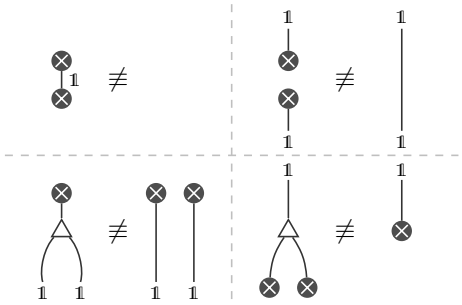
When defining the equational theory for $\mathbf{TP}^R$, we simply take the equational theory for $\mathbf{FTP}^R$ and add the few equations of Figure 12, proven sound in **??**. The last equation, (Can), will be irrelevant as soon as R is cancellative, as it follows that $s = t$ whenever $s + 1 = t + 1$. We write $\mathbf{TP}^R_{\equiv}$ for $\mathbf{TP}^R$ quotiented by $\equiv$.

Quite importantly, we can use $\equiv$ for both equational theories without ambiguity:

Proposition IV.3. *For $d, e \in \mathbf{FTP}^R(\mathcal{X}, \mathcal{Y})$, $d \equiv e$ for the equational theory of $\mathbf{FTP}^R$ if and only if $d \equiv e$ for the equational theory of $\mathbf{TP}^R$.*

Proof. The direct implication follows from the fact that we only added equations. The indirect implication follows from the fact that the semantics of $\mathbf{TP}^R$ is conservative with respect to the semantics of $\mathbf{FTP}^R$, hence it will not equate morphisms that had distinct semantics as morphisms of $\mathbf{FTP}^R$. Since the equational theory of $\mathbf{FTP}^R$ is complete, it means the additional equations do not equate morphisms that are distinct within $\mathbf{FTP}^R_{\equiv}$. □

Note that outside of those equations, the Unit generator is ill-behaved, and in particular assuming a non-trivial R we have the following inequations[8]:



This behavior is actually in line with our semantics, as the completeness result also extends, as proved in **????** using a slight generalization of the previous normal form.

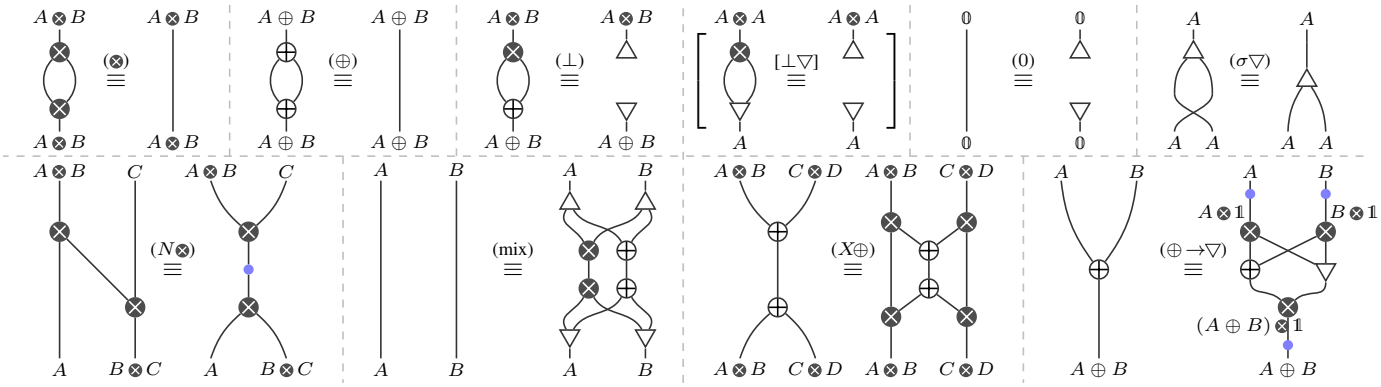
Therefore, soundness (Proposition IV.1) and completeness (Theorem IV.2) extend in this setting.

C. Back to the Examples

In this section, we revisit earlier examples in order to illustrate the equational theory.

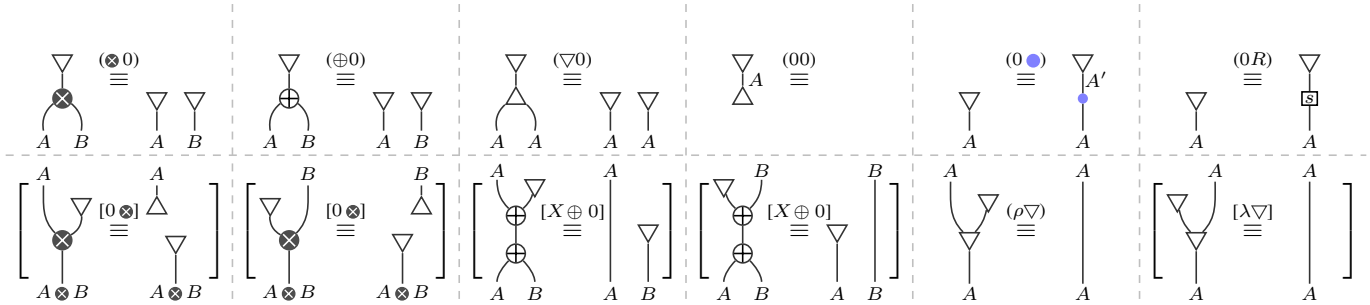
Example IV.4. We start by revisiting Example II.1. In fact, the functional part of the calculus is enough already to express part of the behavior of the ORs, for instance applying a Null

[8]More precisely, the first and last inequations require that R satisfies $1+1 \neq 1$, and the other two require the even weaker property $1 \neq 0$.



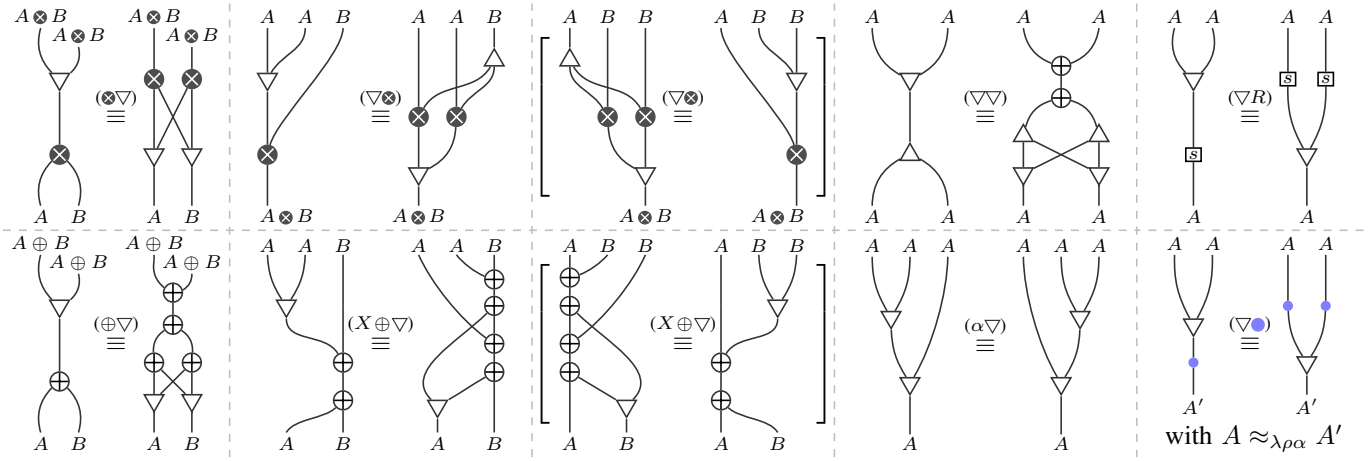
and their up-down mirrored versions.

Fig. 7: Main equations for $\mathbf{FTP}^R$



and their up-down mirrored versions.

Fig. 8: Additional equations describing the commutations of **zero** in $\mathbf{FTP}^R$



and their up-down mirrored versions.

Fig. 9: Additional equations describing the commutations of ▽ in $\mathbf{FTP}^R$

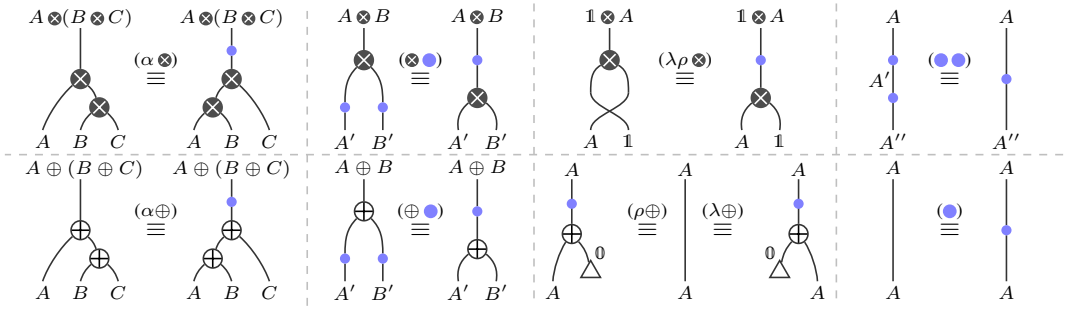


Fig. 10: Additional equations representing $\approx_{\lambda\rho\alpha}$ in $\mathbf{FTP}^R$

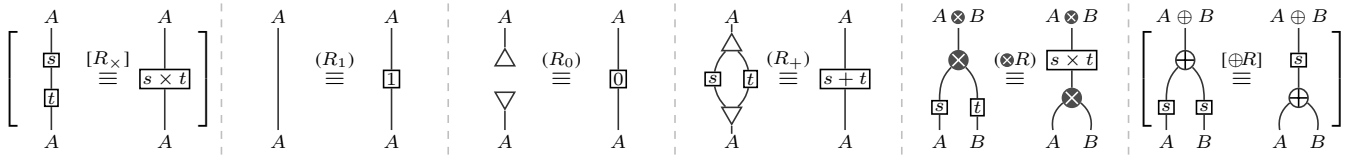


Fig. 11: Additional equations representing the semiring $(R, +, 0, \times, 1)$ in $\mathbf{FTP}^R$

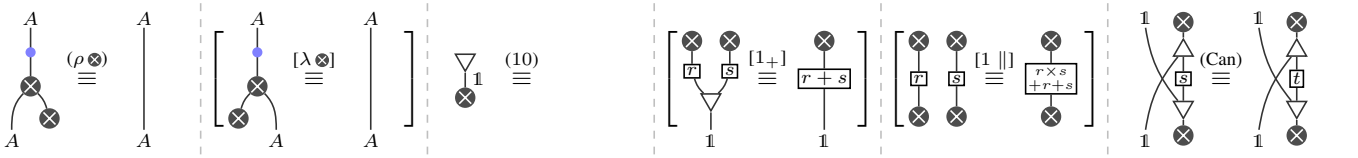


Fig. 12: Additional Equations for the full Tensor-Plus Calculus

to the second input of the strict OR. Since the strict OR forces both inputs to be evaluated to True or False, and since the Null means that the input is not used, the output should not be used either here, i.e. it should be equivalent to throwing away the first input and outputting the Null. This is shown in Figure 14.

In the context of the general, non-functional Tensor-Plus Calculus, we have access to the tensor unit, and we can fully express the behavior of the ORs. For instance, the lazy OR allows the second input to not be evaluated if the first input is evaluated to True. In that case, it should be equivalent to ignoring the second input and outputting True, as in Section IV-B.

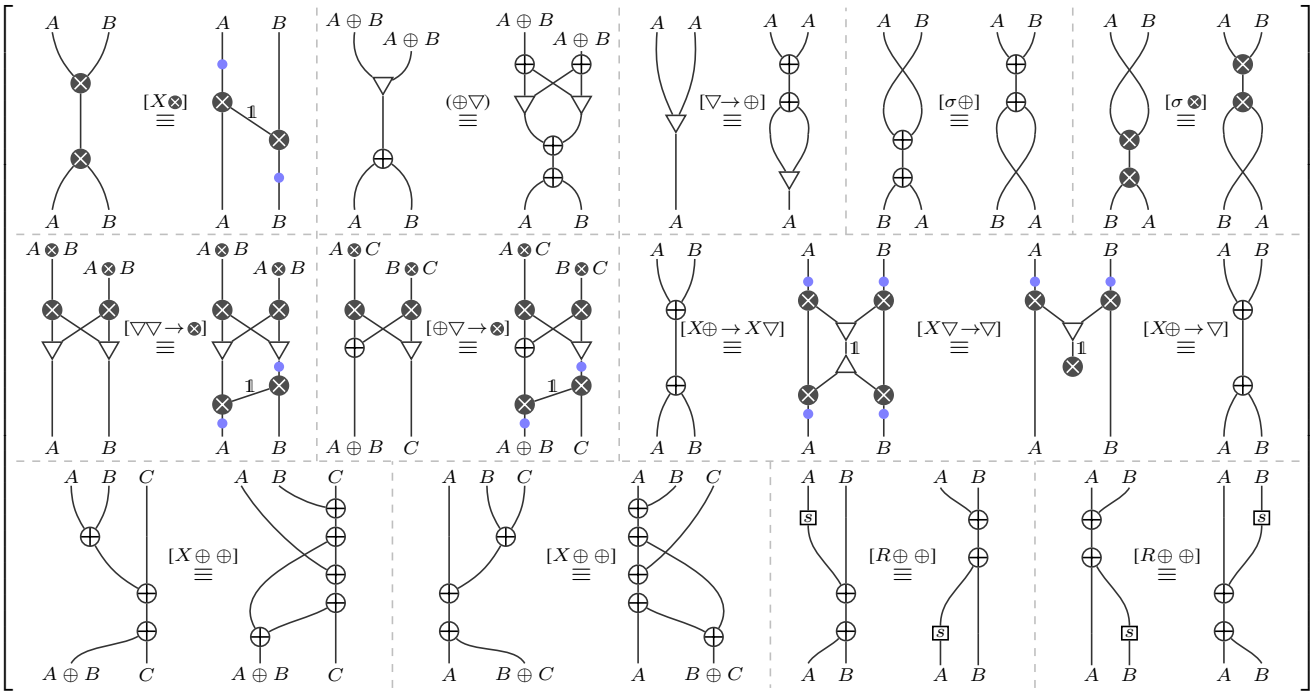
Example IV.5. Let us now revisit Example II.2 and compute the result of the the application of a matrix to a vector: As a first approximation, the two Pluses at the top will eliminate each other, connecting the left-hand side of each Plus together, and similarly for the right-hand side. After that, both the value q and the Unit are duplicated through the Contraction. Finally, p and $\frac{q}{2}$ are summed by the last Contraction. We obtain the final result, corresponding to the vector $\begin{pmatrix} p+\frac{q}{2} \\ \frac{q}{2} \end{pmatrix}$, as shown in Figure 17.

However, while each of the diagrams of the above figure are equivalent to the others, this "first approximation" is only correct because of the presence of the Plus at the very bottom. Without that context, the two head-to-head Pluses would not eliminate each others. In actual practice, instead of eliminating them directly we slide the two head-to-head Pluses downward until they are absorbed by the bottom Plus, as shown in Figure 16.

D. Tensor-Plus Calculus as an Internal Language

In Section III, we built a semantics toward a semiadditive category with an additional symmetric monoidal structure. In this subsection, we claim that **singleFTP**^{R}, the restriction of **FTP**^{R} to morphisms with a single input and a single output, is an *internal language* for "semiadditive categories, with a symmetric monoidal structure distributive over it, and such that the homset of automorphisms over the latter's unit are isomorphic to R", which means the following:

Theorem IV.6. *Let* **singleFTP**^{R} *be the full subcategory of* **FTP**^{R} *with for objects the colors of* **FTP**^{R}*. We can see* **singleFTP**$^R_\equiv$ *as a semiadditive category* $(\mathbf{singleFTP}^R_\equiv, \oplus, \mathbb{0})$ *and a symmetric monoidal category* $(\mathbf{singleFTP}^R_\equiv, \otimes, \mathbb{1})$ *such*



and their up-down mirrored versions.

Fig. 13: A Collection of Various Other Deducible Equations

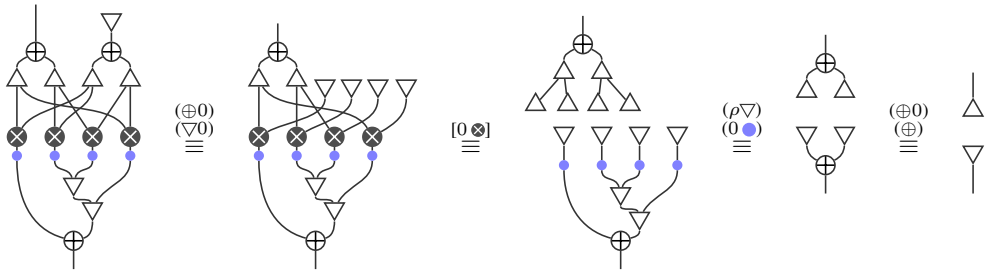


Fig. 14: Rewrite of the strict OR

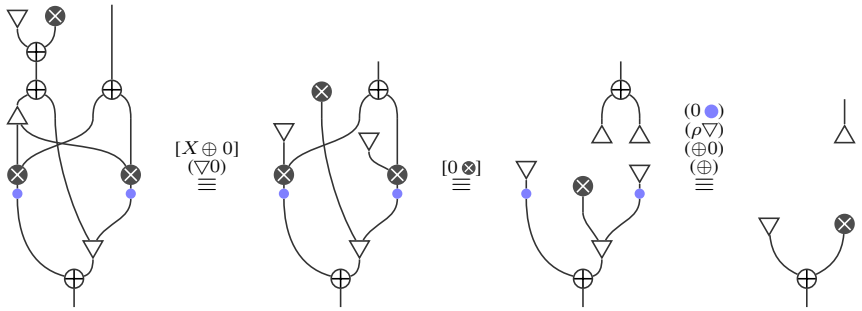


Fig. 15: Rewrite of the lazy OR

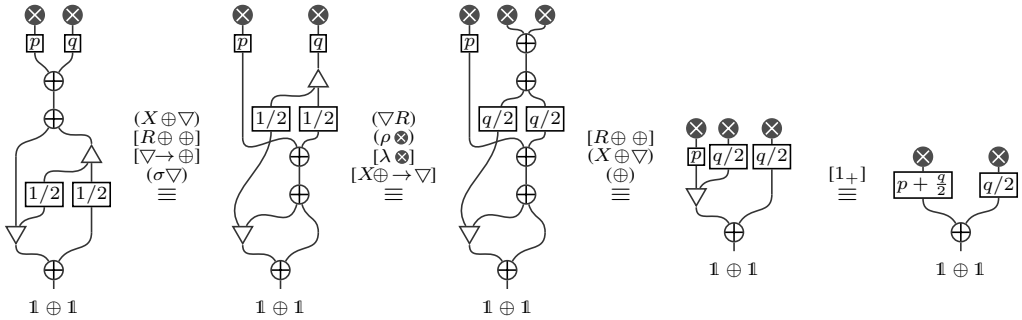


Fig. 16: Rewrite of the Application of a Probabilistic Matrix on a pbit (general case).

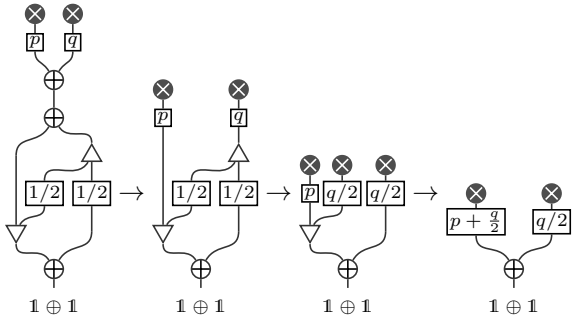


Fig. 17: Probability matrix applied to a pbit (simple case).

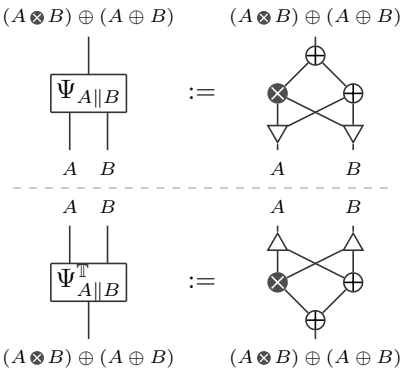


Fig. 18: Isomorphism between $A \parallel B$ and $(A \mid B) := (A \otimes B) \oplus (A \oplus B)$.

that if we consider $\mathbf{H} = \mathbf{singleFTP}^R_{\equiv}$ *(with* $\mathbf{scal}(s) = s$*) then the semantics* $[\![-]\!] : \mathbf{singleFTP}^R_{\equiv} \to \mathbf{H}$ *is the identity.*

We provide a detailed proof in the appendix **??**. For the most part, this is a direct consequence of the completeness result of Theorem IV.2. While the restriction to single input/output might look significant, it is actually minimal as:

Proposition IV.7. *The category* $\mathbf{singleFTP}^R_{\equiv}$ *is equivalent to the category* $\mathbf{FTP}^R_{\equiv}$.

This equivalence follows from the existence of a natural isomorphism $\Psi_{A\|B}$ defined in Figure 18 between the two-colors object $A \parallel B$ and the one-color object $(A \otimes B) \oplus (A \oplus B)$, and is detailed in the appendices **??**. Both results extend to the non-functional case:

Theorem IV.8. *Let* $\mathbf{singleTP}^R$ *be the full subcategory of* $\mathbf{TP}^R$ *with for objects the colors of* $\mathbf{TP}^R$*. We can see* $\mathbf{singleTP}^R_{\equiv}$ *as being equivalent to* $(\mathbf{singleFTP}^R_{\equiv})^{\oplus\mathbb{1}}$*, such that if we consider* $\mathbf{H} = \mathbf{singleFTP}^R$ *(with* $\mathbf{scal}(s) = s$*) then the semantics* $[\![-]\!]^{\oplus\mathbb{1}} : \mathbf{singleTP}^R \to \mathbf{H}^{\oplus 1}$ *is the identity.*

Proposition IV.9. *The category* $\mathbf{singleTP}^R_{\equiv}$ *is equivalent to the category* $\mathbf{TP}^R_{\equiv}$.

V. Conclusion

We introduced a new graphical language unifying additive and multiplicative structure, gave it categorical semantics and an equational theory, and proved the main properties one should expect: universality, soundness, and completeness. We show how this graphical language is an internal language for semiadditive categories with an additional monoidal structure that is distributive over it and parametrized by some algebraic effect.

This language is a first step towards the unification of languages based on the tensor $\otimes$ and those based on the biproduct $\oplus$. The language allows us to reason about both systems in parallel and superpositions of executions, as shown by the encoding of the Switch in Example II.4.

A natural development of the Tensor-Plus Calculus consists in accommodating recursive types in the language. We expect significant difficulties because macros defined by induction on the type – like the cup/caps of Section II-D and the normal form – will no longer be well-defined.

Additional questions also arise when instantiating the language to a specific semiring. For example, we can represent pure quantum computation when considering $R = \mathbb{C}$. However, the question of a complete equational theory for the language on mixed states remains open (one idea could be to rely on the discard construction [24]).

Finally, the study of the graphical interaction between additive and multiplicative connectors has extensive literature within the community of multiplicative-additive proof nets of linear logic [25]. As such, a natural development would be to attempt to split our $\oplus$ and $\otimes$ into two dual connectors each (respectively Plus $\oplus$ and With &, and Tensor $\otimes$ and Par ⅋).

References

[1] J.-Y. Girard, "Linear logic," *Theoretical computer science*, vol. 50, no. 1, pp. 1–101, 1987.

[2] ——, "Proof-nets: the parallel syntax for proof-theory," *Lecture Notes in Pure and Applied Mathematics*, pp. 97–124, 1996.

[3] F. Lamarche, "Quantitative domains and infinitary algebras," *Theoretical computer science*, vol. 94, no. 1, pp. 37–62, 1992.

[4] J. Laird, G. Manzonetto, G. McCusker, and M. Pagani, "Weighted relational models of typed lambda-calculi," in *2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*. IEEE, 2013, pp. 301–310.

[5] M. Pagani, P. Selinger, and B. Valiron, "Applying quantitative semantics to higher-order quantum computing," 2013. [Online]. Available: https://arxiv.org/abs/1311.2290

[6] T. EHRHARD, "Finiteness spaces," *Mathematical Structures in Computer Science*, vol. 15, no. 4, p. 615–646, 2005.

[7] T. Ehrhard, C. Tasson, and M. Pagani, "Probabilistic coherence spaces are fully abstract for probabilistic pcf," *SIGPLAN Not.*, vol. 49, no. 1, p. 309–320, Jan. 2014. [Online]. Available: https://doi.org/10.1145/2578855.2535865

[8] T. Tsukada and K. Asada, "Linear-algebraic models of linear logic as categories of modules over sigma-semirings," in *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science*, ser. LICS '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3531130.3533373

[9] A. Díaz-Caro and O. Malherbe, "A concrete model for a typed linear algebraic lambda calculus," *Mathematical Structures in Computer Science*, vol. 34, no. 1, pp. 1–44, 2024.

[10] K. Chardonnet, M. de Visme, B. Valiron, and R. Vilmart, "The many-worlds calculus," 2023.

[11] C. Comfort, A. Delpeuch, and J. Hedges, "Sheet diagrams for bimonoidal categories," 2020. [Online]. Available: https://arxiv.org/abs/2010.13361

[12] F. Bonchi, A. Di Giorgio, and A. Santamaria, "Deconstructing the calculus of relations with tape diagrams," 2022. [Online]. Available: https://arxiv.org/abs/2210.09950

[13] A. Bucciarelli, T. Ehrhard, and G. Manzonetto, "A relational model of a parallel and non-deterministic λ-calculus," in *Logical Foundations of Computer Science*, S. Artemov and A. Nerode, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 107–121.

[14] M. Pagani, P. Selinger, and B. Valiron, "Applying quantitative semantics to higher-order quantum computing," in *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL'14*, S. Jagannathan and P. Sewell, Eds. ACM, 2014, pp. 647–658.

[15] T. Carette, "Wielding the zx-calculus, flexsymmetry, mixed states, and scalable notations. (manier le zx-calcul, flexsymétrie, systèmes ouverts et limandes)," Ph.D. dissertation, University of Lorraine, Nancy, France, 2021. [Online]. Available: https://tel.archives-ouvertes.fr/tel-03468027

[16] P. Hackney and M. Robertson, "On the category of props," *Appl. Categorical Struct.*, vol. 23, no. 4, pp. 543–573, 2015.

[17] B. Coecke and R. Duncan, "Interacting quantum observables: categorical algebra and diagrammatics," *New Journal of Physics*, vol. 13, no. 4, p. 043016, 2011.

[18] G. Chiribella, G. M. D'Ariano, P. Perinotti, and B. Valiron, "Quantum computations without definite causal structure," *Physical Review A*, vol. 88, no. 2, Aug. 2013. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.88.022318

[19] M. Araújo, F. Costa, and i. c. v. Brukner, "Computational advantage from quantum-controlled ordering of gates," *Phys. Rev. Lett.*, vol. 113, p. 250402, Dec 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.113.250402

[20] M. M. Taddei, J. Cariñe, D. Martínez, T. García, N. Guerrero, A. A. Abbott, M. Araújo, C. Branciard, E. S. Gómez, S. P. Walborn, L. Aolita, and G. Lima, "Computational advantage from the quantum superposition of multiple temporal orders of photonic gates," *PRX Quantum*, vol. 2, no. 1, Feb. 2021. [Online]. Available: http://dx.doi.org/10.1103/PRXQuantum.2.010320

[21] A. A. Abbott, J. Wechs, D. Horsman, M. Mhalla, and C. Branciard, "Communication through coherent control of quantum channels," *Quantum*, vol. 4, p. 333, Sep. 2020. [Online]. Available: http://dx.doi.org/10.22331/q-2020-09-24-333

[22] G. Kelly, "On maclane's conditions for coherence of natural associativities, commutativities, etc." *Journal of Algebra*, vol. 1, no. 4, pp. 397–402, 1964. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021869364900183

[23] A. Joyal and R. Street, "Braided tensor categories," *Advances in Mathematics*, vol. 102, no. 1, pp. 20–78, 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001870883710558

[24] T. Carette, E. Jeandel, S. Perdrix, and R. Vilmart, "Completeness of Graphical Languages for Mixed States Quantum Mechanics," in *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), C. Baier, I. Chatzigiannakis, P. Flocchini, and S. Leonardi, Eds., vol. 132. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, pp. 108:1–108:15. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2019/10684

[25] D. J. D. Hughes and R. J. Van Glabbeek, "Proof nets for unit-free multiplicative-additive linear logic," *ACM Trans. Comput. Logic*, vol. 6, no. 4, p. 784–842, Oct. 2005. [Online]. Available: https://doi.org/10.1145/1094622.1094629