Monoidal Multiplexing

Apiwat Chantawibul and Paweł Sobociński

ECS, University of Southampton

Abstract. Given a classical algebraic structure—e.g. a monoid or group—with carrier set X, and given a positive integer n, there is a canonical way of obtaining the same structure on carrier set X^n by defining the required operations "pointwise". For resource-sensitive algebra (i.e. based on mere symmetric monoidal, not cartesian structure), similar "pointwise" operations are usually defined as a kind of syntactic sugar: for example, given a comonoid structure on X, one obtains a comultiplication on $X \otimes X$ by tensoring two comultiplications and composing with an appropriate permutation. This is a specific example of a general construction that we identify and refer to as *multiplexing*. We obtain a general theorem that guarantees that any equation that holds in the base case will hold also for the multiplexed operations, thus generalising the "pointwise" definitions of classical universal algebra.

Keywords: string diagrams· resource sensitivity · symmetric monoidal categories· props

1 Introduction

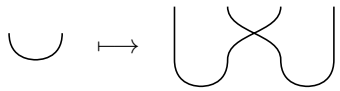
In recent years there has been a significant amount of work that uses *string diagrams* as a compositional syntax for various computational artefacts. A few of the application domains are quantum foundations and quantum computing [1, 12, 13], Petri nets [24, 25], signal flow graphs in control theory [2, 7, 9, 10, 14], electrical circuits [3, 17, 18], game theory [16] and functional programming [19, 23]. In applications, string diagrams are an intuitive, yet formal syntax and often come equipped with an underlying algebraic theory with which one can reason about the specific application domains using *diagrammatic reasoning*. The deeper reason for this trend is that string diagrams are an appropriate graphical representation for the arrows of symmetric monoidal categories, since intuitive topological deformations capture the underlying algebraic laws. But why symmetric monoidal categories?

In categorical universal algebra, following Lawvere [21], categories with finite products are a canonical, categorical setting with which to capture the data of any *classical* algebraic theory. Classical algebraic theories have an implicit assumption that the underlying data is amenable to copying and discarding. Mathematically, this is reflected by the characterisation of cartesian categories as those symmetric monoidal categories where each object is equipped with a cocommutative comonoid structure and all arrows are comonoid homomorphisms [11, 15].

In many applications (e.g. quantum), however, data is not classical. In others (e.g. concurrency, control), it is advisable to make copying and discarding explicit, whenever it is used. This boils down to passing from cartesian categories (Lawvere theories) to mere symmetric monoidal categories (props).

Classical universal algebra dates back to the 1930s, and is a mature subject. On the other hand, "resource-sensitive" universal algebra is still in a state of flux. The upshot of this state of affairs is that the same basic constructions are repeated in different articles, often without a clear picture of their generality. A consolidation effort is only just beginning, e.g. by extending Lawvere's functorial semantics to a suitable class of props [8] and by developing a general theory of rewriting modulo the structure of symmetric monoidal categories [4–6].

An example of a construction that appears in many of the aforementioned applications of string diagrams is "pointwise" definitions: e.g. given an operation such as multiplication ($2 \to 1$), comultiplication ($1 \to 2$), cup ($0 \to 2$) or a cap ($2 \to 0$), it is common to define its n-ary version, i.e. replacing the carrier 1 by n. For instance, two cups can be wired together appropriately to obtain a "2-cup":



In articles, the "obvious" recursive definitions are often given explicitly. Proving that "k-cups" behave as ordinary cups then reduces to a simple induction. This paper is devoted to continuing the consolidation effort through a close examination of such "pointwise" definitions, which we call *monoidal multiplexing*.

We start with an examination of *classical* "pointwise" definitions. A presentation of an algebraic theory is a pair (Σ, E) where Σ is a set of operations, each with an arity, and E is a set of equations between terms constructed from operations and variables. For a concrete example, consider the algebraic theory of a monoid. Its usual presentation is $\Sigma = \{m, e\}$, where m has arity 2 and e arity 0. The set of equations consists of associativity ($m(m(x_1, x_2), x_3) = m(x_1, m(x_2, x_3))$) and unitality ($m(x_1, e) = x_1$, $m(e, x_1) = x_1$). To give a model (a concrete monoid) is to pick a carrier set X and interpretations: $m\colon X^2 \to X$, $e\colon X^0 \to X$, satisfying the required equations, given an implicit universal quantification over the variables that appear within them.

Given $k \in \mathbb{N}$, there is a canonical way to define this structure when the underlying carrier set is X^k: the operation $m{\cdot}k : (X^k)^2 \to X^k$ simply performs m "pointwise" on a pair of k-tuples. So, say letting $k = 3$, the multiplication takes (x_1, x_2, x_3), (y_1, y_2, y_3) to $(m(x_1, y_1), m(x_2, y_2), m(x_3, y_3))$ with unit (e, e, e). This idea is not specific to monoids and can be carried through similarly for any algebraic theory, as we shall see below.

Let us come back to Lawvere theories in more detail: the data of an algebraic theory with presentation (Σ, E) is captured by the Lawvere theory $\mathcal{L}_{\Sigma,E}$. This is a category with finite products where objects are natural numbers; moreover, the categorical product of m and n is $m + n$. A concrete description of an arrow $m \to n$ in $\mathcal{L}_{\Sigma,E}$ is as an n-tuple of terms constructed from operations of Σ

and variables $x_1, x_2, \ldots x_m$, taken modulo the equations of E. Composition is substitution, in the obvious way. An outcome of this is *functorial semantics*: a classical model is a product-preserving functor $\mathcal{L}_{\Sigma,E} \to \mathbf{Set}$.

The "pointwise" construction can be explained concisely using Lawvere theories. Indeed, suppose that $\sigma \in \Sigma$ has arity n and consider some $k \in \mathbb{N}$. Write $n \cdot k$ for $\underbrace{k + k + \cdots + k}_{n \text{ times}}$ for the n-fold product of k in $\mathcal{L}_{\Sigma,E}$; recall that in a Lawvere theory $+$ is the categorical product on objects. The object k is itself a k-fold product of 1; for $1 \leq i \leq k$, denote the ith projection $\pi_i \colon k \to 1$. Note that as a term, π_i is simply the (1-tuple containing the) ith variable $\pi_i = (x_i)$.

Now let $\Pi_i \colon n \cdot k \to n = \underbrace{\pi_i + \pi_i + \cdots + \pi_i}_{n \text{ times}}$, which—concretely—is the n-tuple of variables $(x_i, x_{i+k}, \ldots, x_{i+(n-1)k})$. Together, $\{\Pi_1, \Pi_2, \ldots, \Pi_k\}$ is a complete set of projections of $n \cdot k$. Given this choice of projections, the pointwise definition of σ on k-tuples is the unique arrow $\sigma \cdot k : n \cdot k \to k$ induced by the universal property of products, where for each projection:

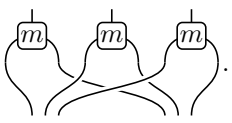
$$\begin{array}{ccc} k & \xrightarrow{\pi_i} & 1 \\ {\scriptstyle \sigma \cdot k}\uparrow & & \uparrow{\scriptstyle \sigma} \\ n \cdot k & \xrightarrow[\Pi_i]{} & n \end{array}$$

The above definition does not rely on the fact that σ is in Σ, and works for any arrow of $\mathcal{L}_{\Sigma,E}$. It is therefore easy to see that it defines a functor

$$(-) \cdot k : \mathcal{L}_{\Sigma,E} \to \mathcal{L}_{\Sigma,E}$$

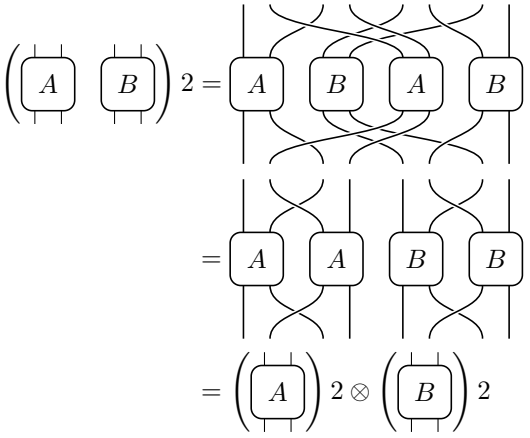
with the heavy lifting taken care of by the fact that $\mathcal{L}_{\Sigma,E}$ has finite products. Now, given a model $M \colon \mathcal{L}_{\Sigma,E} \to \mathbf{Set}$, we obtain a canonical "pointwise" model on k-tuples: $M \circ [(-) \cdot k] \colon \mathcal{L}_{\Sigma,E} \to \mathbf{Set}$.

Although finite products seem to play an important role in the above development, they are in fact not necessary. To see why this is the case it is useful to note that any Lawvere theory is in fact a prop [11]. Then, returning the concrete example of the theory of monoids, $m \cdot 3$ is the string diagram



Our main result is that $(-) \cdot k$ defines a *strict* monoidal functor on any prop, where strict refers to preservation of $\otimes$ on the nose. An example of this for $k = 2$

and the arrow $A \otimes B$, where $A \colon 2 \to 2$ and $B \colon 2 \to 2$, is given below.



In order to define and reason about $(-)\cdot k$ *without* assuming that $\otimes$ is the categorical product, we need to carefully identify the required permutations, which feature in the diagrams above. We rely on the fact that the initial prop is the prop of permutations $\mathbb{P}$, which can be understood as the skeletal version of the category of finite sets and bijections. The latter category embeds faithfully in the category of finite sets and functions, which has both products and coproducts, and whose skeletal version can be presented as the symmetric monoidal theory of commutative monoids $\mathbb{CM}$ [20]. We use the structure of $\mathbb{CM}$ as a useful syntax with which to identify the required permutations.

Structure of the paper After recalling the necessary background definitions and graphical conventions in Section 2, we develop a toolbox of permutations in Section 3. We define the multiplexing operation in Section 4 where we prove our main result, and conclude in Section 5.

2 Preliminaries

2.1 Props

Props or **pro**duct and **p**ermutation categories are special cases of symmetric strict monoidal categories where the objects are generated from repeated monoidal product of a single generator object [22]. The strictness of monoidal categories means that the coherence morphisms (associator, left unitor, and right unitor) that mediate the different ways objects are combined with monoidal product are trivial: they are all identities.

The effect of strictness is that objects in props can be harmlessly identified with finite ordinals where the monoidal product on objects is addition and the monoidal unit is 0. Morphisms between props are symmetric strict monoidal functors, as described in Definition 2 that are, moreover, also identity-on-objects.

A common use of props is as a carrier of the data of an algebraic theory. Such "algebraic" props are often called *symmetric monoidal theories*. They strictly generalise Lawvere theories, which in turn can be identified with *cartesian* props where the monoidal product is also the categorical product.

2.2 Symmetric Monoidal Theory

By a symmetric monoidal theory we mean a prop that is generated from a *presentation*: a pair (Σ, E) of signature set Σ and a equation set E. As opposed to classical presentation, the elements of Σ are equipped with both arity and coarity. A presentation of particular relevance for us is the theory of commutative monoids, which appears at the beginning of Section 3.

2.3 Symmetric monoidal functors

Symmetric monoidal functors are structure-preserving maps between symmetric monoidal categories. They are typically defined with extra conditions ensuring their compatibility with the coherence conditions of monoidal categories. However, since the paper only concerns props which are symmetric strict monoidal categories, there are no further coherence conditions that the monoidal functors need to satisfy. The definition then reduces to:

Definition 1 (symmetric monoidal functor). *Let $\mathbb{C}$ and $\mathbb{D}$ be props. A symmetric monoidal functor $\mathcal{F}\colon \mathbb{C} \to \mathbb{D}$ consists of*

- *a functor*
$$F\colon \mathbb{C} \to \mathbb{D}$$
- *an isomorphism*
$$\varepsilon^{\mathcal{F}}\colon 0 \to F(0)$$
- *a natural isomorphism*
$$\mu^{\mathcal{F}}_{a,b}\colon F(a) \otimes F(b) \to F(a \otimes b)$$
for all objects $a, b \in \mathbb{C}$.

satisfying the preservation of symmetry condition:

$$\mu^{\mathcal{F}}_{b,a} \circ \sigma_{Fa,Fb} = F(\sigma_{a,b}) \circ \mu^{\mathcal{F}}_{a,b}$$

where σ denotes the symmetry natural transformation of the props.

The strictness of symmetric monoidal functors refers to the additional property that the preservation of symmetric monoidal structure is, in fact, on the nose.

Definition 2 (symmetric strict monoidal functor). *A symmetric monoidal functor $\mathcal{F}\colon \mathbb{C} \to \mathbb{D}$ is* strict *if $\varepsilon^{\mathcal{F}}$ is the identity morphism on 0, i.e.,*

$$0 = F(0)$$

and $\mu^{\mathcal{F}}$ is the identity natural transformation, i.e.,

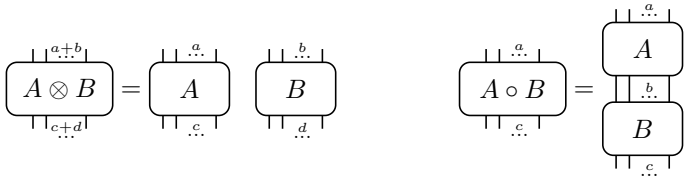
$$F(a) \otimes F(b) = F(a \otimes b)$$

thus satisfying the strict preservation of symmetry condition:

$$\sigma_{Fa,Fb} = F(\sigma_{a,b})$$

2.4 Graphical conventions

Props admit a particularly simple and topologically intuitive string diagrammatic notation. The objects (which, as we previously mentioned, can be considered as finite ordinals) are drawn as an ordered list of wires. We will draw a morphism $A\colon n \to m$ as an A-labelled box with n strings originating from the bottom and m strings coming out from the top. Sometimes, in specific cases such as $\mathbb{CM}$, a custom graphical notation is used instead to represent generators. The monoidal product of two morphisms is represented by juxtaposing two diagrams side-by-side and the composition of two morphisms is drawn by connecting the diagrams with matching number of strings vertically, as shown below.



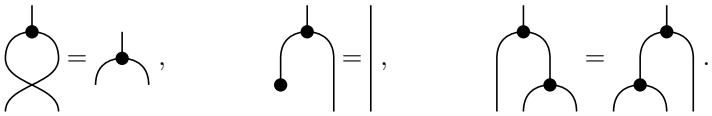
3 Permutations structured by $\mathbb{CM}$

The goal of this section is to assemble a toolbox of definitions and results about permutations, which are needed for a proper account of multiplexing. By permutations in an arbitrary prop $\mathbb{X}$, we refer to the morphisms of $\mathbb{X}$ contained within the image of the unique (but possibly non-faithful) morphism of props $\mathbb{P} \to \mathbb{X}$ where $\mathbb{P}$ is the initial prop which is equivalent to the category of finite ordinals and bijections. To manage the class of relevant permutations, we first note that $\mathbb{P}$ embeds in the prop of commutative monoids $\mathbb{CM}$ which is also equivalent to the category of finite ordinals and (all, i.e. possibly non-monotone) functions.

Remark 1. The embedding $\mathbb{P} \to \mathbb{CM}$ implies that we are able to use $\mathbb{CM}$ as a "sound and complete calculus" for permutations in $\mathbb{P}$ — it is "sound" because equations involving the permutations in $\mathbb{CM}$ are reflected in $\mathbb{P}$ due to faithfulness of the embedding, and it is "complete" because equations involving permutations in $\mathbb{P}$ hold also in $\mathbb{CM}$ due to functoriality. Unlike $\mathbb{P}$, $\mathbb{CM}$ has finite (categorical) products and coproducts which, on objects, are the multiplication and addition of finite ordinals respectively; this is enough structure for description of the class of permutations of interest.

In § 4, we will define the multiplexing operation on prop $\mathbb{X}$ by using the aforementioned class of permutations. Given the above embedding, we are able to do this without loss of generality.

In order to retain the "syntactic-flavour" of working with string diagrams, we use the well-known presentation [20] of $\mathbb{CM}$. The generators of $\mathbb{CM}$ are multiplication  and unit while the commutative monoid equations are:

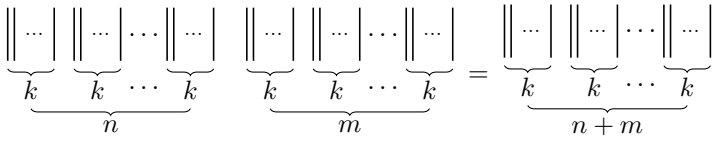


The permutations of interest follow from the universal properties of a *particular* choice of products and coproducts in $\mathbb{CM}$. Of course, the object part of products and coproducts is forced on us since $\mathbb{CM}$ is skeletal: the only choice is the projections and injections. In fact, these are determined by the following two conditions, which follow from usual conventions in diagrammatic reasoning:

1. the monoidal product of $\mathbb{CM}$ is diagrammatically represented by juxtaposing string diagrams side-by-side. Thus the left injection ought to "pick out" the left hand side of the composite diagram, the right the right hand side.
2. the product is *strictly right-distributive* over the coproduct, i.e., the canonical morphism:

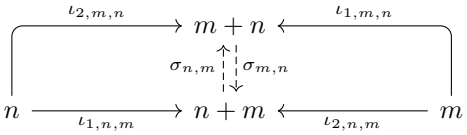
$$n{\cdot}k + m{\cdot}k \longrightarrow (n+m){\cdot}k \tag{SRD}$$

is required to be the identity. Informally, this translates to the identification of the following two ways of grouping of identity string diagrams:



The informal use of ellipses, as above, is part of what this work intends to eliminate.

Coproducts and injections. From the above conditions, the inductive characterisations of projections and injections can be deduced. Fix the notation $\iota_{1,n,m} \colon n \to n+m$ and $\iota_{2,n,m} \colon m \to n+m$ for the left and right injections, respectively. Write $\sigma_{n,m} \colon n+m \to m+n$ for the isomorphism obtained from the universal property (of coproducts), which coincides with the symmetry of $\mathbb{CM}$:

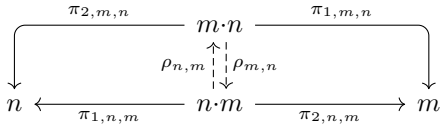


(1)

The left injection can be given inductively as:

$$\begin{aligned} \iota_{1,0,0} &= id_0, \\ \iota_{1,a,b+1} &= \iota_{1,a,b} \otimes \bullet, \\ \iota_{1,1+a,b} &= id_1 \otimes \iota_{1,a,b}. \end{aligned}$$

Products and projections. We fix notation $\pi_{1,m,n} \colon m{\cdot}n \to m$ and $\pi_{2,m,n} \colon m{\cdot}n \to n$ for the left and right projections, respectively. Let $\rho_{n,m} \colon n{\cdot}m \to m{\cdot}n$ be defined by the universal property of products, as illustrated below:



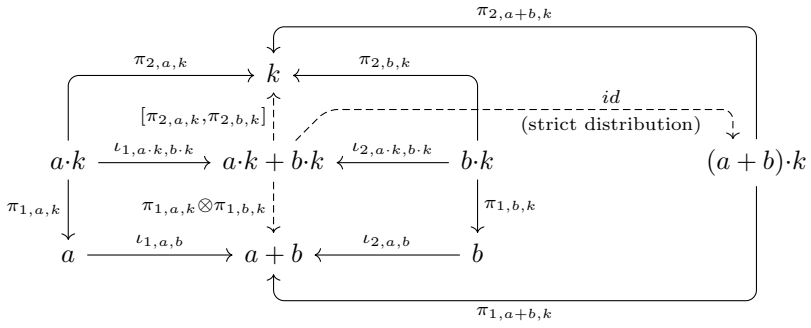
(2)

Note that $\rho_{n,m} = \rho_{m,n}^{-1}$. It is easy to check that $\rho_{m,n}$ is a natural transformation from $-_1{\cdot}-_2 \colon \mathbb{CM} \times \mathbb{CM} \to \mathbb{CM}$ to $-_2{\cdot}-_1 \colon \mathbb{CM} \times \mathbb{CM} \to \mathbb{CM}$. Similarly, the projections are natural transformations from $-_1{\cdot}-_2 \colon \mathbb{CM} \times \mathbb{CM} \to \mathbb{CM}$ to the two projection functors. In subsequent mentions, the subscript of these natural transformations are omitted as they are implied by expressions of source and target objects.

Lemma 1. *The strict right-distributivity condition uniquely determines the inductive characterisation of the left projection π_1 as:*

$$\pi_{1,0,k} = id_0 \tag{3}$$
$$\pi_{1,1,1} = id_1 \tag{4}$$
$$\pi_{1,1,1+k} = \curvearrowleft \circ (id_1 \otimes \pi_{1,1,k}) \tag{5}$$
$$\pi_{1,a+b,k} = \pi_{1,a,k} \otimes \pi_{1,b,k} \tag{6}$$

Proof. Projections given by (3), (4), and (5) are imposed by the universal properties of initial object 0 and terminal object 1. Lastly, (6) follows from fixing identity as the canonical right-distributor in the commutative diagram defining it as seen below:



More explicitly, the derivation starts from noting that strict distribution condition equates the canonical right-distributor with identity:

$$id = (\pi_{1,a,k} \otimes \pi_{1,b,k}, [\pi_{2,a,k}, \pi_{2,b,k}])$$

Post-composing with the first projection of $(a + b)\cdot k$ on both sides results in

$$\pi_{1,a+b,k} = \pi_{1,a,k} \otimes \pi_{1,b,k}$$

□

3.1 Product functor; left and right multiplication

We take a closer look at the product functor that follows from our particular choice of projections and note the intuitive relationship it has with the desired multiplexing operation on arbitrary props. The induced product functor $-_1\cdot-_2 \colon \mathbb{CM}\times\mathbb{CM} \to \mathbb{CM}$ maps $(A \colon a' \to a, B \colon b' \to b)$ to the morphism induced by the universal property of $a\cdot b$:

$$\begin{array}{ccccc} a & \xleftarrow{\pi_1} & a\cdot b & \xrightarrow{\pi_2} & b \\ {\scriptstyle A}\uparrow & & {\scriptstyle A\cdot B}\uparrow & & \uparrow{\scriptstyle B} \\ a' & \xleftarrow{\pi_1} & a'\cdot b' & \xrightarrow{\pi_2} & b' \end{array}$$

Writing id_a as just a for brevity, $A\cdot B$ can be factorised using the universal property of products as:

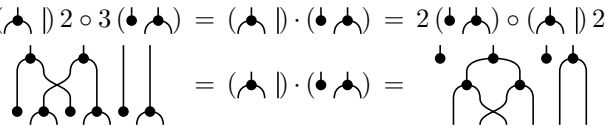
$$\begin{array}{ccc} a\cdot b' & \xrightarrow{a\cdot B} & a\cdot b \\ {\scriptstyle A\cdot b'}\uparrow & \nearrow{\scriptstyle A\cdot B} & \uparrow{\scriptstyle A\cdot b} \\ a'\cdot b' & \xrightarrow{a'\cdot B} & a'\cdot b \end{array} \tag{7}$$

We demonstrate in Lemma 2 that left multiplication $k\cdot(-) \colon \mathbb{CM} \to \mathbb{CM}$ is much easier to describe, namely as "k-fold monoidal product" and thus simple to define in arbitrary props. The right multiplication $(-)\cdot k \colon \mathbb{CM} \to \mathbb{CM}$, however, is used to define the multiplex operation in § 4. To this end, we note the natural symmetry of product ρ can be used to express right multiplication in terms of left-multiplication instead as shown by the following commutative diagram:

$$\begin{array}{ccccccc} b'\cdot a & \xrightarrow{\rho} & a\cdot b' & \xrightarrow{a\cdot B} & a\cdot b & \xleftarrow{\rho} & b\cdot a \\ {\scriptstyle b'\cdot A}\uparrow & & {\scriptstyle A\cdot b'}\uparrow & \nearrow{\scriptstyle A\cdot B} & \uparrow{\scriptstyle A\cdot b} & & \uparrow{\scriptstyle b\cdot A} \\ b'\cdot a' & \xleftarrow{\rho} & a'\cdot b' & \xrightarrow{a'\cdot B} & a'\cdot b & \xrightarrow{\rho} & b\cdot a' \end{array} \tag{8}$$

$$A\cdot B = \rho_{b,a} \circ (b\cdot A) \circ \rho_{a'\cdot b} \circ (a'\cdot B)$$
$$A\cdot B = (a\cdot B) \circ \rho_{b',a} \circ (b'\cdot A) \circ \rho_{a'\cdot b'}$$

The bifunctor $-_1\cdot-_2 \colon \mathbb{CM}\times\mathbb{CM} \to \mathbb{CM}$ provides a useful tool for manipulation of $\mathbb{CM}$ as a string diagram, for example:



(Note that the diagrams are only equal w.r.t. the equational theory of $\mathbb{CM}$.) For any natural number k, the intuitive diagrammatic description of

- left multiplication $k{\cdot}(-)$ is k-fold monoidal product of the argument.
- right multiplication $(-){\cdot}k$ is k copies of the argument 'placed in an overlapping cascade'.

Lemma 2. *Given any $A\colon a' \to a$ in $\mathbb{CM}$, the left multiplication $k{\cdot}(-)\colon \mathbb{CM} \to \mathbb{CM}$ satisfies:*

$$0{\cdot}A = 0 \qquad\qquad k{\cdot}A = A \otimes (k-1){\cdot}A$$

Proof. The first equation is forced by initiality of 0. The second equation follows from strict right-distributivity (SRD) inducing a natural identity:

$$(1 + (k-1)){\cdot}(-) \Rightarrow 1{\cdot}(-) \otimes (k-1){\cdot}(-)$$

3.2 Natural permutations structured by $\mathbb{CM}$

We summarise the relationships between the natural family of permutations structured by $\mathbb{CM}$ here, ready to be transferred into arbitrary props on which multiplexing will be defined.

Let $\xi_{k,a,b}\colon (k{\cdot}a) + (k{\cdot}b) \to k{\cdot}(a+b)$ be the natural isomorphism defined by the canonical left-distribution of product over the coproduct (as opposed to the right-distribution which is required to be identity in (SRD)). Together with the symmetry of coproduct σ, the symmetry of product ρ, and the product functor defined previously, we obtain the following commutative diagram:

$$\begin{array}{ccccccc}
 & & & \overset{k\cdot(\sigma)}{\frown} & & & \\
k{\cdot}(a+b) & \xrightarrow{\xi} & k{\cdot}a + k{\cdot}b & \xrightarrow{\sigma} & k{\cdot}b + k{\cdot}a & \xrightarrow{\xi} & k{\cdot}(b+a) \\
\rho\downarrow & & \rho\otimes\rho\downarrow & & \rho\otimes\rho\downarrow & & \rho\downarrow \\
(a+b){\cdot}k & \xrightarrow{id} & a{\cdot}k + b{\cdot}k & \xrightarrow{\sigma} & b{\cdot}k + a{\cdot}k & \xrightarrow{id} & (b+a){\cdot}k \\
 & & & \underset{(\sigma)\cdot k}{\smile} & & &
\end{array} \tag{9}$$

which commutes because they are all canonical isomorphisms. In the diagram above, we omit the arrowheads and subscripts to emphasise that these are all isomorphisms.

Lemma 3. *ξ has an inductive characterisation with base case $\xi_{0,a,b} = id_0$ and inductive case as shown by the following commutative diagram:*

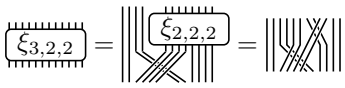
$$\begin{array}{ccc}
(1+k)\cdot(a+b) & \xleftarrow{id} & a+b+k\cdot(a+b) \\
\uparrow{\scriptstyle \xi_{(1+k),a,b}} & & \uparrow{\scriptstyle id_a\otimes id_b\otimes\xi_{k,a,b}} \\
 & & a+b+k\cdot a+k\cdot b \\
 & & \uparrow{\scriptstyle id_a\otimes\sigma_{k\cdot a,b}\otimes id_{k\cdot b}} \\
(1+k)\cdot a+(1+k)\cdot b & \xrightarrow{id} & a+k\cdot a+b+k\cdot b
\end{array}$$

Proof. The lemma is a special case where $n = 1$ of

$$\begin{array}{ccccc}
(n+m)\cdot a+(n+m)\cdot b & \xrightarrow{\xi} & (n+m)\cdot(a+b) & \xrightarrow{id} & n\cdot(a+b)+m\cdot(a+b) \\
\downarrow{\scriptstyle id} & & & & \downarrow{\scriptstyle \xi\otimes\xi} \\
n\cdot a+m\cdot a+n\cdot b+m\cdot b & & \xrightarrow{id_{n\cdot a}\otimes\sigma_{n\cdot b,m\cdot a}\otimes id_{m\cdot b}} & & n\cdot a+n\cdot b+m\cdot a+m\cdot b
\end{array}$$

where the diagram commutes because they mediate canonical ways to distribute $(n+m)\cdot(a+b)$. □

Example 1. From the inductive definition of ξ, the string diagram of $\xi_{3,2,2}$ is:



4 Multiplexing

We have seen in the previous section that $(-)\cdot k\colon \mathbb{CM} \to \mathbb{CM}$ maps diagrams to our desired "pointwise" k-fold version. To define this as a functor on an arbitrary prop $\mathbb{X}$, we define it not through the product functor (which may not exists in $\mathbb{X}$) but through repeated tensor and permutations. With Remark 1 in mind, we abuse the notation and denote permutations in arbitrary props with the same symbols — ξ, ρ, σ — as the corresponding permutations defined in $\mathbb{CM}$.

Definition 3 (multiplexing map). *For an arbitrary prop $\mathbb{X}$, a morphism $A\colon a' \to a$ in $\mathbb{X}$, and any natural number k, define $k\cdot(-)\colon \mathbb{X}[a', a] \to \mathbb{X}[k\cdot a', k\cdot a]$ by recursion as:*

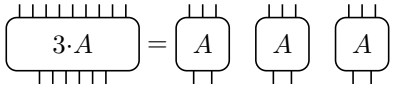
$$0\cdot A = id_0 \qquad\qquad k\cdot A = A \otimes (k-1)\cdot A.$$

Next, we define $(-)\cdot k\colon \mathbb{X}[a', a] \to \mathbb{X}[a'\cdot k, a\cdot k]$ as:

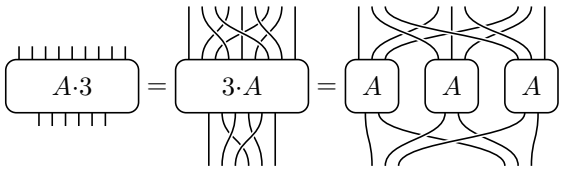
$$A\cdot k = \rho_{k,a} \circ (k\cdot A) \circ \rho_{a',k}$$

where $\rho_{a,k}\colon a\cdot k \to k\cdot a$ denotes the permutation from (2). Call $k\cdot A$ the k-fold monoidal product of A and $A\cdot k$ the k-multiplex of A.

Example 2. Let $A\colon 2 \to 3$ be a morphism in $\mathbb{X}$, then



and



The above defines $k{\cdot}(-)$ and $(-){\cdot}k$ as functions on homsets. On objects, we let $[k{\cdot}(-)](m) = k{\cdot}m = m{\cdot}k = [(-){\cdot}k](m)$.

Lemma 4. *Both $k{\cdot}(-)$ and $(-){\cdot}k$ strictly preserve the monoidal unit, i.e., on objects:*

$$0{\cdot}k = 0 = k{\cdot}0$$

Lemma 5. *In the case $\mathbb{X} = \mathbb{CM}$, the definitions of $k{\cdot}(-)$ and $(-){\cdot}k$ as defined inductively in Definition 3 agree with their definition as a one-argument product functor given in § 3.1.*

Next, we verify that both $k{\cdot}(-)$ and $(-){\cdot}k$ (strictly) preserve composition, i.e., are endofunctors on $\mathbb{X}$ as a plain category.

Lemma 6. *$k{\cdot}(-)$ strictly preserves composition.*

Proof. Using induction on k, the base case is derived by:

$$\begin{aligned}(0{\cdot}A) \circ (0{\cdot}B) &= id_0 \circ id_0 \\ &= id_0 \\ &= 0{\cdot}(A \circ B)\end{aligned}$$

and the inductive case is derived by:

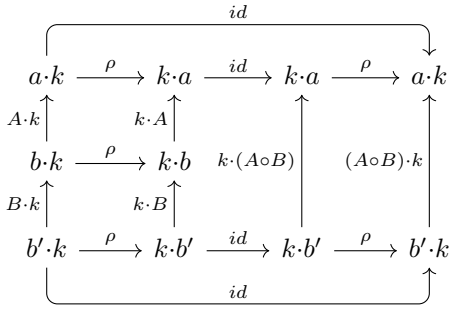
$$\begin{aligned}(k{\cdot}A) \circ (k{\cdot}B) &= (A \otimes (k-1){\cdot}A) \circ (B \otimes (k-1){\cdot}B) && \text{; distributivity} \\ &= (A \circ B) \otimes ((k-1){\cdot}A) \circ ((k-1){\cdot}B) && \text{; interchange law} \\ &= (A \circ B) \otimes ((k-1){\cdot}(A \circ B)) && \text{; hypothesis} \\ &= k{\cdot}(A \circ B) && \text{; distributivity}\end{aligned}$$

□

Lemma 7. *$(-){\cdot}k$ strictly preserves composition, i.e., the following diagram commutes in $\mathbb{X}$ for all $A\colon b \to a$ and $B\colon b' \to b$.*

$$\begin{array}{ccc} a{\cdot}k & \xrightarrow{id} & a{\cdot}k \\ {\scriptstyle A{\cdot}k}\uparrow & & \\ b{\cdot}k & & \uparrow{\scriptstyle (A\circ B){\cdot}k} \\ {\scriptstyle B{\cdot}k}\uparrow & & \\ b'{\cdot}k & \xrightarrow{id} & b'{\cdot}k \end{array}$$

Proof. The following commutes by diagram pasting:



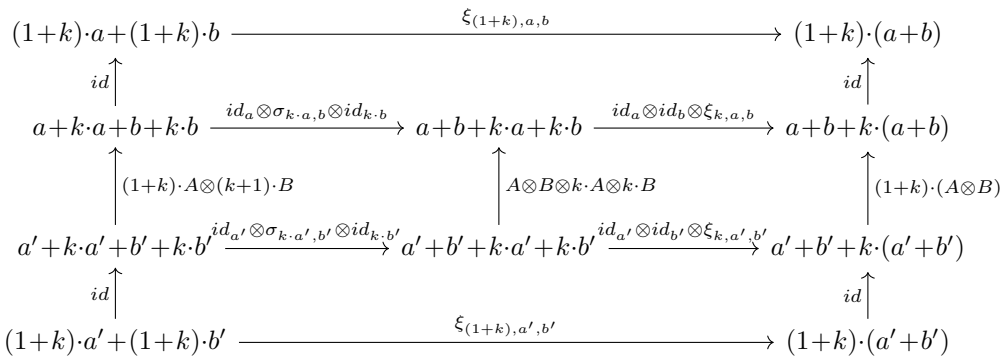
where the middle rectangle commutes by Lemma 6; the top and bottom rectangle commutes by (2); and other rectangles to the side commute as direct consequences of Definition 3. □

The next two results demonstrate a significant difference between $k{\cdot}(-)$ and $(-){\cdot}k$: whereas the former preserves monoidal product only up to isomorphism, the latter preserves it on the nose.

Lemma 8. *$k{\cdot}(-)$ preserves tensor up to isomorphism via the naturality of ξ.*

Proof. The proof is by induction on k, relying on the inductive characterisation of ξ given in Lemma 3. The base case satisfies naturality condition because $(0{\cdot}A) \otimes (0{\cdot}B) = id_0 \otimes id_0 = id_0 = 0{\cdot}(A \otimes B)$ by Definition 3.

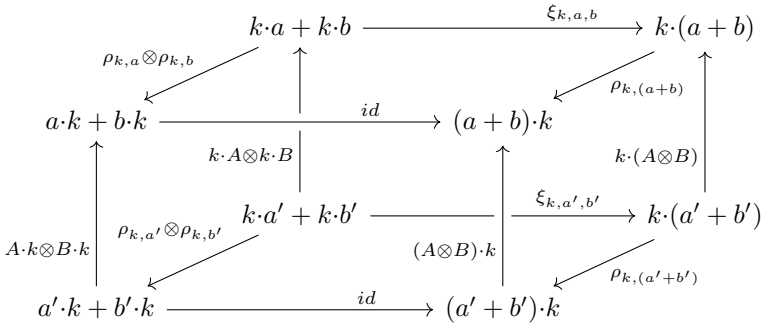
The inductive case is given by the commutativity of the outer perimeter of



which is obtained by pasting commutative diagrams where the top and bottom rectangles commute by Lemma 3; the middle-left rectangle commutes by naturality of symmetry σ; and the middle-right rectangle commutes by induction hypothesis. □

Lemma 9. *$(-){\cdot}k$ strictly preserves tensor.*

Proof. The lemma is represented by the front face of the following diagram



which commutes by diagram pasting with back face from Lemma 8, left and right faces from Definition 3, and top and bottom faces from the left rectangle in (9). □

Lemma 10. *$(-){\cdot}k$ strictly preserves symmetry, i.e., the following commutes:*

$$\begin{array}{ccc} (b+a){\cdot}k & \xrightarrow{id} & b{\cdot}k + a{\cdot}k \\ {\scriptstyle (\sigma_{a,b}){\cdot}k}\uparrow & & \uparrow{\scriptstyle \sigma_{a{\cdot}k,b{\cdot}k}} \\ (a+b){\cdot}k & \xrightarrow{id} & a{\cdot}k + b{\cdot}k \end{array}$$

Proof. Directly from the commutativity of the bottom rectangle in (9). □

Theorem 1. *$(-){\cdot}k$ is a symmetric strict monoidal functor.*

Proof. Follows directly from Lemma 4, Lemma 7, Lemma 9, and Lemma 10. □

The fact that $(-){\cdot}k \colon \mathbb{X} \to \mathbb{X}$ is a strict monoidal functor is the main technical result of our work and it is worthwhile to examine its significance. First, its action on arrows gives us a concise definition of multiplexing: given an arrow $A \colon m \to n$ in $\mathbb{X}$, $A{\cdot}k$ is its k-multiplexed version. Moreover, functoriality means that any equation $A = B$ that holds for arrows in $\mathbb{X}$ will also hold for its k-multiplexed variation, i.e. $A{\cdot}k = B{\cdot}k$. Finally, given a notion of model as a symmetric monoidal functor $\mathbb{X} \to \mathbf{C}$, precomposition with $(-){\cdot}k$ yields a model on which any algebraic structure of $\mathbb{X}$ is defined "pointwise", generalising the situation for classical models outlined in the Introduction.

5 Conclusions and future work

We showed that "pointwise" definitions of classical universal algebra generalise to resource-sensitive theories. Our main result shows that this operation defines a strict monoidal functor $(-){\cdot}k$ on any prop $\mathbb{X}$. By identifying a suitable categorical setting in which to define and reason about the required permutations, we showed that although the similar operation on Lawvere theories seemingly requires the presence of categorical products, they are actually not necessary.

We note that this construction can be extended to braided monoidal categories: in fact, every string diagram for symmetry drawn in this article is already shown in compatible braiding scheme.

Our work fits into the recent trend of consolidating disparate strands of theory and applications of string diagrams in computer science and related fields, and through it, the crystallisation of a "resource-sensitive universal algebra".

References

1. S. Abramsky and B. Coecke. A categorical semantics of quantum protocols. In *Logic in Computer Science (LiCS '04)*. IEEE Press, 2004.
2. J. C. Baez and J. Erbele. Categories in control. Technical report, arXiv:1405.6881, 2014.
3. J. C. Baez and B. Fong. A compositional framework for passive linear networks. *arXiv preprint arXiv:1504.05625*, 2015.
4. F. Bonchi, F. Gadducci, A. Kissinger, P. Sobociński, and F. Zanasi. Rewriting modulo symmetric monoidal structure. In *Thirty-first annual ACM/IEEE symposium on Logic and Computer Science (LiCS 2016)*, pages 710–719, 2016.
5. F. Bonchi, F. Gadducci, A. Kissinger, P. Sobociński, and F. Zanasi. Confluence of graph rewriting with interfaces. In *European Symposium on Programming (ESOP 2017)*, 2017.
6. F. Bonchi, F. Gadducci, A. Kissinger, P. Sobociński, and F. Zanasi. Rewriting with Frobenius. In *Thirty-third annual ACM/IEEE symposium on Logic and Computer Science (LiCS 2018)*, 2018. To appear.
7. F. Bonchi, J. Holland, D. Pavlovic, and P. Sobociński. Refinement for signal flow graphs. Concurrency Theory - 28th International Conference, (CONCUR 2017), 2017.
8. F. Bonchi, D. Pavlovic, and P. Sobocinski. Functorial semantics for relational theories. *arXiv preprint arXiv:1711.08699*, 2017.
9. F. Bonchi, P. Sobociński, and F. Zanasi. Full abstraction for signal flow graphs. In *42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, (POPL 2015)*, pages 515–526. ACM, 2015.
10. F. Bonchi, P. Sobociński, and F. Zanasi. The calculus of signal flow diagrams I: Linear relations on streams. *Inf. Comput.*, 252:2–29, 2017.
11. F. Bonchi, P. Sobociński, and F. Zanasi. Deconstructing lawvere with distributive laws. *Journal of Logical and Algebraic Methods in Programming*, 2018. Accepted for publication.
12. B. Coecke and R. Duncan. Interacting quantum observables. In *ICALP'08*, pages 298–310, 2008.
13. B. Coecke and A. Kissinger. *Picturing Quantum Processes - A first course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017.
14. B. Fong, P. Rapisarda, and P. Sobociński. A categorical approach to open and interconnected dynamical systems. In *Thirty-first annual ACM/IEEE symposium on Logic and Computer Science (LiCS 2016)*, pages 495–504, 2016.
15. T. Fox. Coalgebras and cartesian categories. *Communications in Algebra*, 4(7):665–667, 1976.
16. N. Ghani, J. Hedges, V. Winschel, and P. Zahn. Compositional game theory. In *Thirty-third annual ACM/IEEE symposium on Logic and Computer Science (LiCS 2018)*, 2018. To appear.

17. D. R. Ghica. Diagrammatic reasoning for delay-insensitive asynchronous circuits. In *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky*, pages 52–68. Springer, 2013.
18. D. R. Ghica and A. Jung. Categorical semantics of digital circuits. In *16th Formal Methods in Computer-Aided Design (FMCAD 2016)*, pages 41–48, 2016.
19. R. Hinze. Kan extensions for program optimisation or: Art and dan explain an old trick. In J. Gibbons and P. Nogueira, editors, *Mathematics of Program Construction*, pages 324–362, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
20. S. Lack. Composing PROPs. *Theor. App. Categories*, 13(9):147–163, 2004.
21. F. W. Lawvere. Functorial semantics of algebraic theories. In *Proceedings, National Academy of Sciences*, volume 50, pages 869–873, 1963.
22. S. Mac Lane. Categorical algebra. *Bull. Amer. Math. Soc.*, 71:40–106, 1965.
23. M. Piróg and N. Wu. String diagrams for free monads (functional pearl). In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, ICFP 2016, pages 490–501, New York, NY, USA, 2016. ACM.
24. P. Sobociński. Representations of Petri net interactions. In *Concurrency Theory, 21th International Conference, (CONCUR 2010)*, number 6269 in LNCS, pages 554–568. Springer, 2010.
25. P. Sobociński. Nets, relations and linking diagrams. In *Algebra and Coalgebra in Computer Science - 5th International Conference, (CALCO 2013)*, volume 8089 of *LNCS*, pages 282–298. Springer, 2013.