Categorifying the zx-calculus

Daniel Cicala
University of California, Riverside
Department of Mathematics
cicala@math.ucr.edu

We build a symmetric monoidal and compact closed bicategory by combining spans and cospans inside a topos. This can be used as a framework in which to study open networks and diagrammatic languages. We illustrate this framework with Coecke and Duncan's zx-calculus by constructing a bicategory with the natural numbers for 0-cells, the zx-calculus diagrams for 1-cells, and rewrite rules for 2-cells.

1 Introduction

Compositionality is increasingly becoming recognized as a viable point of view from which to study complex systems such as those found in physics [1], computer science [37], and biology [6]. The focus is on connecting together smaller, simpler systems. The word 'compositionality' suggests the relevancy of category theory. This is indeed the case. In fact, this paper fits into a larger project of establishing suitable categorical frameworks in which to study composable systems [4, 5, 6, 25, 8, 9, 35]. Open diagrams and diagrammatic languages are typical players in compositional approaches. In our context, the adjective 'open' is an established term [29, 31, 35] referring to a structure equipped with chosen inputs and outputs. The primary advantage of open diagrams is the ability to work within a more intuitive syntax. Such diagrams are typically constructed with graph or topological string-like objects. Occasionally, additional data is attached to nodes, edges, or strings as needed. For example, open Markov chains [35] use graphs whose nodes are labeled with a *population* and edges with the rate at which the population of the source node shifts to the target node.

Another common feature shared between diagrammatic languages is the notion of equality. Formal languages are often equipped with a collection of rewrite rules. For us, a rewrite rule is an equivalence relation on diagrams stating when we can replace a diagram D with diagram D'. This is our equality.

Currently, syntax for diagrammatic calculi are usually captured with 1-categories by encoding diagrams as morphisms and diagram connection by composition. As mentioned above, rewrite rules provide a notion of equality between diagrams. However, 1-categorical frameworks squash the information contained in a rewrite rule. That is, there is no way to reconstruct a rewrite rule from an equality. To more fully capture a system, rewrite rules ought to have better representation in our syntax. We accomplish this is by including them as 2-cells in a bicategory.

What should such a bicategory look like? The 0-cells should communicate whether a pair of diagrams can be connected. Taking 0-cells to be sets, then a 1-cell $D\colon x \to y$ is a diagram whose set of inputs is x and set of outputs is y. Hence, we can connect to D any diagram whose inputs are y or outputs are x. The 2-cells $D \Rightarrow D'$ are rules that rewrite D into D'.

To better envision such a bicategory, consider a hypothetical system modeled by a directed graph D. Suppose that we would like D to have inputs I and outputs O, each subsets of the D-nodes. We can build this information into a cospan $I \to D \leftarrow O$ of graphs by taking I and O to be edgeless graphs.

Submitted to:
QPL 2017

Our 1-cells are cospans like this. This means that the inputs and outputs are 0-cells. Rewrite rules are included through *double pushout rewriting* [17]. This presents a rule rewriting D to D' as a span of graphs $D \leftarrow K \rightarrow D'$, through some intermediary graph K. As 2-cells in our bicategory, rewrite rules are isomorphism classes of spans of cospans. These are depicted in Figure 2. Kissinger [27] also modeled rewriting using spans of cospans under the term *cospan rewrites*.

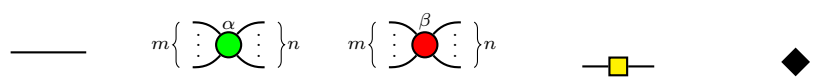
The author proved that this construction actually gives a bicategory [8]. In particular, starting with a topos T, there is a bicategory $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$ whose 0-cells are the T-objects, 1-cells are cospans in T, and 2-cells are isomorphism classes of monic legged spans of cospans in T. A topos and monic span legs are required to ensure that the interchange law holds. This is not overly restrictive, because the spans used in double pushout rewriting are often assumed to have monic legs [24]. Though not discussed in that paper, we can bypass both needs by taking coarser classes of 2-cells. Specifically, we consider the bicategory $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ where $\mathbf{C}$ is a category with finite limits and colimits. This differs from $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$ by taking 2-cells to be all spans of cospans up to *having the same domain and codomain*.

The reason for constructing $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$ and $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ is to provide syntactic bicategories for diagrammatic languages. Which bicategory we use depends on the nature of the diagrammatic language of interest. Regardless of which bicategory we use, we typically start by letting $\mathbf{T}$ or $\mathbf{C}$ be the topos **Graph** of directed graphs or, perhaps, the topos consisting of some other flavor of graphs. For now, we look at $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{Graph}))$ and $\mathbf{Sp}(\mathbf{Csp}(\mathbf{Graph}))$ and consider, in each, the sub-bicategory that is 1-full and 2-full on the edgeless graphs. The 1-cells this sub-bicategory are open graphs and the 2-cells are ways to rewrite one open graph to another. Because we are currently painting with broad strokes, distinguishing between this sub-bicategory in $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{Graph}))$ or $\mathbf{Sp}(\mathbf{Csp}(\mathbf{Graph}))$ is inconsequential. Hence, we commit the sin of referring to this bicategory as **Rewrite** regardless of where it lives.

Suppose we have a diagrammatic language L given by some presentation. We must find a suitable way to identity the given generators and relations of L with 1-cells and 2-cells, respectively, of **Rewrite**. These 1-cells and 2-cells, in turn, generate a sub-bicategory of **Rewrite** that gives a bicategorical syntax for L. It was shown by the author and Courser [9] that $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$ is symmetric monoidal and compact closed in the sense of Stay [40]. We employ a similar argument showing the same is true of $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$. Therefore **Rewrite** and all of sub-bicategories we generate within **Rewrite** are symmetric monoidal and compact closed.

Due to using isomorphism classes for 2-cells instead of any other equivalence classes, it would seem that beginning with $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$ is the natural construction. Indeed, it is suitable for working with systems admitting a graphical syntax such as the open Markov processes mentioned above. However, systems whose syntax has topological information, like string diagrams, introduce the challenge of conveying topological information with only graphs. To contend with this problem, we begin with $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ because it has a 2-cell not present in $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$. This 2-cell rewrites an edge into a single node, thus forces an analogy between an edge and a string that behaves like an identity. As we will see, this rewrite rule is given by a span with a non-monomorphic leg, leading us to use $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ instead of $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$.

The purpose of this paper is to illustrate our framework with the zx-calculus. The backstory of the zx-calculus dates to Penrose's tensor networks [34] and, more recently, to the relationship between graphical languages and monoidal categories [26, 38]. Abramsky and Coecke capitalized on this relationship when inventing a categorical framework for quantum physics [1]. Soon after, Coecke and Duncan introduced a diagrammatic language in which to reason about *complementary quantum observables* [10]. After a fruitful period of development [13, 16, 20, 21, 22, 33], a full presentation of the zx-calculus was published



(a) Wire (b) Green spider (c) Red spider (d) Hadamard (e) Diamond

Figure 1: Generators for the category **zx**

(a) A span of cospans (b) A span of cospans morphism

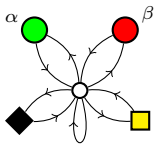
Figure 2: A generic 2-cell in $\mathbf{Sp}(\mathbf{Csp}(C))$

[11]. The completeness of the zx-calculus for stabilizer quantum mechanics was later shown by Backens [3].

The zx-calculus begins with the five diagrams depicted in Figure 1. The dangling wires on the diagrams' left are *inputs* and those on the right are *outputs*. By connecting inputs to outputs, we can form larger diagrams. Formalizing this perspective, we let these diagrams generate the morphisms of a dagger compact category **zx** whose objects, the non-negative integers, count the inputs and outputs of a diagram. Section 2 contains a presentation of **zx** along with a brief discussion on the origins of the generating morphisms (Figure 1) and relations (Figure 3). We also mention relevant software, Quantomatic [1, 28] and Globular [7].

Our goal in this paper is to generate a symmetric monoidal and compact closed (SMCC) bicategory $\underline{\mathbf{zx}}$ that provides a syntax for the zx-calculus. Our first steps towards constructing $\underline{\mathbf{zx}}$ is in Section 3 where we fit open graphs into an SMCC bicategory. To this end, we slightly modify recent work by Courser and the author [8, 9] in order to produce an SMCC bicategory with graphs as 0-cells, cospans of graphs as 1-cells, and certain equivalence classes of spans of cospans of graphs as 2-cells (see Figure 2). As discussed above, this has an SMCC sub-bicategory **Rewrite** that provides an ambient space in which to generate systems modeled on open graphs.

However, this version of **Rewrite** does not contain everything we need. In Section 4, we fill the gap by introducing *open graphs over* S_{zx}. That is, we pick a graph S_{zx}



$\alpha, \beta \in [-\pi, \pi)$

whose nodes coincide with the node types found in the zx-calculus diagrams, with one exception: the white node in the center. This node replaces the dangling edges in the zx-diagrams. A graph morphism $G \to S_{\text{zx}}$ then corresponds to a zx-morphism by transporting the node types to G via the fibres of the map. In his thesis [27], Kissinger also colored graphs this way. We then form an SMCC bicategory with graphs over S_{zx} as 0-cells, cospans of graphs over S_{zx} as 1-cells, and spans of cospans as 2-cells. These spans

of cospans are taken up to the equivalence relation obtained by relating 2-cells with the same domain and codomain. In analogy to the formation of **Rewrite**, we find a sub-bicategory **zxRewrite** that can be thought of as containing all open graphs over S_{zx} and their rewrites.

The bicategory **zxRewrite** is a space in which we can generate SMCC sub-bicategories. In Section 5, we give a presentation for a sub-bicategory $\underline{\mathbf{zx}}$ of **zxRewrite** whose 1-cells correspond to zx-calculus diagrams and 2-cells to the relations between them. After constructing $\underline{\mathbf{zx}}$, we decategorify it to a 1-category $||\underline{\mathbf{zx}}||$ by identifying 1-cells whenever there is a 2-cell between them. Though this seems asymmetrical, we actually get an equivalence relation because of the dual nature of spans. In the main result, Theorem 5.4, we construct a dagger compact functor $||\underline{\mathbf{zx}}|| \to \mathbf{zx}$ witnessing an equivalence of categories. It is in this sense that we are categorifying the zx-calculus.

The author would like to thank John Baez for many helpful ideas and discussions that contributed to this paper. A debt of gratitude is also owed to three anonymous referees for their insightful comments on an earlier version of this paper written for the 2017 Quantum Physics and Logic conference in Nijmegen, Netherlands.

2 The zx-calculus

One of the most fascinating features of quantum physics is the incompatibility of observables. Roughly, an observable is a measurable quantity of some system, for instance the spin of a photon. Incompatibility is in stark contrast to classical physics where measurable quantities are compatible in that, we can obtain arbitrarily precise values at the same time. Arguably, the most famous example of incompatibility is Heisenberg's uncertainty principal which places limits to the precision that one can simultaneously measure a pair of observables: position and momentum. There are different levels of incompatibility amongst pairs of observables. When such a pair is maximally incompatible, meaning that knowing one with complete precision implies total uncertainty of the other, we say they are *complementary observables*.

Hilbert spaces are, historically, the typical framework in which one might study observables. This formalism has been quite successful despite involving difficult calculations and non-intuitive notation.

The zx-calculus was developed by Coecke and Duncan [11] as a high-level language to facilitate such computation between complementary observables. It was immediately used to generalize both *quantum circuits* [32] and the *measurement calculus* [18]. Its validity was further justified when Duncan and Perdrix presented a non-trivial method of verifying measurement-based quantum computations [21]. At its core, the zx-calculus is an intuitive graphical language in which to reason about complementary observables.

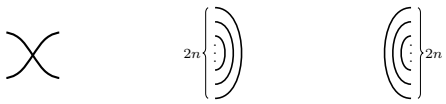
The five *basic diagrams* in the zx-calculus are depicted in Figure 1 and are to be read from left to right. They are

- a *wire* with a single input and output,
- *green spiders* with a non-negative integer number of inputs and outputs and paired with a phase $\alpha \in [-\pi, \pi)$,
- *red spiders* with a non-negative integer number inputs and outputs and paired with a phase $\beta \in [-\pi, \pi)$,
- the *Hadamard node* with a single input and output, and
- a *diamond node* with no inputs or outputs.

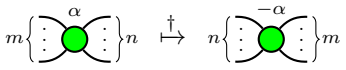
The wire plays the role of an identity, much like a plain wire in an electrical circuit, or straight pipe in a plumbing system. The green and red spiders arise from a pair of complementary observables. Incredibly, observables correspond to certain commutative Frobenius algebras A living in a dagger symmetric monoidal category $\mathbf{C}$. Moreover, a pair of complementary observables gives a pair of Frobenius algebras whose operations interact via laws like those of a Hopf algebra [14, 15]. This is particularly nice because Frobenius algebras have beautiful string diagram representations. If I is the monoidal unit of $\mathbf{C}$, there is an isomorphism $\mathbf{C}(I,A) \to \mathbf{C}(A,A)$ of commutative monoids that gives rise to a group structure on A known as the *phase group*. The spider phases arise from this group. The Hadamard node embodies the Hadamard gate. The diamond is a scalar obtained when connecting a green and red node together. A deeper exploration of these notions goes beyond the scope of this paper. For those interested, the original paper on the topic [11] is an excellent place read more.

In the spirit of compositionality, we present a category **zx** below whose morphisms are generated by the five basic diagrams. To anticipate the shift in terminology, we will refer to zx-calculus diagrams as **zx**-*morphisms* and continue to use the qualifier '*basic*' in the same way.

Observe that there is a non-negative number of wires dangling on the left and right side of each basic **zx**-morphism. Those on the left, we call *inputs* and those on the right *outputs*. These basic **zx**-morphisms generate the morphisms of a dagger compact category **zx** whose objects are the non-negative integers. This category was introduced by Coecke and Duncan [11] and further studied by Backens [3]. To compose in **zx**, connect compatible diagrams along an enumeration of the the inputs and the outputs. A monoidal structure is given by adding numbers and taking the disjoint union of **zx**-morphisms. Relations between the morphisms are given below, but we note here that the wire is the identity on 1. The identity on n is the disjoint union of n wires. The symmetry and compactness of the monoidal product provide a braiding, evaluation, and coevaluation morphisms: respectively,



The evaluation and coevalutation maps are of type $2n \to 0$ and $0 \to 2n$ for each object $n \geq 1$ and the empty diagram for $n = 0$. On the spider diagrams, the dagger structure swapps inputs and outputs then, multiplies the phase by -1:



The dagger acts trivially on the wire, Hadamard, and diamond elements.

Thus far, we have a presentation for a free dagger compact category. However, there are relations between **zx**-morphisms. These are given in Figure 3, though we also include equations obtained by exchanging red and green nodes, daggering, and taking diagrams up to ambient isotopy in 4-space. These listed relations are called *basic*. Spiders with no phase indicated have a phase of 0. The emergence of these relations goes beyond the scope of this paper and we point the interested reader to the genesis of the zx-calculus [11] for an explanation.

A major advantage of using string diagrams, apart from their intuitive nature, is that computations are more easily programmed into computers. Indeed, graphical proof assistants like Quantomatic [7, 19] and Globular [7] were tailor made for such graphical reasoning. The logic of these programs are encapsulated by double pushout rewrite rules. However, the algebraic structure of **zx** and other graphical calculi do not contain the rewrite rules as explicit elements. Perhaps, conceiving of rewrite rules as actual elements in the syntax can prove beneficial for software programmers.

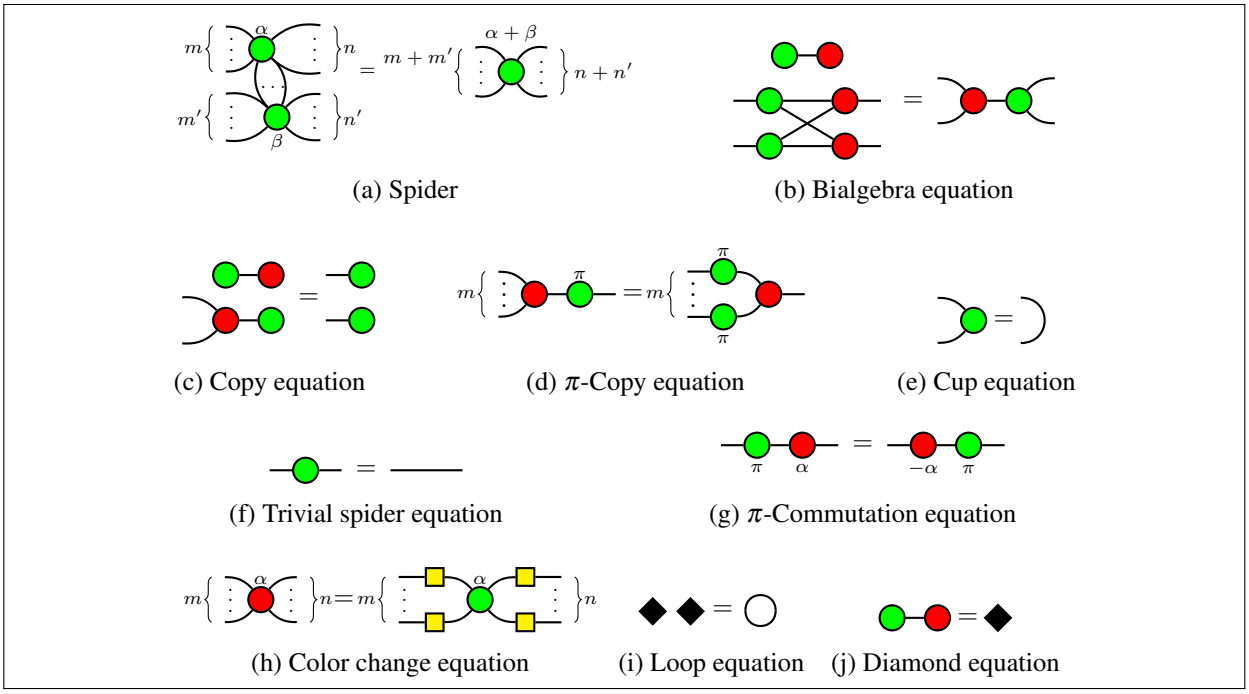
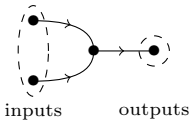


Figure 3: Relations in the category **zx**

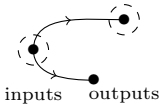
3 Rewriting open graphs

Surely, the **zx**-morphisms are reminiscent of directed graphs. Hence there is a reasonable optimism that we can model the zx-calculus with graphs. However, our hope is tempered by some clear differences between graphs and **zx**-morphisms. For one, graphs do not have inputs or outputs. In this section, we reconcile this particular difference by using open graphs.

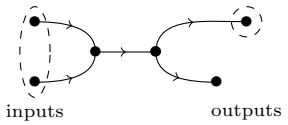
Open graphs and their morphisms have been considered by Dixon, Duncan, and Kissinger [29] though our conceit is slightly different. Conceptually, open graphs are quite simple. Take a directed graph and declare some of the nodes to be inputs and others to be outputs, for example



Given open graphs G and G', if the set of inputs in G and the set of outputs in G' have the same cardinality, we can glue them together along a bijection. This gives a way to turn a pair of compatible open graphs into a single open graph. For instance, to the above open graph, we can connect



to form



We make this precise with cospans and pushouts.

Definition 3.1. Consider the functor $N\colon \mathbf{FinSet}_0 \to \mathbf{Graph}$, on a skeleton of **FinSet**, defined by the letting $N(X)$ be the edgeless graph with nodes X. An *open graph* is then a cospan in the category **Graph** of the form $N(X) \to G \leftarrow N(Y)$ for sets X and Y.

The left leg $N(X)$ of the cospan gives the *input* and the right leg $N(Y)$ the *outputs*. Suppose we have another open graph G' with inputs $N(Y)$ and outputs $N(Z)$. Then we can compose cospans

$$N(X) \to G \leftarrow N(Y) \to G' \leftarrow N(Z).$$

by pushing out over $G \leftarrow N(Y) \to G'$ to get

$$N(X) \to G +_{N(Y)} G' \leftarrow N(Z).$$

By taking isomorphism classes of these pushouts, we obtain a category whose objects are those in the image of N and morphisms are open graphs. But we can do better!

Thus far, we have only just described the first layer of bicategory introduced by the author under the name **Rewrite** [8]. It was shown in a joint work with Courser [9] that **Rewrite** is symmetric monoidal and compact closed. The monoidal structure is induced from the coproduct of graphs. Here, we take Stay's definition of compact closedness for bicategories [40]. As discussed in the introduction, we will work instead with the slightly modified version of **Rewrite** described in Definition 3.3.

Construction on this modified bicategory begins with the theorem below. First, we introduce some needed terminology. A *span of cospans* is a commuting diagram as illustrated in Figure 2. A *parallel class* of spans of cospans is formed by the equivalence relation given by identifying spans of cospans with same domain and codomain.

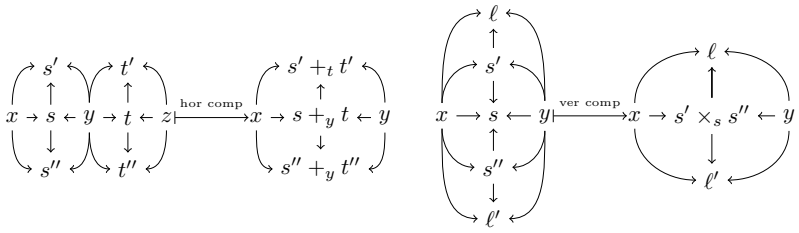
Theorem 3.2. *Let $\mathbf{C} = (\mathbf{C}_0, \otimes, I)$ be a finitely complete and cocomplete (braided, symmetric) monoidal category such that $\otimes$ preserves colimits. There is a (braided, symmetric) monoidal bicategory $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ whose 0-cells are $\mathbf{C}$-objects, 1-cells are cospans in $\mathbf{C}$, and 2-cells are parallel classes of spans of cospans. In case $\mathbf{C}$ is cocartesian, then $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ is also compact closed.*

We prove this theorem in Appendix A. It follows from taking parallel classes of 2-cells that hom-categories in $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ are groupoids.

Using parallel classes of 2-cells instead of isomorphism classes has several advantages. First, it removes two conditions required of $\mathbf{C}$ in $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{C}))$: that $\mathbf{C}$ be a topos and that the legs in the span of cospans be monic. It also allows us, when $\mathbf{C}$ is sufficiently like **Graphs**, to rewrite an edge into a node in analogy to deforming a topological string into a point. Moreover, these 2-cells give us unitary 1-cells.

Recall that there are two ways to compose 2-cells in a bicategory. Horizontal composition in $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$

uses pushouts and vertical composition uses pullbacks:



(1)

Definition 3.3. Define **Rewrite** to be the 1-full and 2-full SMCC sub-bicategory of $\mathbf{Sp}(\mathbf{Csp}(\mathbf{Graph}))$ whose 0-cells are exactly those graphs in the image of the functor $N\colon \mathbf{Set}_0 \to \mathbf{Graph}$.

The conceit of **Rewrite** is that the 1-cells are open graphs whose inputs and outputs are chosen by the 0-cells, and the 2-cells are rewrite rules that preserves the input and output nodes. By rewrite rules, we mean those taken from the double pushout graph rewriting approach [17]. Our open graphs are a different formulation of what amounts to the same concept explored by Dixon, Duncan, and Kissinger [29] though we go a bit further, getting an SMCC bicategory of open graphs instead of a 1-category.

Our motivation for constructing **Rewrite** is not to study it directly, but for it to serve as an ambient context in which to generate SMCC sub-bicategories on some collection of open graphs and rewriting rules. Presenting categories by open graphs and rewrite rules is common enough [29, 23, 35] to warrant finding a common framework in which to fit such categories. However, there are drawbacks to this approach. For example, working with open graphs is only useful to model graphical languages whose terms are equal up to ambient isotopy in 4-space. This limits the current approach to only symmetric monoidal (bi)categories as Selinger's work shows [38].

Employing open graphs is not quite enough for us to fully capture the zx-calculus. We still need to color our open graphs in a way that corresponds to the zx-diagram node types.

4 Open graphs over $S_{\mathbf{zx}}$

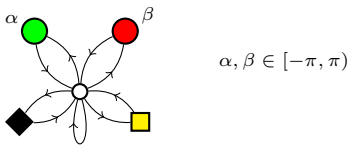
Last section, we began the process of modeling the zx-calculus with graphs by introducing open graphs and fitting them into a bicategory **Rewrite**. This overcame the issue of graphs lacking inputs and outputs. In this section, we face a different issue. Unlike open graphs, **zx**-morphisms have multi-sorted nodes. In this section, we equip open graphs with multi-sorted nodes by working with a slice category of **Graph**. Kissinger used a similar method to give graphs multi-sorted nodes in his thesis [27].

Definition 4.1. Let S be a graph. By a *graph over* S, we mean a graph morphism $G \to S$. A morphism between graphs over S is a graph morphism $G \to G'$ such that the following diagram commutes

$$\begin{array}{ccc} G & \longrightarrow & G' \\ & \searrow \; \swarrow & \\ & S & \end{array}$$

Every graph morphism $G \to G'$ contains a map between corresponding nodes sets. The fibre of this map colors the G-nodes with the G'-nodes. We illustrate this with the following example.

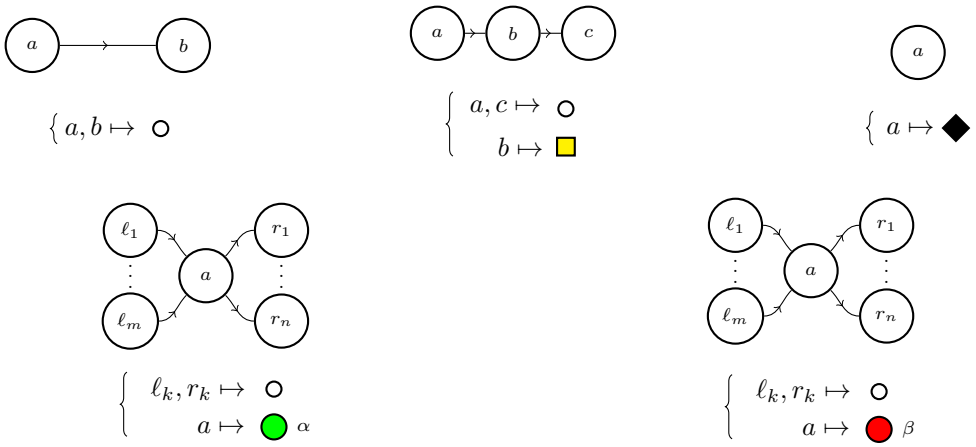
Example 4.2. Let S_{zx} be the graph



(2)

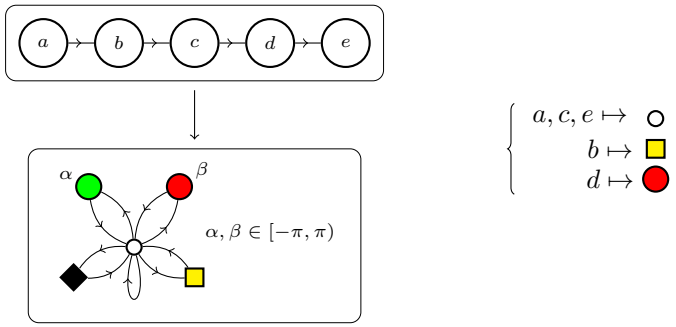
We have not drawn the entirety of S_{zx}. The green and red nodes actually run through $[-\pi, \pi)$ and all of them have a single arrow to and from node ○.

Most of the structure of the basic **zx**-morphisms is captured by graphs over S_{zx}. Consider the following graphs over S_{zx}



where the diagrams give the domain of each graph over S_{zx} and the map is described directly underneath each diagram. The behavior of each map is determined by the image of the nodes because there is at most one arrow between any two nodes in S_{zx}. The role played by each node of S_{zx} in providing our desired structure is evident except, perhaps, for the node ○. Observe that four of the basic **zx**-morphisms have dangling wires on either end. Because edges of directed graphs must be attached to a pair of nodes, we use this node to anchor the dangling edges.

Example 4.3. At this point, we can interpret the basic zx-diagrams as graphs over S_{zx}. This extends nicely to a translation of any **zx**-morphism, such as



which corresponds to the **zx**-morphism

The graphs over S_{zx} in Examples 4.2 and 4.3 capture most of the structure of the basic **zx**-morphisms. The ability to compose is still missing. Composition becomes possible with *open graphs over* S_{zx}. Again, we use cospans to make this precise, though combining these two structure introduces new considerations.

Start with the slice category $\mathbf{Graph} \downarrow S_{\text{zx}}$ of graphs over S_{zx}. By Theorem 3.2, this gives us an SMCC bicategory $\mathbf{Sp}(\mathbf{Csp}(\mathbf{Graph} \downarrow S_{\text{zx}}))$ within which we want to construct a sub-bicategory analogous to **Rewrite**. However, there is a problem. Recall that the objects of **Rewrite** have form $N(X)$ where $N\colon \mathbf{Set}_0 \to \mathbf{Graph}$ is the functor sending a set to the edgeless graph on that set. In **Graph**, there is a unique, up to isomorphism, way to be an edgeless graph. But in $\mathbf{Graph} \downarrow S_{\text{zx}}$, there may be many ways to be edgeless. This depends on the number of graph morphisms to S_{zx}. For instance, a graph with n nodes and no edges can be a graph over S_{zx} in 5^n ways. We rectify this issue by functorially turning a set into an edgeless graph. Recall that $\mathbf{FinSet}_0$ is a skeleton of **FinSet**.

Definition 4.4. Define a functor $N_{\text{zx}}\colon \mathbf{FinSet}_0 \to \mathbf{Graph} \downarrow S_{\text{zx}}$ by

$$X \mapsto (N_{\text{zx}}(X) \to S_{\text{zx}})$$

where $N_{\text{zx}}(X)$ is the edgeless graph with nodes X that are constant over $\circ$. An *open graph over* S_{zx} is a cospan in $\mathbf{Graph} \downarrow S_{\text{zx}}$ of the form

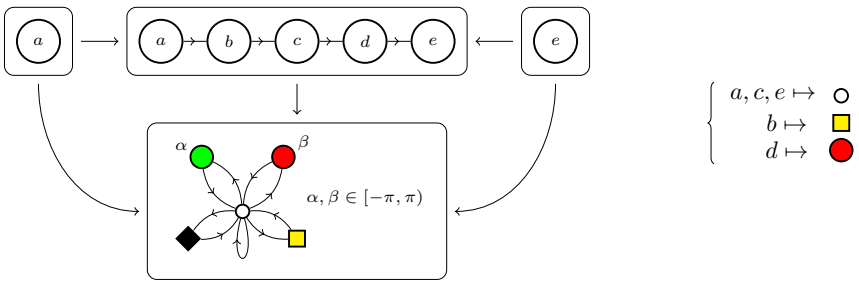
$$N_{\text{zx}}(X) \to G \leftarrow N_{\text{zx}}(Y).$$

With this definition of open graphs over S_{zx}, we propose the analogue to **Rewrite**.

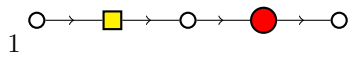
Definition 4.5. Define **zxRewrite** to be the symmetric monoidal and compact closed sub-bicategory of $\mathbf{Sp}(\mathbf{Csp}(\mathbf{Graph} \downarrow S_{\text{zx}}))$ that is 1-full and 2-full on objects of the form $N_{\text{zx}}(X)$ for finite sets X.

Unpacking this definition, the 0-cells of **zxRewrite** are those edgeless graphs over S_{zx} in the image of N_{zx}. The 1-cells are exactly the open graphs over S_{zx}. The 2-cells are the rewritings of one open graph over S_{zx} into that preserve the inputs and outputs. To better understand this bicategory, we give an example of an open graph over S_{zx}. Along with this example, we present a new notation that allow us to draw the remaining diagrams more compactly.

Example 4.6. Consider the graph over S_{zx} in Example 4.3. Make this an open graph as follows:



There is a single input, node a, and a single output, node e. Denote this by

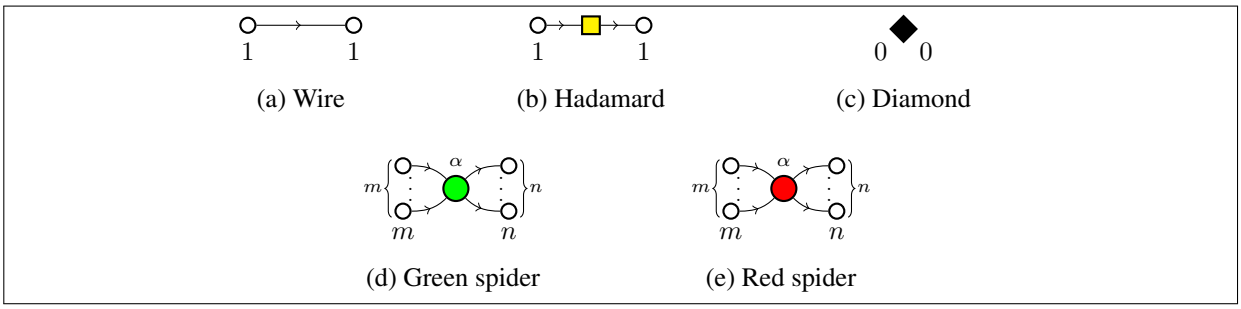


The input nodes are aligned on the far left and the output nodes on the far right. The 1's in the corners refer to the cardinality of the input and output node sets. This may seem unnecessary or redundant, but it will clarify several situations arising later on. Thus, we side with consistency and always write the cardinalities. Of course, this notation strips a fair amount information regarding the graph morphisms involved in the cospan. However, any missing information should be evident in context.

Recall that our interest in **Rewrite** is as an ambient space in which to generate syntactical bicategories for graphical languages. The same is true of our new bicategory **zxRewrite**. Presently, we are interested in 1-cells corresponding to the basic **zx**-morphisms and 2-cells to the basic relations depicted in Figure 3. We claim that the bicategory generated by these 1-cells and 2-cells categorifies **zx**.

5 A categorification of zx

Section 4 describes a translation of the basic **zx**-morphisms into open graphs over S_{zx}. These are depicted in Figure 4.2 and are referred to as *basic open graphs over* S_{zx}. To clarify the double instances of m and n written in the spider diagrams, those below the diagram refer to the cardinalities of the cospan legs, and those beside the brackets count how many nodes are there (cf. Example 4.6).

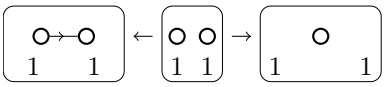


(a) Wire (b) Hadamard (c) Diamond

(d) Green spider (e) Red spider

Figure 4: Generating 1-cells for the bicategory **zx**

Just as the basic open graphs over S_{zx} capture the generating **zx**-morphisms, we must also include the basic relations into our framework. Figure 5 depicts our representation of the basic relations as spans of open graphs. In addition to the basic relations listed explicitly, we add those obtained by exchanging red and green nodes, swapping inputs and outputs, turning the spans around, as well as



(3)

The last is added to ensure that the wire 1-cell behaves as the identity, a property we lose when using graphs. All of these 2-cells, we call *basic*.

It is important to emphasize that the basic 2-cells are representatives of an equivalence class. That is, we have made a decision to present these 2-cells as a span of cospans whose apex is the edgeless graph whose node set is the disjoint union of the inputs and outputs. This is certainly not the only representative we could have chosen, though it does seem to be the most natural choice.

Forget for a moment that our 2-cells are classes and think of only the representatives. When we compose a span of cospans with its dagger, we get a non-trivial way to rewrite a 1-cell into itself. This ought to be distinct from the identity rewrite, which does nothing. However, our choice of equivalence classes render these the same. This hints that a higher rewriting structure is hiding in the background. Indeed, conveying rewrite rules as spans of cospans has the advantage of including higher level rewrite rules in by iterating the process of taking spans. Currently, we content ourselves to work within bicategories and leave an exploration for higher structure for another time.

We now define the bicategory which categorifies the zx-calculus.

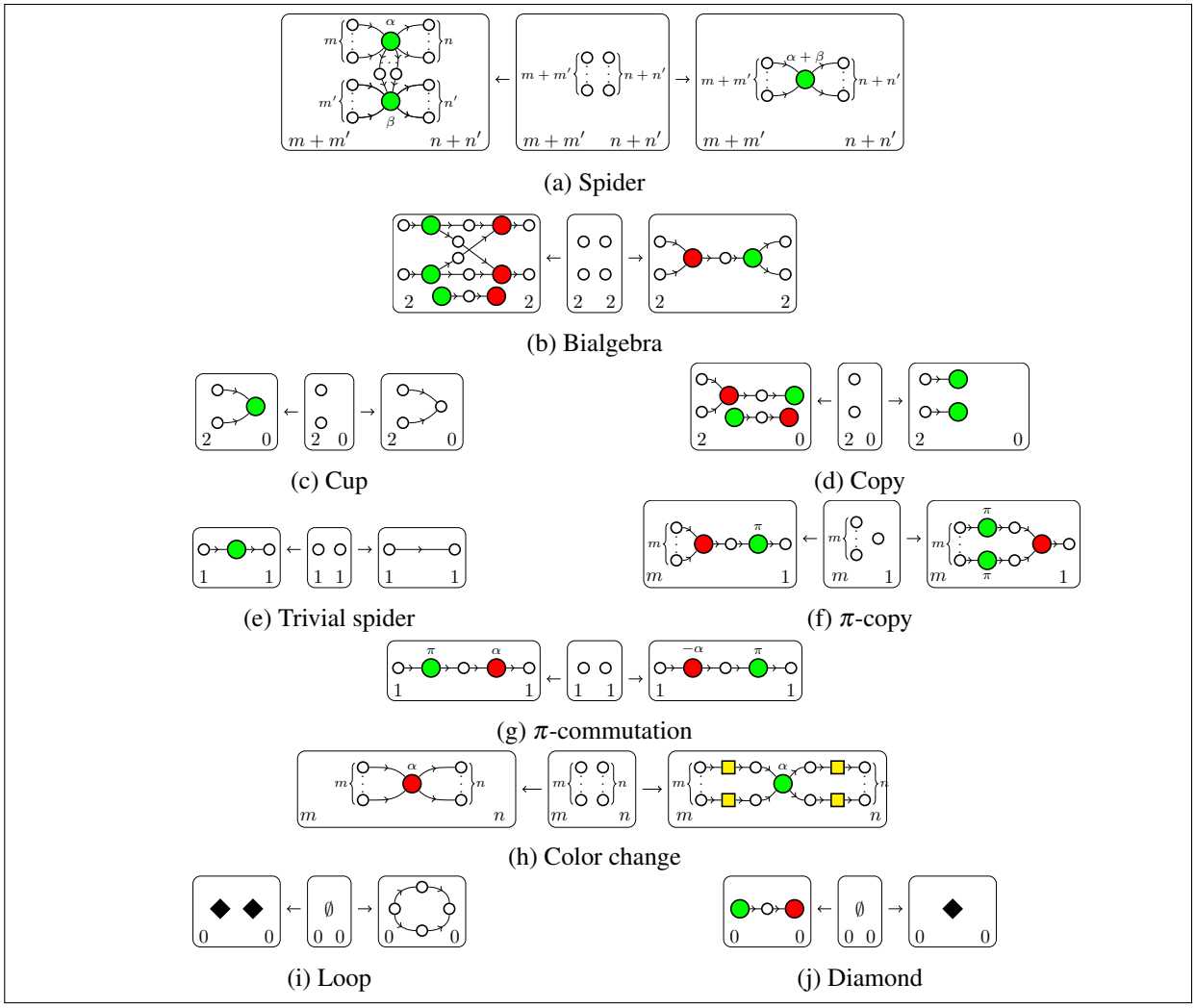
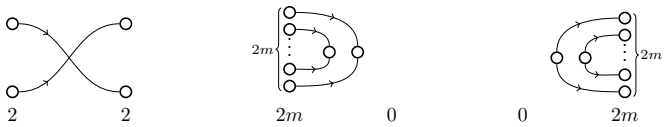


Figure 5: Generating 2-cells for the bicategory **zx**

Definition 5.1. Define **zx** to be the symmetric monoidal and compact closed sub-bicategory of **zxRewrite** generated by the basic 1-cells and basic 2-cells.

Working within **zxRewrite** allows us to generate **zx** as an SMCC in this way. Without having this ambient space, we cannot be sure that we obtain an SMCC bicategory simply by giving a presentation.

Because **zx** is symmetric monoidal and compact closed, it contains twist, evaluation, and coevaluation 1-cells

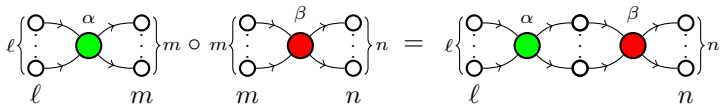


(4)

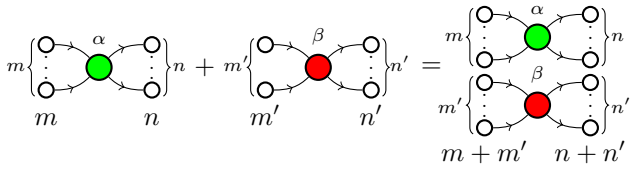
witnessing symmetry and the compact structure on m.

Horizontal and vertical composition are the same as in 1. For example, we can compose spider

diagrams with the same number of inputs and outputs:



We tensor 1-cells by disjoint union, such as

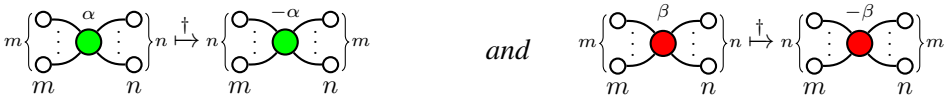


With **zx** defined, we turn our focus towards presenting the main theorem. We start by giving a category that is a *decategorification* or *truncation* of **zx**.

Definition 5.2. Define $||\mathbf{zx}||$ to be the category whose objects are the 0-cells of **zx** and whose arrows are the 1-cells of **zx** modulo the equivalence relation $\sim$ given by: $f \sim g$ if and only if there is a 2-cell $f \Rightarrow g$ in **zx**.

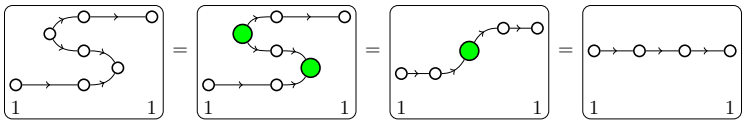
To be clear, $\sim$ is an equivalence relation and doesn't merely generate one. This follows from the symmetry of spans and vertical composition.

Theorem 5.3. *The category* $||\mathbf{zx}||$ *is dagger compact via the identity-on-objects functor* † *given by*



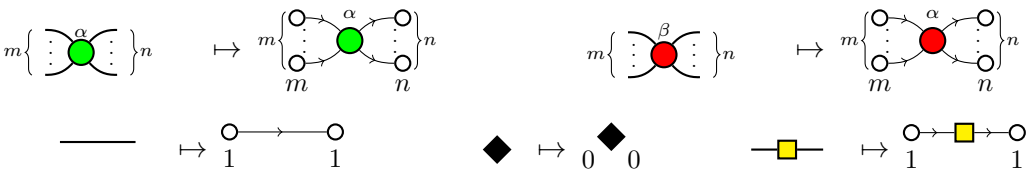
as well as by identity on the wire, Hadamard, and diamond morphisms.

Proof. Compact closedness follows from the self duality of objects via the evaluation and coevaluation maps from (4). The snake equation is derived by



where the equalities follow from the evident 2-cells in **zx**. The extra relation (3) ensures that the string of wires is the identity. Showing that † is a dagger functor is a matter of checking some easily verified details. □

Theorem 5.4. *The identity on objects, dagger compact functor* $E\colon \mathbf{zx} \to ||\mathbf{zx}||$ *given by*



is an equivalence of categories.

Proof. That E is identity-on-objects implies essential surjectivity. Fullness holds because the generating morphisms for $||\underline{\mathbf{zx}}||$ are all in the image of E.

Proving faithfulness is more involved. Let f and g be **zx**-morphisms. Consider representatives $\widetilde{Ef}$, $\widetilde{Eg}$ of Ef, Eg obtained by translating directly the graphical representation of f, g to open graphs over S_{zx} as in Examples 4.3 and 4.6. It suffices to show that the existence of a 2-cell $\widetilde{Ef} \Rightarrow \widetilde{Eg}$ in $\underline{\mathbf{zx}}$ implies that $f = g$.

Observe that any 2-cell α in $\underline{\mathbf{zx}}$ can be written, not necessarily uniquely, as a sequence $\alpha_1 \square \cdots \square \alpha_n$ where each α_i is a basic 2-cell, each box is filled with '$\circ_{\mathrm{h}}$', '$\circ_{\mathrm{v}}$', or '$+$', and parentheses are right justified. By '$\circ_{\mathrm{h}}$' and '$\circ_{\mathrm{v}}$', we mean horizontal and vertical composition. We induct on sequence length. If $\alpha \colon \widetilde{Ef} \Rightarrow \widetilde{Eg}$ is a basic 2-cell, then there is clearly a corresponding basic relation equating f and g. Suppose we have a sequence of length $n+1$ such that the left-most square is a '$+$'. Then we have a 2-cell $\alpha_1 + \alpha_2 \colon Ef \Rightarrow Eg$ where α_1 is a basic 2-cell and α_2 can be written with length n. By fullness, we can write $\alpha_1 + \alpha_2 \colon Ef_1 + Ef_2 \Rightarrow Eg_1 + Eg_2$ where $\alpha_i \colon Ef_i \Rightarrow Eg_i$. This gives that $f_i = g_i$ and the result follows. A similar argument handles the cases when the left-most operation is vertical or horizontal composition. □

6 Conclusion

The main advantage of fitting the zx-calculus into a bicategory is that the rewrite rules are now explicitly included into the mathematical structure. That is, we are now capturing a larger portion of the full picture that is the zx-calculus.

However, categorifying the zx-calculus is only part of the story. The methods used here are general with only slight tweaks made to accommodate the case at hand. Indeed, similarly to how we use **zxRewrite** to capture zx-diagrams, we can construct modified versions of **Rewrite** to frame open Markov chains, resistor networks, internal Frobenius algebras, etc into this framework.

A A symmetric monoidal bicategory of spans of cospans

In this section, we prove Theorem 3.2. Let $\mathbf{C} = (\mathbf{C}_0, \otimes, I)$ be a finitely complete and cocomplete (braided, symmetric) monoidal category such that $\otimes$ preserves colimits. The category **Graph** together with its coproduct is one example. The proof consists of two parts: that $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ is a (braided, symmetric) monoidal bicategory and that it is compact closed.

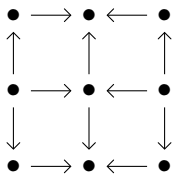
We first show $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ is a (braided, symmetric) monoidal bicategory with a result from Shulman.

Theorem A.1. *[39, Theorem 5.1] Let $\mathbb{D}$ be an isofibrant (braided, symmetric) monoidal double category. There is a (braided, symmetric) monoidal bicategory* $\mathbf{D}$ *whose objects are those of $\mathbb{D}$ and whose hom-categories $\mathbf{D}(x,y)$ have as objects the horizontal arrows in $\mathbb{D}$ of type $x \to y$ and as morphisms the 2-cells in $\mathbb{D}$ of type*

$$\begin{array}{ccc} x & \longrightarrow & y \\ {\scriptstyle id}\downarrow & \Downarrow & \downarrow{\scriptstyle id} \\ x & \longrightarrow & y \end{array}$$

The same paper [39] also contains the definitions used in this section. To use this theorem, we begin construction on a double category $\mathbb{S}\mathbf{p}(\mathbf{Csp}(\mathbf{C}))$. This requires *cubical spans of cospans*, which are

commuting diagrams of shape



We get an equivalence relation on these by relating cubical spans of cospans that share the same outside square. The induced classes are called *parallel classes*.

Lemma A.2. *There is a double category* $\mathbb{S}\mathbf{p}(\mathbf{Csp}(\mathbf{C}))$ *whose objects are the* $\mathbf{C}$*-objects, vertical morphisms are given by isomorphism classes of spans in* $\mathbf{C}$ *with invertible legs, horizontal morphisms are given by cospans in* $\mathbf{C}$*, and 2-morphisms are parallel classes of cubical spans of cospans in* $\mathbf{C}$*.*

Proof. Define the object category $\widetilde{\mathbb{C}}_0$ to have as objects the $\mathbf{C}$-objects and as morphisms the isomorphism classes of spans in $\mathbf{C}$ with invertible legs. Define the arrow category $\widetilde{\mathbb{C}}_1$ to have as objects the cospans in $\mathbf{C}$ and as morphisms the parallel classes of cubical spans of cospans in $\mathbf{C}$.

The structure functor $U\colon \widetilde{\mathbb{C}}_0 \to \widetilde{\mathbb{C}}_1$ acts on objects by mapping x to the identity cospan on x and on morphisms by mapping $x \leftarrow y \rightarrow z$, whose legs are isomorphisms, to

$$\begin{array}{ccccc} x & \longrightarrow & y & \longleftarrow & z \\ \uparrow & & \uparrow & & \uparrow \\ x & \longrightarrow & y & \longleftarrow & z \\ \downarrow & & \downarrow & & \downarrow \\ x & \longrightarrow & y & \longleftarrow & z \end{array}$$

The source functor $S\colon \widetilde{\mathbb{C}}_1 \to \widetilde{\mathbb{C}}_0$ acts on objects by sending $x \rightarrow y \leftarrow z$ to x and on morphisms by sending a cubical span of cospans to the span occupying the left vertical side. The target functor T is defined similarly.

The horizontal composition functor $\odot\colon \widetilde{\mathbb{C}}_1 \times_{\widetilde{\mathbb{C}}_0} \widetilde{\mathbb{C}}_1 \to \widetilde{\mathbb{C}}_1$ acts on objects by composing cospans with pushouts in the usual way. It acts on morphisms by

$$\begin{array}{ccccccccc} a & \longrightarrow & b & \longleftarrow & c & \longrightarrow & d & \longleftarrow & e \\ \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow \\ a' & \longrightarrow & b' & \longleftarrow & c' & \longrightarrow & d' & \longleftarrow & e' \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ a'' & \longrightarrow & b'' & \longleftarrow & c'' & \longrightarrow & d'' & \longleftarrow & e'' \end{array} \quad \overset{\odot}{\mapsto} \quad \begin{array}{ccccc} a & \longrightarrow & b +_c d & \longleftarrow & e \\ \uparrow & & \uparrow & & \uparrow \\ a' & \longrightarrow & b' +_{c'} d' & \longleftarrow & e' \\ \downarrow & & \downarrow & & \downarrow \\ a'' & \longrightarrow & b'' +_{c''} d'' & \longleftarrow & e'' \end{array}$$

This respects identities and we separate the proof that $\odot$ respects composition into the next lemma. It is straightforward to check that the required equations are satisfied. The associators and unitors arise from universal properties. □

Lemma A.3. *The assignment* $\odot$ *from Lemma A.2 preserves composition. In particular,* $\odot$ *is a functor.*

Proof. Let $\alpha, \alpha', \beta, \beta'$ be the following 2-morphisms

$$\alpha = \begin{array}{ccccc} a & \longrightarrow & b & \longleftarrow & c \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ a' & \longrightarrow & b' & \longleftarrow & c' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ \ell & \longrightarrow & m & \longleftarrow & n \end{array} \qquad \alpha' = \begin{array}{ccccc} c & \longrightarrow & d & \longleftarrow & e \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ c' & \longrightarrow & d' & \longleftarrow & e' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ n & \longrightarrow & p & \longleftarrow & q \end{array}$$

$$\beta = \begin{array}{ccccc} \ell & \longrightarrow & m & \longleftarrow & n \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ v' & \longrightarrow & w' & \longleftarrow & x' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ v & \longrightarrow & w & \longleftarrow & x \end{array} \qquad \beta' = \begin{array}{ccccc} n & \longrightarrow & p & \longleftarrow & q \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ x' & \longrightarrow & y' & \longleftarrow & z' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ x & \longrightarrow & y & \longleftarrow & z \end{array}$$

Our goal is to show that

$$(\alpha \odot \alpha') \circ (\beta \odot \beta') = (\alpha \circ \beta) \odot (\alpha' \circ \beta'). \tag{5}$$

The left hand side of this equation corresponds to horizontal composition before vertical composition. The right hand side corresponds to composing in the opposite order.

First, compute the left hand side of (5). Composing horizontally, $\alpha \odot \alpha'$ and $\beta \odot \beta'$ are, respectively,

$$\begin{array}{ccccc} a & \longrightarrow & b +_c d & \longleftarrow & e \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ a' & \longrightarrow & b' +_{c'} d' & \longleftarrow & e' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ \ell & \longrightarrow & m +_n p & \longleftarrow & q \end{array} \qquad \begin{array}{ccccc} \ell & \longrightarrow & m +_n p & \longleftarrow & q \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ v' & \longrightarrow & w' +_{x'} y' & \longleftarrow & z' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ v & \longrightarrow & w +_x y & \longleftarrow & z \end{array}$$

The vertical composite $(\alpha \odot \alpha') \circ (\beta \odot \beta')$ is equal to

$$\begin{array}{ccccc} a & & \longrightarrow & & b +_d d & & \longleftarrow & & e \\ {\scriptstyle\cong}\uparrow & & & & \uparrow & & & & {\scriptstyle\cong}\uparrow \\ a' \times_\ell v' & & \longrightarrow & & (b' +_{c'} d') \times_{(m+_n p)} (w' +_{x'} y') & & \longleftarrow & & e' +_q z' \\ {\scriptstyle\cong}\downarrow & & & & \downarrow & & & & {\scriptstyle\cong}\downarrow \\ v & & \longrightarrow & & w +_x y & & \longleftarrow & & z \end{array} \tag{6}$$

Now solving for the right hand side of (5), $\alpha \circ \beta$ and $\alpha' \circ \beta'$ are respectively

$$\begin{array}{ccccc} a & \longrightarrow & b & \longleftarrow & c \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ a' \times_\ell v' & \longrightarrow & b' \times_m w' & \longleftarrow & c' \times_n x' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ v & \longrightarrow & w & \longleftarrow & x \end{array} \qquad \begin{array}{ccccc} c & \longrightarrow & d & \longleftarrow & e \\ {\scriptstyle\cong}\uparrow & & \uparrow & & {\scriptstyle\cong}\uparrow \\ c' \times_n x' & \longrightarrow & d' \times_p y' & \longleftarrow & e' \times_q z' \\ {\scriptstyle\cong}\downarrow & & \downarrow & & {\scriptstyle\cong}\downarrow \\ x & \longrightarrow & y & \longleftarrow & z \end{array}$$

The vertical composite $(\alpha \circ \beta) \odot (\alpha' \circ \beta')$ is equal to

$$\begin{array}{ccccc} a & & \longrightarrow & & b +_c d & & \longleftarrow & & e \\ {\scriptstyle\cong}\uparrow & & & & \uparrow & & & & {\scriptstyle\cong}\uparrow \\ a' \times_\ell v' & & \longrightarrow & & (b' \times_m w') +_{(c' \times_n x')} (d' \times_p y') & & \longleftarrow & & e' \times_q z' \\ {\scriptstyle\cong}\downarrow & & & & \downarrow & & & & {\scriptstyle\cong}\downarrow \\ v & & \longrightarrow & & w +_x y & & \longleftarrow & & z \end{array} \tag{7}$$

Since (6) and (7) have coinciding outer squares, they represent the same parallel class, hence (5) holds. □

Our next step is to show that the (braided, symmetric) monoidal structure from $\mathbf{C}$ lifts to $\mathbb{S}\mathbf{p}(\mathbf{Csp}(\mathbf{C}))$. We point to [39, Def. 2.9] for the definition of a monoidal double category.

Lemma A.4. *The (braided, symmetric) monoidal structure of* $\mathbf{C}$ *lifts to* $\mathbb{S}\mathbf{p}(\mathbf{Csp}(\mathbf{C}))$.

Proof. Again, denote $\mathbb{S}\mathbf{p}(\mathbf{Csp}(\mathbf{C}))$ by $\widetilde{\mathbb{C}}$. The object $\widetilde{\mathbb{C}}_0$ and arrow $\widetilde{\mathbb{C}}_1$ categories are (braided, symmetric) monoidal by taking $\otimes$ pointwise. The monoidal structure for $\widetilde{\mathbb{C}}_0$-objects follows from that on $\mathbf{C}$ and for $\widetilde{\mathbb{C}}_0$-morphisms is

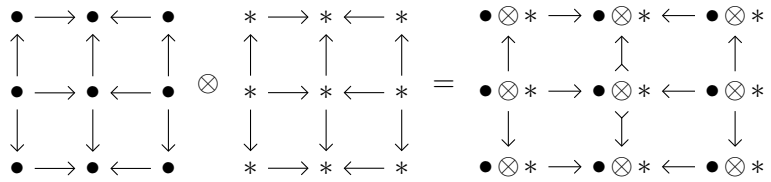
$$(a \leftarrow b \rightarrow c) \otimes (a' \leftarrow b' \rightarrow c') = (a \otimes a' \leftarrow b \otimes b' \rightarrow c \otimes c').$$

Universal properties provide the associator, unitors, and coherence axioms. It is clear that $\widetilde{\mathbb{C}}_0$ is also braided or symmetric monoidal whenever $\mathbf{C}$ is.

We obtain a monoidal structure for $\widetilde{\mathbb{C}}_1$-objects by

$$(a \rightarrow b \leftarrow c) \otimes (a' \rightarrow b' \leftarrow c') = (a \otimes a' \rightarrow b \otimes b' \leftarrow c \otimes c')$$

and for $\widetilde{\mathbb{C}}_1$-morphisms by



The monoidal unit for $\widetilde{\mathbb{C}}_1$ is the identity cospan on I which is exactly U_I. Universal properties again provide the associator, unitors, and coherence axioms. The braiding and symmetry of the monoidal structure clearly lifts from $\mathbf{C}$.

It is straightforward to check that the source and target functors are strict monoidal and respect the associator and unitors. It remains to find two invertible globular 2-cells: one witnessing interchange

$$r\colon (M_1 \otimes N_1) \odot (M_2 \otimes N_2) \rightarrow (M_1 \odot M_2) \otimes (N_1 \odot N_2)$$

for $\widetilde{\mathbb{C}}_1$-objects M_i and N_i, and another witnessing units

$$u\colon U_{f \otimes g} \rightarrow U_f \otimes U_g$$

for $\widetilde{\mathbb{C}}_0$-arrows f and g. Moreover, r and u must satisfy certain axioms [39, Def. 2.9].

If $M_1 = (a \rightarrow b \leftarrow c)$, $M_2 = (c \rightarrow d \leftarrow e)$, $N_1 = (v \rightarrow w \leftarrow x)$, and $N_2 = (x \rightarrow y \leftarrow z)$, then r has domain

$$a \otimes v \rightarrow (b \otimes w) +_{(c \otimes x)} (d \otimes y) \leftarrow (e \otimes z)$$

and codomain

$$a \otimes v \rightarrow b +_c d \otimes w +_x y \leftarrow (e \otimes z).$$

We must find a 2-cell whose outer square is formed by the domain and codomain of r on the top and bottom plus identity $\widetilde{\mathbb{C}}_0$-morphisms on the left and right. Let $J\colon \mathbf{D} \rightarrow \mathbf{C} \times \mathbf{C}$ be the functor on the category

$$\mathbf{D} = \{\bullet \rightarrow \bullet \leftarrow \bullet \rightarrow \bullet \leftarrow \bullet\}.$$

whose image is of the form

$$(a \to b \leftarrow c \to d \leftarrow e) \times (w \to v \leftarrow x \to y \leftarrow z).$$

Then the domain of r is $\mathrm{colim}(\Delta \circ \otimes)$ and the codomain is $\otimes(\mathrm{colim}(\Delta))$. But these are isomorphic by assumption. This gives r.

To define u, let f be the $\widetilde{\mathbb{C}}_0$-morphism $a \leftarrow b \to c$ and let g be $x \leftarrow y \to z$. It is easy to check that both $U_{f \otimes g}$ and $U_f \otimes U_g$ are

$$\begin{array}{ccccc}
a \otimes x & \longrightarrow & b \otimes y & \longleftarrow & c \otimes z \\
\uparrow & & \uparrow & & \uparrow \\
a \otimes x & \longrightarrow & b \otimes y & \longleftarrow & c \otimes z \\
\downarrow & & \downarrow & & \downarrow \\
a \otimes x & \longrightarrow & b \otimes y & \longleftarrow & c \otimes z
\end{array}$$

where the legs of the horizontal cospans are built using the inverses of the legs of f and g. The legs of the vertical spans are identities.

As for the remaining axioms, they are straightforward though tedious to check and are left to the reader. □

Lemma A.5. $\mathbb{S}\mathbf{p}(\mathbf{Csp}(\mathbf{C}))$ *is isofibrant.*

Proof. Take a vertical morphism $f = (a \leftarrow b \to c)$. The legs of the companion $\widehat{f} = (a \to b \leftarrow c)$, are the inverses of those from f. The companion is equipped with the 2-morphisms

$$\begin{array}{ccccc}
a & \longrightarrow & b & \longleftarrow & c \\
\uparrow & & \uparrow & & \uparrow \\
b & \longrightarrow & c & \longleftarrow & c \\
\downarrow & & \downarrow & & \downarrow \\
c & \longrightarrow & c & \longleftarrow & c
\end{array}
\quad \text{and} \quad
\begin{array}{ccccc}
a & \longrightarrow & a & \longleftarrow & a \\
\uparrow & & \uparrow & & \uparrow \\
a & \longrightarrow & a & \longleftarrow & b \\
\downarrow & & \downarrow & & \downarrow \\
a & \longrightarrow & b & \longleftarrow & c
\end{array}$$

The reader may check that the required equations hold. The conjoint $\check{f}$ of f is $\widehat{f}^{\mathrm{op}}$. □

Theorem A.6. $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ *is a (braided, symmetric) monoidal bicategory.*

Proof. Apply Theorem A.1. □

This proves the first half of Theorem 3.2. To prove the second half, we assume that $\mathbf{C}$ is a cocartesian monoidal bicategory.

Note that we take Stay's definition of a compact closed bicategory [40].

Lemma A.7. *The diagram*

$$\begin{array}{ccc}
x+x+x & \xrightarrow{x+\nabla} & x+x \\
{\scriptstyle \nabla+x}\downarrow & & \downarrow{\scriptstyle \nabla} \\
x+x & \xrightarrow{\nabla} & x
\end{array}$$

in $\mathbf{C}$ *is a pushout square.*

Proof. Suppose $f, g\colon x+x \to y$ form a cocone over the above diagram. Let $\iota\colon x \to x+x+x$ be an inclusion into the middle copy of x. Observe that $\ell := (\nabla + x) \circ \iota$ and $r := (x+\nabla)\circ\iota$ are the left and right inclusions $x \to x+x$. Then $f \circ \ell = g \circ r$ is a map $x \to y$, which we claim to be the unique map making the required diagram commute. Indeed, given $h\colon x \to y$ such that $f = h \circ \nabla = g$, then $g \circ r = f \circ \ell = h \circ \nabla \circ \ell = h$. □

Theorem A.8. $\mathbf{Sp}(\mathbf{Csp}(\mathbf{C}))$ *is compact closed.*

Proof. The objects are self dual. To show this, start with an object x. Define the evaluation morphism and coevaluation morphism by

$$e = (x+x \xrightarrow{\nabla} x \leftarrow 0), \qquad c = (0 \to x \xleftarrow{\nabla} x+x).$$

We next define the cusp isomorphisms, α and β. The domain for α is the composite

$$x \xrightarrow{\ell} x+x \xleftarrow{x+\nabla} x+x+x \xrightarrow{\nabla+x} x+x \xleftarrow{r} x$$

and the codomain for β is

$$x \xrightarrow{r} x+x \xleftarrow{\nabla+x} x+x+x \xrightarrow{x+\nabla} x+x \xleftarrow{\ell} x.$$

That these are both identity cospans on x follows from Lemma A.7 and the equations $\nabla + x = \ell \circ \nabla$ and $x+\nabla = r \circ \nabla$ Take α and β each to be the identity 2-morphism on x. This gives a dual pair $(x, x, e, c, \alpha, \beta)$ which we can complete to a coherent dual pair [36, p. 22]. □

References

[1] S. Abramsky & B. Coecke (2004): *A categorical semantics of quantum protocols*. In: *Logic in Computer Science. Proceedings of the 19th Annual IEEE Symposium*, pp. 415–425, doi:10.1109/LICS.2004.1319636.

[2] Bob B. Coecke & B. Edwards (2011): *Toy quantum categories*. *Electron. Notes Theor. Comput. Sci.* 270(1), doi:10.1016/j.entcs.2011.01.004. Available at `https://arxiv.org/abs/0808.1037`.

[3] M. Backens (2016): *Completeness and the ZX-calculus*, doi:10.1109/LICS.2004.1319636. Available at `https://arxiv.org/abs/1602.08954`.

[4] J. Baez, B. Coya & F. Rebro (2017): *Props in network theory*. Available at `https://arxiv.org/abs/1707.08321`.

[5] J. Baez & B. Fong (2015): *A compositional framework for passive linear networks*. Available at `https://arxiv.org/abs/1504.05625`.

[6] J. Baez, B. Fong & B. Pollard (2016): *A compositional framework for Markov processes*. *J. Math. Phys.* 57, doi:10.1063/1.4941578.

[7] K. Bar, A. Kissinger & J. Vicary (2016): *Globular: an online proof assistant for higher-dimensional rewriting*. Available at `http://globular.science`.

[8] D. Cicala (2016): *Spans of cospans*. Available at `https://arxiv.org/abs/1611.07886`.

[9] D. Cicala & K. Courser (2017): *Spans of cospans in a topos*. Available at `https://arxiv.org/abs/1707.02098`.

[10] B. Coecke & R. Duncan (2008): *Interacting quantum observables*. In: *Automata, languages and programming. Part II*, *Lecture Notes in Comput. Sci.* 5126, Springer, Berlin, pp. 298–310, doi:10.1007/978-3-540-70583-3_25.

[11] B. Coecke & R. Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics*. *New J. Phys.* 13, doi:10.1088/1367-2630/13/4/043016.

[12] B. Coecke & B. Edwards (2012): *Spekkens's toy theory as a category of processes*. In: *Mathematical foundations of information flow*, *Proc. Sympos. Appl. Math.* 71, Amer. Math. Soc., pp. 61–68, doi:10.1090/psapm/071/602.

[13] B. Coecke, B. Edwards & R. Spekkens (2011): *Phase groups and the origin of non-locality for qubits*. *Electron. Notes Theor. Comput. Sci.* 270(2), doi:10.1016/j.entcs.2011.01.021.

[14] B. Coecke & D. Pavlovic (2008): *Quantum measurements without sums*. In: *Mathematics of quantum computation and quantum technology*, Chapman & Hall CRC Appl. Math. Nonlinear Sci. Ser., Chapman & Hall/CRC, Boca Raton, FL, pp. 559–596, doi:10.1201/9781584889007.ch16.

[15] B. Coecke, D. Pavlovic & J. Vicary (2013): *A new description of orthogonal bases*. *Math. Structures Comput. Sci.* 23(3), doi:10.1017/S0960129512000047.

[16] B. Coecke & S. Perdrix (2012): *Environment and classical channels in categorical quantum mechanics*. *Log. Methods Comput. Sci.* 8(4), doi:10.2168/LMCS-8(4:14)2012.

[17] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel & M. Lowe (1997): *Algebraic approaches to graph transformation. Basic concepts and double pushout approach*. In: *Handbook of graph grammars and computing by graph transformation, Vol. 1*, World Sci. Publ., River Edge, NJ, doi:10.1142/9789812384720_0003.

[18] V. Danos, E. Kashefi & P. Panangaden (2007): *The measurement calculus*. *J. ACM* 54(2), doi:10.1145/1219092.1219096.

[19] L. Dixon, R. Duncan & A. Kissinger: *Quantomatic*. Available at `https://sites.google.com/site/quantomatic/`.

[20] R. Duncan & S. Perdrix (2009): *Graph states and the necessity of Euler decomposition*. In: *Mathematical theory and computational practice*, *Lecture Notes in Comput. Sci.* 5635, Springer, Berlin, pp. 167–177, doi:10.1007/978-3-642-03073-4_18.

[21] R. Duncan & S. Perdrix (2010): *Rewriting measurement-based quantum computations with generalised flow*. *Automata, Languages and Programming*, doi:10.1007/978-3-642-14162-1_24.

[22] J. Evans, R. Duncan, A. Lang & P. Panangaden (2009): *Classifying all mutually unbiased bases in Rel*. Available at `https://arxiv.org/abs/0909.4453`.

[23] B. Fong (2016): *The Algebra of Open and Interconnected Systems*. Available at `https://arxiv.org/abs/arXiv:1609.05382`.

[24] A. Habel, J. Muller & D. Plump (2001): *Double-pushout graph transformation revisited*. *Math. Structures Comput. Sci.* 11(5), doi:10.1017/S0960129501003425.

[25] John J. Baez & B. Pollard (2017): *A compositional framework for reaction networks*. *Rev. Math. Phys.* 29(9), doi:10.1142/S0129055X17500283.

[26] A. Joyal & R. Street (1991): *The geometry of tensor calculus. I*. *Adv. Math.* 88(1), doi:10.1016/0001-8708(91)90003-P.

[27] A. Kissinger (2012): *Pictures of processes: automated graph rewriting for monoidal categories and applications to quantum computing*. *Ph.D. Thesis, University of Oxford*. Available at `https://arxiv.org/abs/1203.0202`.

[28] A. Kissinger & V. Zamdzhiev (2015): *Quantomatic: a proof assistant for diagrammatic reasoning*. In: *Automated deduction—CADE 25*, *Lecture Notes in Comput. Sci.* 9195, Springer, Cham, pp. 326–336, doi:10.1007/978-3-319-21401-6_22.

[29] Lucas L. Dixon, R. Duncan & A. Kissinger (2010): *Electron. Proc. Theor. Comput. Sci.* 26, doi:10.4204/EPTCS.26.16.

[30] S. MacLane & I. Moerdijk (2012): *Sheaves in geometry and logic: A first introduction to topos theory*. Springer Science & Business Media.

[31] A. Merry (2014): *Reasoning with !-Graphs*. *CoRR* abs/1403.7828. Available at `http://arxiv.org/abs/1403.7828`.

[32] M. Nielsen & I. Chuang (2010): *Quantum computation and quantum information*. Cambridge University Press, Cambridge, doi:10.1017/CBO9780511976667.

[33] D. Pavlovic (2009): *Quantum and classical structures in nondeterministic computation*. In: *Quantum interaction*, *Lecture Notes in Comput. Sci.* 5494, Springer, Berlin, pp. 143–157, doi:10.1007/978-3-642-00834-4_13.

[34] R. Penrose (1971): *Applications of negative dimensional tensors*. In: *Combinatorial Mathematics and its Applications (Proc. Conf., Oxford, 1969)*, Academic Press, London, pp. 221–244. Available at `http://homepages.math.uic.edu/~kauffman/Penrose.pdf`.

[35] B. Pollard (2016): *Open Markov Processes: A Compositional Perspective on Non-Equilibrium Steady States in Biology*. *Entropy* 18(4), doi:10.3390/e18040140.

[36] P. Pstragowski (2014): *On dualizable objects in monoidal bicategories, framed surfaces and the Cobordism Hypothesis*. Available at `https://arxiv.org/abs/1411.6691`.

[37] V. Sassone & P. Sobocinski (2005): *A congruence for Petri nets*. *Electronic Notes in Theoretical Computer Science* 127(2), doi:10.1016/j.entcs.2005.02.008.

[38] P. Selinger (2011): *A survey of graphical languages for monoidal categories*. In: *New structures for physics*, *Lecture Notes in Phys.* 813, Springer, Heidelberg, pp. 289–355, doi:10.1007/978-3-642-12821-9_4.

[39] M. Shulman (2010): *Constructing symmetric monoidal bicategories*. Available at `http://arxiv.org/abs/1004.0993`.

[40] M. Stay (2016): *Compact closed bicategories*. *Theory Appl. Categ.* 31. Available at `https://arxiv.org/abs/1301.1053`.