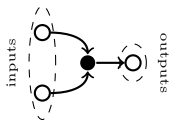
REWRITING ON OPEN STRUCTURED GRAPHS

DANIEL CICALA

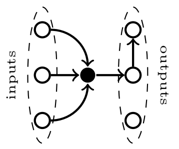
Abstract. It was shown in [2] that given a topos T, we can form a bicategory **MonicSp**(**Csp**(**T**)) whose 0-cells are the T-objects, 1-cells are cospans in T, and 2-cells are spans of cospans with monic legs. Taking T to be the category **Graph** of (directed) graphs and graph morphisms, we consider the full sub-bicategory **Rewrite** of **MonicSp**(**Csp**(**Graph**)) on the edgeless graphs. This bicategory ought to be considered as having open graphs and ways of rewriting them. We then demonstrate that we can capture categories of graphs with extra structure by considering certain slice categories of **Graph**. We can then construct a symmetric monoidal, compact closed bicategory of open structured graphs and their rewritings. Finally, we apply this construction to categorify the PROP whose morphisms are diagrams in the zx-calculus developed by Coecke and Duncan.

1. Rewriting open graphs

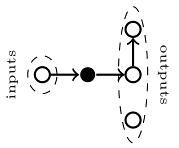
Intuitively, the notion of an open graph is rather simple. Take a directed graph and declare some of the nodes to be inputs and others to be outputs, for instance



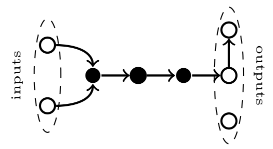
Whenever there is a bijection between the inputs of one graph and the outputs of another, we can connect them in a way described by the bijection. This process is provides a way to turn a pair of compatible open graphs into a single open graph. Indeed, we cannot connect



to the above open graph, though we can connect



to form the open graph



This is made precise using cospans and pushout as follows. Consider the functor $N\colon \mathbf{Set} \to \mathbf{Graph}$ that assigns the edgeless graph with node set X to a set 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. We will denote this open graph by G when the legs of the cospan do not need to be explicit. Also, call the left leg $N(X)$ of the cospan the *inputs* of G and the right leg $N(Y)$ the *outputs* of G. Suppose we have another open graph G' with inputs $N(Y)$ and outputs $N(Z)$. Then we can compose the cospans

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

This is certainly not an open graph, but pushing out over the span $G \leftarrow N(Y) \to G'$ induces the open graph

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

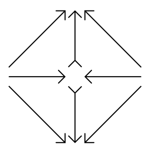
By taking isomorphism classes of these pushouts, we obtain a category whose objects are those in the image of N and morphisms are open graphs.

But we can do better! Indeed, we have only just described the first layer of a symmetric monoidal and compact closed bicategory. This bicategory was introduced by the author in [2] under the name **Rewrite**. It was shown that **Rewrite** is symmetric monoidal and compact closed in a joint work with Courser [3]. Before moving on to consider graphs equipped with some additional structure, we briefly recall the story of **Rewrite**.

Given a topos **T**, there is a symmetric monoidal and compact closed bicategory $\mathbf{MonicSp}(\mathbf{Csp}(\mathbf{T}))$, or $\mathbf{MSC}(\mathbf{T})$. for short, consisting of

- (0-cells) objects of **T**,
- (1-cells) cospans in **T**, and
- (2-cells) isoclasses of monic spans of cospans in **T**.

The 2-cells are diagrams



where '$\rightarrowtail$' denotes a monic **T**-morphism.

Letting **T** be the topos **Graph** of directed graphs, there is a symmetric monoidal and compact closed sub-bicategory of $\mathbf{MSC}(\mathbf{Graph})$ called **Rewrite** whose 0-cells are exactly the edgeless graphs, and full in 1-cells and 2-cells. The idea of **Rewrite** is that the 1-cells are open graphs and the 2-cells are the ways to rewrite a graph into another while preserving the input and output nodes. By rewriting, we mean double pushout graph rewriting [5]. This means that restricting the 2-cells to spans with monic legs is not overly restrictive since many authors only consider monic double pushout rewriting rules.

The motivation for constructing **Rewrite** is not necessarily to study it directly, but rather for it to serve as an ambient context in which to generate symmetric monoidal and compact closed sub-bicategories on some collection of open graphs and rewriting rules. Of course, the collection that one would use depends on their interest. For instance, if one were interested in topological quantum field theories, Courser and the author categorify the category **2Cob**, the category of smooth compact 1-dimensional manifolds without boundary and smooth compact oriented cobordisms [3].

The interest in **Rewrite** is that graphical calculi typically use some version of open graphs as a semantics and equations between graphical terms are given by rewrite rules [10]. However, equations are evil when looking through a categorical lens. Morally, equations ought to be replaced by isomorphisms. The process of replacing equations with isomorphisms and sets by categories is known as categorification. In this program, we are interested in categorifying certain categories into bicategories. The categories of interest are those whose morphisms are associated to open graphs of some sort. For instance there are various categories of open networks [6, 7, 9] which we seek to categorify by interpreting rewrite rules as 2-cells instead of as providing equations between morphisms. However, this requires some work.

There are drawbacks to this approach. For example, working with open graphs is only useful to model graphical calculi whose terms are equal up to ambient isotopy in 4-space. Selinger's work [10] excellently describes various graphical calculi and the role ambient isotopy plays in their use. Amending **Rewrite** to account for this shortcoming is a direction of future research. Currently, our interest lies in expanding the idea of **Rewrite** to categorify the zx-calculus. However, first we will briefly cover the zx-calculus.

2. The zx-calculus

The zx-calculus was developed by Coecke and Duncan [4] as an intuitive graphical language in which to reason about quantum observables.

say more here

There are five 'generating elements' to the zx-calculus. They are depicted in Figure 1 and are to be read from left to right. In the spirit of compositionality, we will soon present a category **zx** whose morphisms are generated by these five diagrams, so we will go ahead and call them **zx**-morphisms. In particular, we will call the five **zx**-morphisms listed below as *basic*:

- a wire with a single input and output,
- green nodes with a non-negative integer number of inputs and outputs and paired with a phase $\alpha \in [-\pi, \pi)$,
- red nodes with a non-negative integer number inputs and outputs and paired with a phase $\beta \in [-\pi, \pi)$,
- a yellow node with a single input and output, and

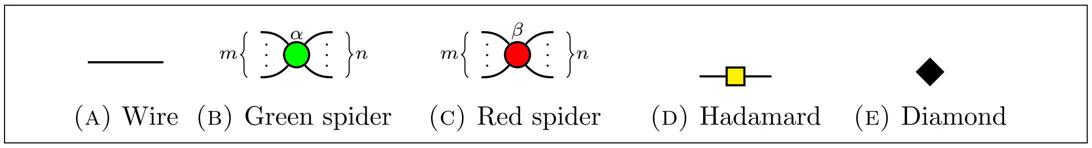
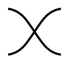


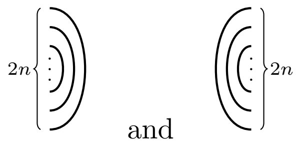
FIGURE 1. Generators for the category **zx**

- a black box node with no inputs or outputs

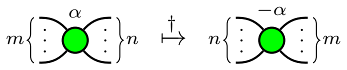
Observe that there is a non-negative number of dangling wires on the right and left sides of these basic **zx**-morphisms. We will refer to those wires on the left as *inputs* and those on the right as *outputs*. With this in mind, we can present the dagger compact category **zx** introduced by Coecke and Duncan [4]. The objects of **zx** are the non-negative integers. The morphisms are generated by those depicted in Figure 1, though we take the wire to be the identity on 1. Composition in **zx** is performed by enumerating the inputs and outputs of a pair of compatible diagrams and connecting the outputs of the first diagram to the inputs of the second diagram. The monoidal product on **zx** is given by addition on the objects and the disjoint union of elements on the morphisms. From this, we obtain the identity on all n by taking the disjoint union of n wires. The symmetry of the monoidal product gives us a morphism



The compactness provides evaluation and coevaluation morphisms



from $2n \to 0$ and $0 \to 2n$ for each object $n \geq 1$ and the empty diagram for $n = 0$. The dagger structure is obtained by swapping inputs and outputs then, for the red and green spider diagrams, multiplying the phase by -1. For instance,



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

Thus far, we have a presentation for a free dagger compact category. To obtain **zx**, we also include the relations depicted in Figure 2. Note that, to this list of relations, we also include equations obtained

- by exchanging red and green nodes,
- by daggering, and
- up to ambient isotopy in 4-space.

Note that we denote the empty graph, which is the monoidal unit, by $\emptyset$. Also, red and green nodes with no phase indicated have a phase of 0.

talk on 2cells here or intro? use word 'phase'?

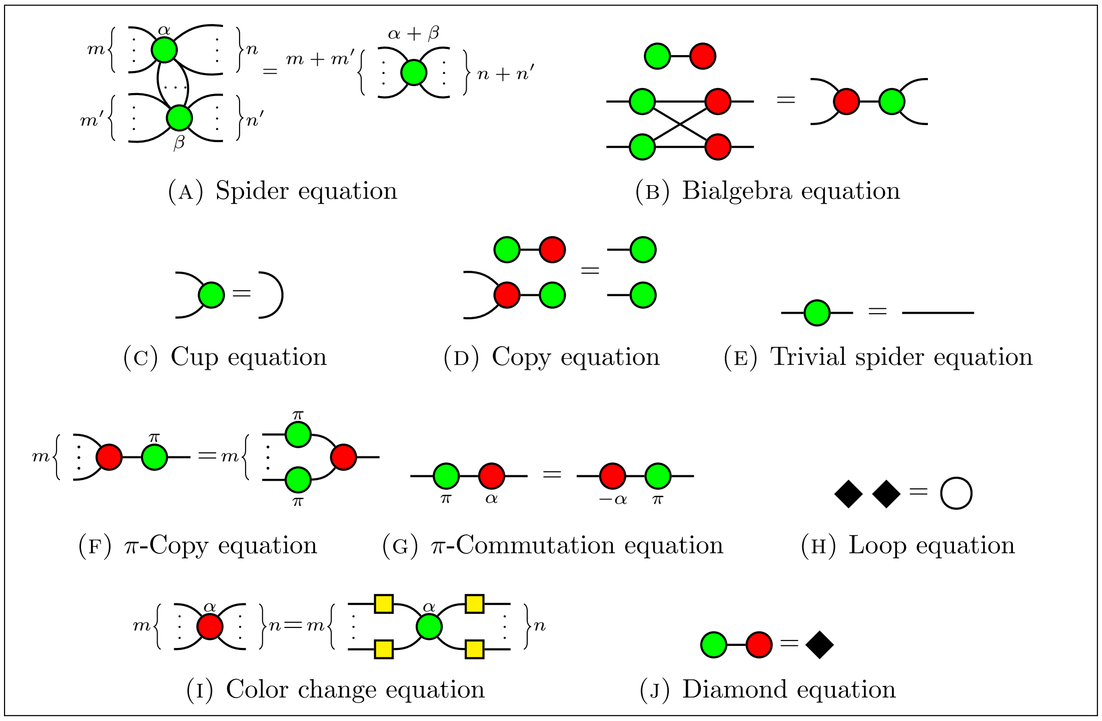


FIGURE 2. Relations in the category **zx**

3. OPEN STRUCTURED GRAPHS

In Section 1, we saw that there is a symmetric monoidal and compact closed bicategory **MSC**(**T**) for any topos **T**. In particular, we presented a nice symmetric monoidal and compact closed sub-bicategory **Rewrite** of **MSC**(**Graph**). Because many graphical languages use graphs with additional structure , it would be nice to give a similar construction for these cases. To this end, we first discuss a method of providing additional structure in a manner that is amenable to this construction.

cite some

Let S be a graph. By a *graph over* S, we mean a graph morphism $G \to S$. Then a morphism between graphs over S is a graph morphism $G \to G'$ such that

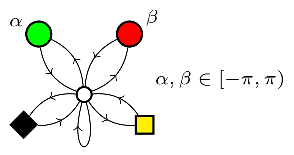
$$\begin{tikzcd} G \arrow[r] \arrow[dr] & G' \arrow[d] \\ & S \end{tikzcd}$$

The way to think about how a graph G over S gives structure to G is that S classifies the nodes and edges of G via the fibres of the morphism. This method is being investigated in more generality by Baez and Courser in an upcoming paper . In the meantime, we illustrate it by giving a graph the structure of an **zx**-morphism.

cite it

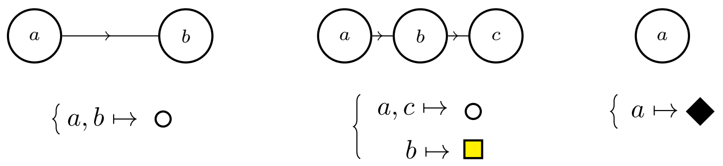
Example 3.1. Let S_{zx} be the graph

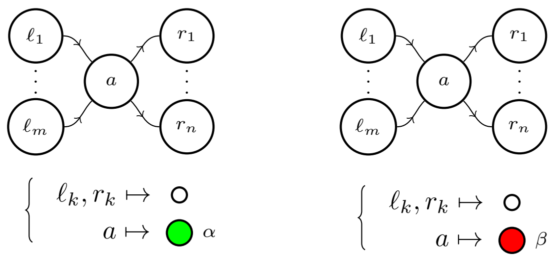
(1)



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

We can draw all of the basic zx-calculus elements from Figure 1 as graphs over S_{zx} as follows:

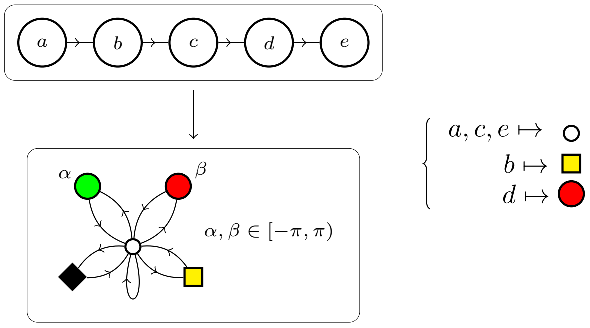




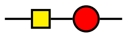
Note that the functions are completely determined by their behavior on the nodes because there is only one possible arrow between nodes in S_{zx}. The role each of the nodes in S_{zx} plays in providing structure is evident except, perhaps, for the node ○. Observe that four of the basic zx-elements have 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.

At this point, we can think of the basic elements of the zx-calculus as graphs over S_{zx}. But we can do this for any **zx**-morphism, such as

(2)



which corresponds with the **zx**-morphism



With the basic **zx**-morphisms, we can form larger **zx**-morphisms by composition. Therefore, if we aim to describe **zx**-morphisms with graphs over S_{zx}, we need to be able to connect graphs over S_{zx} in a way that captures composition. A way to accomplish this is to equip graphs over S_{zx} with inputs and outputs as we did for open graphs. Again, we use cospans, though the added structure introduces new considerations.

We will do things slightly backwards here and introduce a bicategory before discussing its 0-cells, 1-cells, and 2-cells. Recall that an over category in a topos is still a topos [8, Thm. IV.7.1]. And so, there is a topos $\mathbf{Graph} \downarrow S_{\text{zx}}$ of graphs over S_{zx}. As discussed earlier, this gives us a symmetric monoidal and compact closed bicategory $\mathbf{MSC}(\mathbf{Graph} \downarrow S_{\text{zx}})$. Now, we would like to construct a sub-bicategory in a similar vein to **Rewrite**. However, there is a problem. Recall that the objects of **Rewrite** have the form $N(X)$ where $N \colon \mathbf{Set} \to \mathbf{Graph}$ is the functor sending a set to the edgeless graph on that set. In **Graph**, there is a unique way to be an edgeless graph. But in $\mathbf{Graph} \downarrow S_{\text{zx}}$ there is not a unique (up to isomorphism) way to be an edgeless graph because there may be more than one graph morphism to S_{zx}. For instance, the graph with two nodes and no edges can be a graph over S_{zx} in $5^2 = 25$ ways. This issue is rectified by functorially choosing which edgeless graphs over S_{zx} will serve as inputs and outputs. This lets us expand the notion of graphs over S_{zx} to 'open graphs over S_{zx}'.

Definition 3.2. Define a functor $N_{\text{zx}} \colon \mathbf{Set} \to \mathbf{Graph} \downarrow S_{\text{zx}}$ by assigning $N_{\text{zx}}(X) \to S_{\text{zx}}$ to each set X where $N_{\text{zx}}(X)$ is the edgeless graph with nodes X and each node is mapped to ○. An *open graph over S_{zx}* is a cospan in the topos $\mathbf{Graph} \downarrow S_{\text{zx}}$ of the form

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

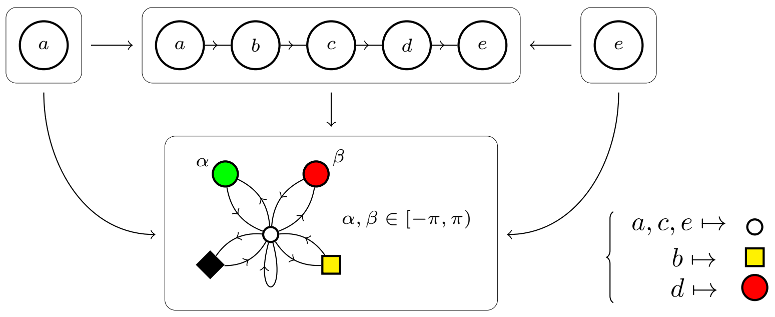
At this point, we are ready to define the analogue to **Rewrite**.

Definition 3.3. Define a bicategory **zxRewrite** as the symmetric monoidal and compact closed sub-bicategory of $\mathbf{MSC}(\mathbf{Graph} \downarrow S_{\mathrm{zx}})$ that is 1-full and 2-full on the objects of the form $N_{\mathrm{zx}}(X)$ for sets X.

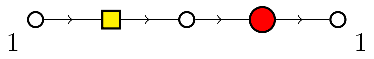
The 0-cells of **zxRewrite** are those edgeless graphs over S_{zx} in the image of N_{zx}. The 1-cells are exactly the open graphs over S_{zx}. The 2-cells are all the ways to rewrite one open graph over S_{zx} into another in a way that preserves the inputs and outputs.

To better understand this bicategory, we give an example of an open graph over S_{zx}. Along with this example, we introduce new notation in order to keep the remaining diagrams rather compact. This notation includes the information of an open graph $G \to S_{\mathrm{zx}}$ completely in the nodes of G. Draw each node of G in the same way as its image is drawn in S_{zx}. Moreover, the legs of the cospans have associated cardinalities κ and κ' which we will write in the lower left and right corners of the graph. The following example should clarify this notation.

Example 3.4. Consider the graph (2) over S_{zx}. Make this an open graph as follows:



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



Of course, this notation sheds away a fair amount information regarding the graph morphisms involved in the cospan. To gain a bit of the information back, we will align all input nodes on the far left and the output nodes on the far right. Assume that the left and right hand cospan maps are in bijection with these input nodes and output nodes, respectively. This should cause no confusion for our purposes here.

As mentioned earlier, our interest in the category **Rewrite** is as an ambient context in which to generate symmetric monoidal and compact closed bicategories from some collection of 1-cells and 2-cells. The same can be said for our new bicategory **zxRewrite**. In particular, we want to have generating 1-cells that correspond to the generating **zx**-morphisms from Figure 1 and generating 2-cells that correspond to the relations from Figure 2. Our claim is that the resulting bicategory categorifies the category **zx**.

4. A categorification of **zx**

The basic **zx**-morphisms in Figure 1 are naturally presented as open graphs over S_{zx} as shown in Figure 3. At first glance, there seems to be a redundancy in the green and red spiders: m and n have each been written twice. However, this is because each instance of m and n have different meanings. The m and n written beside the brackets counts the number of nodes. The m and n underneath the diagrams indicate how many nodes are in the edgeless graphs in the left and right legs, respectively, of the cospan the diagram represents. We are not requiring the cospan maps to be injective even though this happens to be the case for our generators.

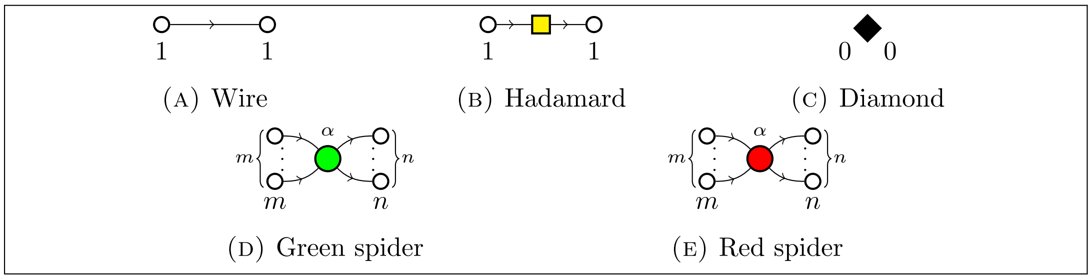
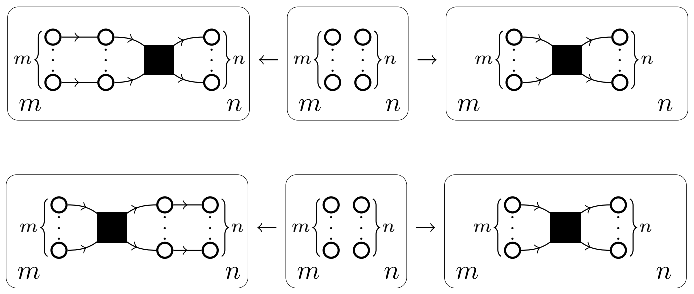


Figure 3. Generating 1-cells for the bicategory $\underline{\mathbf{zx}}$

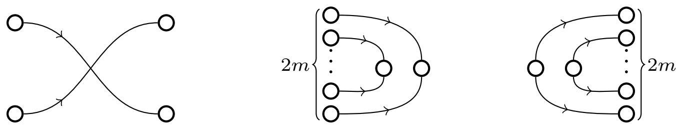
Just as the open graphs over S_{zx} in Figure 3 capture the generating **zx**-morphisms, we also want to introduce the relations from Figure 2 into our framework. These relations are presented in Figure 4.

Definition 4.1. Define $\underline{\mathbf{zx}}$ to be the symmetric monoidal and compact closed sub-bicategory of **zxRewrite** generated by the 1-cells depicted in Figure 3. The 2-cells are generated by those spans of spans Figure 4. In addition, we also include 2-cells obtained by exchanging red and green nodes, swapping inputs and outputs for each graph over S_{zx}, or by turning the spans around. There is also an additional family

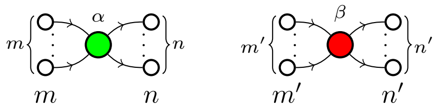


of 2-cells added to account for the fact that the wire 1-cell is not the identity on 1 here. The black box stands in for any 1-cell of type $m \to n$.-v -

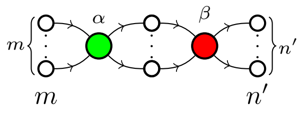
Because $\underline{\mathbf{zx}}$ is symmetric monoidal and compact closed, it also contains a twist 1-cell, plus evaluation and coevaluation 1-cells for the compact structure on m

(3)


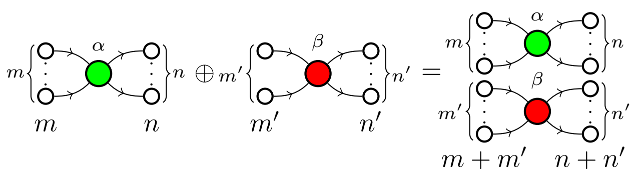
There are two ways to make the new 1-cells from old in $\underline{\mathbf{zx}}$: composing and tensoring. Composing, of course, is made precise with pushouts. For example, we can compose the red and green spider diagrams



only when $n = m'$. In that case the composite 1-cell is

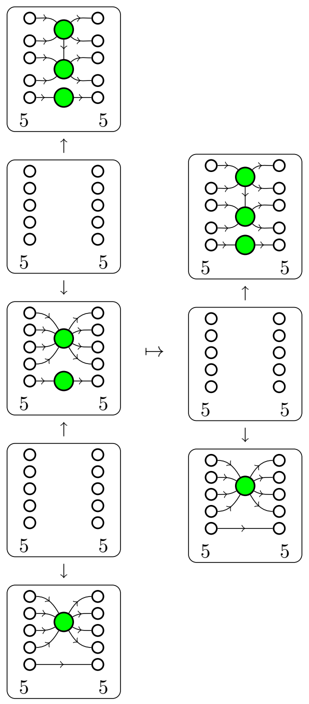


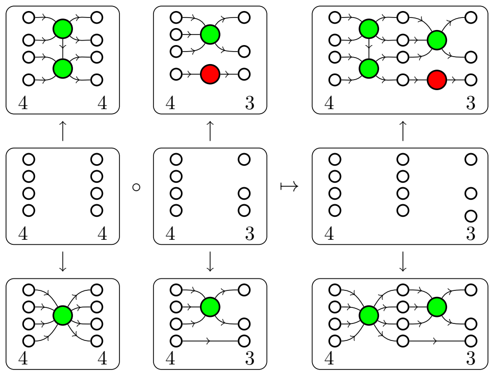
Tensoring 1-cells is simply a matter of taking the disjoint union, as seen here:



Because $\underline{\mathbf{zx}}$ is a bicategory, 2-cells are closed under vertical and horizontal composition, a detailed account of of which was presented while originally defining **Rewrite** [2]. Instead of rehashing this, we provide a brief example to illustrate each composition.

mock up a couple examples

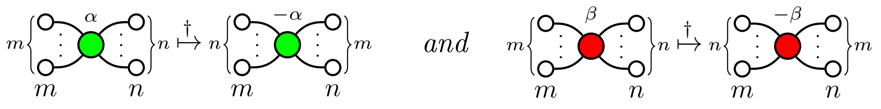




Having defined $\underline{\mathbf{zx}}$, we can turn our focus towards presenting the main theorem showing that it categorifies $\mathbf{zx}$. We start with the following definition.

Definition 4.2. Define decat($\underline{\mathbf{zx}}$) to be the category whose objects are the 0-cells of $\underline{\mathbf{zx}}$ and whose arrows are generated by the 1-cells of $\underline{\mathbf{zx}}$. Additionally, we include the relations $f = g$ on arrows f,g if there is a 2-cell $f \Rightarrow g$ in $\underline{\mathbf{zx}}$.

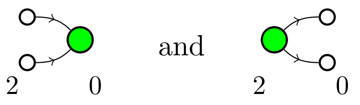
Lemma 4.3. *The category* decat($\underline{\mathbf{zx}}$) *is dagger compact via the functor described by*



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

draw the 2cell realizing snake eq

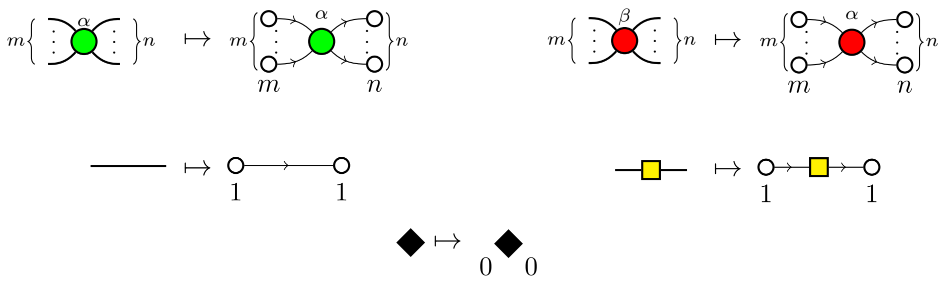
Proof. On compactness, the objects are self dual with the evaluation maps and coevaluation maps from (3). The snake equations follow from the fact that there is a 2-cell in $\underline{\mathbf{zx}}$. Showing that the described functor is a dagger functor is a matter of checking some easy to verify details. For instance, † behaves as required on the compact evaluation and coevaluation maps by recognizing that those can be drawn



$\square$

Theorem 4.4. *The identity on objects, dagger compact functor given by*

$$E\colon \mathbf{zx} \to \mathrm{decat}(\underline{\mathbf{zx}})$$



is an equivalence of categories.

Proof. Essential surjectivity is follows immediately from E being identity on objects. Fullness follows from the fact that the morphism generators for decat($\underline{\mathbf{zx}}$) are all in the image of E. Faithfulness follows from there being a bijective correspondence between the relations for the morphisms in $\mathbf{zx}$ and the morphisms in $\underline{\mathbf{zx}}$, where the additional relation from (**??**) in $\underline{\mathbf{zx}}$ corresponds to the fact that $\mathbf{zx}$ uses topological strings instead of graph edges. $\square$

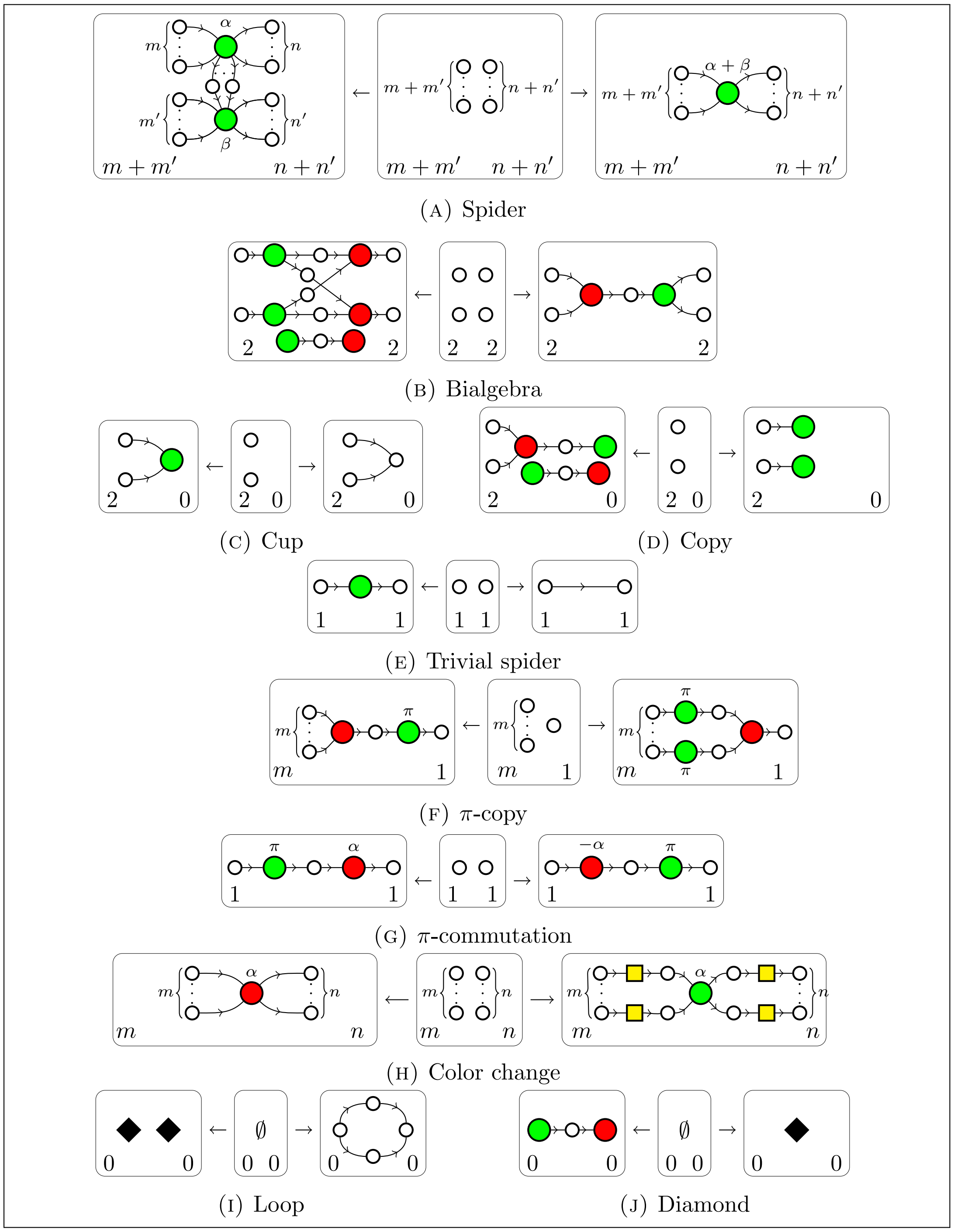
References

[1] M. Backens, *Completeness and the ZX-calculus.* (2016). Available as arXiv:1602.08954.
[2] D. Cicala, *Spans of cospans.* Available as arXiv:1611.07886.
[3] D. Cicala and K. Courser, *Bicategories of spans and cospans.* In preparation.
[4] B. Coecke, and R. Duncan, *Interacting quantum observables: categorical algebra and diagrammatics.* New Journal of Physics, 13(4), 043016. (2011).
[5] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, & M. Löwe, *Algebraic Approaches to Graph Transformation-Part I: Basic Concepts and Double Pushout Approach.* In Handbook of Graph Grammars, pp. 163-246. (February 1997).
[6] L. Dixon, R. Duncan, and A. Kissinger, *Open graphs and computational reasoning.* (2010). Available as arXiv:1007.3794.
[7] B. Fong, *The Algebra of Open and Interconnected Systems.* (2016). Available as arXiv:1609.05382.

[8] S. MacLane, and I. Moerdijk, *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Springer Science & Business Media. (2012).

[9] B. Pollard, *Open Markov processes: A compositional perspective on non-equilibrium steady states in biology*. Entropy, 18(4), p.140. (2016).

[10] P. Selinger, *A survey of graphical languages for monoidal categories*. New Structures for Physics, pp. 289-355. Springer Berlin Heidelberg. (2010).



(A) Spider

(B) Bialgebra

(C) Cup

(D) Copy

(E) Trivial spider

(F) π-copy

(G) π-commutation

(H) Color change

(I) Loop

(J) Diamond

FIGURE 4. Generating 2-cells for the bicategory $\underline{\mathbf{zx}}$