REWRITING OPEN OBJECTS

DANIEL CICALA

Abstract. open graphs, their rewrites, and an application

1. A motivating example

This section serves two functions. First, we discuss the example that motivates this paper. Within our discussion, we take the opportunity to set both notation and language used in the sequel.

When reading network theory literature written from the compositional perspective, one comes across the notion of an open graph. The level of formality this definition is given varies between authors, but the core idea is that an *open graph* is a **Set**-diagram $E \rightrightarrows N$ together with a subset $\partial \subseteq N$ equipped with a partition $\partial = \partial_{\text{in}} + \partial_{\text{out}}$. The conceit is that the subset of nodes ∂ is a boundary that is accessible to other open graphs. Elements of ∂_{in} and ∂_{out} are thought of as inputs and outputs, respectively. Given two open graphs $(E \rightrightarrows N, \partial_{\text{in}} + \partial_{\text{out}})$ and $(E' \rightrightarrows N', \partial'_{\text{in}} + \partial'_{\text{out}})$, such that $\partial_{\text{out}} = \partial'_{\text{in}}$, then we can construct the graph $(E + E' \rightrightarrows (N + N')/\partial_{\text{out}} = \partial'_{\text{in}}, \partial_{\text{in}} + \partial'_{\text{out}})$. For example, . We casually add that by appropriately modifying the definition of a graph morphism, one can define a morphism of open graphs.

cite

insert diagram D1-open graph

insert diagram D2-glueing open graphs

A primary motivation behind this construction is to model the process of connecting networks together. Although, some networks contain additional information that cannot be conveyed by an open graph as described above. To accommodate such demands, we generalize the notion of an open graph to that of an *open object* and develop some basic theory for open objects.

cite examples: circ, zx-calc, petri nets, etc

One feature that distinguishes this work from other related work is our preference for reflexive graphs over directed graphs. Before mentioning our reasons, let us clarify exactly what we mean by these two sorts of graphs. Denote by $\bullet \rightleftarrows \bullet$ the category with two objects $[0]$ and $[1]$ with two arrows $s, t \colon [1] \to [0]$ and an arrow $r \colon [0] \to [1]$ that is a section to both s and t. Throughout this text, the category of reflexive graphs is $\mathbf{RGraph} \coloneqq [\bullet \rightleftarrows \bullet, \mathbf{Set}]$ and the category of directed graphs is $\mathbf{Graph} \coloneqq [\bullet \rightarrow \bullet, \mathbf{Set}]$. The reasons for working with reflexive graphs are myriad. For one, elements $1 \to \Gamma$ of a graph Γ are not just nodes, but nodes with a loop attached. It follows that the underlying nodes functor $\mathbf{RGraph} \to \mathbf{Set}$ is representable by the terminal graph. This is not the case for the underlying nodes functor of type $\mathbf{Graph} \to \mathbf{Set}$. Also, the objects of **RGraph** are truncated simplicial sets. The advantage of this goes, perhaps, beyond the scope of this paper. Suffice to say, when working with graph relations, particularly those homotopical in nature, we desire a well-known model structure to work with. Having said this, let it be known that from this point, any reference to a "graph" will mean a "reflexive graph" unless specified otherwise. This includes cases when we modify "graph"

with an adjective. For instance, by "open graph" we actually mean "open reflexive graph".

Recall that the terms "discrete" and "codiscrete" usually refer to the, respectively, left and right adjoints of a global sections functor. We will use these terms a bit more loosely here. Namely, given a functor $\mathbf{A} \to \mathbf{X}$ forgetting structure, we call a left adjoint **discrete** and a right adjoint **codiscrete**, assuming their existence. For example, the functor $\mathbf{Set} \to \mathbf{RGraph}$ sending given by sending a set x to the graph $x \rightrightarrows x$ with only the required loops is the discrete graphs functor, as it is left adjoint to $\mathbf{RGraph}(1, -)$. Also, the functor $\mathbf{Set} \to \mathbf{RGraph}$ sending a set to the complete graph is the codiscrete functor, as it is right adjoint to $\mathbf{RGraph}(1, -)$.

is this right?

Because open graphs are the archetypal example of an open object, it behooves us to formalize that which we have so far glossed over.

Definition 1.1. Let $\partial\colon \mathbf{Set} \to \mathbf{RGraph}$ be the discrete graph functor. An **open graph** is a cospan of the form $\partial x \to \gamma \leftarrow \partial y$.

Immediately, there emerges two perspectives on open graphs, both alluded to above. The first is that we want to be able to glue together suitable open graphs to form new open graphs. This leads one to consider a category with open graphs $\partial x \to \gamma \leftarrow \partial y$ as arrows from ∂x to ∂yb. As we discuss below in further detail, composition of such arrows uses pushouts. Heuristically, this can be thought of as a "categorical gluing". The second perspective is that open graphs are mathematical objects which deserve their own morphisms. Indeed, a morphism

$$(\partial x \to \gamma \leftarrow \partial y) \to (\partial x' \to \gamma' \leftarrow \partial y')$$

of open graphs is a triple (f, g, h) that fits into a commuting diagram

$$\begin{array}{ccccc} \partial x & \longrightarrow & \gamma & \longleftarrow & \partial y \\ {\scriptstyle \partial f}\downarrow & & {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle \partial h} \\ \partial x' & \longrightarrow & \gamma' & \longleftarrow & \partial y' \end{array}$$

This leads to another category where open graphs are objects, as opposed to arrows.

Having two categories featuring open graphs—one as arrows, the other as objects—one thinks to construct a double category containing all of this structure. In fact, we do this in Section BLAH. In the following section, we generalize open graphs to 'open objects' and construct a pair of categories, one with open objects as arrows and the other with open objects as objects.

input appropriate section

2. Open objects

Our definition of an open graph is ripe for generalization. In the present section, we start by defining *open objects* and constructing a pair of categories emphasizing them. We discover that the open graphs arises as a special case of the open objects. The main result of this section is that the category of open objects and their morphisms form a topos.

Definition 2.1. Fix a functor $\partial\colon \mathbf{A} \to \mathbf{X}$. A **∂-open object**, or **open object** when ∂ is understood, is a cospan of the form $\partial a \to x \leftarrow \partial a'$. We will refer to $\mathbf{A}$ as the category of *boundary types* and to $\mathbf{X}$ as the category of *object types*.

Remark 2.2. It may not seem that this construction warrants a definition, as we are merely renaming a cospan. A fairer portrayal is not that we are renaming a cospan, but that we are reframing a cospan. Calling such a thing an "open object" places us into a network theory context, whereas "cospan" lacks such connotative powers. There is a precedence for this in category theory, particularly in naming functors of a certain sort. For example, a topological quantum field theory is defined to be a functor of a certain sort.

The idea behind an open object $\partial a \to x \leftarrow \partial a'$ is that x is the "closed" object which we open to interaction with other open objects via its *inputs* ∂a and *outputs* $\partial a'$. Specifically, the way in which a pair of open objects can interact is by fusing together to form a new open object. This is modeled by setting open objects as arrows in a category. The fusion of two open objects into one is then composition. But composing cospans requires pushouts, and so henceforth we require that the category of object types has pushouts.

Definition 2.3. Let $\mathbf{X}$ be a category with chosen pushouts and $\partial\colon \mathbf{A} \to \mathbf{X}$ a functor. Denote by $\mathbf{Csp}(\partial)$ the category whose objects are those of $\mathbf{A}$ and whose arrows $a \to b$ are cospans $\partial a \to x \leftarrow \partial b$.

Chosen pushouts are necessary to compose in $\mathbf{Csp}(\partial)$: Compare this to composition in the category, mentioned above, whose morphisms are open graphs.

input diagram D3-dCsp composition

Example 2.4. In the case that we are working in particular with the discrete graph functor $\partial\colon \mathbf{Set} \to \mathbf{RGraph}$