A Foundation for Ledger Structures

Chad Nester
Tallinn University of Technology, Estonia

Abstract

This paper introduces an approach to constructing ledger structures for cryptocurrency systems with basic category theory. Compositional theories of resource convertibility allow us to express the material history of virtual goods, and ownership is modelled by a free construction. Our notion of ownership admits an intuitive graphical representation through string diagrams for monoidal functors.

2012 ACM Subject Classification Theory of computation → Categorical semantics

Keywords and phrases String Diagrams, Category Theory, Blockchains

Digital Object Identifier 10.4230/OASIcs.Tokenomics.2020.7

Funding *Chad Nester*: This research was supported by the ESF funded Estonian IT Academy research measure (project 2014-2020.4.05.19-0001).

1 Introduction

Modern cryptocurrency systems consist of two largely orthogonal parts: A consensus protocol, and the ledger structure it is used to maintain. While consensus protocols have received a lot of attention (see e.g. [10, 7]), the design space of the accompanying ledger structures is barely explored. The recent interest in smart contracts has led to the development of sophisticated ledger structures with complex behaviour (see e.g. [1, 13]). These efforts have been largely *ad hoc*, and the resulting ledger structures are difficult to reason about. This difficulty also manifests in the larger system, which has contributed to several unfortunate incidents involving blockchain technology [2].

A strong mathematical foundation for ledger structures would enable more rigorous development of sophisticated blockchain systems. Further, the ability to reason about the ledger at a high level of abstraction would facilitate analysis of system behaviour. This is important: users of the system must understand it in order to use it with confidence. The formalism we propose has an intuitive graphical representation, which would make this kind of rigorous operational understanding possible on a far wider scale that it would otherwise be.

Blockchain systems are largely concerned with recording the material history of virtual objects, with a particular focus on changes in ownership. The resource theoretic interpretation of string diagrams for symmetric monoidal categories gives a precise mathematical meaning to this sort of material history. Building on this, we consider string diagrams augmented with extra information concerning the ownership of resources. We give these diagrams a precise mathematical meaning in terms of strong monoidal functors, drawing heavily on the work of [9], where our augmented diagrams originated. We show that an augmented resource theory has the same categorical structure as the original, in the sense that the two corresponding categories are equivalent. Finally, we give a simple example of a ledger structure using our machinery.

2 Monoidal Categories as Resource Theories

We assume familiarity with some basic category theory, in particular with symmetric monoidal categories. A good reference is [8]. Throughout, we will write composition in diagrammatic order. That is, the composite of $f : X \to Y$ and $g : Y \to Z$ is written $fg : X \to Z$. We may also write $g \circ f : X \to Z$, but we will *never* write $gf : X \to Z$. We will make heavy use of string diagrams for monoidal categories (see e.g. [11]), which we read from top to bottom (for composition) and left to right (for the monoidal tensor). Our string diagrams for ownership are in fact the string diagrams for monoidal functors of [9].

2.1 Resource Theories

We begin by observing (after [4]) that a symmetric strict monoidal category can be interpreted as a theory of resource convertibility: Each object corresponds to collection of resources with $A \otimes B$ denoting the collection composed of both A and B and the unit I denoting the empty collection. Morphisms $f : A \to B$ are then understood as a way to convert the resources of A to those of B.

For example, consider the free symmetric strict monoidal category on the set

$$\{\texttt{bread}, \texttt{dough}, \texttt{water}, \texttt{flour}, \texttt{oven}\}$$

of atomic objects, subject to the following additional axioms:

$$\texttt{mix} : \texttt{water} \otimes \texttt{flour} \to \texttt{dough} \qquad \texttt{knead} : \texttt{dough} \to \texttt{dough}$$

$$\texttt{bake} : \texttt{dough} \otimes \texttt{oven} \to \texttt{bread} \otimes \texttt{oven}$$

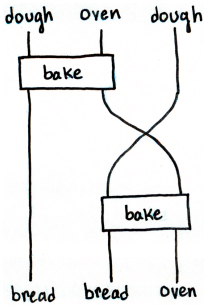
This category can be understood as a theory of resource convertibility for baking bread. The morphism `mix` represents the process of combining water and flour to form a bread dough, `knead` the process of kneading the dough, and `bake` the process of baking the dough in an oven to yield bread (and an oven). While this model has many failings as a theory of bread, it suffices to illustrate the idea. The axioms of a symmetric strict monoidal category provide a natural scaffolding for this theory to live in. For example, consider the morphism

$$(\texttt{bake} \otimes 1_{\texttt{dough}})(1_{\texttt{bread}} \otimes \sigma_{\texttt{oven},\texttt{dough}}\texttt{bake})$$

where $\sigma_{A,B} : A \otimes B \xrightarrow{\sim} B \otimes A$ is the braiding. This morphism has type

$$\texttt{dough} \otimes \texttt{oven} \otimes \texttt{dough} \to \texttt{bread} \otimes \texttt{bread} \otimes \texttt{oven}$$

and describes the transformation of two pieces of dough into two loaves of bread by baking them one after the other in an oven. We obtain a string diagram for this morphism by drawing our objects as wires, and our morphisms as boxes with inputs and outputs. Composition is represented by connecting output wires to input wires, and we represent the tensor product of two morphisms by placing them beside one another. Finally, the braiding is represented by crossing the involved wires. For the morphism in question, we obtain:



We will think of our ledger systems in terms of such string diagrams: The state of the system is a string diagram describing the *material history* of the resources involved, the available resources correspond to the output wires, and changes are effected by appending resource conversions to the bottom of the diagram. From now on we understand a *resource theory* to be a symmetric strict monoidal category with an implicit resource-theoretic interpretation.

2.2 How to Read Equality

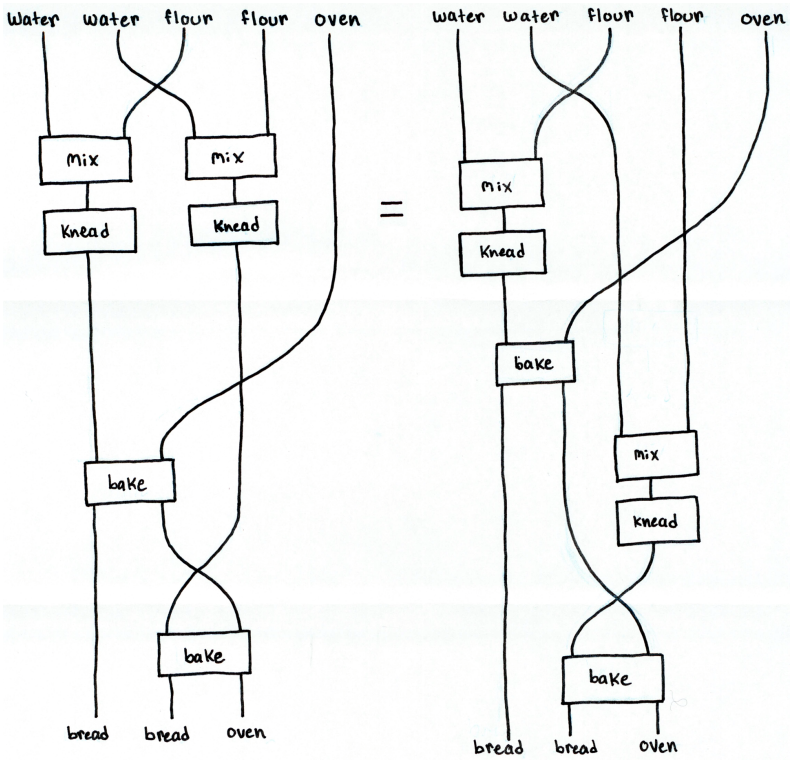
Suppose we have a resource theory $\mathbb{X}$, and two resource transformations $f, g : A \to B$. Each of f and g expresses a different way to transform an instance of resource A into an instance of resource B, but these may not have the same effect. For example, consider $\texttt{knead} : \texttt{dough} \to \texttt{dough}$ and $1_{\texttt{dough}} : \texttt{dough} \to \texttt{dough}$ from our resource theory of bread. Clearly these should not have the same effect on the input dough. This is reflected in our resource theory in the sense that they are not made equal by its axioms. For contrast, we can imagine a (somewhat) reasonable model of baking bread in which there is no difference between kneading the dough once and kneading it many times. We could capture this in our resource theory of baking bread by imposing the equation

$$\texttt{knead} = \texttt{knead} \circ \texttt{knead}$$

In this new resource theory, our equation tells us that kneading dough once has the same effect as kneading it twice, or three times, and so on, since the corresponding morphisms of the resource theory are made equal by its axioms. Of course, the material history described by $\texttt{knead} \circ \texttt{knead}$ is not identical to that described by $\texttt{knead}$. In the former case, the kneading process has been carried out twice in sequence, while in the latter case it has only been carried out once. That these morphisms are equal merely means that the effect of each sequence of events on the dough involved is the same.

We adopt the following general principle in our design and understanding of resource theories: *Two transformations should be equal precisely when they have the same effect on the resources involved.*

We further illustrate this by observing that, by the axioms of a symmetric monoidal category (specifically, by naturality of braiding), the following two transformations in the resource theory of baking (expressed as string diagrams) are equal. The transformation on the left describes baking two loaves of bread by first mixing and kneading two batches of dough before baking them in sequence, while the transformation on the right describes baking two loaves of bread by mixing, kneading, and baking the first batch of dough, and *then* mixing, kneading, and baking the second batch. Thus, according to our resource theory the two procedures will yield the same result – not an entirely unreasonable conclusion!



3 String Diagrams for Ownership

Ledgers used by blockchain systems are largely concerned with *ownership*. For example, in the Bitcoin system, each coin is associated with a computable function called the *validator*, which is used to control access to it. Anyone who wishes to use the coin must supply input data, called a *redeemer*, and the system only allows them to use the coin in question in case running the validator on the redeemer terminates in a fixed amount of time. If the validator is defined only on the data that results from `Alice` digitally signing a nonce generated by the system, then that coin can only be used by `Alice`, who then effectively owns it.

Different use cases call for different authentication schemes. For example, a proposed application of blockchain technology is to improve supply chain accountability by requiring participants to log any transfers and transformations of material on a public ledger (see e.g. [5, 12]). Here ownership implies responsibility, and so for Alice to log the transfer of, say, a ton of steel to Bob, *both* Alice and Bob must ratify the transfer via digital signature.

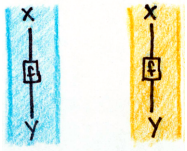
What different use cases have in common is that the resources of the ledger system are associated with ownership data. We leave the interpretation of this ownership data, including the specific details of the authentication scheme unspecified, instead giving a structural account of resource ownership. We develop our account of resource ownership intuitively, and somewhat informally, by introducing addtional features to string diagrams. This is made fully formal in the next section.

3.1 Ownership and Collection Management

Begin by assuming a theory of resources $\mathbb{X}$, and a collection $\mathcal{C}$ of potential resource owners, each of which we associate with a colour for use in our diagrams. Suppose for the remainder that `Alice`, `Bob`, and `Carol` range over $\mathcal{C}$, and are associated with colours as follows:

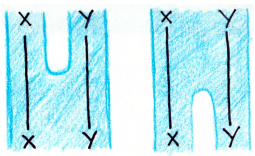
Our goal will be to construct a new theory of resources in which resources and transformations are associated with (owned and carried out by) elements of $\mathcal{C}$. The objects of our new resource theory will be collections of owned objects of $\mathbb{X}$. That is, for each object X of $\mathbb{X}$ and each $\texttt{Alice} \in \mathcal{C}$ we have an object $X^{\texttt{Alice}}$, which we interpret as an instance of resource X owned by Alice, along with the empty collection I and composite collections $X^{\texttt{Alice}} \otimes Y^{\texttt{Bob}}$, in which Alice's instance of X exists alongside an instance of Y owned by Bob.

Similarly, for each transformation $f : X \to Y$ in $\mathbb{X}$, we ask for transformations $f^{\texttt{Alice}} : X^{\texttt{Alice}} \to Y^{\texttt{Alice}}$ and $f^{\texttt{Bob}} : X^{\texttt{Bob}} \to Y^{\texttt{Bob}}$ for all $\texttt{Alice}, \texttt{Bob} \in \mathcal{C}$, whose presence we interpret as the ability of each owner to effect all possible transformations of resources they own. We draw these annotated transformations as, respectively:



Since we are building a theory of resources we must end up with a symmetric monoidal category, so we also assume the presence of the associated morphisms, such as $f^{\texttt{Alice}} \otimes g^{\texttt{Bob}}$ and $\sigma_{X^{\texttt{Alice}}, Y^{\texttt{Bob}}}$.

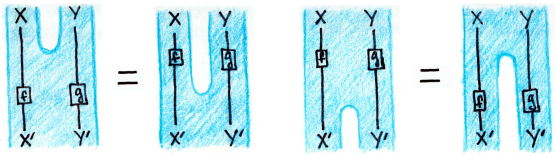
Next, we account for the formal difference between $X^{\texttt{Alice}} \otimes Y^{\texttt{Alice}}$ and $(X \otimes Y)^{\texttt{Alice}}$. In both situations Alice owns an X and a Y, but in the former they are formally grouped together, while in the latter they are formally separated. We understand this formal grouping of Alice's assets by analogy with physical currency. The situation in which Alice's assets are separated is like Alice having two coins worth one euro, while the situation in which they are grouped together is like Alice having one coin worth two euros. In both cases, Alice posesses two euros, but the difference is important: Alice cannot give Bob half of the two euro coin, but can easily give Bob one of the two one euro coins. This distinction is also present in cryptocurrency systems, where there is an operational difference between having funds spread across many addresses and having them collected at one address. Reflecting both the reality of such systems and the principle that one ought to be able to freely reconfigure the formal grouping of things that they own, we ask that for each X, Y objects of $\mathbb{X}$ and each $\texttt{Alice} \in \mathcal{C}$ our new resource theory has morphisms $\phi_{X,Y} : X^{\texttt{Alice}} \otimes Y^{\texttt{Alice}} \to (X \otimes Y)^{\texttt{Alice}}$ and $\psi_{X,Y} : (X \otimes Y)^{\texttt{Alice}} \to X^{\texttt{Alice}} \otimes Y^{\texttt{Alice}}$. We draw these morphisms, respectively, as follows:



These changes of formal grouping should not interact with the resource transformations of our original theory $\mathbb{X}$, since it ought not matter whether Alice combines (splits) her resources before or after transforming them. That is, we we require:

[G.1] $\phi^{\texttt{Alice}}_{X,Y}(f \otimes g)^{\texttt{Alice}} = (f^{\texttt{Alice}} \otimes g^{\texttt{Alice}})\phi^{\texttt{Alice}}_{X',Y'}$,

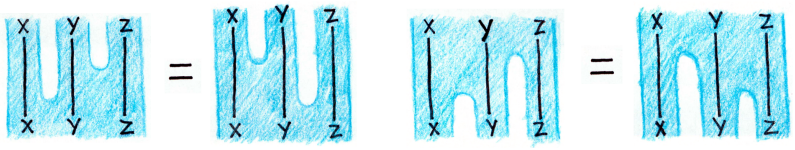
[G.2] $(f \otimes g)^{\texttt{Alice}}\psi^{\texttt{Alice}}_{X',Y'} = \psi^{\texttt{Alice}}_{X,Y}(f^{\texttt{Alice}} \otimes g^{\texttt{Alice}})$



As it stands, there are many non-equal ways for `Alice` to reconfigure the formal grouping of their assets. Since these should all have the same effect, we need them all to be equal as morphisms in our resource theory. It suffices to ask that the $\phi^{\texttt{Alice}}$ and $\psi^{\texttt{Alice}}$ maps give, respectively, associative and coassociative operations, and that they are mutually inverse. That is (associativity and coassociativity):

[G.3] $(\phi^{\texttt{Alice}}_{X,Y} \otimes 1^{\texttt{Alice}}_{Z})\phi^{\texttt{Alice}}_{X\otimes Y,Z} = (1^{\texttt{Alice}}_{X} \otimes \phi^{\texttt{Alice}}_{Y,Z})\phi^{\texttt{Alice}}_{X,Y\otimes Z}$

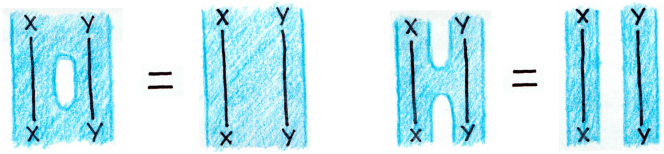
[G.4] $\psi^{\texttt{Alice}}_{X\otimes Y,Z}(\psi^{\texttt{Alice}}_{X,Y} \otimes 1^{\texttt{Alice}}_{Z}) = \psi^{\texttt{Alice}}_{X,Y\otimes Z}(1^{\texttt{Alice}}_{X} \otimes \psi^{\texttt{Alice}}_{Y,Z})$



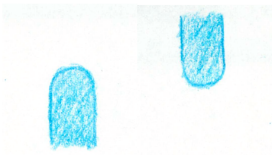
and (mutually inverse):

[G.5] $\psi^{\texttt{Alice}}_{X,Y}\phi^{\texttt{Alice}}_{X,Y} = 1^{\texttt{Alice}}_{X\otimes Y}$

[G.6] $\phi^{\texttt{Alice}}_{X,Y}\psi^{\texttt{Alice}}_{X,Y} = 1^{\texttt{Alice}}_{X} \otimes 1^{\texttt{Alice}}_{Y}$



To complete our treatment of these formal resource groupings, we must deal with the empty case $I^{\texttt{Alice}}$. We insist that `Alice` may freely create and destroy such empty collections via morphisms $\phi^{\texttt{Alice}}_{I} : I \to I^{\texttt{Alice}}$ and $\psi^{\texttt{Alice}}_{I} : I^{\texttt{Alice}} \to I$:



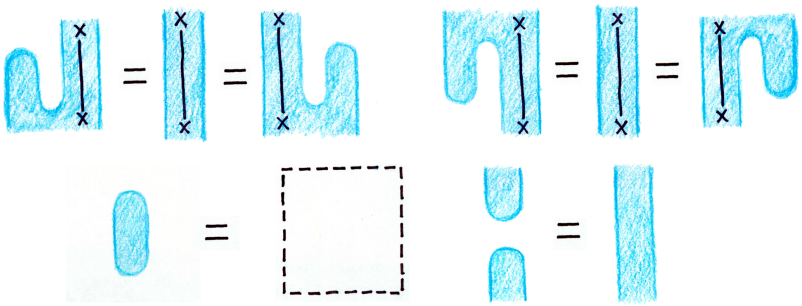
subject to the following axioms, which state that adding or removing nothing from a group or resources has the same effect as doing nothing, and that ϕ_I and ψ_I are mutually inverse, which together ensure that even with ϕ_I and ψ_I in the mix, any two formal regroupings with the same domain and codomain are equal.

[G.7] $(\phi^{\texttt{Alice}}_{I} \otimes 1^{\texttt{Alice}}_{X})\phi^{\texttt{Alice}}_{I,X} = 1^{\texttt{Alice}}_{X} = (1^{\texttt{Alice}}_{X} \otimes \phi^{\texttt{Alice}}_{I})\phi^{\texttt{Alice}}_{X,I}$

[G.8] $\psi^{\texttt{Alice}}_{I,X}(\psi^{\texttt{Alice}}_{I} \otimes 1^{\texttt{Alice}}_{X}) = 1^{\texttt{Alice}}_{X} = \psi^{\texttt{Alice}}_{X,I}(1^{\texttt{Alice}}_{X} \otimes \psi^{\texttt{Alice}}_{I})$

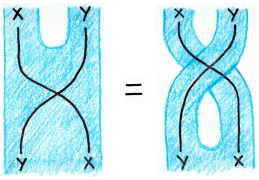
[G.9] $\phi^{\texttt{Alice}}_{I}\psi^{\texttt{Alice}}_{I} = 1_I$

[G.10] $\psi^{\texttt{Alice}}_{I}\phi^{\texttt{Alice}}_{I} = 1^{\texttt{Alice}}_{I}$



Finally, we ask that ϕ and ψ behave coherently with respect to the symmetry maps. It suffices to require that

[**G.11**] $\phi^{\mathtt{Alice}}_{X,Y}\sigma^{\mathtt{Alice}}_{X,Y} = \sigma_{X^{\mathtt{Alice}},Y^{\mathtt{Alice}}}\phi^{\mathtt{Alice}}_{Y,X}$

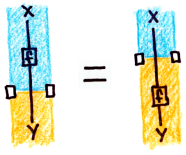


3.2 Change of Ownership

Of course, ownership is not static over time. We require the ability the *change* the owner of a given collection of resources. To this end we add morphisms $\gamma^{\mathtt{Alice},\mathtt{Bob}}_X : X^{\mathtt{Alice}} \to X^{\mathtt{Bob}}$ to our new resource theory for each object X of $\mathbb{X}$, each $\mathtt{Alice}, \mathtt{Bob} \in \mathcal{C}$. We depict these new morphisms in our string diagrams as follows:

As with regrouping, change of ownership should not interact with resource transformations, in the sense that:

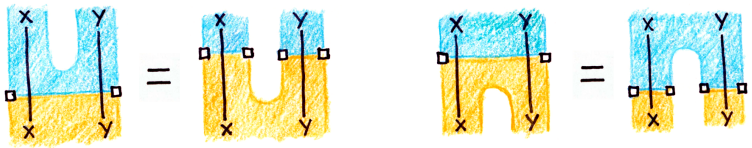
[**O.1**] $f^{\mathtt{Alice}}\gamma^{\mathtt{Alice},\mathtt{Bob}}_Y = \gamma^{\mathtt{Alice},\mathtt{Bob}}_X f^{\mathtt{Bob}}$



Further, change of ownership must behave coherently with respect to the regrouping morphisms in the sense that:

[**O.2**] $\phi^{\mathtt{Alice}}_{X,Y}\gamma^{\mathtt{Alice},\mathtt{Bob}}_{X\otimes Y} = (\gamma^{\mathtt{Alice},\mathtt{Bob}}_X \otimes \gamma^{\mathtt{Alice},\mathtt{Bob}}_Y)\phi^{\mathtt{Bob}}_{X,Y}$

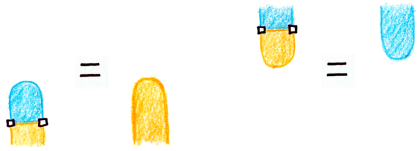
[**O.3**] $\gamma^{\mathtt{Alice},\mathtt{Bob}}_{X\otimes Y}\psi^{\mathtt{Bob}}_{X,Y} = \psi^{\mathtt{Alice}}_{X,Y}(\gamma^{\mathtt{Alice},\mathtt{Bob}}_X \otimes \gamma^{\mathtt{Alice},\mathtt{Bob}}_Y)$



For completeness, we axiomatize the interaction of change of ownership with empty collections by requiring that:

[O.4] $\phi_I^{\texttt{Alice}}\gamma_I^{\texttt{Alice,Bob}} = \phi_I^{\texttt{Bob}}$

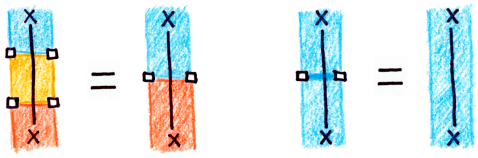
[O.5] $\gamma_I^{\texttt{Alice,Bob}}\psi_I^{\texttt{Bob}} = \psi_I^{\texttt{Alice}}$



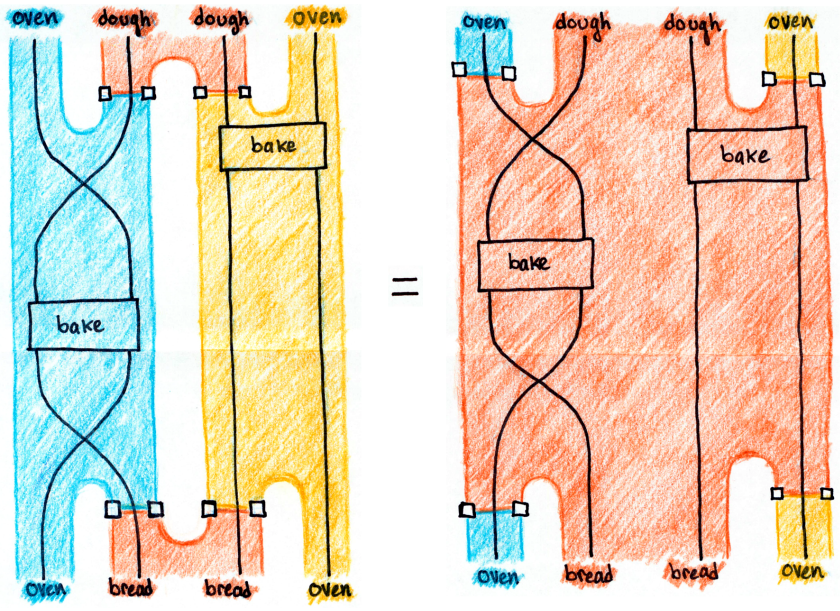
Finally, we insist that if `Alice` gives something to `Bob`, and `Bob` then gives it to `Carol`, this has the same effect as `Alice` giving the thing directly to `Carol`. Similarly, if `Alice` gives something to `Alice`, we insist that this has no effect.

[O.6] $\gamma_X^{\texttt{Alice,Bob}}\gamma_X^{\texttt{Bob,Carol}} = \gamma_X^{\texttt{Alice,Carol}}$

[O.7] $\gamma_X^{\texttt{Alice,Alice}} = 1_X^{\texttt{Alice}}$



We end up with a rather expressive diagrammatic language. For example, if we begin with the resource theory of bread, then our new resource theory is powerful enough to show:



which captures the fact that the sequence of events on the left in which `Carol` gives `Alice` and `Bob` each a portion of dough to bake in their ovens, after which they give the resulting bread to `Carol` *has the same effect* as the sequence of events on the right in which `Alice` and `Bob` give their ovens to `Carol`, who bakes the portions of dough herself before returning the ovens to their original owners. Notice that our diagrammatic representation of this is *much* easier to understand than the corresponding terms in linear syntax!

4 Categorical Semantics

In this section we show how our augmented string diagrams can be given precise mathematical meaning. Specifically, from a resource theory and a set whose elements we think of as entities capable of owning resources, we construct a new resource theory in which all resources are owned by some entity. We finish by showing how to model a simple cyrptocurrency ledger with our machinery.

4.1 Interpreting String Diagrams with Ownership

If $\mathbb{X}$ is a theory of resources and $\mathcal{C}$ is our set, we treat $\mathcal{C}$ as the corresponding discrete category, writing $A : A \to A$ for the identity maps, and form the product category $\mathbb{X} \times \mathcal{C}$. Write objects and maps of this product category as $X^A = (X, A)$ and $f^A = (f, A)$ respectively. Now, define $\mathcal{C}(\mathbb{X})$ to be the free strict symmetric monoidal category on $\mathbb{X} \times \mathcal{C}$ subject to the following additional axioms:

$$\frac{A \in \mathcal{C} \qquad X, Y \text{ objects of } \mathbb{X}}{\phi^A_{X,Y} : X^A \otimes Y^A \to (X \otimes Y)^A \text{ in } \mathcal{C}(\mathbb{X})} \qquad \frac{A \in \mathcal{C}}{\phi^A_I : I \to I^A \text{ in } \mathcal{C}(\mathbb{X})}$$

$$\frac{A \in \mathcal{C} \qquad X, Y \text{ objects of } \mathbb{X}}{\psi^A_{X,Y} : (X \otimes Y)^A \to X^A \otimes Y^A \text{ in } \mathcal{C}(\mathbb{X})} \qquad \frac{A \in \mathcal{C}}{\psi^A_I : I^A \to I \text{ in } \mathcal{C}(\mathbb{X})}$$

$$\frac{A, B \in \mathcal{C} \qquad X \text{ an object of } \mathbb{X}}{\gamma^{A,B}_X : X^A \to X^B \text{ in } \mathcal{C}(\mathbb{X})}$$

and subject to equations **[G.1–11]** and **[O.1–7]** for $\texttt{Alice}, \texttt{Bob}, \texttt{Carol} \in \mathcal{C}$, X, Y, Z objects of $\mathbb{X}$, and f, g morphisms of $\mathbb{X}$.

Clearly, $\mathcal{C}(\mathbb{X})$ is the new resource theory our coloured string diagrams live in. We think of objects X^A and morphisms f^A as being owned and carried out, respectively, by $A \in \mathcal{C}$. The free monoidal structue gives us the ability to compose such transformations sequentially and in parallel, and the additional axioms ensure our ownership interpretation of $\mathcal{C}(\mathbb{X})$ is reasonable.

We can characterize the category-theoretic effect of axioms **[G.1–11]** and **[O.1–5]** as follows:

▶ **Proposition 1.** *For any symmetric monoidal category $\mathbb{X}$ and any set $\mathcal{C}$, there is a strong symmetric monoidal functor*

$$A : \mathbb{X} \to \mathcal{C}(\mathbb{X})$$

for each $A \in \mathcal{C}$. Further, there is a monoidal and comonoidal natural transformation

$$\gamma^{A,B} : A \to B$$

between the functors corresponding to any two $A, B \in \mathcal{C}$.

Proof. Define $A : \mathbb{X} \to \mathcal{C}(\mathbb{X})$ by $A(X) = (X, A)$ on objects, and $A(f) = (f, A)$ on maps. For identity maps, $A(1_X) = (1_X, A) = 1_{(X,A)} = 1_{A(X)}$ since $(1_X, A)$ is the identity on (X, A) in $\mathbb{X} \times \mathcal{C}$. For composition, $A(fg) = (fg, A) = (f, A)(g, A) = A(f)A(g)$.Thus A defines a functor. A is strong symmetric monoidal via the ϕ^A and ψ^A maps together with **[G.1]** through **[G.11]**. Consider $A, B : \mathbb{X} \to \mathcal{C}(\mathbb{X})$ corresponding to $A, B \in \mathcal{C}$. Define $\gamma^{A,B} : A \to B$ to have components $\gamma^{A,B}_X$. Then $\gamma^{A,B}$ is a monoidal and comonoidal via **[O.1]** through **[O.5]**. ◀

Notice that we did not use **[O.6–7]** above. These axioms are motivated by our desire to model resource ownership, but they have an important, if subtle, effect on the theory: they allow us to show that $\mathbb{X}$ and $\mathcal{C}(\mathbb{X})$ are equivalent as categories. This means that any suitably categorical structure is present in $\mathbb{X}$ if and only if it is present in $\mathcal{C}(\mathbb{X})$ as well. For example, products in $\mathbb{X}$ manifest as products in $\mathcal{C}(\mathbb{X})$, morphisms that are monic in $\mathbb{X}$ remain monic in $\mathcal{C}(\mathbb{X})$, and so on. We may be confident that our addition of ownership information has not broken any of the structure of $\mathbb{X}$, or added anything superfluous!

▶ **Proposition 2.** *There is an adjoint equivalence between $\mathbb{X}$ and $\mathcal{C}(\mathbb{X})$ for each functor corresponding to some $A \in \mathcal{C}$.*

Proof. We show that each $A : \mathbb{X} \to \mathcal{C}(\mathbb{X})$ is fully faithful, and essentially surjective, beginning with the latter. To that end, suppose that P is an object of $\mathcal{C}(\mathbb{X})$. We proceed by structural induction: If P is I, then ϕ_0 witnesses $I \simeq A(I)$. If P is an atom (X, A), then $(X, A) = A(X)$. If P is $Q \otimes R$ for some Q, R, then by induction we have that $Q \simeq A(X_1)$ and $R \simeq A(X_2)$ for some objects X_1, X_2 of $\mathbb{X}$. We may now form

$$Q \otimes R \simeq A(X_1) \otimes A(X_2) \xrightarrow{\phi^A_{X_1,X_2}} A(X_1 \otimes X_2)$$

which witnesses $P \simeq A(X_1 \otimes X_2)$. Thus, A is essentially surjective. To see that A is fully faithful, let $U : \mathcal{C}(\mathbb{X}) \to \mathbb{X}$ be the obvious forgetful functor. The required bijection $\mathbb{X}(X, Y) \simeq \mathcal{C}(\mathbb{X})(A(X), A(Y))$ is given by A in one direction and U in the other. It sufffices to show that any morphism $h : A(X) \to A(Y)$ with $U(h) = f$ is such that $h = A(f)$. Notice that since each $\gamma^{A,B}$ is a monoidal and comonoidal natural transformation, there is a term equal to h in which all γ morphisms occur before all other morphisms (in the sense that f occurs before g in fg). Since $h : A(X) \to A(Y)$ we know that in this equal term the composite of the γ must have type $A(X) \to A(X)$, and must therefore be the identity by repeated application of **[O.6]** and **[O.7]**. This gives a term h' containing no γ maps with $h' = h$. Similarly, since the various ϕ and ψ morphisms are natural transformations, we may construct a term h'' by collecting all instances of ϕ and ψ terms at the beginning of h'. Once collected there, the composite of all the ϕ and ψ must have type $A(X) \to A(X)$, and is therefore equal to the identity. At this point we know that $h'' : A(X) \to A(Y)$ is such that $h'' = A(f_1) \cdots A(f_n)$ for some $f_1, \ldots, f_n$ in $\mathbb{X}$. By assumption $f = U(h) = U(h'') = f_1 \cdots f_n$, and therefore $h'' = A(f)$. ◀

4.2 A Simple Example

In this section we attempt to demonstrate the relevance of the above techniques to the cryptocurrency world by building a resource theory that models a simple ledger structure along the lines of Bitcoin [10]. Let $\mathbb{1}$ be the trivial category, with one object, 1, and one morphism, the identity 1_1. Define $\mathbb{N}$ to be the free symmetric strict monoidal category on $\mathbb{1}$, write 0 for the monoidal unit of $\mathbb{N}$, and n for the n-fold tensor product of 1 with itself for all natural numbers $n \geq 1$. Notice that $n + m$ is $n \otimes m$. We will think of the objects n of $\mathbb{N}$ where $n \geq 1$ as *coins*. Of course, $0 = I$ represents the situation in which no coin in present.

Define $\mathbb{N}_\nu$ to be the result of formally adding a morphism $\nu : 0 \to 1$ to $\mathbb{N}$, write $\nu_0 = 1_0 : 0 \to 0$, and $\nu_n : 0 \to n$ for the n-fold tensor product of ν with itself for $n \geq 1$. These morphisms confer the ability to create new coins, so we imagine their use would be restricted in practice. We will not ask for the ability to destroy coins, although there would be no theoretical obstacle to doing so.

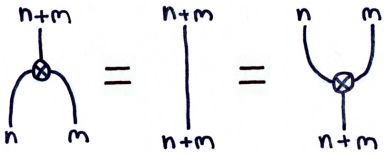
Now, let $\mathcal{C}$ be a collection of colours, which we can think of as standing in for cryptographic key pairs, or simply entities capable of owning coins. Consider $\mathcal{C}(\mathbb{N}_\nu)$. Objects are lists $n_1^{c_1} \otimes \cdots \otimes n_k^{c_k}$, which we interpret as lists of coins, where $n_i^{c_i}$ is a coin of value n_i belonging to

$c_i \in \mathcal{C}$. The morphisms are either ν_n^c for some $c \in \mathcal{C}$, the structural morphisms of a monoidal category, or the ϕ, ψ, and γ morphisms added by our construction. For $n, m \in \mathbb{N}$ and $\mathtt{Alice}, \mathtt{Bob} \in \mathcal{C}$, the maps $\phi_{n,m}^{\mathtt{Alice}} : n^{\mathtt{Alice}} \otimes m^{\mathtt{Alice}} \to (n+m)^{\mathtt{Alice}}$ and $\psi_{n,m}^{\mathtt{Alice}} : (n+m)^{\mathtt{Alice}} \to n^{\mathtt{Alice}} \otimes m^{\mathtt{Alice}}$ allow users to combine and split their coins in a value-preserving manner, and the $\gamma_n^{\mathtt{Alice},\mathtt{Bob}}$ maps allow them to exchange coins.

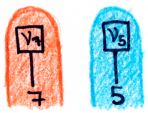
Now, a ledger is a (syntactic) morphism $a : I \to A$ of $\mathcal{C}(\mathbb{N}_\nu)$. A transaction to be included in a consists of a transformation $f : X \to Y$ of $\mathcal{C}(\mathbb{N}_\nu)$ along with information about which outputs of a are to be the inputs of the transformation, which we package as $t = \pi_t(1 \otimes f \otimes 1) : A \to B$. The result of including transaction t in ledger a is then the composite ledger $t \circ a : I \to B$. Put another way, a ledger is given by a list of transformations in $\mathcal{C}(\mathbb{N}_\nu)$:

$$I \xrightarrow{t_1} A_1 \xrightarrow{t_2} \cdots \xrightarrow{t_k} A_k$$

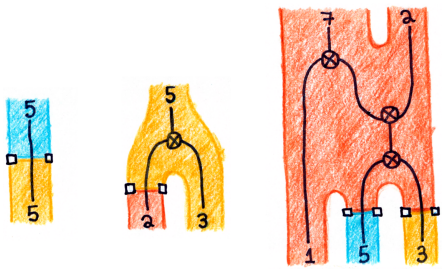
For the purpose of illustration, we differentiate between $m+n$ and $m \otimes n$ in our string diagrams for $\mathbb{N}_\nu$. We do so by means of the string diagrams for (not necessarily strict) monoidal categories (see e.g. [3]), as in:



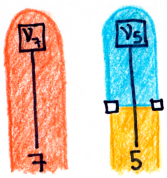
Now, suppose we have a ledger $a : I \to \nu_7^{\mathtt{Carol}} \otimes \nu_5^{\mathtt{Alice}}$:



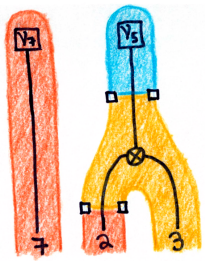
and resource transformations f_1, f_2, f_3 defined, respectively, by:



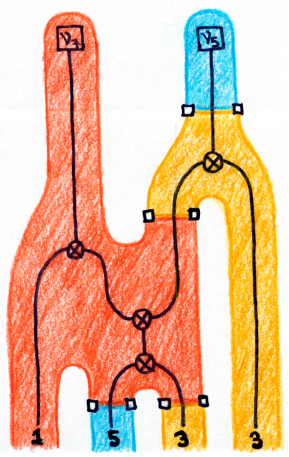
Now, form transaction $t_1 = (1_7^{\mathtt{Carol}} \otimes f_1)$ and append it to a to obtain $t_1 \circ a$



Next, form transaction $t_2 = (1_7^{\mathtt{Carol}} \otimes f_2)$ and append it to obtain $t_2 \circ t_1 \circ a$



Finally, form transaction $t_3 = (f_3 \otimes 1_3^{\mathsf{Bob}})$ and append it to obtain $t_3 \circ t_2 \circ t_1 \circ a$



In this manner, we capture the evolution of the ledger over time. Of course, we can also reason about whether two sequences of transactions result in the same ledger state by comparing the corresponding morphisms for equality, although in the case of $\mathcal{C}(\mathbb{N}_\nu)$ there isn't much point, since all morphisms $A \to B$ are necessarily equal.

5 Conclusions and Future Work

We have seen how the resource theoretic interpretation of monoidal categories, and in particular their string diagrams, captures the sort of material history that concerns ledger structures for blockchain systems. Additionally, we have shown how to freely add a notion of ownership to such a resource theory, and that the resulting category is equivalent to the original one. We have also shown that these resource theories with ownership admit an intuitive graphical calculus, which is more or less that of monoidal functors and natural transformations. Finally, we have used our machinery to construct a simple ledger structure and show how it might be used in practice.

While we do not claim to have solved the problem of providing a rigorous foundation for the development of ledger structures in its entirety, we feel that our approach shows promise. There are a few diffrent directions for future research. One is the development of categorical models for more sophisticated ledger structures, with the eventual goal being to give a rigorous formal account of smart contracts. Another is to explore the connections of the current work with formal treatments of accounting, such as [6].

References

1. N. Atzei, M. Bartoletti, T. Cimoli, S. Lande, and R. Zunino. Unravelling bitcoin smart contracts. In *POST 2018*, volume 10804 of *LNCS*, pages 217–242, 2018.
2. N. Atzei, M. Bartolietti, and T. Cimoli. A survey of attacks on ethereum smart contracts. In *POST 2017*, volume 10204 of *LNCS*, pages 164–186, 2017.
3. J.R.B. Cockett and R.A.G. Seely. Proof theory of the cut rule. In E. Landry, editor, *Categories for the Working Philosopher*, pages 223–261. Oxford University Press, 2017.
4. B. Coecke, T. Fritz, and R.W. Spekkens. A mathematical theory of resources. *Information and Computation*, 250:59–86, 2016.
5. K. Jabbar and P. Bjorn. Infrastructural grind: Introducing blockchain technology in the shipping domain. In *GROUP 2018*, 2018.
6. P. Katis, N. Sabadini, and R.F.C. Walters. On partita doppia. *CoRR*, 1998.
7. A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. *CRYPTO 2017, Part I, volume 10401 of LNCS*, 2017.
8. S. Mac Lane. *Categories for the Working Mathematician*. Springer, 1971.
9. M.B. McCurdy. Graphical methods for tannaka duality of weak bialgebras and weak hopf algebras. *Theory and Applications of Categories*, 26:233–280, 2011.
10. S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL: `https://bitcoin.org/bitcoin.pdf`.
11. Peter Selinger. A survey of graphical languages for monoidal categories. In *New Structures for Physics*, pages 289–355. Springer, 2010.
12. M. Staples, S. Chen, S. Falamaki, A. Ponomarev, P. Rimba, A.B. Tran, I. Weber, X. Xu, and J. Zhu. *Risks and Opportunities for Systems Using Blockchain and Smart Contracts*. Data61 (CSIRO), Sydney, 2017.
13. Gavin Wood. Ethereum: A secure decentralized generalised transaction ledger, 2014. URL: `https://gavwood.com/paper.pdf`.