1

Physics, Topology, Logic and Computation: A Rosetta Stone DRAFT VERSION ONLY

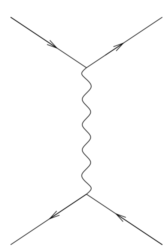
John Baez[1]
Michael Stay[2]

[1]Department of Mathematics, University of California, Riverside CA 92521, USA
[2]Google, 1600 Amphitheatre Pkwy, Mountain View CA 94043, USA

1.1 Introduction

Category theory is a very general formalism, but there is a certain special way that physicists use categories which turns out to have close analogues in topology, logic and computation. A category has *objects* and *morphisms*, which represent *things* and *ways to go between things*. In physics, the objects are often *physical systems*, and the morphisms are *processes* turning a state of one physical system into a state of another system — perhaps the same one. In quantum physics we often formalize this by taking *Hilbert spaces* as objects, and *linear operators* as morphisms.

Sometime around 1949, Feynman [53] realized that in quantum field theory it is useful to draw linear operators as diagrams:



This lets us reason with them pictorially. We can warp a picture without changing the operator it stands for: all that matters is the topology, not the geometry. By the 1980s, people realized that this idea sets up a powerful analogy between quantum physics and topology! This analogy is now being intensively exploited in string theory, loop quantum gravity, and especially topological quantum field theory.

Meanwhile, quite separately, logicians had begun using categories where the objects represent *propositions* and the morphisms represent *proofs*. The idea is that a proof is a process going us go from one proposition (the hypothesis) to another (the conclusion). Later, computer scientists started using categories where the objects represent *data types* and the morphisms represent *programs*. They also started using 'flow charts' to describe programs. Abstractly, these are very much like Feynman diagrams!

The logicians and computer scientists were never very far from each other. Indeed, the 'Curry–Howard correspondence' relating proofs to programs has been well-known at least since the early 1970s, with roots stretching back earlier [33, 50]. But, it is only rather recently that the logicians and computer scientists bumped into the physicists and topologists. One reason is the current interest in quantum cryptography, quantum computation and other forms of quantum information processing. For example, the 'topological quantum computers' currently envisaged by researchers at Microsoft [41] use the analogies between physics, topology, and computation so thoroughly that it is impossible to say where one subject ends and the other starts.

Regardless of whether useful quantum computers are ever built, it is worth laying out these analogies in one place. They suggest that seemingly disparate fields of research are really just branches of a science yet to be built: a general science of systems and processes. Building this science will be difficult. There are good reasons for this, but also bad ones. One bad reason is that different fields use different terminology and notation.

The original Rosetta Stone, created in 196 BC, contains versions of the same text in three languages: demotic Egyptian, hieroglyphic script and classical Greek. Its rediscovery by Napoleon's soldiers let modern Egyptologists decipher the hieroglyphs. Eventually this led to a vast increase in our understanding of Egyptian culture.

At present, the deductive systems in mathematical logic look like hieroglyphs to most physicists. Similarly, quantum field theory is Greek to most computer scientists, and so on. So, there is a need for a new Rosetta Stone to aid researchers attempting to translate between fields. Table 1.1 gives our guess as to what this Rosetta Stone might look like.

	object	morphism
Physics	system	process
Topology	manifold	cobordism
Logic	proposition	proof
Computation	data type	program

Table 1.1. The Rosetta Stone (pocket version)

The rest of this paper expands on this tables by comparing how how categories are used in physics, topology, logic, and computation. Unfortunately, these different fields focus on slightly different kinds of categories. Though most physicists don't know it, quantum physics has long made use of 'compact symmetric monoidal categories'. Topology — especially knot theory — uses 'compact braided monoidal categories', which are slightly more general. However, it became clear by the 1990s that these more general gadgets are useful in physics too. Logic and computer science used to focus on 'cartesian closed categories' — where 'cartesian' can be seen, roughly, as an antonym of 'quantum'. However, thanks to work on linear logic and quantum computation, some logicians and computer scientists have dropped their insistence on cartesianness: now they study more general sorts of 'closed symmetric monoidal categories'.

In Section 1.2 we explain all these concepts, how they illuminate the analogy between physics and topology, and how to work with them using string diagrams. We assume no prior knowledge of category theory, only a willingness to learn some. We give precise definitions, but leave most of the calculations as exercises for the reader.

In Section 1.3 we explain how closed symmetric monoidal categories correspond to a small fragment of ordinary propositional logic, which also happens to be a fragment of Girard's 'linear logic' [45]. In Section 1.4 we explain how closed symmetric monoidal categories correspond to a simple model of computation: a version of the lambda calculus that allows for quantum effects. In Section 1.5, we summarize by presenting a larger version of the Rosetta Stone.

Our treatment of all four subjects — physics, topology, logic and computation — is bound to seem sketchy and idiosyncratic to practitioners of these subjects. Our excuse is that we wish to emphasize certain analogies while saying no more than absolutely necessary. To make up for this, we include many references for those who wish to dig deeper.

1.2 The Analogy Between Physics and Topology

1.2.1 Overview

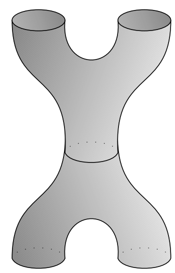
Currently our best theories of physics are general relativity and the Standard Model of particle physics. The first describes gravity without taking quantum theory into account; the second describes all the other forces taking quantum theory into account, but ignores gravity. So, our world-view is deeply schizophrenic. The field where physicists struggle to solve this problem is called *quantum gravity*, since it is widely believed that the solution requires treating gravity in a way that takes quantum theory into account.

Nobody is sure how to do this, but there is a striking similarity between two of the main approaches: string theory and loop quantum gravity. Both rely on the analogy between physics and topology shown in Table 1.2.

On the left we have a basic ingredient of quantum theory: the category Hilb whose objects are Hilbert spaces, used to describe physical *systems*, and whose morphisms are linear operators, used to describe physical *processes*. On the right we have a basic structure in differential topology: the category nCob, whose objects are $(n-1)$-dimensional manifolds, used to describe , and whose morphisms are n-dimensional cobordisms, used to describe *spacetime*. We give precise definitions below; for now, just to dispel any possible terror caused by the term 'cobordism', here is a picture of one when $n = 2$:

Physics	Topology
Hilbert space (system)	$(n-1)$-dimensional manifold (space)
operator between Hilbert spaces (process)	cobordism between $(n-1)$-dimensional manifolds (spacetime)
composition of operators	composition of cobordisms
identity operator	identity cobordism

Table 1.2. Analogy between physics and topology



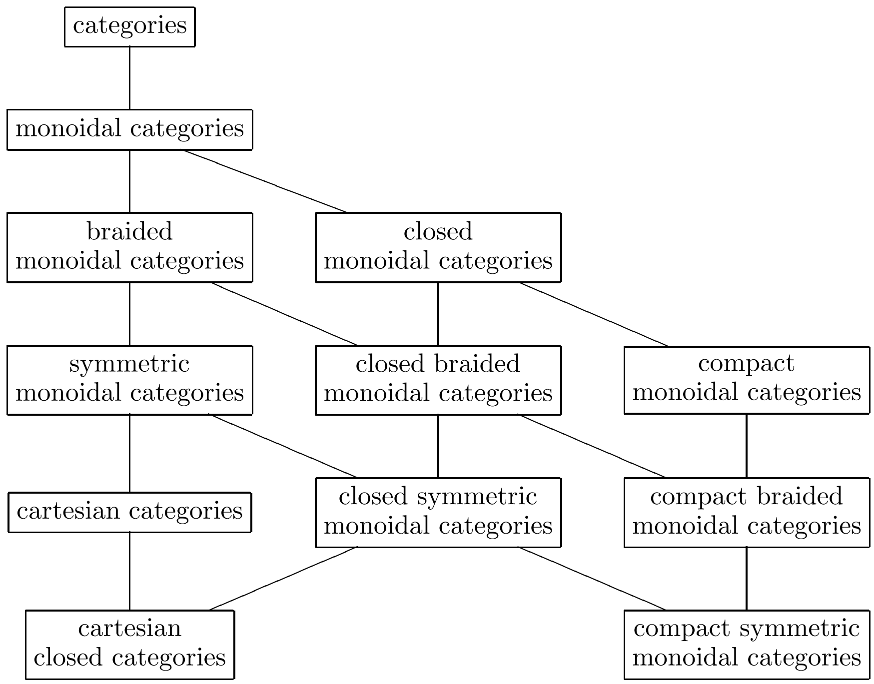
We can think of this as a 2-dimensional 'spacetime' going between 1-dimensional manifolds describing 'space'.

As we shall see, Hilb and nCob share many structural features. Moreover, both are very different from the more familiar category Set, whose objects are sets and whose morphisms are functions. Elsewhere we have argued at great length that this is important for better understanding quantum mechanics [8] and even quantum gravity [7]. The idea is that if Hilb is more like nCob than Set, maybe we should stop thinking of a quantum process as a function from one set of states to another. Instead, maybe we should think of it as resembling a 'spacetime' going between spaces of dimension one less.

This idea sounds strange, but the simplest example is something very practical, used by physicists every day: a Feynman diagram. This is a 1-dimensional graph going between 0-dimensional collections of points, with edges and vertices labelled in certain ways. Feynman diagrams are topological entities, but they describe linear operators. String theory and loop quantum gravity use higher-dimensional versions of Feynman diagrams to do a similar job.

Here we will not focus on the puzzles of quantum mechanics or quantum gravity. Instead we take a different tack, simply explaining some basic concepts from category theory and showing how Set, Hilb, nCob and categories of tangles give examples. A recurring theme, however, is that Set is very different from the other examples.

To help the reader safely navigate the sea of jargon, here is a chart of the concepts we shall explain in this section:



The category Set is cartesian closed, while Hilb and nCob are compact symmetric monoidal.

1.2.2 Categories

Category theory was born around 1945, with Eilenberg and Mac Lane [38] giving the definitions of 'categories', 'functors' between categories, and 'natural transformations' between functors. By now there are many introductions to the subject [32, 65, 69], including some available for free online [16, 47]. However, we begin at the beginning:

Definition 1. *A* **category** C *consists of:*

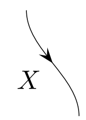
- *a collection of* **objects***, where if* X *is an object of* C *we write* $X \in C$*, and*
- *for every pair of objects* (X, Y)*, a set* $\hom(X, Y)$ *of* **morphisms** *from* X *to* Y*. We call this set* $\hom(X, Y)$ *a* **homset***. If* $f \in \hom(X, Y)$*, then we write* $f\colon X \to Y$*.*

such that:

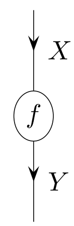
- *for every object* X *there is an* **identity morphism** $1_X\colon X \to X$*;*
- *morphisms are composable: given* $f\colon X \to Y$ *and* $g\colon Y \to Z$*, there is a* **composite morphism** $gf\colon X \to Z$*; sometimes also written* $g \circ f$*.*
- *an identity morphism is both a* **left and a right unit** *for composition: if* $f\colon X \to Y$*, then* $f 1_X = f = 1_Y f$*; and*

- *composition is* **associative***:* $(hg)f = h(gf)$ *whenever either side is well-defined.*

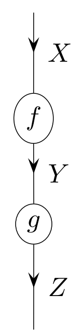
A category is the simplest framework where we can talk about systems (objects) and processes (morphisms). To visualize these, we can use 'Feynman diagrams' of a very primitive sort, which mathematicians call 'string diagrams'. The term 'string' here has little to do with string theory: instead, the idea is that objects of our category label 'strings', or 'wires':



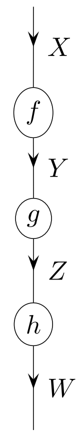
and morphisms $f\colon X \to Y$ are 'black boxes' with an input wire of type X and an output wire of type Y:



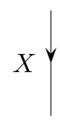
We compose two morphisms by connecting the output wire of one black box to the input wire of the next. So, the composite of $f\colon X \to Y$ and $g\colon Y \to Z$ looks like this:



Associativity of composition is then implicit:



is our notation for both $h(gf)$ and $(hg)f$. Similarly, if we draw the identity morphism $1_X\colon X \to X$ as a piece of wire of type X:



then the left and right unit laws are also implicit.

There are countless examples of categories, but we will focus on four:

- Set: the category where objects are sets.
- Hilb: the category where objects are finite-dimensional Hilbert spaces.
- nCob: the category where morphisms are n-dimensional cobordisms.
- Tang_k: the category where morphisms are k-codimensional tangles.

As we shall see, all four are closed symmetric monoidal categories, at least when k is big enough. However, the most familiar of the lot, namely Set, is the odd man out: it is 'cartesian'.

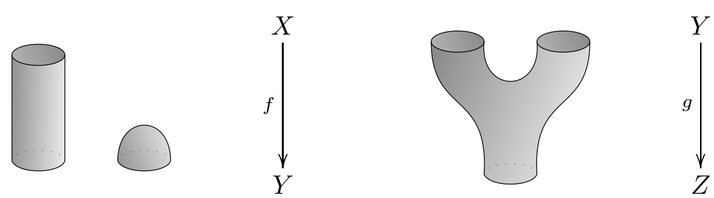
Traditionally, mathematics has been founded on the category Set, where the objects are *sets* and the morphisms are *functions*. So, when we study systems and processes in physics, it is tempting to specify a system by giving its set of states, and a process by giving a function from states of one system to states of another.

However, in quantum physics we do something subtly different: we use categories where objects are *Hilbert spaces* and morphisms are *bounded linear operators*. We specify a system by giving a Hilbert space, but this Hilbert space is not really the set of states of the system: a state is actually a ray in Hilbert space. Similarly, a bounded linear operator is not precisely a function from states of one system to states of another.

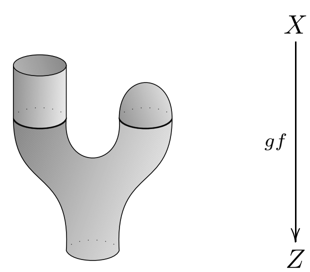
In the day-to-day practice of quantum physics, what really matters is not sets of states and functions between them, but Hilbert space and operators. One of the virtues of category theory is that it frees us from the 'Set-centric' view of traditional mathematics and lets us view quantum physics on its own terms. As we shall see, this sheds new light on the quandaries that have always plagued our understanding of the quantum realm [8].

To avoid technical issues that would take us far afield, let us define Hilb to be the category where objects are *finite-dimensional Hilbert spaces* and morphisms are *linear operators* (automatically bounded in this case). This should be fine for those interested in quantum information theory and some aspects of quantum computation, but it should upset experts on quantum field theory. See Section 1.6.1 for some remarks on the infinite-dimensional case.

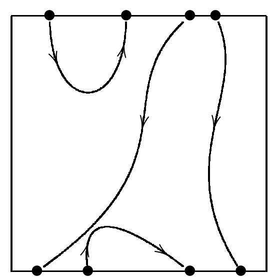
In physics we also use categories where the objects represent choices of *space*, and the morphisms represent choices of *spacetime*. The simplest is nCob, where the objects are $(n-1)$-*dimensional manifolds*, and the morphisms are n-*dimensional cobordisms*. We explain nCob in more detail in Section 1.6.2, but roughly speaking, a cobordism $f\colon X \to Y$ is an n-dimensional manifold whose boundary is the disjoint union of the $(n-1)$-dimensional manifolds X and Y. Here are a couple of cobordisms in the case $n = 2$:



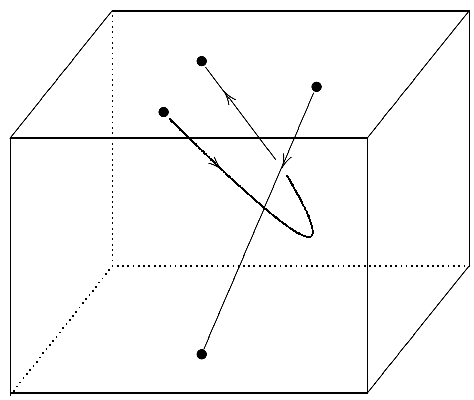
We compose them by gluing the 'output' of one to the 'input' of the other. So, in the above example $gf\colon X \to Z$ looks like this:



Another kind of category important in physics has objects representing *collections of particles*, and morphisms representing their *worldlines and interactions*. Feynman diagrams are the classic example, but in these diagrams the 'edges' are not taken literally as particle trajectories. An example with closer ties to topology is Tang_k. We defer the details to Section 1.6.3, but very roughly speaking, objects in Tang_k are collections of points in a k-dimensional cube, while morphisms are 'framed oriented tangles' in a $(k+1)$-dimensional cube. Since a picture is worth a thousand words, here is a picture of a morphism in Tang_1:



and here is a picture of a morphism in Tang_2:



In these pictures we have drawn the 'orientation' as a little arrow on each curve in the tangle. In applications to physics, the curves are worldlines of particles, and the arrows say whether each particle is going forwards or backwards in time, following Feynman's idea that antiparticles are particles going backwards in time. We have not drawn the 'framing'. If we did, each curve would be replaced by a 'ribbon'. In applications to physics, this keeps track of how each particle twists. This is especially important for fermions, where a 2π twist acts nontrivially.

It is difficult to do much with categories without discussing the maps between them. A map between categories is called a 'functor':

Definition 2. *A* **functor** $F: C \to D$ *from a category* C *to a category* D *is map sending:*

- *any object* $X \in C$ *to an object* $F(X) \in D$*,*
- *any morphism* $f: X \to Y$ *in* C *to a morphism* $F(f): F(X) \to F(Y)$ *in* D*,*

in such a way that:

- F **preserves identities***: for any object* $X \in C$*,* $F(1_X) = 1_{F(X)}$*;*

- F **preserves composition**: *for any pair of morphisms* $f: X \to Y$, $g: Y \to Z$ *in* C, $F(gf) = F(g)F(f)$.

Functors are good for many things, but here is one: we can think of a functor $F: C \to D$ as a 'representation' of C in D. The idea here is that objects and morphisms of some 'abstract' category C are sent to objects and morphisms in some more 'concrete' category D. For example, consider an abstract group G, which we can think of as a category with one object and all morphisms invertible. Then a representation G on a finite-dimensional Hilbert space is just a functor $F: G \to \mathrm{Hilb}$. Similarly, an action of G on a set is a functor $F: G \to \mathrm{Set}$.

Ever since Lawvere's 1963 thesis on functorial semantics [62], the idea of functors as representations has become pervasive in modern logic. However, the terminology is different! In logic, the category C is called a 'theory', and the functor $F: C \to D$ is called a 'model' of this theory.

Though Lawvere was interested in theories in physics as well as in logic, his way of thinking caught on in physics only much later, around 1988, with Segal's work on conformal field theories [74] and Atiyah's work on topological field theories [5]. Again, the terminology is a bit different: physicists prefer to call the functor $F: C \to D$ a 'theory', rather than a model of a theory.

If functors are models, natural transformations are maps between models:

Definition 3. *Given two functors* $F, G: C \to D$, *a* **natural transformation** $\alpha: F \Rightarrow G$ *assigns to every object* X *in* C *a morphism* $\alpha_X: F(X) \to G(X)$ *such that for any morphism* $f: X \to Y$ *in* C, *the equation* $\alpha_Y\, F(f) = G(f)\, \alpha_X$ *holds in* D. *In other words, this square commutes:*

$$\begin{tikzcd} F(X) \arrow[r, "F(f)"] \arrow[d, "\alpha_X"'] & F(Y) \arrow[d, "\alpha_Y"] \\ G(X) \arrow[r, "G(f)"'] & G(Y) \end{tikzcd}$$

(Going across and then down equals going down and then across.)

Definition 4. *A* **natural isomorphism** *between functors* $F, G: C \to D$ *is a natural transformation* $\alpha: F \Rightarrow G$ *such that* α_X *is an isomorphism for every* $X \in D$.

1.2.3 Monoidal Categories

In physics, it is often useful to think of two systems sitting side by side as forming a single system. In topology, the disjoint union of two manifolds is again a manifold in its own right. In logic, the conjunction of two statement is again a statement. In programming we can combine two data types into a single 'product type'. The concept of 'monoidal category' unifies all these examples in a single framework.

A monoidal category C has a functor $\otimes : C \times C \to C$ that takes two objects X and Y and puts them together to give a new object $X \otimes Y$. To make this precise, we need the cartesian product of categories:

Definition 5. *The* **cartesian product** $C \times C'$ *of categories* C *and* C' *is the category where:*

- *an object is a pair* (X, X') *consisting of an object* $X \in C$ *and an object* $X' \in C'$*;*
- *a morphism from* (X, X') *to* (Y, Y') *is a pair* (f, f') *consisting of a morphism* $f : X \to Y$ *and a morphism* $f' : X' \to Y'$*;*
- *composition is done componentwise:* $(g, g')(f, f') = (gf, g'f')$*;*
- *identity morphisms are defined componentwise:* $1_{(X,X')} = (1_X, 1_{X'})$.

Mac Lane [64] defined monoidal categories in 1963. The subtlety of the definition lies in the fact that $(X \otimes Y) \otimes Z$ and $X \otimes (Y \otimes Z)$ are not usually equal. Instead, we should specify an isomorphism between them, called the 'associator'. Similarly, while a monoidal category has a 'unit object' I, it is not usually true that $I \otimes X$ and $X \otimes I$ equal X. Instead, we should specify isomorphisms $I \otimes X \cong X$ and $X \otimes I \cong X$. To be manageable, all these isomorphisms must then satisfy certain equations:

Definition 6. *A* **monoidal category** *consists of:*

- *a category* C,
- *a* **tensor product** *functor* $\otimes : C \times C \to C$,
- *a* **unit object** $I \in C$*,*
- *a natural isomorphism called the* **associator***, assigning to each triple of objects* $X, Y, Z \in C$ *an isomorphism*
$$a_{X,Y,Z} : (X \otimes Y) \otimes Z \xrightarrow{\sim} X \otimes (Y \otimes Z),$$
- *natural isomorphisms called the* **left** *and* **right unitors***, assigning to each object* $X \in C$ *isomorphisms*
$$l_X : I \otimes X \xrightarrow{\sim} X$$
$$r_X : X \otimes I \xrightarrow{\sim} X,$$

such that:

- *for all* $X, Y \in C$ *the* **triangle equation** *holds:*

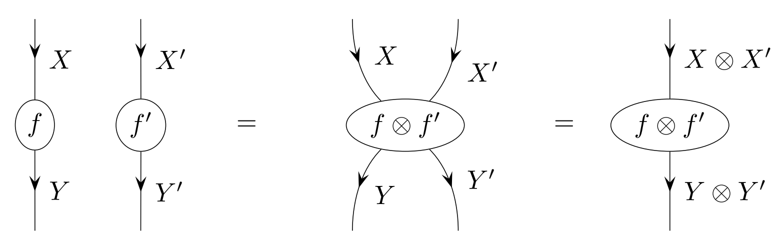
$$\begin{tikzcd}
(X \otimes I) \otimes Y \arrow[rr, "a_{X,I,Y}"] \arrow[dr, "r_X \otimes 1_Y"'] & & X \otimes (I \otimes Y) \arrow[dl, "1_X \otimes l_Y"] \\
& X \otimes Y &
\end{tikzcd}$$

- *for all* $W, X, Y, Z \in C$, *the* **pentagon equation** *holds:*

$$
\begin{tikzcd}
 & ((W \otimes X) \otimes Y) \otimes Z \arrow[ddl, "a_{W \otimes X, Y, Z}"'] \arrow[dr, "a_{W,X,Y} \otimes 1_Z"] & \\
 & & (W \otimes (X \otimes Y)) \otimes Z \arrow[dd, "a_{W, X \otimes Y, Z}"] \\
(W \otimes X) \otimes (Y \otimes Z) \arrow[ddr, "a_{W,X,Y \otimes Z}"'] & & \\
 & & W \otimes ((X \otimes Y) \otimes Z) \arrow[dl, "1_W \otimes a_{X,Y,Z}"] \\
 & W \otimes (X \otimes (Y \otimes Z)) &
\end{tikzcd}
$$

When we have a tensor product of four objects, there are five ways to parenthesize it, and at first glance the associator lets us build two isomorphisms from $W \otimes (X \otimes (Y \otimes Z))$ to $((W \otimes X) \otimes Y) \otimes Z$. But, the pentagon equation says these isomorphisms are equal. When we have tensor products of even more objects there are even more ways to parenthesize them, and even more isomorphisms between them built from the associator. However, Mac Lane showed that the pentagon identity implies these isomorphisms are all the same. Similarly, if we also assume the triangle equation, all isomorphisms with the same source and target built from the associator, left and right unit laws are equal.

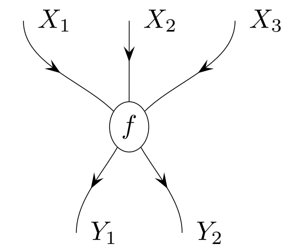
In a monoidal category we can do processes in 'parallel' as well as in 'series'. Doing processes in series is just composition of morphisms, which works in any category. But in a monoidal category we can also tensor morphisms $f\colon X \to Y$ and $f'\colon X' \to Y'$ and obtain a 'parallel process' $f \otimes f'\colon X \otimes X' \to Y \otimes Y'$. We can draw this in various ways:



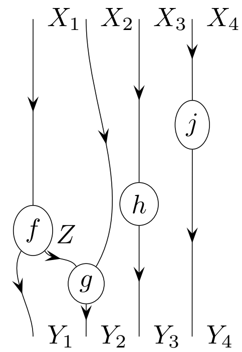
More generally, we can draw any morphism

$$f\colon X_1 \otimes \cdots \otimes X_n \to Y_1 \otimes \cdots \otimes Y_m$$

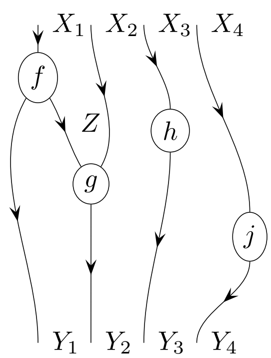
as a black box with n input wires and m output wires:



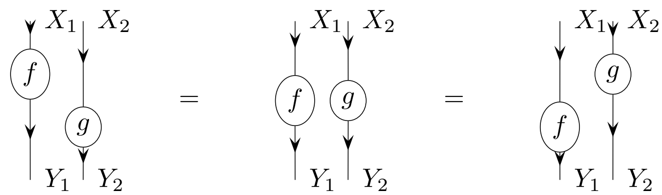
By composing and tensoring these morphisms, we can build up elaborate pictures resembling Feynman diagrams:



The laws governing a monoidal category allow us to neglect associators and unitors when drawing such pictures, without getting in trouble. We can also deform the picture in a wide variety of ways without changing the morphism it describes. For example, the above morphism equals this one:



Everyone who uses string diagrams for calculations in monoidal categories starts by worrying about the rules of the game: *precisely how* can we deform these pictures without changing the morphisms they describe? Instead of stating the rules precisely — which gets a bit technical — we urge you to explore for yourself what is allowed and what is not. For example, show that we can slide black boxes up and down like this:



For a formal treatment of the rules governing string diagrams, try the original papers by Joyal and Street [51] and the book by Yetter [87].

Now let us turn to examples. Here it is crucial to realize that the same category can often be equipped with different tensor products, resulting in different monoidal categories:

- There is a way to make Set into a monoidal category where $X \otimes Y$ is the cartesian product $X \times Y$ and the unit object is any one-element set. Note that this tensor product is not strictly associative, since $(x,(y,z)) \neq ((x,y),z)$, but there's a natural isomorphism $(X \times Y) \times Z \cong X \times (Y \times Z)$, and this is our associator. Similar considerations give the left and right unitors. In this monoidal category, the tensor product of $f\colon X \to Y$ and $f'\colon X' \to Y'$ is the function

$$\begin{array}{rl} f \times f' : X \times X' & \to Y \times Y' \\ (x, x') & \mapsto (f(x), f'(x')). \end{array}$$

  There is also a way to make Set into a monoidal category where $X \otimes Y$ is the disjoint union of X and Y, which we shall denote by $X + Y$. Here the unit object is the empty set. Again, as indeed with all these examples, the associative law and left/right unit laws hold only up to natural isomorphism. In this monoidal category, the tensor product of $f\colon X \to Y$ and $f'\colon X' \to Y'$ is the function

$$\begin{array}{rl} f + f'\colon X + X' & \to Y + Y' \\ x & \mapsto \begin{cases} f(x) & \text{if } x \in X, \\ f'(x) & \text{if } x \in X'. \end{cases} \end{array}$$

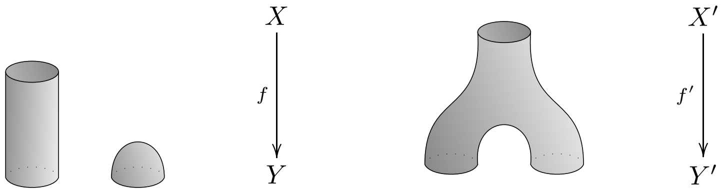
  However, in what follows, when we speak of Set *as a monoidal category, we always use the cartesian product!*

- There is a way to make Hilb into a monoidal category with the usual tensor product of Hilbert spaces: $\mathbb{C}^n \otimes \mathbb{C}^m \cong \mathbb{C}^{nm}$. In this case the unit object I can be taken to be an 1-dimensional Hilbert space, for example $\mathbb{C}$.

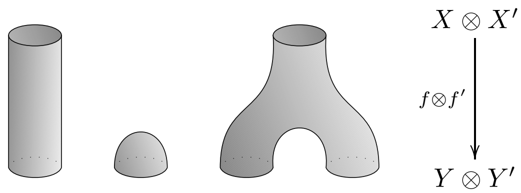
  There is also way to make Hilb into a monoidal category where the tensor product is the direct sum: $\mathbb{C}^n \oplus \mathbb{C}^m \cong \mathbb{C}^{n+m}$. In this case the unit object is the zero-dimensional Hilbert space, $\{0\}$.

  However, in what follows, when we speak of Hilb *as a monoidal category, we always use the usual tensor product!*

- The tensor product of objects and morphisms in nCob is given by disjoint union. For example, the tensor product of these two morphisms:



is this:



- The category Tang_k is monoidal when $k \geq 1$, where the the tensor product is given by disjoint union. For example, given these two tangles:

  two tangles, side by side, $f\colon X \to Y$ and $f'\colon X' \to Y'$.

  their tensor product is this:

  picture of their tensor product

The example of Set with its cartesian product is different from our other three main examples, because the cartesian product of sets $X \times X'$ comes equipped with functions called 'projections' to the sets X and X':

$$X \xleftarrow{p'} X \times X' \xrightarrow{p} X'$$

Our other main examples lack this feature — though Hilb made into a monoidal category using $\oplus$ has projections. Also, every set has a unique function to the the one-element set:

$$!_X\colon X \to I.$$

Again, our other main examples lack this feature, though Hilb made into a monoidal category using $\oplus$ has it. A fascinating feature of quantum mechanics is that we make Hilb into a monoidal category using $\otimes$ instead of $\oplus$, even though the latter approach would lead to a category more like Set.

We can isolate the special features of the cartesian product of sets and its projections, obtaining a definition that applies to any category:

Definition 7. *Given objects X and X' in some category, we say an object $X \times X'$ equipped with morphisms*

$$X \xleftarrow{p} X \times X' \xrightarrow{p'} X'$$

is a **cartesian product** *(or simply* **product***) of X and X' if for any object Q and morphisms*

$$\begin{tikzcd} & Q \arrow[dl, "f"'] \arrow[dr, "f'"] & \\ X & & X' \end{tikzcd}$$

there exists a unique morphism $g\colon Q \to X \times X'$ making the following diagram commute:

$$\begin{tikzcd} & Q \arrow[dl, "f"'] \arrow[d, "g"] \arrow[dr, "f'"] & \\ X & X \times X' \arrow[l, "p"] \arrow[r, "p'"'] & X' \end{tikzcd}$$

(That is, $f = pg$ and $f' = p'g$.) We say a category has **binary products** *if every pair of objects has a product.*

The product may not exist, and it may not be unique, but when it exists it is unique up to a canonical isomorphism. This justifies our speaking of 'the' product of objects X and Y when it exists, and denoting it as $X \times Y$.

The definition of cartesian product, while absolutely fundamental, is a bit scary at first sight. To illustrate its power, let us do something with it: combine two morphisms $f\colon X \to Y$ and $f'\colon X' \to Y'$ into a single morphism

$$f \times f'\colon X \times X' \to Y \times Y'.$$

The definition of cartesian product says how to build a morphism of this sort out of a pair of morphisms: namely, morphisms from $X \times X'$ to Y and Y'. If we take these to be fp and $f'p'$, we obtain $f \times f'$:

$$\begin{tikzcd} & X \times X' \arrow[dl, "fp"'] \arrow[d, "f \times f'"] \arrow[dr, "f'p'"] & \\ Y & Y \times Y' \arrow[l, "p"'] \arrow[r, "p'"] & Y' \end{tikzcd}$$

Next, let us isolate the special features of the one-element set:

Definition 8. *An object 1 in a category C is* **terminal** *if for any object $Q \in C$ there exists a unique morphism from Q to 1, which we denote as $!_Q\colon Q \to 1$.*

Again, a terminal object may not exist and may not be unique, but it is unique up to a canonical isomorphism. This is why we can speak of 'the' terminal object of a category, and denote it by a specific symbol, 1.

We have introduced the concept of binary products. One can also talk about n-ary products for other values of n, but a category with binary products has n-ary products for all $n \geq 1$, since we can construct these as iterated binary products. The case $n = 1$ is trivial, since the product of one object is just that object itself (up to canonical isomorphism). The remaining case is $n = 0$. The zero-ary product of objects, if it exists, is just the terminal object. So, we make the following definition:

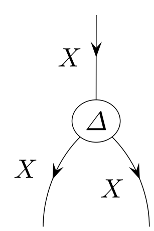
Definition 9. *A category has* **finite products** *if it has binary products and a terminal object.*

A category with finite products can always be made into a monoidal category by choosing a specific product $X \times Y$ to be to the tensor product $X \otimes Y$, and choosing a specific terminal object to be the unit object. It takes a bit of work to show this! A monoidal category of this form is called **cartesian**.

In a cartesian category, we can 'duplicate and delete information'. In general, the definition of cartesian products gives a way to take two morphisms $f_1 : Q \to X$ and $f_2 : Q \to Y$ and combine them into a single morphism from Q to $X \times Y$. If we take $Q = X = Y$ and take f_1 and f_2 to be the identity, we obtain the **diagonal** or **duplication** morphism:

$$\Delta_X : X \to X \times X.$$

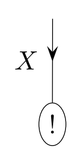
In the category Set one can check that this maps any element $x \in X$ to the pair (x, x). In general, we can draw the diagonal as follows:



Similarly, we call the unique map to the terminal object

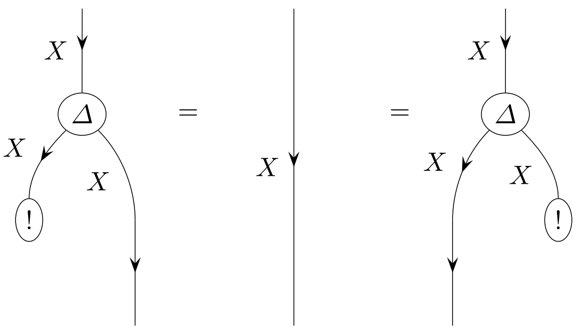
$$!_X : X \to 1$$

the **deletion** morphism, and draw it as follows:



Note that we draw the unit object as an empty space.

A fundamental fact about cartesian categories is that duplicating something and then deleting either copy is the same as doing nothing at all! In string diagrams, this says:



We leave the proof as an exercise for the reader.

1.2.4 Braided Monoidal Categories

In physics, there is often a process that lets us 'switch' two systems by moving them around each other. In topology, there is a tangle that describes the process of switching two points:

basic braid in a box

In logic, we can switch the order of two statements in a conjunction: the statement 'X and Y' is isomorphic to 'Y and X'. In computation, there is a simple program that switches the order of two pieces of data. A monoidal category in which we can do this sort of thing is called 'braided':

Definition 10. *A* **braided monoidal category** *consists of:*

- *a monoidal category C,*
- *a natural isomorphism called the* **braiding** *that assigns to every pair of objects $X, Y \in C$ an isomorphism*

$$b_{X,Y} \colon X \otimes Y \to Y \otimes X,$$

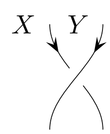
such that the **hexagon equations** *hold:*

$$\begin{tikzcd}
X \otimes (Y \otimes Z) \arrow[r, "a^{-1}_{X,Y,Z}"] \arrow[d, "b_{X,Y \otimes Z}"'] & (X \otimes Y) \otimes Z \arrow[r, "b_{X,Y} \otimes 1_Z"] & (Y \otimes X) \otimes Z \arrow[d, "a_{Y,X,Z}"] \\
(Y \otimes Z) \otimes X & Y \otimes (Z \otimes X) \arrow[l, "a^{-1}_{Y,Z,X}"] & Y \otimes (X \otimes Z) \arrow[l, "1_Y \otimes b_{X,Z}"]
\end{tikzcd}$$

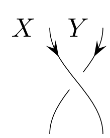
$$\begin{tikzcd}
(X \otimes Y) \otimes Z \arrow[r, "a_{X,Y,Z}"] \arrow[d, "b_{X \otimes Y,Z}"'] & X \otimes (Y \otimes Z) \arrow[r, "1_X \otimes b_{Y,Z}"] & X \otimes (Z \otimes Y) \arrow[d, "a^{-1}_{X,Z,Y}"] \\
Z \otimes (X \otimes Y) & (Z \otimes X) \otimes Y \arrow[l, "a_{Z,X,Y}"] & (X \otimes Z) \otimes Y \arrow[l, "b_{Z,X} \otimes 1_Y"]
\end{tikzcd}$$

The first hexagon equation says that switching the object X past $Y \otimes Z$ all at once is the same as switching it past Y and then past Z (with some associators thrown in to move the parentheses). The second one is similar: it says switching $X \otimes Y$ past Z all at once is the same as doing it in two steps.

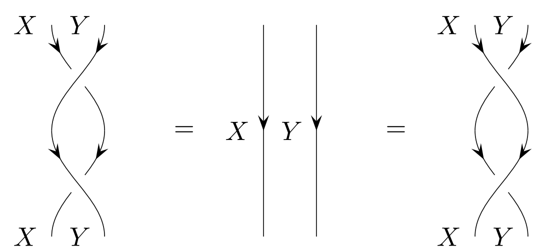
In string diagrams, we draw the braiding $b_{X,Y}\colon X \otimes Y \to Y \otimes X$ like this:



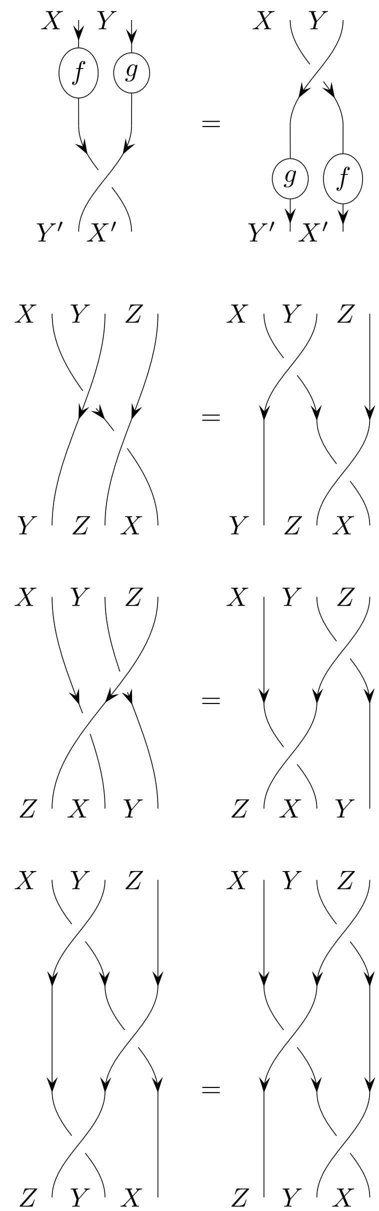
We draw its inverse $b^{-1}_{X,Y}$ like this:



This is a nice notation, because it makes the equations saying that $b_{X,Y}$ and $b^{-1}_{X,Y}$ are inverses 'topologically true':



We urge you to prove the following equations:



If you get stuck, here are some hints. The first equation follows from the naturality of the braiding. The next two follow from the hexagon equations. The last is called the **Yang–Baxter equation** and follows from a combination of naturality and the hexagon equations [52].

Next, here are some examples. There can be many different ways to give a monoidal category a braiding, or none. However, most of our favorite examples come with well-known 'standard' braidings:

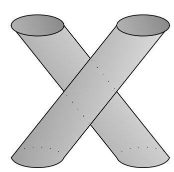
- Any cartesian category automatically becomes braided, and in Set with its cartesian product, this standard braiding is given by:

$$\begin{aligned} b_{X,Y} : X \times Y &\to Y \times X \\ (x, y) &\mapsto (y, x). \end{aligned}$$

- In Hilb with its usual tensor product, the standard braiding is given by:

$$\begin{aligned} b_{X,Y} : X \otimes Y &\to Y \otimes X \\ x \otimes y &\mapsto y \otimes x. \end{aligned}$$

- The monoidal category nCob has a standard braiding where $b_{X,Y}$ is diffeomorphic to the disjoint union of cylinders $X \times [0, 1]$ and $Y \times [0, 1]$. For 2Cob this braiding looks as follows when X and Y are circles:



$$\begin{tikzcd} X \otimes Y \arrow[d, "b_{X,Y}"'] \\ Y \otimes X \end{tikzcd}$$

- The monoidal category Tang_k has a standard braiding when $k \geq 2$. For $k = 2$ this looks as follows when X and Y are each a single point:

basic braid in a box (again)

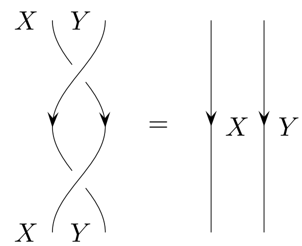
The example of Tang_k illustrates an important pattern. Tang_0 is just a category, because in 0-dimensional space we can only do processes in 'series': that is, compose morphisms. Tang_1 is a monoidal category, because in 1-dimensional space we can also do processes in 'parallel': that is, tensor morphisms. Tang_2 is a braided monoidal category, because in 2-dimensional space there is room to move one object around another. Next we shall see what happens when space has 3 or more dimensions!

1.2.5 Symmetric Monoidal Categories

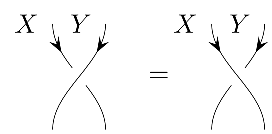
Sometimes switching two objects and switching them again is the same as doing nothing at all. Indeed, this situation is very familiar. So, the first braided monoidal categories to be discovered were 'symmetric' ones [64]:

Definition 11. *A* **symmetric monoidal category** *is a braided monoidal category where the braiding satisfies* $b_{X,Y} = b^{-1}_{Y,X}$.

So, in a symmetric monoidal category,



or equivalently,



Any cartesian category automatically becomes a symmetric monoidal category, so Set is symmetric. It is also easy to check that Hilb, nCob are symmetric monoidal categories. So is Tang_k for $k \geq 3$.

Interestingly, Tang_k 'stabilizes' at $k = 3$: increasing the value of k beyond this value merely gives a category equivalent to Tang_3. The reason is that we can already untie all knots in 4-dimensional space; adding extra dimensions has no real effect. This is part of a conjectured larger pattern called the 'Periodic Table' of n-categories [10], shown in Table 1.3.

An n-category has not only morphisms going between objects, but 2-morphisms going between morphisms, 3-morphisms going between 2-morphisms and so on up to n-morphisms. In topology we can use n-categories to describe tangled higher-dimensional surfaces [11], and in physics we can use them to describe not just particles but also strings and higher-dimensional membranes [10, 12]. The Rosetta Stone we are describing concerns only the $n = 1$ column of the Periodic Table — thus, particles with 1-dimensional worldlines, or tangles in the traditional sense. So, it is probably just a fragment of a larger, still buried n-categorical Rosetta Stone.

1.2.6 Closed Categories

In quantum mechanics, one can encode a linear operator $f\colon X \to Y$ into a quantum state using a technique called 'gate teleportation' [48]. In topology, there is a way to take any tangle $f\colon X \to Y$ and bend the input back around to make it part of the output. In logic, we can take a proof that goes from some assumption X to some conclusion Y and turn it into a proof that goes from no assumptions to the conclusion 'X implies Y'. In computer science,

	$n = \mathbf{0}$	$n = \mathbf{1}$	$n = \mathbf{2}$
$k = \mathbf{0}$	sets	categories	2-categories
$k = \mathbf{1}$	monoids	monoidal categories	monoidal 2-categories
$k = \mathbf{2}$	commutative monoids	braided monoidal categories	braided monoidal 2-categories
$k = \mathbf{3}$	‘’	symmetric monoidal categories	sylleptic monoidal 2-categories
$k = \mathbf{4}$	‘’	‘’	symmetric monoidal 2-categories
$k = \mathbf{5}$	‘’	‘’	‘’
$k = \mathbf{6}$	‘’	‘’	‘’

Table 1.3. The Periodic Table: conjectured descriptions of $(n + k)$-categories with only one j-morphism for $j < k$.

we can take any program that takes input of type X and produces output of type Y, and think of it as a piece of data of a new type: a ‘function type’. The underlying concept that unifies all these examples is the concept of a ‘closed category’.

Given objects X and Y in any category C, there is a *set* of morphisms from X to Y, denoted $\hom(X, Y)$. In a closed category there is also an *object* of morphisms from X to Y, which we denote by $X \multimap Y$. (Many other notations are also used.) In this situation we speak of an ‘internal hom’, since the object $X \multimap Y$ lives inside C, instead of ‘outside’, in the category of sets.

Closed categories were introduced in 1966, by Eilenberg and Kelly [37]. While these authors were able to define a closed structure for any category, it turns out that the internal hom is most easily understood for monoidal categories. The reason is that when our category has a tensor product, it is closed precisely when morphisms from $X \otimes Y$ to Z are in natural one-to-one correspondence with morphisms from X to $Y \multimap Z$. In other words, it is closed when we have a natural isomorphism

$$\begin{array}{c} \hom(X \otimes Y, Z) \cong \hom(X, Y \multimap Z) \\ f \mapsto \tilde{f} \end{array}$$

For example, in the category Set, if we take $X \otimes Y$ to be the cartesian product $X \times Y$, then $Y \multimap Z$ is just the set of functions from Y to Z, and we have a one-to-one correspondence between

- functions f that eat elements of $X \times Y$ and spit out elements of Z

and

- functions $\tilde{f}$ that eat elements of X and spit out functions from Y to Z.

This correspondence goes as follows:

$$\tilde{f}(x)(y) = f(x, y).$$

Before considering other examples, we should make the definition of 'closed monoidal category' completely precise. For this we must note that for any category C, there is a functor

$$\mathrm{hom}\colon C^{\mathrm{op}} \times C \to \mathrm{Set}.$$

Definition 12. *The* **opposite category** C^{op} *of a category* C *has the same objects as* C*, but a morphism* $f\colon x \to y$ *in* C^{op} *is a morphism* $f\colon y \to x$ *in* C*, and the composite* gf *in* C^{op} *is the composite* fg *in* C*.*

Definition 13. *For any category* C*, the* **hom functor**

$$\mathrm{hom}\colon C^{\mathrm{op}} \times C \to \mathrm{Set}$$

sends any object $(X, Y) \in C^{\mathrm{op}} \times C$ *to the set* $\mathrm{hom}(X, Y)$*, and sends any morphism* $(f, g) \in C^{\mathrm{op}} \times C$ *to the function*

$$\begin{array}{rcl} \mathrm{hom}(f, g)\colon \mathrm{hom}(X, Y) & \to & \mathrm{hom}(X', Y') \\ h & \mapsto & ghf \end{array}$$

when $f\colon X' \to X$ *and* $g\colon Y \to Y'$ *as morphisms in* C*.*

Definition 14. *A monoidal category* C *is* **closed** *if there is an* **internal hom** *functor*

$$\multimap\colon C^{\mathrm{op}} \times C \to C$$

together with a natural isomorphism c *called* **currying** *that assigns to any objects* $X, Y, Z \in C$ *a bijection*

$$\begin{array}{rcl} c_{X,Y,Z}\colon \mathrm{hom}(X \otimes Y, Z) & \xrightarrow{\sim} & \mathrm{hom}(Y, X \multimap Z) \\ f & \mapsto & \tilde{f}. \end{array}$$

The term 'currying' is mainly used in computer science, after the work of Curry [33]. We are working with **left closed** monoidal categories: there are also **right closed** ones, where currying goes like this:

$$c_{X,Y,Z}\colon \mathrm{hom}(X \otimes Y, Z) \xrightarrow{\sim} \mathrm{hom}(X, Y \multimap Z)$$

We shall ignore this subtlety, since the difference between left and right closed evaporates for a braided monoidal category: the braiding gives an isomorphism $X \otimes Y \cong Y \otimes X$.

All our examples of monoidal categories are closed, but we shall see that, yet again, Set is different from the rest:

- The cartesian category Set is closed, where $X \multimap Y$ is just the set of functions from X to Y. In Set or any other cartesian closed category, the internal hom $X \multimap Y$ is usually denoted Y^X. To minimize the number of different notations and emphasize analogies between different contexts, we shall not do this: we shall always use $X \multimap Y$.
- The symmetric monoidal category Hilb with its usual tensor product is closed, where $X \multimap Y$ is the set of linear operators from X to Y, made into a Hilbert space in a standard way. In this case we have

$$X \multimap Y \cong X^* \otimes Y$$

  where X^* is the dual of the Hilbert space X, that is, the set of linear operators $f\colon X \to \mathbb{C}$, made into a Hilbert space in the usual way.
- The monoidal category Tang_k ($k \geq 1$) is closed... need information on orientations here!!!
- The symmetric monoidal category $n\mathrm{Cob}$ is also closed.

Except for Set, all these examples are actually 'compact'. This basically means that $X \multimap Y$ is isomorphic to $X^* \otimes Y$, where X^* is some object called the 'dual' of X. To be precise, we should say this isomorphism is natural:

Definition 15. *A monoidal closed category C is* **compact** *if there is a* **dualizing** *functor*

$$\begin{aligned} *\colon C^{\mathrm{op}} &\to C \\ X &\mapsto X^* \end{aligned}$$

such that the internal hom functor is naturally isomorphic to this composite:

$$\begin{array}{ccccc} C^{\mathrm{op}} \times C & \xrightarrow{*\times 1} & C \times C & \xrightarrow{\otimes} & C \\ (X,Y) & \mapsto & (X^*,Y) & \mapsto & X^* \otimes Y \end{array}$$

This definition is elegant but quite compressed. To unravel its consequences, note that in a compact monoidal category we can curry the right unitor

$$r_X\colon X \otimes I \to X$$

and obtain a morphism called the **unit** of X:

$$i_X\colon I \to X^* \otimes X.$$

Since currying is invertible, we can also 'uncurry' the inverse of the right unitor

$$r_{X^*}\colon X^* \to X^* \otimes I$$

and obtain a morphism called the **counit** of X:

$$e_X\colon X \otimes X^* \to I.$$

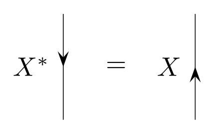
With some strenuous calculations, the reader can show these satisfy two equations called the **zig-zag equations**, which say these diagrams commute:

$$\begin{tikzcd}
X \otimes I \arrow[r, "1_X \otimes i_X"] \arrow[d, "r_X"'] & X \otimes (X^* \otimes X) \arrow[r, "a^{-1}_{X,X^*,X}"] & (X \otimes X^*) \otimes X \arrow[d, "e_X \otimes 1_X"] \\
X & & I \otimes X \arrow[ll, "l_X"]
\end{tikzcd}$$

$$\begin{tikzcd}
I \otimes X^* \arrow[r, "i_X \otimes 1_X"] \arrow[d, "l_X"'] & (X^* \otimes X) \otimes X^* \arrow[r, "a_{X^*,X,X^*}"] & X^* \otimes (X \otimes X^*) \arrow[d, "1_{X^*} \otimes e_X"] \\
X^* & & X^* \otimes I \arrow[ll, "r_{X^*}"]
\end{tikzcd}$$

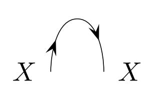
With even more work, one can show the converse: any monoidal category in which every object X has an object X^* equipped with unit and counit satisfying the zig-zag equations is compact!

The point of this reformulation, and the reason for the name 'zig-zag', become clear if we borrow some ideas from Feynman. In physics, if X is the Hilbert space of internal states of some particle, X^* is the Hilbert space for the corresponding antiparticle. Feynman realized that it is enlightening to think of antiparticles as particles going backwards in time. So, we draw a wire labelled by X^* as a wire labelled by X, but with an arrow pointing 'backwards in time': that is, up instead of down:

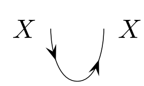


(Here we should admit that most physicists use the opposite convention, where time marches up the page. Since we read from top to bottom, we prefer to let time run down the page.)

Given this, we should draw the unit as a **cap**:

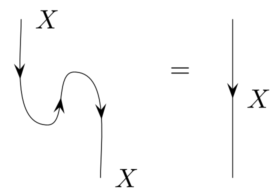


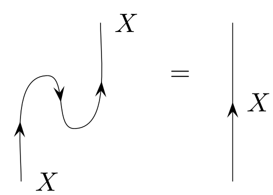
and the counit as a **cup**:



In Feynman diagrams, these describe the *creation* and *annihilation* of virtual particle-antiparticle pairs!

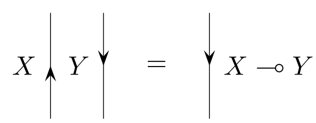
In this notation, the zig-zag equations look like this:



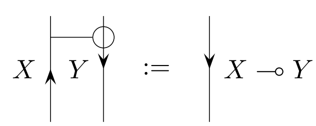


They really describe two ways of straightening out a zig-zag. This is especially vivid in examples from topology, such as Tang_k and $n\mathrm{Cob}$.

In a compact monoidal category, the internal hom $X \multimap Y$ is naturally isomorphic $X^* \otimes Y$. So, it is harmless to redefine the internal hom to *equal* $X^* \otimes Y$, and then we have:

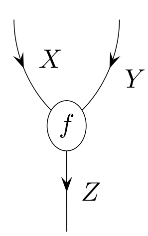


In general, closed monoidal categories don't allow arrows pointing up, so drawing the internal hom is more of a challenge. We can use the same style of notation as long as we add a decoration — a **clasp** — that binds two strings together:

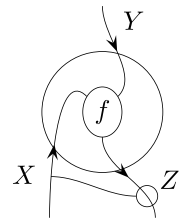


Only when our closed monoidal category happens to be compact can we eliminate the clasp.

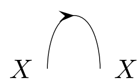
Then, since we draw a morphism $f\colon X \otimes Y \to Z$ like this:



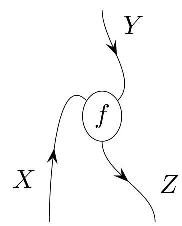
we can draw its curried version $\tilde{f}: Y \to X \multimap Z$ by bending down the input wire labelled X to make it part of the output:



Note that where we bent back the wire labelled X, a cap like this appeared



Closed monoidal categories don't really have a cap unless they are compact. So, we drew a **bubble** enclosing f and the cap, to keep us from doing any illegal manipulations. In the compact case, both the bubble and the clasp are unecessary, so we can draw $\tilde{f}$ like this:



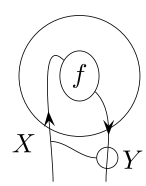
An important special case of currying gives the **name** of a morphism $f: X \to Y$,

$$\ulcorner f \urcorner : I \to X \multimap Y.$$

This is obtained by currying the morphism

$$f r_x : I \otimes X \to Y.$$

In string diagrams, we draw $\ulcorner f \urcorner$ as follows:



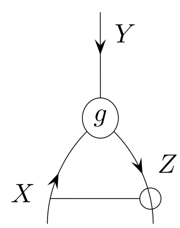
In the category Set, the unit object I is the one-element set. So, a morphism from I to any set Q picks out a point of Q. In particular, the name $\ulcorner f \urcorner : I \to X \multimap Y$ picks out the element of $X \multimap Y$ corresponding to the function $f: X \to Y$. More generally, in any cartesian closed category, a morphism from 1 to an object Q is called a **point** of Q. So, even in this case, we can say the name of a morphism $f: X \to Y$ is a point of $X \multimap Y$.

Something similar works for Hilb, though this example is compact rather than cartesian. In Hilb, the unit object I is just $\mathbb{C}$. So, a nonzero morphism from I to any Hilbert space Q picks out a nonzero vector in Q, which we can normalize to obtain a **state** in Q: that is, a unit vector. In particular, the the name of a nonzero morphism $f\colon X \to Y$ gives a state of $X^* \otimes Y$. This method of encoding operators as states is the basis of 'gate teleportation' [48].

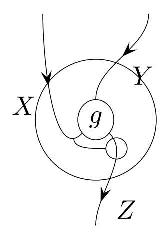
Currying is a bijection, so we can also **uncurry**:

$$\begin{aligned} c^{-1}_{X,Y,Z}\colon \hom(Y, X \multimap Z) &\xrightarrow{\sim} \hom(X \otimes Y, Z) \\ g &\mapsto \underset{\sim}{g}. \end{aligned}$$

Since we draw a morphism $g\colon Y \to X \multimap Z$ like this:



we draw its 'uncurried' version $\underset{\sim}{g}\colon X \otimes Y \to Z$ by bending the output X up to become an input:



Again, we must put a bubble around the 'cup' formed when we bend down the wire labelled Y, unless we are in a compact monoidal category.

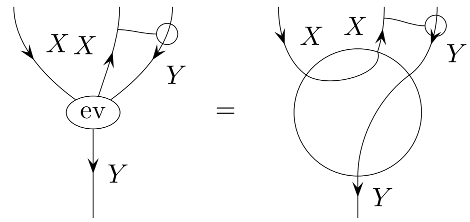
A good example of uncurrying is the **evaluation** morphism:

$$\mathrm{ev}_{X,Y}\colon X \otimes (X \multimap Y) \to Y.$$

This is obtained by uncurrying the identity

$$1_{X \multimap Y}\colon (X \multimap Y) \to (X \multimap Y).$$

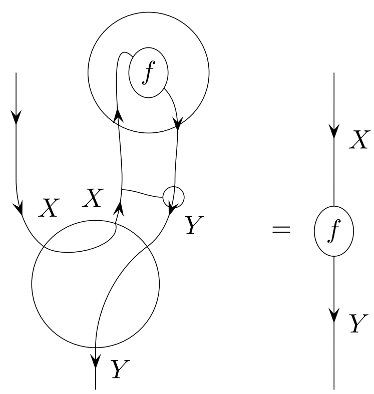
In Set, $\mathrm{ev}_{X,Y}$ takes any function from X to Y and evaluates it at any element of X to give an element of Y. In terms of string diagrams, the evaluation morphism looks like this:



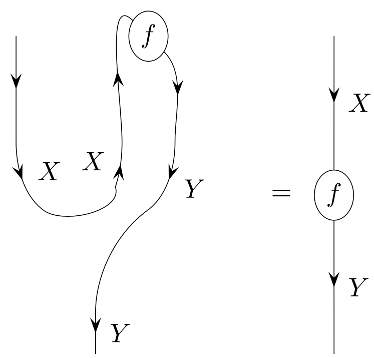
In any closed monoidal category, we can recover a morphism from its name using evaluation. More precisely, this diagram commutes:

$$\begin{tikzcd}
X \otimes I \arrow[d, "1_X \otimes \ulcorner f \urcorner"'] & X \arrow[l, "r^{-1}"'] \arrow[d, "f"] \\
X \otimes (X \multimap Y) \arrow[r, "\mathrm{ev}_{X,Y}"'] & Y
\end{tikzcd}$$

Or, in terms of string diagrams:



We leave the proof of this as an exercise. In general, one must use the naturality of currying. In the special case of a compact monoidal category, there is a nice picture proof! Simply pop the bubbles and remove the clasps:



The result then follows from one of the zig-zag identities. (However, we never proved the zig-zag identities! In fact, the exercise left for the reader here yields one of the zig-zag identities as a special case when $f = 1_X$. The other is similar.)

In our rapid introduction to string diagrams, we have not had time to illustrate how these diagrams are a powerful tool for solving concrete problems. So, here are some starting points for further study:

- Representations of Lie groups play a fundamental role in quantum physics, especially gauge field theory. Every Lie group has a compact symmetric monoidal category of finite-dimensional representations. In his book *Group Theory*, Cvitanovic [34] develops detailed string diagram descriptions of these representation categories for the classical Lie groups $\mathrm{S}U(n)$, $\mathrm{S}O(n)$, $\mathrm{S}U(n)$ and also the more exotic 'exceptional' Lie groups. His book also illustrates how this technology can be used to simplify difficult calculations in gauge field theory.
- Quantum groups are a generalization of groups which show up in 2d and 3d physics. The big difference is that a quantum group has compact *braided* monoidal category of finite-dimensional representation. Kauffman's *Knots and Physics* [55] is an excellent introduction to how quantum groups show up in knot theory and physics; it is packed with string diagrams. For more details on quantum groups and braided monoidal categories, see the book by Kassel [54].
- Kauffman and Lins [56] have written a beautiful string diagram treatment of the category of representations of the simplest quantum group, $SU_q(2)$. They also use it to construct some famous 3-manifold invariants associated to 3d and 4d topological quantum field theories: the Witten–Reshetikhin–Turaev, Turaev–Viro and Crane–Yetter invariants. For generalizations of these theories to other quantum groups, see the more advanced books by Turaev [82] and by Bakalov and Kirillov [13].
- Kock [58] has written a nice introduction to 2d topological quantum field theories which makes heavy use of string diagram methods for studying 2Cob.

- Abramsky, Coecke and collaborators [1, 2, 3, 28, 30, 31] have developed string diagrams for a certain crucial class of symmetric monoidal compact categories as a tool for understanding quantum computation. The easiest introduction is Coecke's 'Kindergarten quantum mechanics' [29].

1.3 Logic

1.3.1 Overview

Proof theory is the branch of logic that studies proofs as mathematical entities in their own right. Modern proof theory studies proofs in many different systems of logic, of which 'classical logic' is just one [46].

In Hilbert's approach to proof there are many axioms and just one rule to deduce new theorems: *modus ponens*, which says that from X and 'X implies Y' we can deduce Y. Most of modern proof theory focuses on another approach, due to Gentzen [44]. In this approach there are few axioms but many inference rules.

A nonexpert might be surprised that proof theorists often focus on systems of logic that are *weaker* than classical logic — systems where it is harder or even impossible to prove things we normally take for granted. These are sometimes called 'substructural logics' [71]. One reason these logics are interesting is that that they allow a fine-grained study of precisely which methods of reasoning are able to prove which results. A deeper reason is that they shed light on the connection between proof theory and category theory.

1.3.2 Proof Theory

In Section 1.2 we described categories with various amounts of extra structure, starting from categories pure and simple, and working our way up to monoidal categories, braided monoidal categories, symmetric monoidal categories, and so on. Our treatment only scratched the surface of an enormously rich taxonomy. Now we shall see that each kind of category with extra structure corresponds to a system of logic with its own inference rules!

In a nutshell, the idea is to think of *propositions* as *objects* in some category, and *proofs* as giving *morphisms*. Suppose X and Y are propositions. Then, we can think of a proof starting from the assumption X and leading to the conclusion Y as giving a morphism $f: X \to Y$. For reasons of convenience, we may want to think of slightly different proofs as giving the same morphism — soon we shall see why. So, morphisms are really equivalence classes of proofs.

Proof theorists write $X \vdash Y$ when, starting from the assumption X, there is a proof leading to the conclusion Y. An inference rule is a way to get new proofs from old. For example, in almost every system of logic, if there is a proof leading from X to Y, and a proof leading from Y to Z, then there is a proof leading from X to Z. Proof theorists write this inference rule as follows:

$$\frac{X \vdash Y \qquad Y \vdash Z}{X \vdash Z}$$

This is called the **cut rule**, since it lets us 'cut out' the intermediate step Y. It should remind us of composition of morphisms in a category: if we have a morphism $f\colon X \to Y$ and a morphism $g\colon Y \to Z$, we get a morphism $gf\colon X \to Z$.

Also, in almost every system of logic there is a proof leading from X to X. We can write this as an inference rule that starts with *nothing* and concludes the existence of a proof of X from X:

$$\frac{}{X \vdash X}$$

This rule should remind us of how every object in category has an identity morphism: for any object X, we automatically get a morphism $1_X\colon X \to X$. Indeed, this rule is sometimes called the **identity rule**.

If we pursue this line of thought, we can take the definition of a closed symmetric monoidal category and extract a collection of inference rules. Each rule is a way to get new morphisms from old in a closed symmetric monoidal category. There are various superficially different but ultimately equivalent ways to list these rules. Here is one:

$$\frac{}{X \vdash X}\ \text{(i)} \qquad\qquad \frac{X \vdash Y \qquad Y \vdash Z}{X \vdash Z}\ (\circ)$$

$$\frac{W \vdash X \qquad Y \vdash Z}{W \otimes Y \vdash X \otimes Z}\ (\otimes) \qquad\qquad \frac{W \vdash (X \otimes Y) \otimes Z}{\overline{W \vdash X \otimes (Y \otimes Z)}}\ \text{(a)}$$

$$\frac{X \vdash I \otimes Y}{\overline{X \vdash Y}}\ \text{(l)} \qquad\qquad \frac{X \vdash Y \otimes I}{\overline{X \vdash Y}}\ \text{(r)}$$

$$\frac{W \vdash X \otimes Y}{\overline{W \vdash Y \otimes X}}\ \text{(b)} \qquad\qquad \frac{X \otimes Y \vdash Z}{\overline{Y \vdash X \multimap Z}}\ \text{(c)}$$

Double lines mean that the inverse rule also holds. We have given each rule a name, written to the right in parentheses. As already explained, rules (i) and ($\circ$) come from the presence of identity morphisms and composition in any category. Rules ($\otimes$), (a), (l), and (r) come from tensoring, the associator, and the left and right unitors in a monoidal category. Rule (b) comes from the braiding in a braided monoidal category, and rule (c) comes from currying in a closed monoidal category.

Now for the big question: *what does all this mean in terms of logic?* These rules describe a small fragment of classical logic. To see this, we should read the connective $\otimes$ as 'and', the connective $\multimap$ as 'implies', and the proposition I as 'true'.

In this interpretation, rule (c) says we can turn a proof leading from the assumption 'Y and X' to the conclusion Z into a proof leading from X to 'Y implies Z'. It also says we can do the reverse. This is true in classical logic, and so are all the other rules. Rules (a) and (b) say that 'and' is associative and commutative. Rule (l) says that any proof leading from the assumption X to the conclusion 'true and Y' can be converted to a proof leading from X to Y, and vice versa. Rule (r) is similar.

What do we do with these rules? We use them to build 'deductions'. Here is an easy example:

$$\dfrac{\dfrac{}{X \multimap Y \vdash X \multimap Y}\,(\mathrm{i})}{X \otimes (X \multimap Y) \vdash Y}\,(\mathrm{c}^{-1})$$

First we use the identity rule, and then the inverse of the currying rule. At the end, we obtain

$$X \otimes (X \multimap Y) \vdash Y.$$

This should remind us of the evaluation morphisms we have in a closed monoidal category:

$$\mathrm{ev}_{X,Y} \colon X \otimes (X \multimap Y) \to Y.$$

In terms of logic, the point is that we can prove Y from X and 'X implies Y'. This fact comes in handy so often that we may wish to abbreviate the above deduction as an extra inference rule — a rule derived from our basic list:

$$\dfrac{}{X \otimes (X \multimap Y) \vdash Y}\,(\mathrm{ev})$$

This rule is none other than **modus ponens**.

In general, a deduction is a tree built from inference rules. Branches arise when we use the $(\circ)$ or $(\otimes)$ rules. Here is an example:

$$\dfrac{\dfrac{\dfrac{}{(A \otimes B) \otimes C \vdash ((A \otimes B) \otimes C}\,(\mathrm{i})}{(A \otimes B) \otimes C \vdash A \otimes (B \otimes C)}\,(\mathrm{a}) \qquad A \otimes (B \otimes C) \vdash D}{(A \otimes B) \otimes C \vdash D}\,(\circ)$$

Again we can abbreviate this deduction as a derived rule. In fact, this rule is reversible:

$$\dfrac{A \otimes (B \otimes C) \vdash D}{(A \otimes B) \otimes C \vdash D}\,(\alpha)$$

For a more substantial example, suppose we want to show

$$(X \multimap Y) \otimes (Y \multimap Z) \vdash X \multimap Z.$$

The deduction leading to this will not even fit on the page unless we use our abbreviations:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{X \otimes (X \multimap Y) \vdash Y}\,(\mathrm{ev}) \qquad \dfrac{}{Y \multimap Z \vdash Y \multimap Z}\,(\mathrm{id})}{(X \otimes (X \multimap Y)) \otimes (Y \multimap Z) \vdash Y \otimes (Y \multimap Z)}\,(\otimes) \qquad \dfrac{}{Y \otimes (Y \multimap Z) \vdash Z}\,(\mathrm{ev})}{\dfrac{(X \otimes (X \multimap Y)) \otimes (Y \multimap Z) \vdash Z}{X \otimes ((X \multimap Y) \otimes (Y \multimap Z)) \vdash Z}\,(\alpha^{-1})}}{(X \multimap Y) \otimes (Y \multimap Z) \vdash X \multimap Z}\,(\mathrm{c})$$

Since each of the rules used in this deduction came from a way to get new morphisms from old in a closed monoidal category (we never used the braiding), it follows that in every such category we have **internal composition** morphisms:

$$\bullet_{X,Y,Z} : (X \multimap Y) \otimes (Y \multimap Z) \to X \multimap Z.$$

These play the same role for the internal hom that ordinary composition

$$\circ \colon \hom(X,Y) \times \hom(Y,Z) \to \hom(X,Z)$$

plays for the ordinary hom.

We can go ahead making further deductions in this fragment of classical logic, but the really interesting thing is what it omits. For starters, it omits the connective 'or' and the proposition 'false'. It also omits two inference rules we normally take for granted — namely, **contraction**:

$$\frac{X \vdash Y}{X \vdash Y \otimes Y} \, (\Delta)$$

and **weakening**:

$$\frac{X \vdash Y}{X \vdash I} \, (!)$$

These are closely related to duplication and deletion in a cartesian category. Omitting these rules is a distinctive feature of 'linear logic' [35, 45]. The word 'linear' should remind us of the category Hilb. As noted in Section 1.2.3, this category with its usual tensor product is noncartesian, so it does not permit duplication and deletion. But, what does omitting these rules mean *in terms of logic?*

Ordinary logic deals with propositions, so we have been thinking of the above system of logic in the same way. Linear logic deals not just with propositions, but also other resources — for example, physical things! Unlike propositions in ordinary logic, we typically can't duplicate or delete these other resources. In classical logic, if we know that a proposition X is true, we can use X as many or as few times as we like when trying to prove some proposition Y. But if we have a cup of milk, we can't use it to make cake and then use it again to make butter. Nor can we make it disappear without a trace: even if we pour it down the drain, it must go somewhere.

In fact, these ideas are familiar in chemistry. Consider the following resources:

$$\begin{aligned} H_2 &= \text{one molecule of hydrogen} \\ O_2 &= \text{one molecule of oxygen} \\ H_2O &= \text{one molecule of water} \end{aligned}$$

We can burn hydrogen, combining one molecule of oxygen with two of hydrogen to obtain two molecules of water. A category theorist might describe this reaction as a morphism:

$$f \colon O_2 \otimes (H_2 \otimes H_2) \to H_2O \otimes H_2O.$$

A linear logician might write:

$$O_2 \otimes (H_2 \otimes H_2) \vdash H_2O \otimes H_2O$$

to indicate the existence of such a morphism. But, we cannot duplicate or delete molecules, so for example

$$H_2 \nvdash H_2 \otimes H_2$$

and

$$H_2 \nvdash I$$

where I is the unit for the tensor product: not iodine, but 'no molecules at all'.

In short, ordinary chemical reactions are morphisms in a symmetric monoidal category where objects are collections of molecules. As chemists normally conceive of it, this category is not closed. So, it obeys an even more limited system of logic than the one we have been discussing, a system lacking the connective $\multimap$. To get a closed category — in fact a compact one — we need to remember one of the great discoveries of 20th-century physics: *antimatter*. This lets us define $Y \multimap Z$ to by 'anti-Y and Z':

$$Y \multimap Z = Y^* \otimes Z$$

Then the currying rule holds:

$$\frac{Y \otimes X \vdash Z}{\overline{X \vdash Y^* \otimes Z}}$$

Most chemists don't think about antimatter very often — but particle physicists do. They don't use the notation of linear logic or category theory, but they know perfectly well that since a neutrino and a neutron can collide and turn into a proton and an electron:

$$\nu \otimes n \vdash p \otimes e,$$

then a neutron can turn into a neutrino together with a proton and an electron:

$$n \vdash \nu^* \otimes (p \otimes e).$$

This is an instance of the currying rule, rule (c).

1.3.3 Logical Theories from Categories

We have sketched how different systems of logic naturally arise from different types of categories. To illustrate this idea, we introduced a system of logic with inference rules coming from ways to get new morphisms from old in a *closed symmetric monoidal category*. One could substitute many other types of categories here, and get other systems of logic.

To make the connection between proof theory and category tighter, we shall now describe a recipe to get a logical theory from any closed symmetric monoidal category. For this, we shall now use $X \vdash Y$ to denote the *set* of proofs — or actually, equivalence classes of proofs — leading from the assumption X to the conclusion Y. This is a change of viewpoint. Previously we would write $X \vdash Y$ when this set of proofs was nonempty; otherwise we would write $X \nvdash Y$. The advantage of treating $X \vdash Y$ as a set is that this set is precisely what a category theorist would call $\mathrm{hom}(X, Y)$: a homset in a category.

If we let $X \vdash Y$ stand for a homset, an inference rule becomes a function from a product of homsets to a single homset. For example, the cut rule

$$\frac{X \vdash Y \qquad Y \vdash Z}{X \vdash Z}\,(\circ)$$

becomes another way of talking about the composition function

$$\circ_{X,Y,Z}\colon \mathrm{hom}(X, Y) \times \mathrm{hom}(Y, Z) \to \mathrm{hom}(X, Z),$$

while the identity rule

$$\frac{}{X \vdash X}\,(\mathrm{i})$$

becomes another way of talking about the function

$$\mathrm{i}_X\colon 1 \to \mathrm{hom}(X, X)$$

that sends the single element of the set 1 to the identity morphism of X. (Note: the set 1 is a *zero-fold* product of homsets.)

Next, if we let inference rules be certain functions from products of homsets to homsets, deductions become more complicated functions of the same sort built from these basic ones. For example, this deduction:

$$\frac{\dfrac{\dfrac{}{X \otimes I \vdash X \otimes I}\,(\mathrm{i})}{X \otimes I \vdash X}\,(\mathrm{r}) \qquad \dfrac{}{Y \vdash Y}\,(\mathrm{i})}{(X \otimes I) \otimes Y \vdash X \otimes Y}\,(\otimes)$$

specifies a function from 1 to $\mathrm{hom}((X \otimes I) \otimes Y, X \otimes Y)$, built from the basic functions indicated by the labels at each step. This deduction:

$$\frac{\dfrac{\dfrac{}{(X \otimes I) \otimes Y \vdash (X \otimes I) \otimes Y}\,(\mathrm{i})}{(X \otimes I) \otimes Y \vdash X \otimes (I \otimes Y)}\,(\mathrm{a}) \qquad \dfrac{\dfrac{\dfrac{}{I \otimes Y \vdash I \otimes Y}\,(\mathrm{i})}{I \otimes Y \vdash Y}\,(\mathrm{r}) \qquad \dfrac{}{X \vdash X}\,(\mathrm{i})}{X \otimes (I \otimes Y) \vdash X \otimes Y}\,(\otimes)}{(X \otimes I) \otimes Y \vdash X \otimes Y}\,(\circ)$$

gives another function from 1 to $\mathrm{hom}((X \otimes I) \otimes Y, X \otimes Y)$.

If we think of deductions as giving functions this way, the question arises when two such functions are equal. In the example just mentioned, the triangle equation in the definition of monoidal category (Definition 6):

$$\begin{tikzcd}
(X \otimes I) \otimes Y \arrow[rr, "a_{X,I,Y}"] \arrow[dr, "r_X \otimes 1_Y"'] & & X \otimes (I \otimes Y) \arrow[dl, "1_X \otimes l_Y"] \\
& X \otimes Y &
\end{tikzcd}$$

says these two functions *are* equal. Indeed, the triangle equation is precisely the statement that these two functions agree! (We leave this as an exercise for the reader.)

So: even though two deductions may look quite different, they may give the same function from a product of homsets to a homset if we demand that these are homsets in a closed symmetric monoidal category. This is why we think of $X \multimap Y$ as a set of *equivalence classes* of proofs, rather than proofs: it is forced on us by our desire to use category theory. We could get around this by using a 2-category with proofs as morphisms and 'equivalences between proofs' as 2-morphisms [73]. This would lead us further to the right in the Periodic Table (Table 1.3). But let us restrain ourselves and make some definitions formalizing what we have done so far.

From now on we shall call the objects $X, Y, \ldots$ 'propositions', even though we have seen they may represent more general resources. Also, purely for the sake of brevity, we use the term 'proof' to mean 'equivalence class of proofs'.

Definition 16. *A* **closed monoidal theory** *consists of the following:*

- *A collection of* **propositions**. *The collection must contain a proposition I, and if X and Y are propositions, then so are $X \otimes Y$ and $X \multimap Y$.*
- *For every pair of propositions X, Y, a set $X \vdash Y$ of* **proofs** *leading from X to Y. If $f \in X \vdash Y$, then we write $f\colon X \to Y$.*
- *Certain functions, written as* **inference rules**:

$$\frac{}{X \vdash X}\ (\mathrm{i}) \qquad \frac{X \vdash Y \qquad Y \vdash Z}{X \vdash Z}\ (\circ)$$

$$\frac{W \vdash X \qquad Y \vdash Z}{W \otimes Y \vdash X \otimes Z}\ (\otimes) \qquad \frac{W \vdash (X \otimes Y) \otimes Z}{\overline{W \vdash X \otimes (Y \otimes Z)}}\ (\mathrm{a})$$

$$\frac{X \vdash I \otimes Y}{\overline{X \vdash Y}}\ (\mathrm{l}) \qquad \frac{X \vdash Y \otimes I}{\overline{X \vdash Y}}\ (\mathrm{r})$$

$$\frac{X \otimes Y \vdash Z}{\overline{Y \vdash X \multimap Z}}\ (\mathrm{c})$$

A double line means that the function is invertible. So, for example, for each triple X, Y, Z we have a function

$$\circ_{X,Y,Z}\colon (X \vdash Y) \times (Y \vdash Z) \ \to\ (X \vdash Z)$$

and a bijection

$$c_{X,Y,Z}\colon (X \otimes Y \vdash Z) \ \to\ (Y \vdash X \multimap Z).$$

- *Certain equations that must be obeyed by the inference rules. The inference rules* ($\circ$) *and* (i) *must obey equations describing associativity and the left and right unit laws. Rule* ($\otimes$) *must obey an equation saying it is a functor. Rules* (a)*,* (l)*,* (r)*, and* (c) *must obey equations saying they are natural transformations. Rules* (a)*,* (l)*,* (r) *and* ($\otimes$) *must also obey the triangle and pentagon equations.*

Definition 17. *A* **closed braided monoidal theory** *is a closed monoidal theory with this additional inference rule:*

$$\frac{W \vdash X \otimes Y}{W \vdash Y \otimes X} \text{(b)}$$

We demand that this rule give a natural transformation satisfying the hexagon equations.

Definition 18. *A* **closed symmetric monoidal theory** *is a closed braided monoidal theory where the rule* (b) *is its own inverse.*

It should be clear that these are just the definitions of closed monoidal, closed braided monoidal and closed symmetric monoidal category written in a different style. The main advantage is that this style makes it easier to recognize examples coming from various systems of logic. Most of these systems include extra features beyond what we have discussed here, though some *subtract* features. Here are a few examples:

- Monoidal theories [68]
- Algebraic theories. [62]
- Multiplicative intuitionistic linear logic. [19]??? [49].
- Intuitionistic linear logic. [19, 20, 21] [23]
- Linear logic. Practical applications: Wadler [84, 85]. Good general overview: [24].

To conclude, let us say precisely what an inference rule is in this setting. We have said it gives a function from a product of homsets to a homset. While true, that is not the last word on the subject. After all, instead of treating the propositions appearing in an inference rule as *fixed*, we can treat them as *variable*. Then an inference rule is really a 'schema' for getting new proofs from old. How do we formalize this idea?

First we must realize that $X \vdash Y$ is not just a set: it is a set *depending in a functorial way* on X and Y. As noted in Definition 13, there is a functor, the 'hom functor'

$$\mathrm{hom}\colon C^{\mathrm{op}} \times C \to \mathrm{Set},$$

sending (X, Y) to the homset $\mathrm{hom}(X, Y) = X \vdash Y$. To look like logicians, let us write this functor as $\vdash$.

Viewed in this light, most of our inference rules are *natural transformations*. For example, rule (a) is a natural transformation between two functors from $C^{\mathrm{op}} \times C^3$ to Set, namely the functors

$$(W, X, Y, Z) \mapsto W \vdash (X \otimes Y) \otimes Z)$$

and

$$(W, X, Y, Z) \mapsto W \vdash X \otimes (Y \otimes Z)).$$

This natural transformation turns any proof

$$f : W \to (X \otimes Y) \otimes Z)$$

into the proof

$$a_{X,Y,Z} f : W \to X \otimes (Y \otimes Z).$$

The fact that this transformation is *natural* means that it changes in a systematic way as we vary W, X, Y and Z. The commuting square in the definition of natural transformation, Definition 3, makes this precise.

Rules (l), (r), (b) and (c) give natural transformations in a very similar way. The $(\otimes)$ rule gives a natural transformation between two functors from $C^{\mathrm{op}} \times C \times C^{\mathrm{op}} \times C$ to Set, namely

$$(W, X, Y, Z) \mapsto (W \vdash X) \ \times \ (Y \vdash Z)$$

and

$$(W, X, Y, Z) \mapsto W \otimes Y \vdash X \otimes Z$$

This natural transformation sends any element $(f, g) \in \hom(W, X) \times \hom(Y, Z)$ to $f \otimes g$.

The identity and cut rules are different: they *do not* give natural transformations, because the top line of these rules has a different number of variables than the bottom line! Rule (i) says that for each $X \in C$ there is a function

$$\mathrm{i}_X : 1 \ \to \ X \vdash X$$

picking out the identity morphism 1_X. What would it mean for this to be natural in X? Rule $(\circ)$ says that for each triple $X, Y, Z \in C$ there is a function

$$\circ : (X \vdash Y) \ \times \ (Y \vdash Z) \ \to \ X \vdash Z.$$

What would it mean for this to be natural in X, Y and Z? The answer to both questions involves a generalization of natural transformations called 'dinatural' transformations [64].

As noted in Definition 3, a natural transformation $\alpha : F \Rightarrow G$ between two functors $F, G : C \to D$ makes certain squares in D commute. If in fact $C = C_1^{\mathrm{op}} \times C_2$, then we actually obtain commuting cubes in D. Namely, the natural transformation α assigns to each object (X_1, X_2) a morphism α_{X_1,X_2} such that for any morphism $(f_1 : Y_1 \to X_1, f_2 : X_2 \to Y_2)$ in C, the cube shown in Figure 1.1 commutes.

If $C_1 = C_2$, we can choose a single object X and a single morphism $f : X \to Y$ and use it in both slots. As shown in Figure 1.2, there are then two paths from one corner of the cube to the antipodal corner that only involve α for repeated arguments: that is, $\alpha_{X,X}$ and $\alpha_{Y,Y}$, but not $\alpha_{X,Y}$ or $\alpha_{Y,X}$. These paths give a commuting hexagon.

This motivates the following:

$$
\begin{tikzcd}
& G(Y_1, X_2) \arrow[rr, "G(1_{Y_1}, f_2)"] \arrow[dd, "G(f_1, 1_{X_2})"] & & G(Y_1, Y_2) \arrow[dd, "G(f_1, 1_{Y_2})"] \\
F(Y_1, X_2) \arrow[ur, "\alpha_{Y_1, X_2}"] \arrow[rr, "F(1_{Y_1}, f_2)"] \arrow[dd, "F(f_1, 1_{X_2})"] & & F(Y_1, Y_2) \arrow[ur, "\alpha_{Y_1, Y_2}"] \arrow[dd, "F(f_1, 1_{Y_2})"] & \\
& G(X_1, X_2) \arrow[rr, "G(1_{X_1}, f_2)"] & & G(X_1, Y_2) \\
F(X_1, X_2) \arrow[ur, "\alpha_{X_1, X_2}"'] \arrow[rr, "F(1_{X_1}, f_2)"] & & F(X_1, Y_2) \arrow[ur, "\alpha_{X_1, Y_2}"'] &
\end{tikzcd}
$$

Fig. 1.1. A natural transformation between functors $F, G\colon C_1^{\mathrm{op}} \times C_2 \to \mathcal{D}$ gives a commuting cube in D for any morphisms $f_i\colon X_i \to Y_i$ in C_i.

Definition 19. *A* **dinatural transformation** $\alpha\colon F \Rightarrow G$ *between functors* $F, G\colon C^{\mathrm{op}} \times C \to D$ *assigns to every object* X *in* C *a morphism* $\alpha_X\colon F(X,X) \to G(X,X)$ *in* D *such that for every morphism* $f\colon X \to Y$ *in* C*, the hexagon in Figure 1.2 commutes.*

In the case of the identity rule, this commuting hexagon says that the identity morphism is a left and right unit for composition: see Figure 1.3. For the cut rule, this commuting hexagon says that composition is associative: see Figure 1.4.

So, in general, the sort of logical theory we are discussing involves:

- A *category* C of propositions and proofs.
- A *functor* $\vdash\colon C^{\mathrm{op}} \times C \to \mathrm{Set}$ sending any pair of propositions to the set of proofs leading from one to the other.
- A set of *dinatural transformations* describing inference rules.

$$
\begin{tikzcd}
& G(Y,X) & & G(Y,Y) \arrow[dd, "G(f,1_Y)"] \\
F(Y,X) \arrow[rr, "F(1_Y,f)"] \arrow[dd, "F(f,1_X)"'] & & F(Y,Y) \arrow[ur, "\alpha_{Y,Y}"] & \\
& G(X,X) \arrow[rr, "G(1_X,f)"] & & G(X,Y) \\
F(X,X) \arrow[ur, "\alpha_{X,X}"'] & & F(X,Y) &
\end{tikzcd}
$$

Fig. 1.2. A natural transformation between functors $F, G \colon C^{\mathrm{op}} \times C \to \mathcal{D}$ gives a commuting cube in D for any morphism $f \colon X \to Y$, and there are two paths around the cube that only involve α for repeated arguments.

$$
\begin{tikzcd}
& & & \begin{array}{c} Y \vdash Y \\ 1_Y \end{array} \arrow[dd, "- \circ f"] \\
\begin{array}{c} 1 \\ \cdot \end{array} \arrow[rr, "1_1"] \arrow[dd, "1_1"'] & & \begin{array}{c} 1 \\ \cdot \end{array} \arrow[ur, "\mathrm{i}_Y"] & \\
& \begin{array}{c} X \vdash X \\ 1_X \end{array} \arrow[rr, "f \circ -"] & & \begin{array}{c} X \vdash Y \\ f \circ 1_X = 1_Y \circ f \end{array} \\
\begin{array}{c} 1 \\ \cdot \end{array} \arrow[ur, "\mathrm{i}_X"'] & & &
\end{tikzcd}
$$

Fig. 1.3. Dinaturality of the (i) rule, where $f \colon X \to Y$. Here $\cdot \in 1$ denotes the one element of the one-element set.

$$
\begin{tikzcd}
 & & \begin{array}{c} X \vdash Z \\ h(fg) \end{array} \arrow[dd, "1"] \\
\begin{array}{c} (X \vdash W) \times (Y \vdash Z) \\ (g,h) \end{array} \arrow[r, "(f\circ -,1)"] \arrow[dd, "(1,-\circ f)"'] & \begin{array}{c} (X \vdash Y) \times (Y \vdash Z) \\ (fg,h) \end{array} \arrow[ur, "\circ"] & \\
 & \begin{array}{c} X \vdash Z \\ (hf)g \end{array} \arrow[r, "1"] & \begin{array}{c} X \vdash Z \\ (hf)g = h(fg) \end{array} \\
\begin{array}{c} (X \vdash W) \times (W \vdash Z) \\ (g,hf) \end{array} \arrow[ur, "\circ"'] & &
\end{tikzcd}
$$

Fig. 1.4. Dinaturality of the cut rule, where $f\colon W \to Y$, $g\colon X \to W$, $h\colon Y \to Z$.

1.4 Computation

NOT YET

1.5 Conclusions

In this paper we began fleshing out the analogies listed in Table 1.1. Table 1.4 summarizes a bit of what we have seen. However, this is still just the tip of the iceberg. To fully exploit the links between physics, topology, logic and computation we need to more thoroughly understand the analogies between them — and also the special distinctive features of each field.

1.6 Appendix

1.6.1 The Category Hilb

NOT YET.

Category Theory	Physics	Topology	Logic	Computation
object X	Hilbert space X	manifold X	proposition X	data type `X`
morphism $f\colon X \to Y$	operator $f\colon X \to Y$	cobordism $f\colon X \to Y$	proof $f\colon X \to Y$	program `f: X -> Y`
tensor product of objects: $X \otimes Y$	Hilbert space of joint system: $X \otimes Y$	disjoint union of manifolds: $X \otimes Y$	conjunction of propositions: $X \otimes Y$	product of data types: `X` $\otimes$ `Y`
tensor product of morphisms: $f \otimes g$	parallel processes: $f \otimes g$	disjoint union of cobordisms: $f \otimes g$	proofs carried out in parallel: $f \otimes g$	programs executing in parallel: `f` $\otimes$ `g`
internal hom: $X \multimap Y$	Hilbert space of 'anti-X and Y': $X^* \otimes Y$	disjoint union of orientation-reversed X and Y: $X^* \otimes Y$	conditional proposition: $X \multimap Y$	function type: `X -> Y`

Table 1.4. The Rosetta Stone (larger version)

1.6.2 The Category nCob

NOT YET.

1.6.3 The Category Tang$_k$

NOT YET.

Acknowledgements

We would like to thank regulars at the n-Category Café, especially the experts on logic and computation, such as Todd Trimble and Robin Houston. We would also like to thank participants of the seminar at UCR where some of this material was first presented: especially David Ellerman, Larry Harper and Tom Payne, but also Derek Wise, who took notes [9]. MS would also like to thank Google for letting him devote 20% of his time to this research.

References

1. S. Abramsky, Abstract scalars, loops, and free traced and strongly compact closed categories, in *Proceedings of CALCO 2005*, Lecture Notes in Computer Science **3629**, Springer, Berlin, 2005, 1–31. Also available at `http://web.comlab.ox.ac.uk/oucl/work/samson.abramsky/calco05.pdf`.
2. S. Abramsky and B. Coecke, A categorical semantics of quantum protocols, available at `arXiv:quant-ph/0402130`.
3. S. Abramsky, and R. Duncan, A categorical quantum logic, to appear in *Mathematical Structures in Computer Science*, 2006. Also available as `arXiv:quant-ph/0512114`.

4. S. Abramsky, E. Haghverdi and P. Scott, Geometry of interaction and linear combinatory algebras, *Math. Struct. Comp. Sci.* **12** (2002), 625–665. Also available as http://citeseer.ist.psu.edu/491623.html.
5. M. F. Atiyah, Topological quantum field theories, *Publ. Math. IHES Paris* **68** (1989), 175–186. M. F. Atiyah, *The Geometry and Physics of Knots*, Cambridge U. Press, Cambridge, 1990.
6. J. Baez, An introduction to spin foam models of quantum gravity and BF theory, in *Geometry and Quantum Physics*, eds. H. Gausterer and H. Grosse, Springer, Berlin, 2000, pp. 25–93. Also available at `arXiv:gr-qc/9905087`.
7. J. Baez, Higher-dimensional algebra and Planck-scale physics, in *Physics Meets Philosophy at the Planck Length*, eds. C. Callender and N. Huggett, Cambridge U. Press, Cambridge, 2001, pp. 177–195. Also available as `arXiv:gr-qc/9902017`.
8. J. Baez, Quantum quandaries: a category-theoretic perspective, in *Structural Foundations of Quantum Gravity*, eds. S. French, D. Rickles and J. Saatsi, Oxford U. Press, Oxford, 2006, pp. 240-265. Also available as `arXiv:quant-ph/0404040`.
9. J. Baez, Classical versus quantum computation. Seminar notes available at http://math.ucr.edu/home/baez/qg-fall2006/ and http://math.ucr.edu/home/baez/qg-winter2007.
10. J. Baez and J. Dolan, Higher-dimensional algebra and topological quantum field theory, *Jour. Math. Phys.* **36** (1995), 6073–6105. Also available at `arXiv:q-alg/9503002`.
11. J. Baez and L. Langford, Higher-dimensional algebra IV: 2-tangles, *Adv. Math.* **180** (2003), 705–764.
12. J. Baez and A. Lauda, A prehistory of n-categorical physics, to appear in proceedings of Deep Beauty: Mathematical Innovation and the Search for an Underlying Intelligibility of the Quantum World, Princeton, October 3, 2007, ed. Hans Halvorson. Also available at `http://math.ucr.edu/home/baez/history.pdf`.
13. B. Bakalov and A. Kirillov, Jr., *Lectures on Tensor Categories and Modular Functors*, American Mathematical Society, Providence, Rhode Island, 2001.
14. H. Barendregt, *The Lambda Calculus, its Syntax and Semantics*, North–Holland, Amsterdam, 1984.
15. M. Barr and C. Wells, *Toposes, Triples and Theories*, Springer Verlag, Berlin, 1983. Revised and corrected version available at `http://www.cwru.edu/artsci/math/wells/pub/ttt.html`.
16. J. S. Bell, On the Einstein-Podolsky-Rosen paradox, *Physics* **1** (1964), 195–200.
17. J. L. Bell, *The Development of Categorical Logic*, available at http://publish.uwo.ca/ jbell/catlogprime.pdf.
18. C. H. Bennett, P. G'acs, M. Li, P. Vit'anyi, and W. Zurek, Information distance, *IEEE Trans. Inform. Theory*, **44** (1998), 1407–1423. Also available at http://citeseer.ist.psu.edu/166286.html
19. N. Benton, G. M. Bierman, V. de Paiva and J. M. E. Hyland, Linear lambda-calculus and categorical models revisited, in *Computer Science Logic (CSL'92), Selected Papers*, Lecture Notes in Computer Science **702**, Springer, Berlin, 1992, pp. 61–84. Also available at http://citeseer.ist.psu.edu/benton92linear.html
20. N. Benton, G. M. Bierman, V. de Paiva and J. M. E. Hyland, A term calculus for intuitionistic linear logic, *Proceedings of the International Conference on Typed Lambda Calculi and Applications, TLCA '93*, Lecture Notes in Computer Science **664**, Springer, Berlin, 1993. Also available at http://citeseer.ist.psu.edu/16052.html.
21. N. Benton, G. Bierman, V. de Paiva and M. Hyland, *Term Assignment for Intuitionistic Linear Logic*, Technical Report 262, University of Cambridge Computer Laboratory, August 1992.
22. C. Benzmüller, From natural deduction to sequent calculus (and back), lecture notes for CALCULEMUS Autumn School, Pisa, September 2002. Available at `http://www.ags.uni-sb.de/~chris/papers/2002-pisa.pdf`

23. G. Bierman, *On Intuitionistic Linear Logic*, PhD Thesis, Cambridge University. Available at `http://research.microsoft.com/~gmb/Papers/thesis.pdf`
24. R. Blute and P. Scott, Category theory for linear logicians, in *Linear Logic in Computer Science*, eds. T. Ehrhard, J.-Y. Girard, P. Ruet, P. Scott, Cambridge U. Press, Cambridge, 2004, pp. 3–64. Also available as http://www.site.uottawa.ca/ phil/papers/catsurv.web.pdf.
25. V. Chari and A. Pressley, *A Guide to Quantum Groups*, Cambridge University Press, Cambridge, 1995.
26. E. Cheng and A. Lauda, *Higher-Dimensional Categories: an Illustrated Guidebook*. Available at `http://www.dpmms.cam.ac.uk/~elgc2/guidebook/`.
27. A. Church, An unsolvable problem of elementary number theory, *Amer. Jour. Math.* **58** (1936), 345–363.
28. B. Coecke, De-linearizing linearity: projective quantum axiomatics from strong compact closure, *Proceedings of the 3rd International Workshop on Quantum Programming Languages (QPL 2005)*, Elsevier, 2007, pp. 49–72. Also available as `arXiv:quant-ph/0506134`.
29. B. Coecke, Kindergarten quantum mechanics, to appear in *Proceedings of QTRF-III*. Also available as `arXiv:quant-ph/0510032`.
30. B. Coecke and E. O. Paquette, POVMs and Naimark's theorem without sums, to appear in *Proceedings of the 4th International Workshop on Quantum Programming Languages*. Also available as `arXiv:quant-ph/0608072`.
31. B. Coecke and D. Pavlovic, Quantum measurements without sums, to appear in *The Mathematics of Quantum Computation and Technology*, eds. Chen, Kauffman and Lomonaco, Taylor and Francis. Also available as `arXiv:quant-ph/0608035`.
32. R. L. Crole, *Categories for Types*, Cambridge U. Press, Cambridge, 1993.
33. H. B. Curry and R. Feys, *Combinatory Logic* Vol. I, North–Holland, Amsterdam, 1958.
34. P. Cvitanovic, *Group Theory*, Princeton U. Press, Princeton, 2003. Available at `http://www.nbi.dk/GroupTheory/`.
35. R. Di Cosimo and D. Miller, Linear logic, *Stanford Encyclopedia of Philosophy*, available at http://plato.stanford.edu/entries/logic-linear/.
36. M. Dorca and A. van Tonder, Quantum computation, categorical semantics and linear logic, available as `arXiv:quant-ph/0312174`.
37. S. Eilenberg and G. M. Kelly, Closed categories, in *Proceedings of the Conference on Categorical Algebra (La Jolla, 1965)*, Springer, Berlin, 1966, pp. 421–562.
38. S. Eilenberg and S. Mac Lane, General theory of natural equivalences, *Trans. Amer. Math. Soc.* **58** (1945), 231–294.
39. A. Einstein, B. Podolsky, and N. Rosen, Can quantum-mechanical description of physical reality be considered complete?, *Phys. Rev.* **47** (1935) 77.
40. E. Fredkin and T. Toffoli. Conservative logic, *Intl. Jour. Th. Phys.* **21** (1982), 219–253.
41. M. Freedman, A. Kitaev and Z. Wang, Topological quantum computation, available as `arXiv:quant-ph/0101025`.
M. Freedman, A. Kitaev and Z. Wang, Simulation of topological field theories by quantum computers, *Comm. Math. Phys.* **227** (2002), 587–603. Also available as `arXiv:quant-ph/0001071`.
M. Freedman, A. Kitaev and Z. Wang, A modular functor which is universal for quantum computation, *Comm. Math. Phys.* **227** (2002), 605–622. Also available as `arXiv:quant-ph/0001108`.
42. P. Freyd and D. Yetter, Braided compact monoidal categories with applications to low dimensional topology, *Adv. Math.***77** (1989), 156–182.
43. E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-oriented Software*, Addison–Wesley, New York, 1994.
44. G. Gentzen, *Collected Papers of Gerhard Gentzen*, ed. M. E. Szabo, North–Holland, Amsterdam, 1969.

45. J.Y. Girard, Linear logic, *Theor. Comp. Sci.* **50** (1987), 1–102. Also available at http://iml.univ-mrs.fr/ girard/linear.pdf.
46. J.Y. Girard, Y. Lafont and P. Taylor, *Proofs and Types*, Cambridge U. Press, Cambridge, 1990. Also available at http://www.monad.me.uk/stable/Proofs%2BTypes.html.
47. R. Goldblatt, *Topoi: the Categorial Analysis of Logic*, North–Holland, New York, 1984. Also available at `http://cdl.library.cornell.edu/cgi-bin/cul.math/docviewer?did=Gold010`.
48. Gottesman, D. and I. L. Chuang, Quantum teleportation is a universal computational primitive, *Nature* **402** (1999), 390–393. Also available as `arXiv:quant-ph/9908010`.
49. M. Hasegawa, Logical predicates for intuitionistic linear type theories, *Proceedings of the 4th International Conference on Typed Lambda Calculi and Applications (TLCA'99)*, ... Also available at http://citeseer.ist.psu.edu/187161.html.
50. W. A. Howard, The formulae-as-types notion of constructions, in *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, eds. J. P. Seldin and J. R. Hindley, Academic Press, New York, 1980, pp. 479–490.
51. A. Joyal and R. Street, The geometry of tensor calculus I, *Adv. Math.* **88** (1991), 55–113.
A. Joyal and R. Street, The geometry of tensor calculus II. Available at
`http://www.math.mq.edu.au/ street/GTCII.pdf`.
52. A. Joyal and R. Street, Braided monoidal categories, *Macquarie Math Reports* 860081 (1986). Available at `http://rutherglen.ics.mq.edu.au/~street/JS86.pdf`.
A. Joyal and R. Street, Braided tensor categories, *Adv. Math.* **102** (1993), 20–78.
53. D. Kaiser, *Drawing Theories Apart: The Dispersion of Feynman Diagrams in Postwar Physics*, U. Chicago Press, Chicago, 2005.
54. C. Kassel, *Quantum Groups*, Springer, Berlin, 1995.
55. L. H. Kauffman, *Knots and Physics*, World Scientific, Singapore, 1991.
56. L. H. Kauffman and S. Lins, *Temperley–Lieb Recoupling Theory and Invariants of 3-Manifolds*, Princeton U. Press, Princeton, 1994.
57. S. Kleene, λ-definability and recursiveness, *Duke Math. Jour.* **2** (1936), 340–353.
58. J. Kock, *Frobenius Algebras and 2D Topological Quantum Field Theories*, London Mathematical Society Student Texts **59**, Cambridge U. Press, Cambridge, 2004.
59. J. Lambek, From λ-calculus to cartesian closed categories, in *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, eds. J. P. Seldin and J. R. Hindley, Academic Press, New York, 1980, pp. 375–402.
60. J. Lambek and P. J. Scott, *Introduction to Higher-order Categorical Logic*, Cambridge U. Press, Cambridge, 1986.
61. P. Landin, A correspondence between ALGOL 60 and Church's lambda-notation, *Comm. ACM***8** (1965), 89–101, 158–165.
62. F. W. Lawvere, *Functorial Semantics of Algebraic Theories*, Ph.D. Dissertation, Columbia University, 1963. Also available at
`http://www.tac.mta.ca/tac/reprints/articles/5/tr5abs.html`.
63. Tom Leinster, A survey of definitions of n-category, *Th. Appl. Cat.* **10** (2002), 1–70. Also available as `arXiv:math/0107188`.
64. S. Mac Lane, Natural associativity and commutativity, *Rice Univ. Stud.* **49** (1963) 28–46.
65. S. Mac Lane, *Categories for the Working Mathematician*, Springer, Berlin, 1998.
66. S. Mac Lane and I. Moerdijk, *Sheaves in Geometry and Logic: a First Introduction to Topos Theory*, Springer Verlag, Berlin, 1992.
67. M. Markl, S. Shnider and J. Stasheff, *Operads in Algebra, Topology and Physics*, American Mathematical Society, Providence, Rhode Island, 2002.
68. L. Mauri, Algebraic theories in monoidal categories, available at
`http://www.math.rutgers.edu/~mauri`

69. C. McLarty, *Elementary Categories, Elementary Toposes*, Clarendon Press, Oxford, 1995.
70. R. Penrose, Applications of negative dimensional tensors, in *Combinatorial Mathematics and its Applications*, ed. D. Welsh. Academic Press, 1971, pp. 221–244.
R. Penrose, Angular momentum: an approach to combinatorial space-time, in *Quantum Theory and Beyond*, ed. T. Bastin. Cambridge U. Press, 1971, pp. 151–180.
R. Penrose, On the nature of quantum geometry, in *Magic Without Magic*, ed. J. Klauder. Freeman, 1972, pp. 333–354.
R. Penrose, Combinatorial quantum theory and quantized directions, in *Advances in Twistor Theory*, eds. L. Hughston and R. Ward. Pitman Advanced Publishing Program, 1979, pp. 301–317.
71. G. Restall, *An Introduction to Substructural Logics*, Routledge, London, 2000.
G. Restall, Substructural logics, *Stanford Encyclopedia of Philosophy*, available at http://plato.stanford.edu/entries/logic-substructural/.
72. P. Scott, Some aspects of categories in computer science, in *Handbook of Algebra*, Vol. 2, ed. M. Hazewinkel, Elsevier, Amsterdam, 2000. Also available at http://www.site.uottawa.ca/ phil/papers/handbook.ps.
73. R. A. G. Seely, Weak adjointness in proof theory, *Applications of Sheaves*, Lectures Notes in Mathematics **753**, Springer, Berlin, 697–701. Also available at http://www.math.mcgill.ca/∼rags/WkAdj/adj.pdf.
74. G. Segal, The definition of a conformal field theory, in *Topology, Geometry and Quantum Field Theory: Proceedings of the 2002 Oxford Symposium in Honour of the 60th Birthday of Graeme Segal,* ed. U. L. Tillmann, Cambridge U. Press, 2004.
75. P. Selinger, Lecture notes on the lambda calculus, available at http://www.mscs.dal.ca/∼selinger/papers/lambdanotes.pdf.
76. P. Selinger, Dagger compact closed categories and completely positive maps, *Proceedings of the 3rd International Workshop on Quantum Programming Languages (QPL 2005)*, Elsevier, 2007, pp. 139–163. Also available at `http://www.mscs.dal.ca/∼selinger/papers.html/#dagger`.
77. S. Sawin, Links, quantum groups and TQFTs, *Bull. Amer. Math. Soc.* **33** (1996), 413-445. Also available as `arXiv:q-alg/9506002`.
78. M. Schönfinkel, Über die Bausteine der mathematischen Logik, *Math. Ann.* **92** (1924), 305–316. Also available as On the building blocks of mathematical logic, trans. S. Bauer-Mengelberg, in *A Source Book in Mathematical Logic, 1879-1931*, ed. J. van Heijenoort, Harvard U. Press, Cambridge, Massachusetts, 1967, pp. 355–366.
79. M.C. Shum, Tortile tensor categories, *Jour. Pure Appl. Alg.* **93** (1994), 57–110.
80. H. Abelson, G. J. Sussman and J. Sussman, *Structure and Interpretation of Computer Programs*, MIT Press, 1996. Available at http://mitpress.mit.edu/sicp/.
81. T. Toffoli, Reversible computing, *Proceedings of the 7th Colloquium on Automata, Languages and Programming,* Lecture Notes in Computer Science **85**, Springer, Berlin, 1980, pp. 632–644
82. V. G. Turaev, *Quantum Invariants of Knots and 3-Manifolds*, de Gruyter, Berlin, 1994.
83. A. van Tonder, A lambda calculus for quantum computation, *SIAM Jour. Comput.* **33** (2004), 1109–1135. Also available as `arXiv:quant-ph/0307150`.
84. P. Wadler. Linear types can change the world, in *Programming Concepts and Methods*, ed. M. Broy and C. Jones, Sea of Galilee, Israel, April 1990. North Holland, Amsterdam, 1990. Also available at http://homepages.inf.ed.ac.uk/wadler/topics/linear-logic.html
85. P. Wadler, Is there a use for linear logic? *Partial Evaluation and Semantics-Based Program Manipulation (PEPM)*, ACM Press, New Haven, Connecticut, June 1991. http://homepages.inf.ed.ac.uk/wadler/topics/linear-logic.html
86. W. K. Wootters and W. H. Zurek, A single quantum cannot be cloned, *Nature* **299** (1982), 802–803.

87. D. N. Yetter, *Functorial Knot Theory: Categories of Tangles, Coherence, Categorical Deformations, and Topological Invariants*, World Scientific, Singapore, 2001.