Functorial boxes in string diagrams

Paul-André Melliès

Functorial boxes in string diagrams

Paul-André Melliès*

Equipe Preuves, Programmes, Systèmes
CNRS — Université Paris 7 Denis Diderot

July 20, 2006

Abstract

String diagrams were introduced by Roger Penrose as a handy notation to manipulate morphisms in a monoidal category. In principle, this graphical notation should encompass the various pictorial systems introduced in proof-theory (like Jean-Yves Girard's proof-nets) and in concurrency theory (like Robin Milner's bigraphs). This is not the case however, at least because string diagrams do not accomodate *boxes* — a key ingredient in these pictorial systems. In this short tutorial, based on our accidental rediscovery of an idea by Robin Cockett and Robert Seely, we explain how string diagrams may be extended with a notion of *functorial box* to depict a functor separating an inside world (its source category) from an outside world (its target category). We expose two elementary applications of the notation: first, we characterize graphically when a faithful balanced monoidal functor $F : \mathbb{C} \longrightarrow \mathbb{D}$ transports a *trace operator* from the category $\mathbb{D}$ to the category $\mathbb{C}$, and we then exploit this to construct well-behaved *fixpoint operators* in cartesian closed categories generated by models of linear logic; second, we explain how the categorical semantics of linear logic induces that the exponential box of proof-nets decomposes as two enshrined functorial boxes.

*Invited paper at the conference Computer Science Logic 2006 in Szeged, Hungary. To appear in the proceedings of the conference. This research was partially supported by the ANR Project INVAL "Invariants algébriques des systèmes informatiques".

Contents

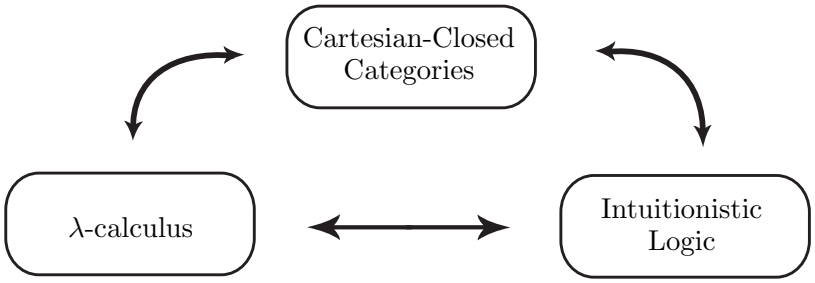
1 Introduction

The origins. Although the process was already initiated in the late 1960s and early 1970s, very few people could have foreseen that Logic and Computer Science would converge so harmoniously and so far in the two areas of *proof theory* and *programming language design*. Today, about fourty years later, the two research fields are so closely connected indeed, that any important discovery in one of them will have, sooner or later, an effect on the other one. The very existence of the conference *Computer Science Logic* bears witness of this important and quite extraordinary matter of fact.

The convergence would not have been as successful without the mediation of *category theory* — which made an excellent matchmaker between the two subjects, by exhibiting the algebraic properties underlying the mathematical models (or denotational semantics) of both proof systems and programming languages. At the end of the 1970s, a few people were already aware that:

- intuitionistic logic as articulated in proof theory,
- the λ-calculus as implemented in programming languages,
- cartesian closed categories as investigated in category theory

are essentially the same object in three different guises — see for instance Jim Lambek and Phil Scott's monograph [34]. The idea circulated widely in the community, and a few years later, in the mid-1980s, the following trilogy of concepts has already become prominent:



A linear world opens. The year 1985 was then a turning point, with the discovery of linear logic by Jean-Yves Girard. This single discovery had the quite extraordinary effect of refurbishing every part of the subject with new tools, new ideas, and new open problems. In particular, each of the three concepts in the trilogy above was reunderstood in a linear fashion. In effect, Jean-Yves Girard [18, 19] introduced simultaneously:

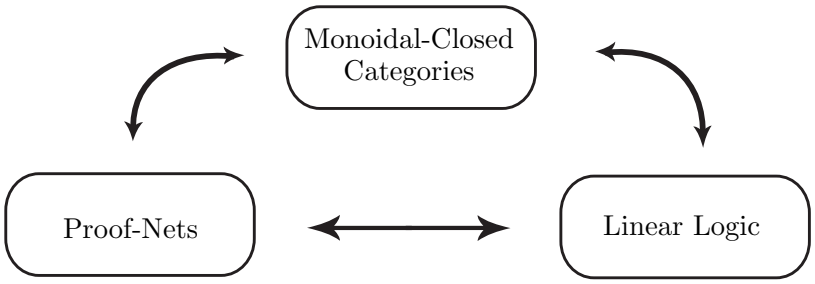
1. a sequent calculus for linear logic, which refines the sequent calculus for intuitionistic logic defined by Gerhard Gentzen in the 1930s — in particular, every derivation rule in intuitionistic logic may be translated as a series of more "atomic" derivation rules in linear logic,

2. a graphical syntax of proofs, called *proof-nets*, which refines the term syntax provided by λ-terms — in particular, every simply-typed λ-term may be translated as a proof-net, in such a way that a β-reduction step on the original λ-term is mirrored as a series of more "atomic" cut-elimination steps in the associated proof-net,

3. a denotational semantics of linear logic, based on *coherence spaces* and *cliques*, which refines the model of dI-domains and stable functions defined by Gérard Berry [7] for the purely functional language PCF — a simply-typed λ-calculus extended with a fixpoint operator, a conditional test on booleans, and the main arithmetic operations. People like Robert Seely [47], Yves Lafont [31] and François Lamarche [33] realized very early that the construction amounts to replacing a cartesian closed category (of dI-domains and stable maps) by a *monoidal*

closed category (of coherence spaces and cliques) equipped with a particular kind of *comonad* to interpret the *exponential modality* (noted !) of linear logic.

From these ideas followed a new and refined "linear" trilogy, which became prominent in our field in the early 1990s:



A puzzle in string diagrams. I started my PhD thesis exactly at that time, but in a quite different topic: Rewriting Theory, with Jean-Jacques Lévy at INRIA Rocquencourt. Although I devoted all my energies to exploring the arcanes of my own subject, this culminating in [38, 39] and the later published [40], I was astonished by the elegance of linear logic, and by the extraordinary perspectives opened by its discovery. Indeed, our emerging field: the semantics of proofs and programs, was suddenly connected to something like mainstream mathematics: linear algebra, representation theory, low-dimensional topology, etc.

My interest was reinforced by a discussion with Yves Lafont, who revealed to me that *multiplicative* proof-nets, and more generally, his own notion of *interaction nets* [32] are specific instances of a graphical notation invented by Roger Penrose [44, 45] to manipulate morphisms in monoidal categories; and that this notation is itself connected to the works by Jean Bénabou on bicategories [4], by Ross Street on computads [49], and by Albert Burroni on polygraphs and higher-dimensional rewriting [13]. Moreover, André Joyal and Ross Street published at about the same time two remarkable papers [26, 27] devoted to *braided monoidal categories* and *string diagrams*. Their elegant work finished to convince me... and I will start this tutorial on string diagrams by giving a very brief and partial account in Section 2 of the two important articles [26, 27].

Now, it is worth recalling that a proof-net is called *multiplicative* when it describes a proof limited to the *multiplicative* fragment of linear logic. Since

multiplicative proof-nets are instances of string diagrams... there remains to understand the "stringy" nature of *general* proof-nets — that is, proof-nets not limited to the multiplicative fragment of linear logic. A serious difficulty arises at this point: general proof-nets admit *exponential boxes* which depict the action of the exponential modality ! on proofs, by encapsulating them. Recall that the purpose of the modality ! is to transform a "linear" proof which must be used exactly once, into a "multiple" proof which may be repeated or discarded during the reasoning. So, by surrounding a proof, the exponential box indicates that this proof may be duplicated or erased. The trouble is that, quite unfortunately, string diagrams do not admit any comparable notion of "box". Consequently, one wishes to extend string diagrams with boxes... But how to proceed?

The lessons of categorical semantics. Interestingly, the solution to this puzzle appears in the categorical semantics of linear logic, in the following way. In the early 1990s, Martin Hyland and Gordon Plotkin initiated together with their students and collaborators Andrew Barber, Nick Benton, Gavin Bierman, Valeria de Paiva, and Andrea Schalk, a meticulous study of the categorical structure defining a model of linear logic [6, 8, 5, 9, 3, 23]. The research was fruitful in many ways. In particular, it disclosed a common pattern behind the various categorical axiomatizations of linear logic. Indeed, every different axiomatization of linear logic generates what appears to be a *symmetric monoidal* adjunction

$$\mathbb{M} \underset{M}{\overset{L}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathbb{L} \tag{1}$$

between a symmetric monoidal closed category $\mathbb{L}$ and a cartesian category $\mathbb{M}$. This important notion was introduced and called a Linear-Non-Linear model by Nick Benton [5, 37]. Here, I will simply call it a *linear adjunction*. The notations L and M are mnemonics for *Linearize* and *Multiply*. Intuitively, a proof of linear logic is interpreted as a morphism in the category $\mathbb{L}$ or in the category $\mathbb{M}$, depending whether it is "linear" or "multiple". Then,

- the functor M transports a "linear" proof into a "multiple" proof, which may be then replicated or discarded inside the cartesian category $\mathbb{M}$,

- conversely, the functor L transports a "multiple" proof into a "linear" proof, which may be then manipulated as a function inside the symmetric monoidal closed category $\mathbb{L}$.

To summarize: there are two "worlds" or "universes of discourse" noted $\mathbb{L}$ and $\mathbb{M}$, each of them implementing a particular policy, and two functors L and M designed to transport proofs from one world to the other.

An early illustration. Interestingly, this pattern traces back to the very origin of linear logic: coherence spaces. Indeed, Max Kelly noticed a long time ago [29, 24] that what one calls "symmetric monoidal adjunction" in (1) is simply an adjunction $L \dashv M$ in the usual sense, in which one requires moreover that the left adjoint functor L transports the cartesian structure of $\mathbb{M}$ to the symmetric monoidal structure of $\mathbb{L}$. The detailed proof of this fact appears for instance in my recent survey on the categorical semantics of linear logic [41]. Such a structure preserving functor L is called *strong monoidal* in the litterature — the precise definition is recalled in Section 4.

Now, the practiced reader will recognize that the linear adjunction (1) describes precisely how the category $\mathbb{M}$ of dI-domains and stable functions is related to the category $\mathbb{L}$ of coherence spaces and cliques. Recall indeed that a coherence space is simply a reflexive graph, and that the functor L transforms every dI-domain D into a coherence space $L(D)$ whose nodes are the compact elements of D, and in which two nodes $x \in D$ and $y \in D$ are connected by an edge (that is, are coherent) precisely when there exists an element $z \in D$ such that $x \leq z \geq y$. Since a compact element in the dI-domain $D \times E$ is the same thing as a pair of compact elements in D and E, and since the tensor product of coherence spaces is the same thing as the usual product of graphs, the equality follows:

$$L(D \times E) \quad = \quad L(D) \otimes L(E).$$

Although one should check carefully the conditions of Section 4, it is quite immediate that the functor L is strict monoidal — hence strong monoidal. At this point, there only remains to define a right adjoint functor M to the functor L in the way exposed in [18, 19, 1] in order to find oneself in the situation of a linear adjunction (1).

The exponential modality decomposed. Although the pattern of linear adjunction (1) looks familiar from a semantic point of view, it appears to

be quite unexpected from the point of view of proof-nets — because the exponential modality ! is not a primitive anymore: it is deduced instead as the *comonad*

$$! \quad = \quad L \circ M \tag{2}$$

generated by the linear adjunction (1) in the category $\mathbb{L}$. In other words, the exponential modality ! factors into a pair of more atomic modalities L and M. Nick Benton [5] mirrors this semantic decomposition into a logic and a term language, which he calls Linear-Non-Linear logic. The decomposition may be transposed instead into the pictorial language of proof-nets: it tells then that the exponential box should decompose into a pair of "boxes" interpreting the two modalities L and M. An important methological point should be raised: this pictorial decomposition of the box ! will have to follow the principles of string diagrams, and be nothing more (and nothing less) than a handy graphical notation for the categorical equality (2).

Functorial boxes. Now, recall that the two modalities L and M in the linear adjunction (1) are *monoidal functors* between the monoidal categories $\mathbb{L}$ and $\mathbb{M}$ — where the monoidal structure of $\mathbb{M}$ is provided by its cartesian structure. Hence: monoidal functors are precisely what one should try to depict as "boxes" in string diagrams. The task of Sections 3 and 4 is precisely to explain how monoidal functors are depicted as *functorial boxes* in string diagrams — and what kind of box depicts a lax, a colax or a strong monoidal functor. I rediscover in this way, ten years later, an idea published by Robin Cockett and Richard Seely [15] in their work on linearly distributive categories and functors. See also the related article written in collaboration with Rick Blute [11]. Obviously, all the credit for the idea should go to these authors. On the other hand, I find appropriate to promote here this graphical notation which remained a bit confidential; and to illustrate how this handy notation for monoidal functors may be applied in other contexts than linear logic or linearly distributive categories.

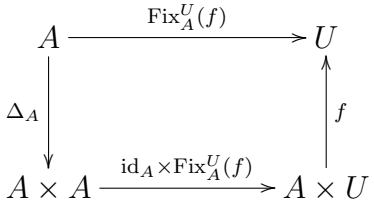
So, I will discuss briefly in Section 8 how the exponential box ! of linear logic decomposes into a functorial box M enshrined inside a functorial box L. Categorical semantics indicates that the functor L is strong monoidal whereas the functor M is lax monoidal — see Section 4 for a definition. Consequently, the two functorial boxes are of a different nature. One benefit of using string diagrams instead of proof-nets is that the graphical notation mirrors *exactly* the underlying categorical semantics. In particular, I will illustrate how the

typical cut-elimination steps in proof-nets are themselves decomposed into sequences of more atomic rewrite steps in string diagrams. Each of these rewrite steps depicts a step in the proof of *soundness* of the categorical semantics of linear logic implemented by Linear-Non-Linear models.

Trace operators in linear logic. In order to interpret recursive calls in a programming language like PCF, one needs a cartesian closed category equipped with a *fixpoint operator*. Recall that a parametric fixpoint operator Fix in a cartesian category $\mathbb{C}$ is a family of functions

$$\mathrm{Fix}_A^U \ : \ \mathbb{C}(A \times U, U) \ \longrightarrow \ \mathbb{C}(A, U)$$

making the diagram below commute



for every morphism $f : A \times U \longrightarrow U$. The diagram expresses that Fix_A^U is a parametric fixpoint of the morphism f. A fixpoint operator should also satisfy a series of naturality properties described in Theorem 3.1 of [20].

A few years ago, Martin Hyland and Masahito Hasegawa [20] have pointed out independently that the notion of fixpoint operator is closely related to the notion of *trace* introduced by André Joyal, Ross Street and Dominic Verity [28] in the context of balanced monoidal categories — a mild refinement of braided monoidal categories, see Section 6 for a definition of trace. More precisely, Martin Hyland and Masahito Hasegawa show that a trace in a cartesian category $\mathbb{C}$ is the same thing as a particularly well-behaved notion of parametric fixpoint, see [20].

Now, it appears that in many existing models of linear logic, formulated here as a linear adjunction (1), the symmetric monoidal closed category $\mathbb{L}$ has a trace. This happens typically when the category $\mathbb{L}$ is *compact-closed* (or more generally, *tortile*) like the category *Rel* of sets and relations (with the usual product of sets as tensor product) or variants recently studied by Nicolas Tabareau [51] of the category of Conway games introduced by André Joyal at the end of the 1970s [25]. An interesting question thus is to

understand when a trace in the category $\mathbb{L}$ may be transported to a trace, and thus a fixpoint operator, in the cartesian category $\mathbb{M}$.

A nice example, suggested to me by Masahito Hasegawa, shows that this is not possible in general. Consider the powerset monad T on the usual category *Set* of sets and functions: the monad associates to every set X the set TX of its subsets. The monad T induces an adjunction

$$\mathit{Rel} \underset{M}{\overset{L}{\rightleftarrows}} \mathit{Set} \qquad (3)$$

between the category *Set* and its kleisli category Set_T — which is isomorphic to the category *Rel* of sets and relations. The powerset monad T being commutative, or equivalently, symmetric monoidal (in the lax sense), the adjunction (3) is symmetric monoidal, see [30]. In particular, the kleisli category Set_T inherits its monoidal structure from the cartesian structure of the category *Set*; and the functor L which sends the category *Set* to the subcategory of functions in *Rel*, is strict monoidal. So, the adjunction (3) is linear, and defines a model of linear logic, in which the category *Set* is moreover isomorphic to the category $\mathit{Rel}_!$ of coalgebras generated by the powerset comonad $! = L \circ M$. Now, the category $\mathbb{L} = \mathit{Rel}$ is compact-closed, and thus has a trace. However, there is no fixpoint operator, and thus no trace, in the cartesian category $\mathbb{M} = \mathit{Set}$.

At this point, it is worth noticing that the functor L is *faithful* in the typical models of linear logic, because the category $\mathbb{M}$ is either equivalent to a subcategory of commutative comonoids in $\mathbb{L}$, or equivalent to a subcategory of coalgebras of the comonad $! = L \circ M$ — in particular, $\mathbb{M}$ is equivalent to the category of free coalgebras when it is the co-kleisli category associated to the comonad. Another equivalent statement is the following one: every component of the unit η of the monad $M \circ L$ is a monomorphism. This observation motivates to characterize in Section 7 when a *faithful* balanced monoidal functor

$$\mathbb{C} \xrightarrow{\quad F \quad} \mathbb{D} \qquad (4)$$

between balanced monoidal categories transports a trace in the target category $\mathbb{D}$ to a trace in the source category $\mathbb{C}$. The proof of this result is perfectly elementary, and offers a nice opportunity to demonstrate how string diagrams and functorial boxes may be manipulated in order to produce purely diagrammatic proofs. Of course, the result specializes then to the strong monoidal

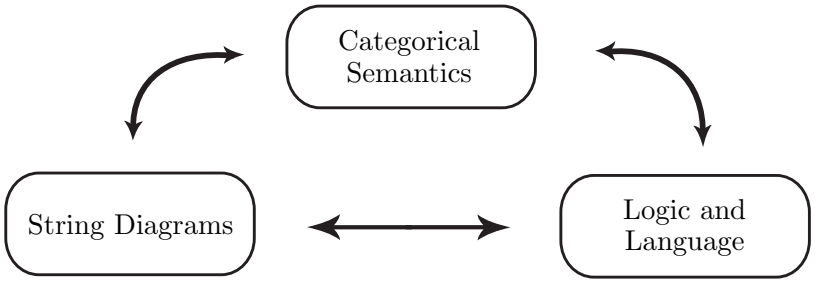
functor L involved in a typical model of linear logic. This enables to transport a trace in the category $\mathbb{L}$ to a well-behaved parametric fixpoint operator in the category $\mathbb{M}$ in several models of interest — including the relational model of linear logic, and the categories of Conway games mentioned earlier.

String diagrams in computer science and logic: a few perspectives. My ambition in writing this elementary tutorial is to demonstrate in a few pictures that categorical semantics is *also* of a diagrammatic nature. Proof-nets were invented by a genial mind, but they remain an ad'hoc and slightly autarchic artefact of proof-theory. On the other hand, string diagrams flourished in the middle of algebra. Categorical semantics is precisely here to connect the two subjects, with benefits on both sides: in logic and computer science, as well as in categorical algebra.

Obviously, much work remains to be done in the area. In many respects, the three concepts appearing in the first trilogy (intuitionistic logic, λ-calculus, cartesian closed categories) were more tightly connected in the mid-1980s than the three concepts appearing in the second trilogy (linear logic, proof-nets, monoidal closed categories) are connected today. From that point of view, the article published recently by Maria Emilia Maietti, Paola Maneggia, Valeria de Paiva, Eike Ritter [37] is extremely clarifying, because it establishes that the Linear-Non-Linear term language introduced by Nick Benton [5] is the *internal language* of the category of linear adjunctions (1). The idea of reformulating this result using string diagrams (extended with functorial boxes) instead of a term language sustains implicitly the discussion in Section 8. Another important work to mention in the area was published by Rick Blute, Robin Cockett, Robert Seely and Todd Trimble [12] about coherence in linearly distributive categories. In particular, the article describes the free linearly distributive category (as well as the free $*$-autonomous category) over a given category $\mathbb{C}$, using equations on a variant of Jean-Yves Girard's multiplicative proof-nets.

I am confident that a broader picture will emerge from the ongoing work at the interface of linear logic and categorical algebra. In the near future, we will certainly find natural to extract a language or a logic as the *internal language* of a particular categorical pattern, similar to the linear adjunction (1) and possibly formulated as a 2-dimensional version of a Lawvere theory [35, 49, 13, 10, 46, 48]. The resulting languages would be equally expressed with string diagrams, for handy manipulation, or with terms, for

easy implementation. The resulting trilogy of concepts:



would be broader in scope and more tightly connected than the current one. It would also integrate the algebraic and pictorial systems formulated for concurrency theory, like Robin Milner's bigraphs [42]. The existing categorical semantics of action calculi [22, 43, 2] indicate for instance a close relationship with the models of linear logic based on linear adjunctions (1) and with the key notion of *fibred* functor between *fibred* categories.

I should conclude this introduction by observing that functorial boxes in string diagrams offer a handy 2-dimensional notation for what could be depicted alternatively using Ross Street's 3-dimensional surface diagrams [50]. Surface diagrams are more perspicuous in several ways: for instance, a functor is depicted as a string — instead of a box. On the other hand, the two notations are not intrinsically different: the practiced reader will easily translate the string diagrams appearing in this tutorial into surface diagrams — in which strings are replaced by ribbons, in order to accomodate the twists of balanced monoidal categories. In that respect, this tutorial should be also an incentive to carry on in the diagrammatic path, and to depict proofs as surface diagrams. The resulting 3-dimensional notation, added to the observation [16] that a $*$-autonomous category is essentially the same thing as a Frobenius algebra in the autonomous category of small categories and "profunctors" or "distributors" — offers a revitalizing point of view on linear logic, which remains largely unexplored today.

Acknowledgments. — This short tutorial was stimulated and inspired by discussions with Martin Hyland and Masahito Hasegawa on trace operators in models of linear logic, held during the summer 2005 and onwards. I thank them for their precious insights and advice. I am also grateful to Nicolas Tabareau for motivating this work by his ongoing study of trace operators

in game semantics. Finally, I thank the Research Institute for Mathematical Science (RIMS) in Kyoto, for hosting me when I wrote this tutorial.

2 String diagrams

In two remarkable papers, André Joyal and Ross Street introduce the notion of *balanced monoidal category* [26] and develop a graphical notation, based on *string diagrams*, to denote morphisms in these categories [27]. Note that from a purely topological point of view, these string diagrams are embedded in the 3-dimensional space. The main task of the second paper [27] is precisely to justify the topological notation, by showing that any two string diagrams equal modulo continuous deformation denote the same morphism in a balanced monoidal category. The interested reader will find the argument in [27].

Recall that a monoidal category [36] is a category $\mathbb{C}$ equipped with a functor

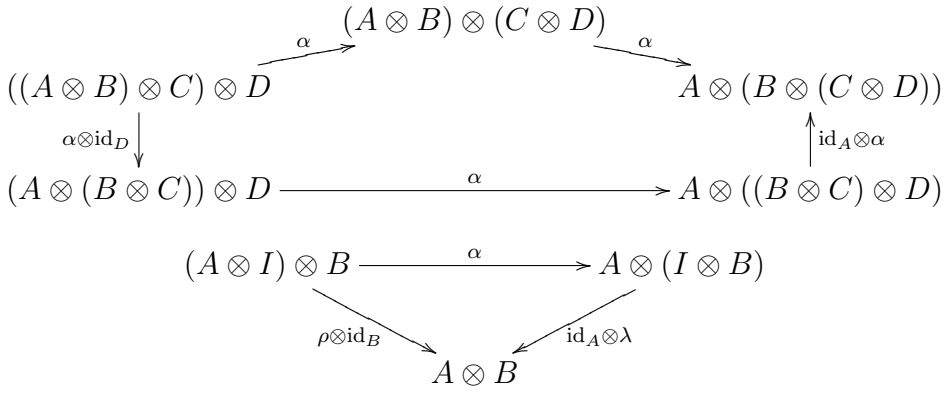
$$\otimes \ : \ \mathbb{C} \times \mathbb{C} \ \longrightarrow \ \mathbb{C}$$

called the *tensor product*, and an object I called the *unit object*; as well as three natural isomorphisms

$$\alpha_{A,B,C} : (A \otimes B) \otimes C \longrightarrow A \otimes (B \otimes C)$$

$$\lambda_A : I \otimes A \longrightarrow A, \qquad \rho_A : A \otimes I \longrightarrow A$$

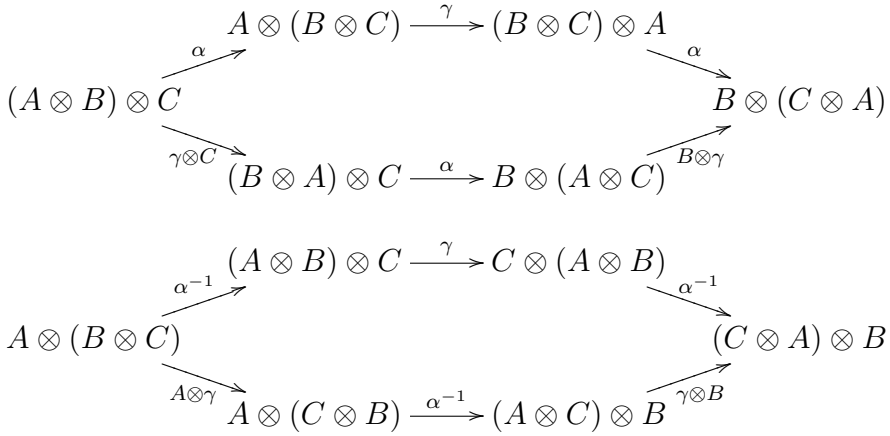
called the *associativity*, the *left* and the *right unit constraints* respectively; such that, for all objects A, B, C and D of the category, the following two diagrams called *MacLane's associativity pentagon* and *triangle for unit*, commute:



A *braiding* is a natural isomorphism

$$\gamma_{A,B} : \ A \otimes B \longrightarrow B \otimes A$$

such that, for all objects A, B and C of the category, the two hexagonal diagrams below commute:



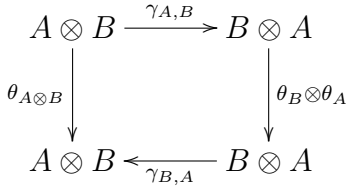
Finally, a *twist* is a natural isomorphism

$$\theta_A : \ A \longrightarrow A$$

such that

$$\theta_I \ = \ \mathrm{id}_I$$

and, for all objects A and B of the category, the diagram below commutes:



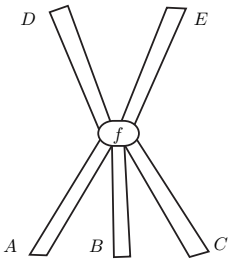
Definition 1 *A balanced monoidal category is a monoidal category equipped with a braiding and a twist.*

Note that a symmetric monoidal category is a balanced category in which, for all objects A and B of the category, the morphism

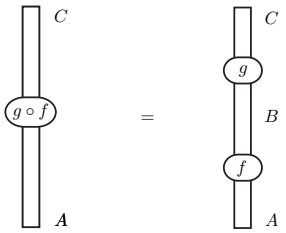
$$A \otimes B \xrightarrow{\gamma_{A,B}} B \otimes A \xrightarrow{\gamma_{B,A}} A \otimes B$$

is equal to the identity morphism $\mathrm{id}_{A\otimes B}$; and the twist morphism θ_A coincides with the identity morphism id_A.

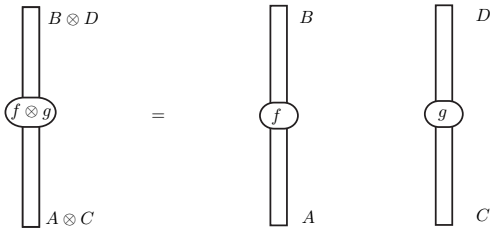
From now on, we suppose for legibility that our balanced monoidal category is *strict*: this means that, for all objects A, B and C of the category, the component $\alpha_{A,B,C}$, λ_A and ρ_A of the the associativity and unit isomorphisms, are identity morphisms. We follow the conventions used in [27] and thus depict a morphism $f : A \otimes B \otimes C \longrightarrow D \otimes E$ in string diagrams as:



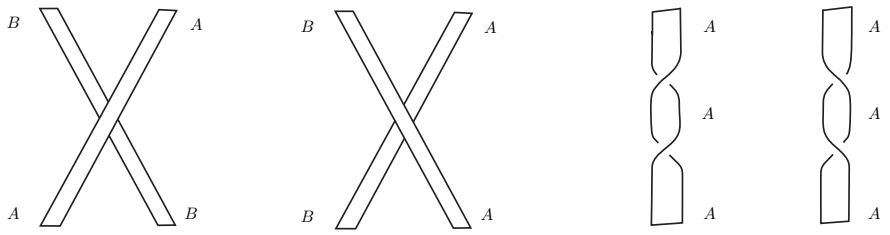
We depict the composite $g \circ f : A \longrightarrow C$ of two morphisms $f : A \longrightarrow B$ and $g : B \longrightarrow C$ as:



and the tensor product $f \otimes g : A \otimes C \longrightarrow B \otimes D$ of two morphisms $f : A \longrightarrow B$ and $g : C \longrightarrow D$ as:



Hence, composition and tensor product are depicted as *vertical* and *horizontal* composition in string diagrams, respectively. Then, the braiding $\gamma_{A,B}$ and its inverse $\gamma^{-1}_{A,B}$, the twist θ_A and its inverse θ^{-1}_A are depicted respectively as:



Note that the third dimension of string diagrams enables to depict the braidings, and that drawing ribbons (instead of strings) is convenient to depict the twists.

3 Functors in string diagrams

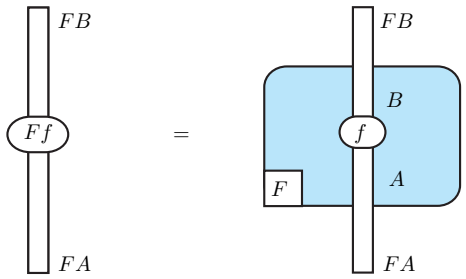
Here, we recall the graphical notation introduced by Robin Cockett and Robert Seely [15] in order to depict a usual functor

$$F \ : \ \mathbb{C} \ \longrightarrow \ \mathbb{D}$$

between balanced monoidal categories. The functor applied to a morphism

$$f \ : \ A \ \longrightarrow \ B$$

of the category $\mathbb{C}$ is represented as a *box* tagged by the label F, and drawn around the morphism f in the following way:



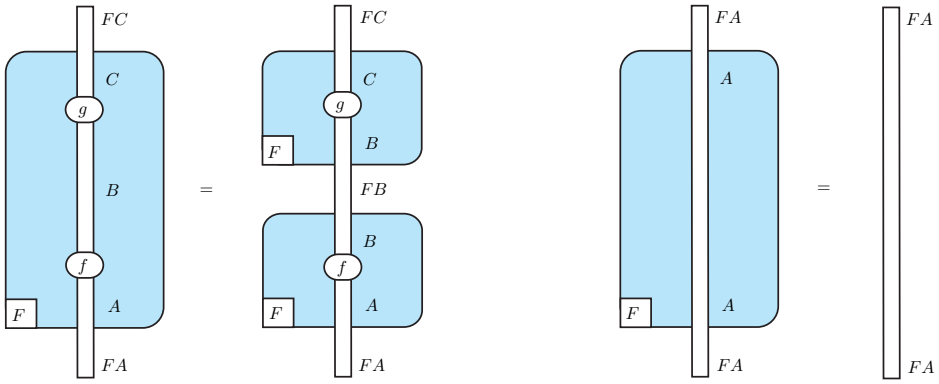
Like any box, the functorial box F is designed to separate an inside world from an outside world: in that case, the inside world is the source category $\mathbb{C}$ and the outside world is the target category $\mathbb{D}$. This explains why a string typed FA outside the box (thus, in the category $\mathbb{D}$) becomes a string typed A (thus, in the category $\mathbb{C}$) when it crosses the frontier and enters the box; and

that a string typed B inside the box (in the category $\mathbb{C}$) becomes a string typed FB (in the category $\mathbb{D}$) when it crosses the frontier and leaves the box.

Given a pair of morphisms $f : A \longrightarrow B$ and $g : B \longrightarrow C$, one depicts the two functorial equalities

$$F(g \circ f) \;=\; Fg \circ Ff \qquad\qquad F(\mathrm{id}_A) \;=\; \mathrm{id}_{FA}$$

in the following way:



Note that exactly one string enters and exits each functorial box F.

4 Monoidal functors in string diagrams

In this section, we recall how the graphical notation for functors introduced in the previous section specializes to monoidal functors, see [15] again. It will appear that a monoidal functor (in the lax sense) implements a particular kind of functorial box in which several strings (possibly none) may enter simultaneously, and from which exactly one string exits. Recall that a lax monoidal functor

$$(F, m) \;:\; \mathbb{C} \;\longrightarrow\; \mathbb{D}$$

between two monoidal categories is a functor F equipped with a morphism

$$m_{[-]} \;:\; I \longrightarrow FI$$

and a natural transformation

$$m_{[A,B]} \;:\; FA \otimes FB \;\longrightarrow\; F(A \otimes B)$$

such that, for all objects A, B and C, the three "coherence" diagrams below commute:

$$\begin{array}{ccc}
(FA \otimes FB) \otimes FC & \xrightarrow{\alpha} & FA \otimes (FB \otimes FC) \\
{\scriptstyle m\otimes FC}\downarrow & & \downarrow{\scriptstyle FA\otimes m} \\
F(A \otimes B) \otimes FC & & FA \otimes F(B \otimes C) \\
{\scriptstyle m}\downarrow & & \downarrow{\scriptstyle m} \\
F((A \otimes B) \otimes C) & \xrightarrow{F\alpha} & F(A \otimes (B \otimes C)
\end{array}$$

$$\begin{array}{ccc}
FA \otimes I & \xrightarrow{\rho} & FA \\
{\scriptstyle FA\otimes m}\downarrow & & \uparrow{\scriptstyle F\rho} \\
FA \otimes FI & \xrightarrow{m} & F(A \otimes I)
\end{array}
\qquad
\begin{array}{ccc}
I \otimes FB & \xrightarrow{\lambda} & FB \\
{\scriptstyle m\otimes FB}\downarrow & & \uparrow{\scriptstyle F\lambda} \\
FI \otimes FB & \xrightarrow{m} & F(I \otimes B)
\end{array}$$

The notion of colax monoidal functor (F, n) is defined in just the same way, except that the coercion morphisms n go in the other direction:

$$n_{[-]} \;:\; FI \;\longrightarrow\; I \qquad\qquad n_{[A,B]} \;:\; F(A \otimes B) \;\longrightarrow\; FA \otimes FB$$

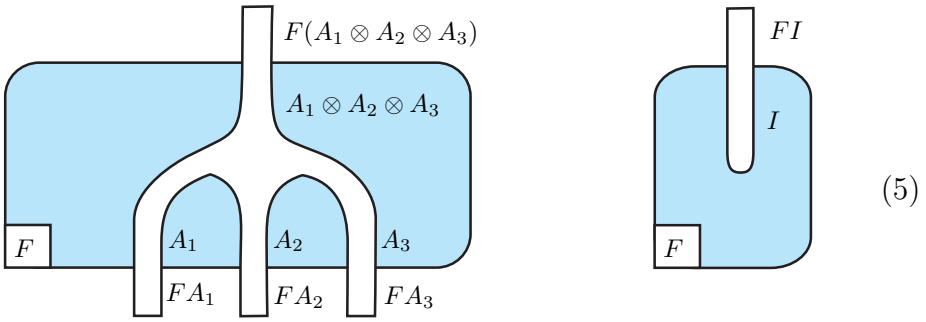
A strong monoidal functor is a lax monoidal functor (F, m) in which the coercion maps m are all isomorphisms; equivalently, it is a colax monoidal functor (F, n) in which the coercion maps n are all isomorphisms.

Now, let us explain how to depict monoidal functors in string diagrams. We will suppose for legibility that the two monoidal categories $\mathbb{C}$ and $\mathbb{D}$ are strict. Given k objects in the category $\mathbb{C}$, there may be several ways to construct a morphism

$$m_{[A_1,\cdots,A_k]} \;:\; FA_1 \otimes \cdots \otimes FA_k \;\longrightarrow\; F(A_1 \otimes \cdots \otimes A_k)$$

by applying a series of structural morphisms m. Then, the definition of a lax monoidal functor, and more specifically the coherence diagrams recalled above, ensure that these various ways define the same morphism $m_{[A_1,\cdots,A_k]}$ in the end. This morphism is depicted in string diagrams as a box F in which k strings labelled $A_1, \cdots, A_k$ enter simultaneously, join together into a unique string labelled $A_1 \otimes \cdots \otimes A_k$, which then exits the box. For instance, the two

structural morphisms $m_{[A_1,A_2,A_3]}$ and $m_{[-]}$ are depicted as follows:



(5)

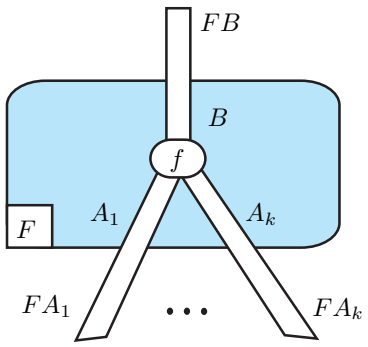
More generally, given a morphism

$$f : A_1 \otimes \cdots \otimes A_k \longrightarrow B$$

in the source category $\mathbb{C}$, one depicts the morphism

$$F(f) \circ m_{[A_1,\cdots,A_k]} \ : \ FA_1 \otimes \cdots \otimes FA_k \ \longrightarrow \ F(A_1 \otimes \cdots \otimes A_k) \ \longrightarrow \ FB$$

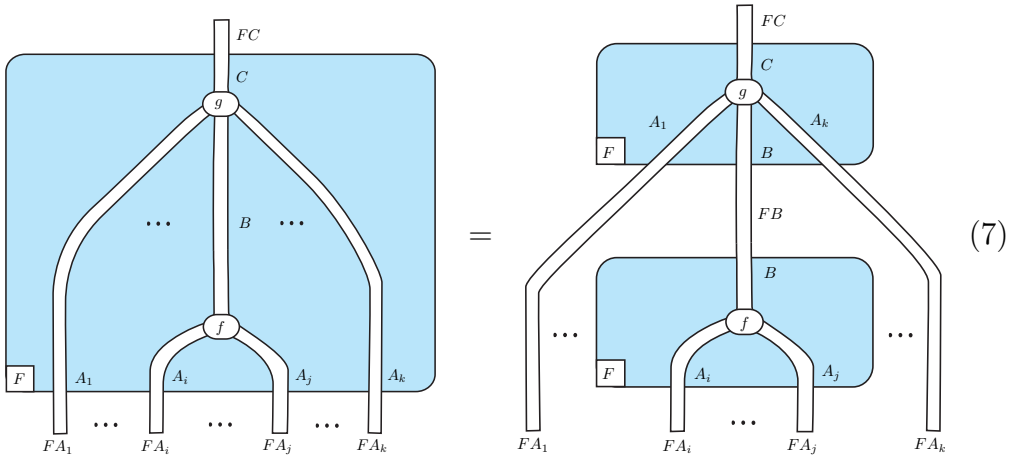
obtained by precomposing the image $F(f)$ with the coercion map $m_{[A_1,\cdots,A_k]}$ in the target category $\mathbb{D}$, as the functorial box below, with k inputs and exactly one output:



(6)

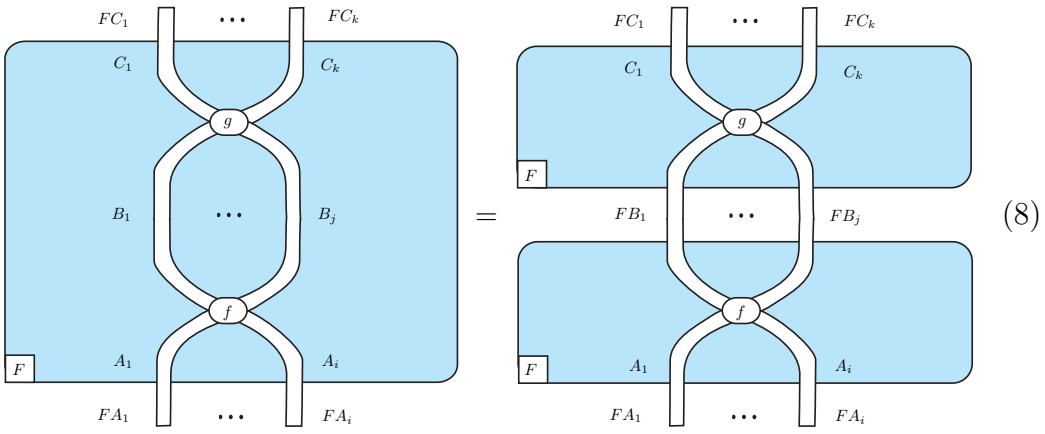
Remark. The definition of lax monoidal functor would permit a more general and *delayed* form of fusion between boxes (think of surface diagrams [50]). Here, we limit ourselves to the specific pattern (5) in which a series of k boxes F, each one encapsulating a unique string labelled A_i, for $1 \leq i \leq k$, join together *simultaneously* in a box F encapsulating a unique string labelled $A_1 \otimes \cdots \otimes A_k$. This specific pattern generates boxes of the shape (6) which are easy to understand and to manipulate, and sufficient to the purpose of this tutorial.

The coherence properties required by the definition of a monoidal functor ensure that we may safely "merge" two monoidal boxes in a string diagram:



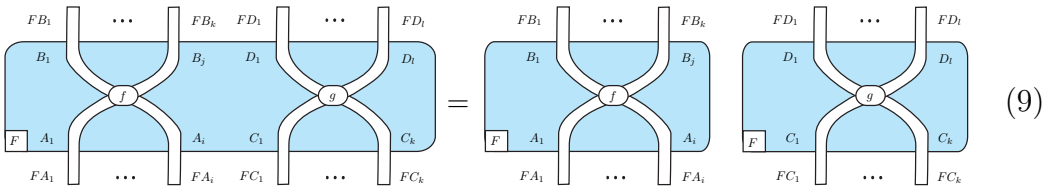
(7)

Note that a colax monoidal functor may be depicated in a similar fashion, as a functorial box in which exactly one string enters, and several strings (possibly none) exit. Now, a strong monoidal functor is at the same time a lax monoidal functor (F, m) and a colax monoidal functor (F, n). It is thus depicted as a functorial box in which several strings may enter, and several strings may exit. Besides, the coercion maps m are inverse to the coercion maps n. Two diagrammatic equalities follow, which enable to split a "strong monoidal" box horizontally:



(8)

as well as vertically:



(9)

These equalities will be illustrated in the series of diagrammatic manipulations exposed in Sections 7 and 8.

5 Natural transformations in string diagrams

Although we do not use *natural transformations* very much in the two elementary exercises exposed in Sections 7 and 8, we find useful to explain briefly how they interact pictorially with functorial boxes. Recall that a natural transformation

$$\theta \quad : \quad F \longrightarrow G \quad : \quad \mathbb{C} \longrightarrow \mathbb{D}$$

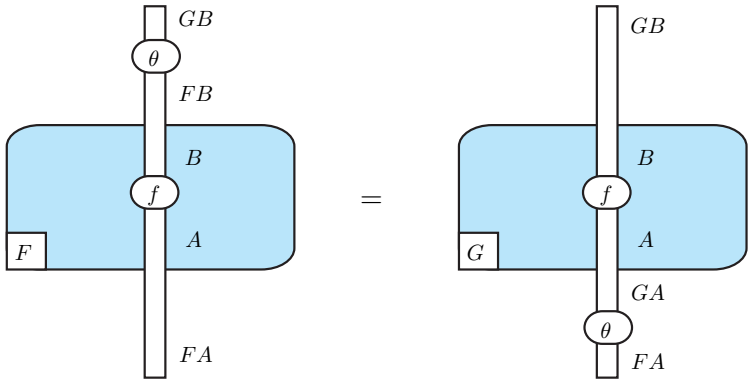
between two functors F and G from the category $\mathbb{C}$ to the category $\mathbb{D}$ is defined as a family of morphisms

$$\theta_A \quad : \quad FA \longrightarrow GA$$

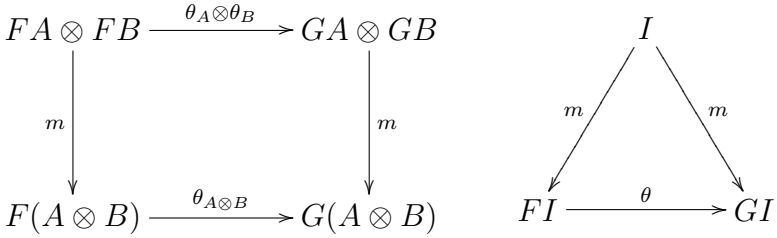
indexed by the objects of the category $\mathbb{C}$, satisfying that the diagram

$$\begin{array}{ccc} FA & \xrightarrow{\theta_A} & GA \\ {\scriptstyle Ff}\downarrow & & \downarrow{\scriptstyle Gf} \\ FB & \xrightarrow{\theta_B} & GB \end{array}$$

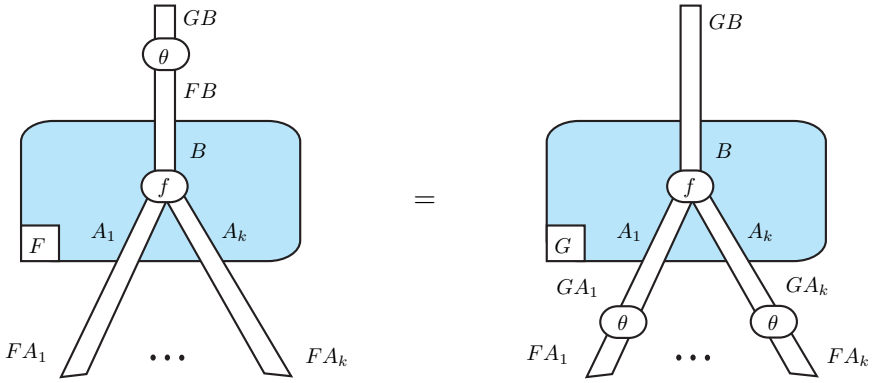
commutes, for every morphism $f : A \longrightarrow B$ in the category $\mathbb{C}$. This is depicted as the diagrammatic equality:



Suppose that the two categories $\mathbb{C}$ and $\mathbb{D}$ are monoidal, and that the two functors F and G are lax monoidal, with structural coercions noted m. By definition, the natural transformation θ is *monoidal* when the diagrams



commute, for all objects A and B of the category $\mathbb{C}$. These coherence diagrams ensure the diagrammatic equality:



in which the natural transformation θ "transforms" the lax monoidal box F into the lax monoidal box G, and "replicates" as one natural transformation θ on each of the k strings $A_1, \ldots, A_k$ entering the lax monoidal boxes F

and G. The notion of monoidal natural transformation between *colax* monoidal functors leads to a similar pictorial equality, which the reader will easily guess by turning the page upside down.

6 Traced monoidal categories

In a remarkable article, André Joyal, Ross Street and Dominic Verity [28] define a *trace* in a balanced monoidal category $\mathbb{C}$ as a natural family of functions

$$\mathrm{Tr}^U_{A,B} \;:\; \mathbb{C}(A \otimes U, B \otimes U) \;\longrightarrow\; \mathbb{C}(A, B)$$

satisfying three axioms:

vanishing (monoidality in U)

$$\mathrm{Tr}^{U\otimes V}_{A,B}(g) \;=\; \mathrm{Tr}^U_{A,B}(\mathrm{Tr}^V_{A\otimes U, B\otimes U}(g)), \qquad \mathrm{Tr}^I_{A,B}(f) \;=\; f.$$

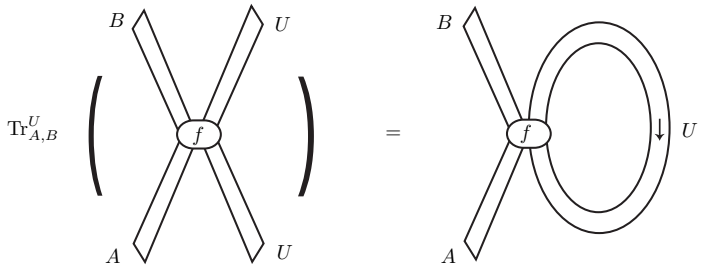
superposing

$$\begin{array}{rcl} \mathrm{Tr}^U_{A,B}(f) \otimes g & = & \mathrm{Tr}^U_{A\otimes C, B\otimes D}((\mathrm{id}_B \otimes \gamma^{-1}_{D,U}) \circ (f \otimes g) \circ (\mathrm{id}_A \otimes \gamma_{C,U})) \\ & = & \mathrm{Tr}^U_{A\otimes C, B\otimes D}((\mathrm{id}_B \otimes \gamma_{D,U}) \circ (f \otimes g) \circ (\mathrm{id}_A \otimes \gamma^{-1}_{C,U})) \end{array}$$

yanking

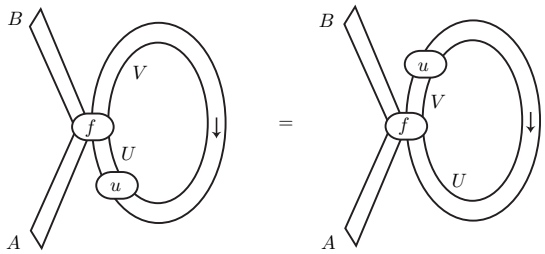
$$\mathrm{Tr}^U_{U,U}(\gamma_{U,U} \circ (\theta^{-1} \otimes \mathrm{id}_U)) \;=\; \mathrm{id}_U \;=\; \mathrm{Tr}^U_{U,U}(\gamma^{-1}_{U,U} \circ (\theta \otimes \mathrm{id}_U)).$$

A balanced monoidal category equipped with a trace is called a *traced monoidal category*. String diagrams for balanced monoidal categories extend to traced monoidal categories by depicting the trace as follows:

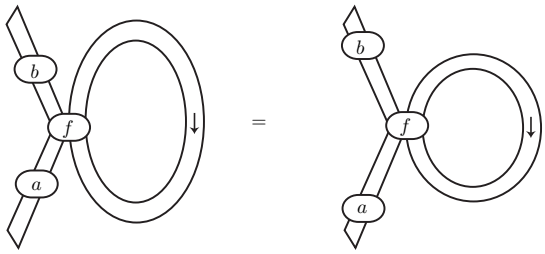


The small arrow embroidered on the ribbon recalls that this part of the string diagram depicts a trace, which expresses intuitively a notion of *feedback*. Thanks to this ingenious notation for traces, the algebraic axioms of a trace are depicted as a series of elementary topological deformations on ribbons, recalled here from [28]:

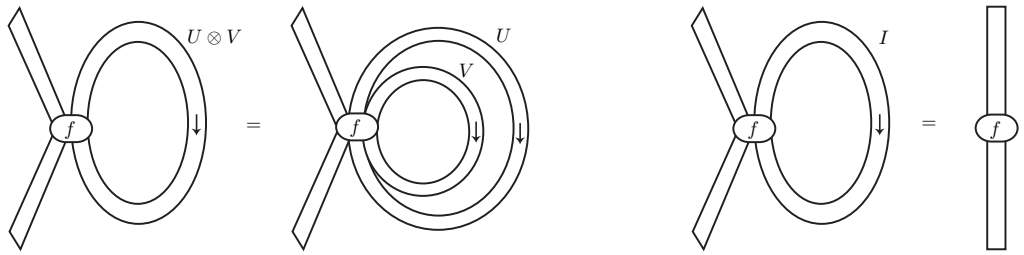
sliding (naturality in U)



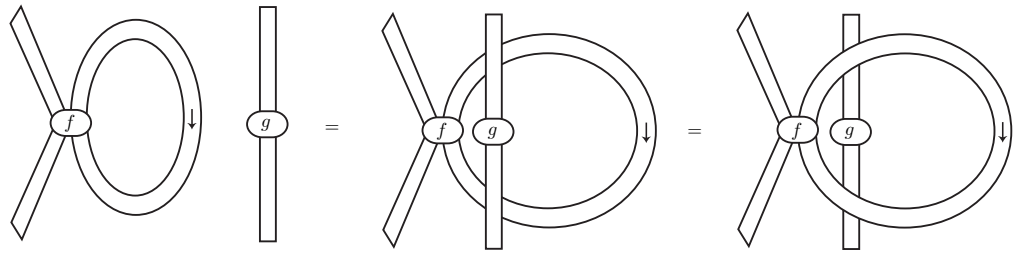
tightening (naturality in A, B)



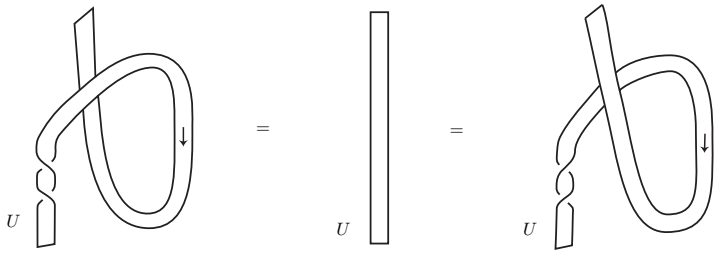
vanishing (monoidality in U)



superposing



yanking

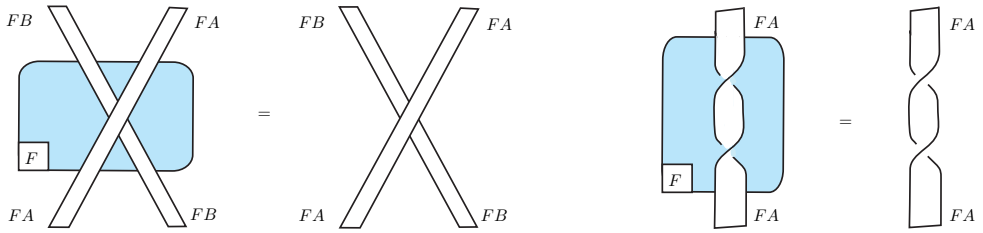


7 Transport of trace along a faithful functor

Recall [28] that a *balanced* monoidal functor $F : \mathbb{C} \longrightarrow \mathbb{D}$ between balanced monoidal categories is a strong monoidal functor satisfying that, for all objects A and B, the diagram below commutes

$$\begin{array}{ccc} FA \otimes FB & \xrightarrow{\gamma_{A,B}} & FB \otimes FA \\ \downarrow{\scriptstyle m_{A,B}} & & \downarrow{\scriptstyle m_{B,A}} \\ F(A \otimes B) & \xrightarrow{F(\gamma_{A,B})} & F(B \otimes A) \end{array}$$

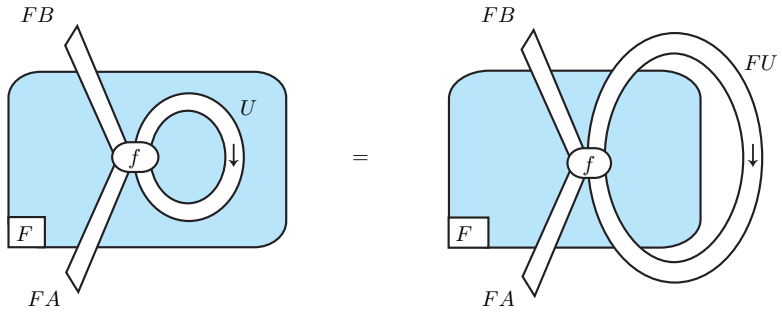
and the equality $F\theta_A = \theta_{FA}$ holds. This may be depicted as the two equalities:



When $\mathbb{C}$ and $\mathbb{D}$ are traced monoidal, one says that $F : \mathbb{C} \longrightarrow \mathbb{D}$ is traced monoidal when F is balanced monoidal, and preserves traces in the expected sense that, for all objects A, B and U and for all morphism $f : A \otimes U \longrightarrow B \otimes U$ of the category $\mathbb{C}$, the following equality holds:

$$F(\mathrm{Tr}^{U}_{A,B}(f)) \;=\; \mathrm{Tr}^{FU}_{FA,FB}(m^{-1}_{[A,B]} \circ Ff \circ m_{[A,B]}).$$

This equality is depicted as follows:



An elementary exercise in string diagrams with functorial boxes follows. It consists in establishing in a purely diagrammatic fashion a mild but useful generalization of a statement (Proposition 2.4) formulated in [28].

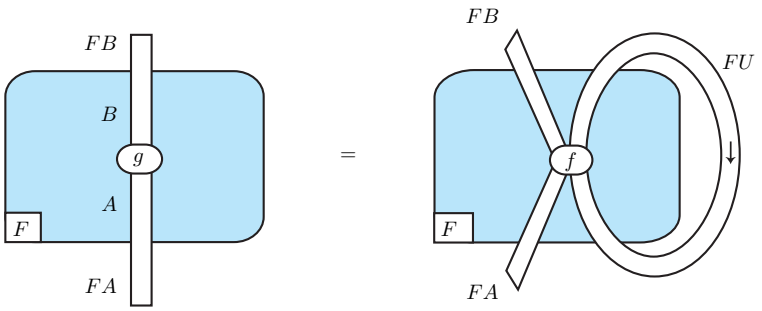
Proposition 1 (Transport along a faithful balanced functor) *Suppose that $F : \mathbb{C} \longrightarrow \mathbb{D}$ is a faithful, balanced monoidal functor with $\mathbb{D}$ traced monoidal. Then, there exists a trace on $\mathbb{C}$ for which F is a traced monoidal functor iff for all objects A, B, U of the category $\mathbb{C}$, and all morphism*

$$f \;:\; A \otimes U \longrightarrow B \otimes U$$

there exists a morphism $g : A \longrightarrow B$ such that

$$F(g) \;=\; \mathrm{Tr}^{FU}_{FA,FB}(m^{-1}_{[A,B]} \circ F(f) \circ m_{[A,B]}) \tag{10}$$

where Tr *denotes the trace in $\mathbb{D}$. The equality is depicted as follows:*

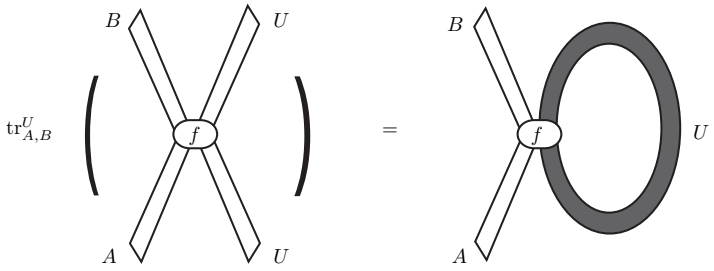


Moreover, if this trace on $\mathbb{C}$ exists, it is unique: it is called the trace on the category $\mathbb{C}$ transported from the category $\mathbb{D}$ along the functor F.

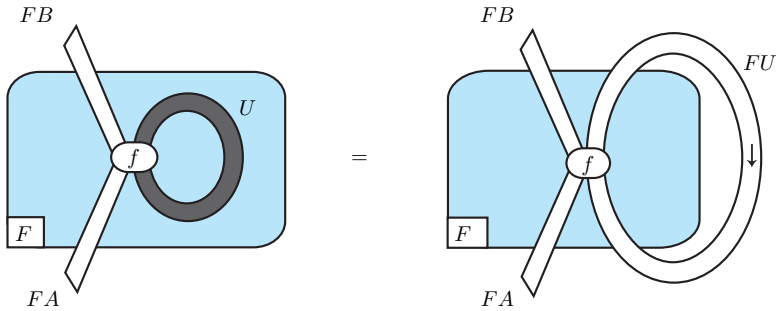
Proof. The direction ($\Rightarrow$) follows immediately from the very definition of a traced monoidal functor. Hence, we only establish here the converse direction ($\Leftarrow$). We suppose from now on that for every morphism $f : A \otimes U \longrightarrow B \otimes U$ there exists a morphism $g : A \longrightarrow B$ satisfying Equation (10). Note that the morphism g is unique because the functor F is faithful. This defines a family of functions noted

$$\mathrm{tr}^U_{A,B} \ : \ \mathbb{C}(A \otimes U, B \otimes U) \ \longrightarrow \ \mathbb{C}(A, B).$$

We establish that tr satisfies the equational axioms of a trace. To that purpose, we introduce a handy notation for the morphism $\mathrm{tr}^U_{A,B}(f)$:

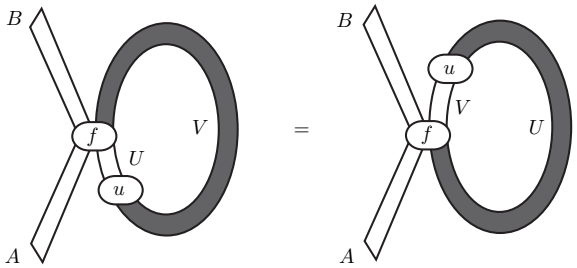


By definition, the morphism $\mathrm{tr}^U_{A,B}(f)$ satisfies the diagrammatic equality:

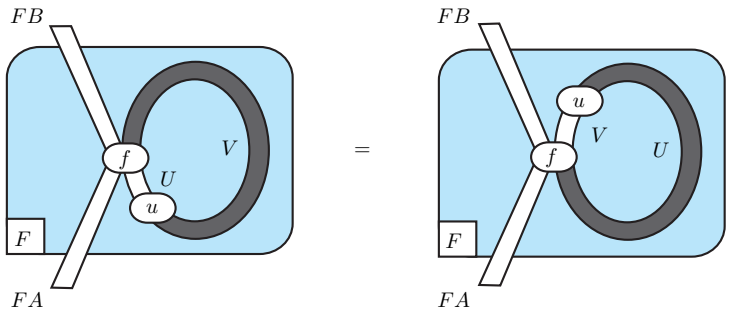


We establish each equational axiom by a series of elementary manipulations on string diagrams. Although the proof is diagrammatic, it is absolutely rigorous, and works for weak monoidal categories $\mathbb{C}$ and $\mathbb{D}$ as well as strict ones.

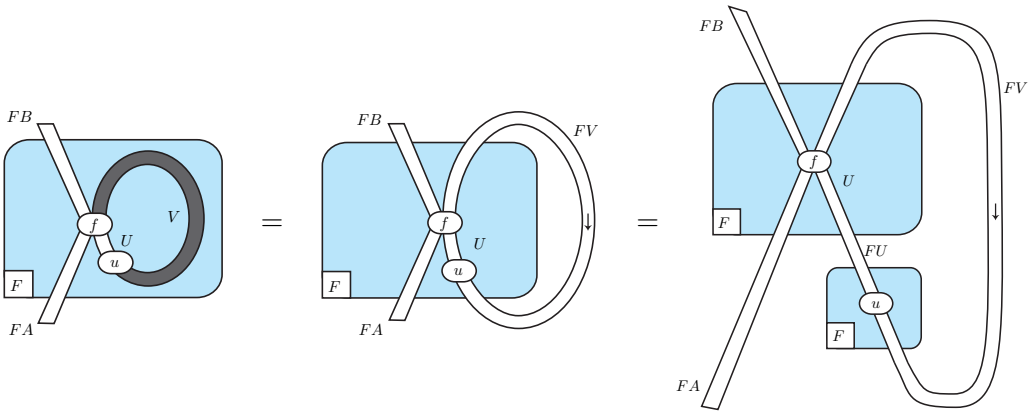
Sliding (naturality in U). We want to show the equality



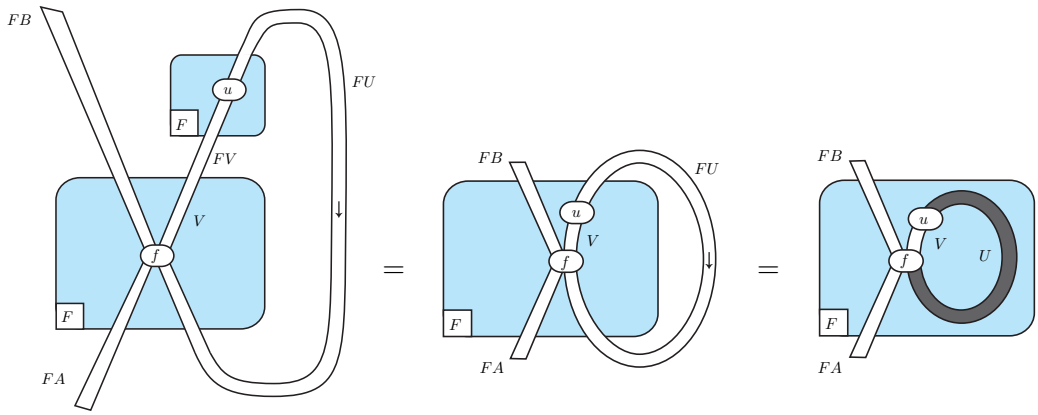
Because the functor F is faithful, it is sufficient to establish that the two morphisms $A \longrightarrow B$ have the same image $FA \longrightarrow FB$ in the target category $\mathbb{D}$:



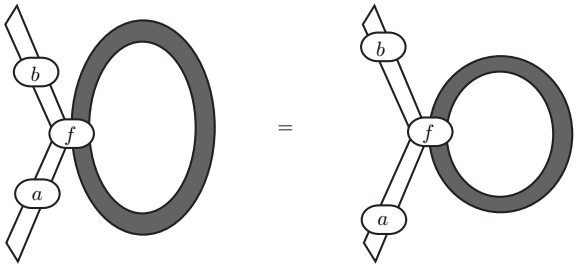
Once the definition of tr applied, we separate the box in two parts, using Equation (7) for the lax monoidal functor F:



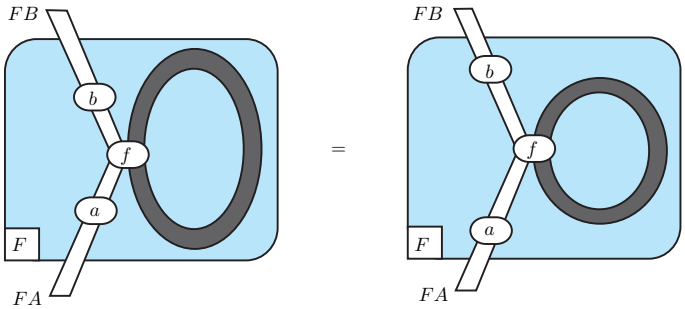
Then, after applying the sliding axiom of the trace Tr in the target category $\mathbb{D}$, we reunify the two separate boxes, using the variant of Equation (7) satisfied by colax monoidal functors. The definition of tr concludes the proof.



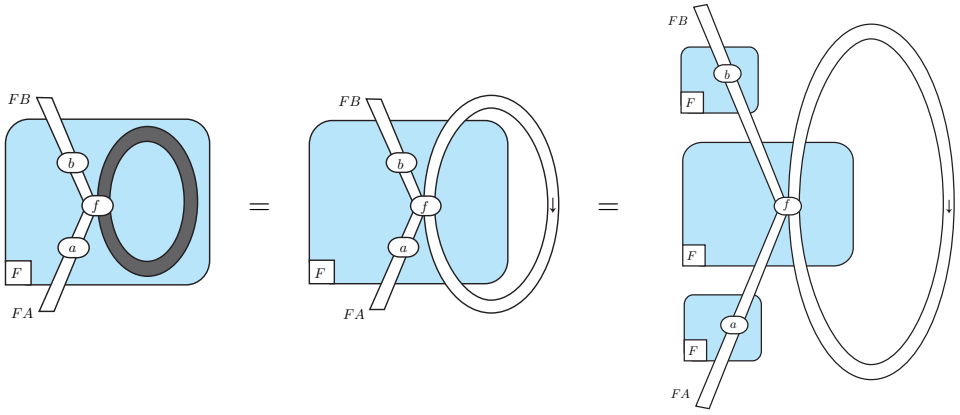
Tightening (Naturality in A and B). The proof is very similar to the proof of the sliding equality. Because the functor F is faithful, we will deduce the equality



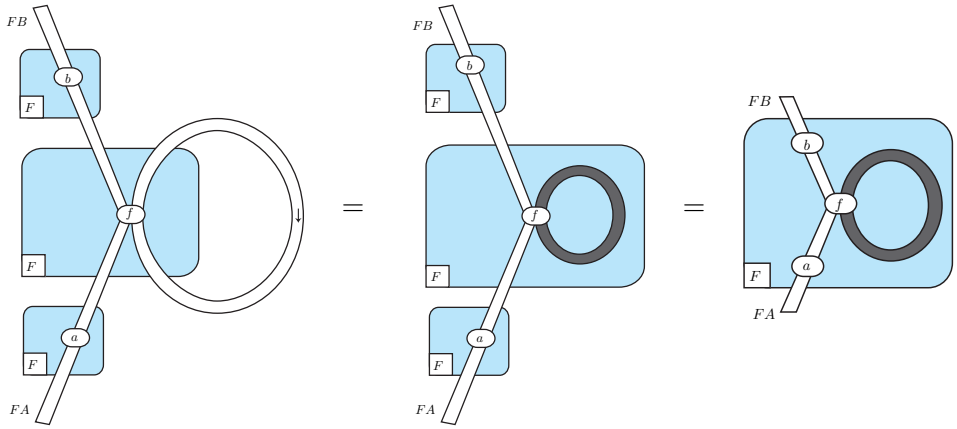
from the equality by F of the image of the two morphisms in the target category:



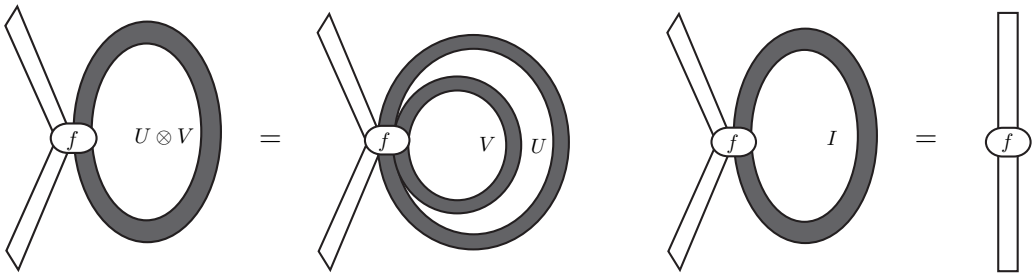
This is established as follows. Once the definition of tr applied, we separate the box in three parts, using Equation (7) for lax monoidal functors, and its colax variant:



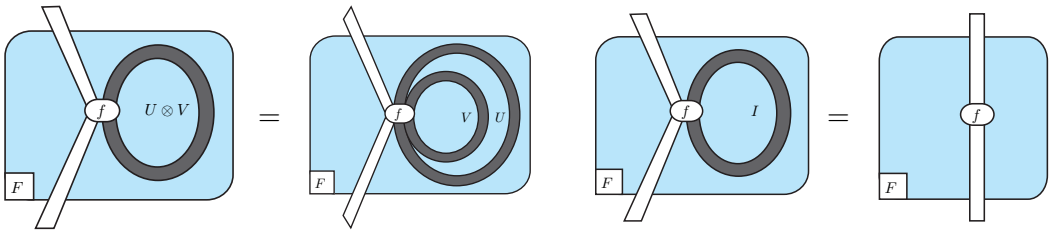
Then, we apply the tightening axiom of the trace Tr in the category $\mathbb{D}$, followed by the definition of tr, and finally reunify the three boxes together, using Equation (7) for lax monoidal functors, and its colax variant.



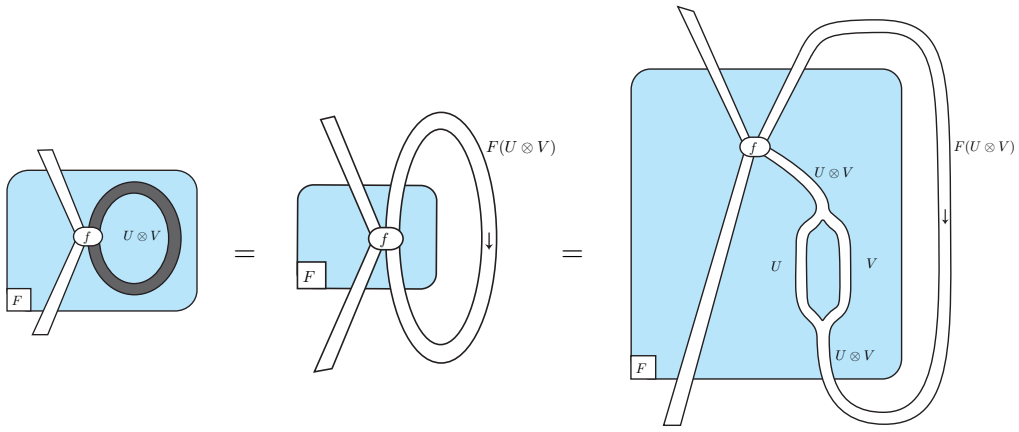
Vanishing (monoidality in U). We will proceed here just as in the two previous proofs, and deduce the two diagrammatic equalities formulated in the source category $\mathbb{C}$,



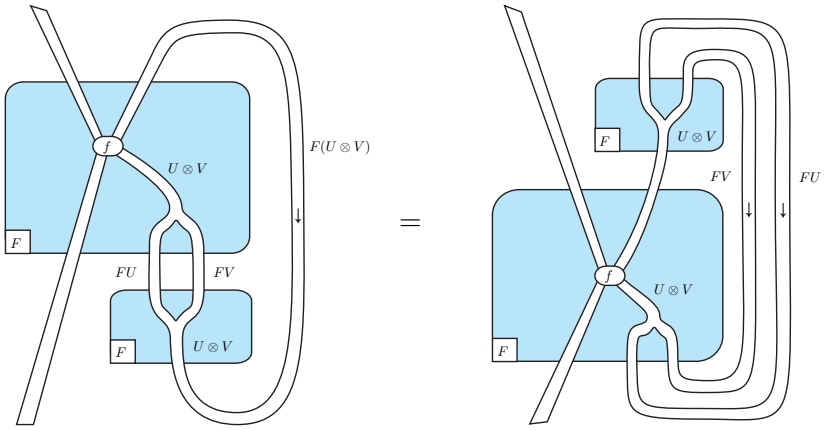
from the two diagrammatic equalities below, formulated in the target category $\mathbb{D}$:



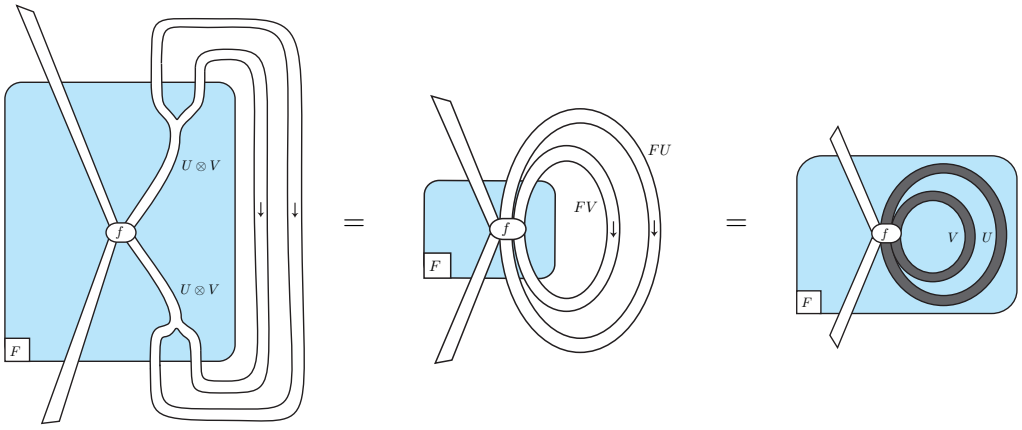
The first equation is established as follows. After applying the definition of tr, we split the string $U \otimes V$ in two strings U and V.



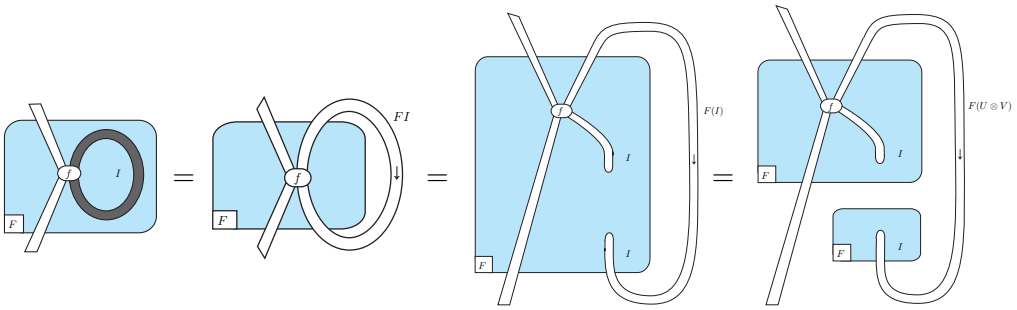
Then, we separate the box in two parts, using Equation (8) for the strong monoidal functor F, and apply together the sliding and vanishing axioms of the trace Tr in the category $\mathbb{D}$:



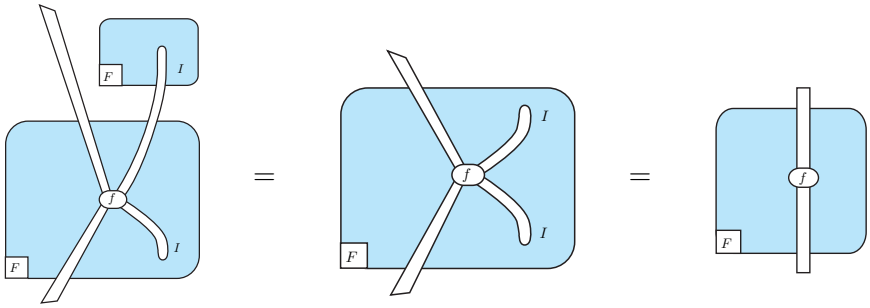
Finally, we reunify the two boxes using Equation (8), and conclude by applying the definition of tr twice.



The second equation is established exactly as the previous one, except that we are dealing now with the nullary case instead of the binary one. After applying the definition of tr, we split the string I, and separate the box in two parts, using Equation (8) for the strong monoidal functor F:

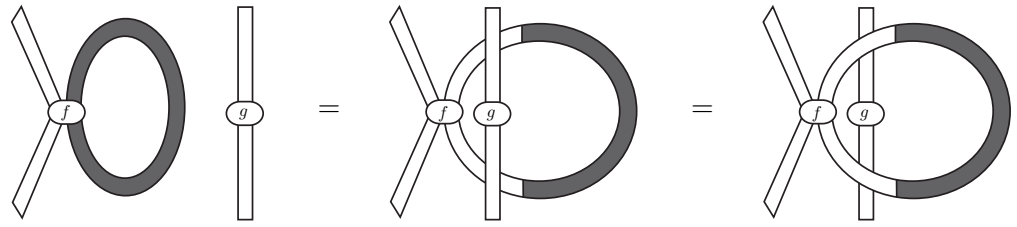


Then, just as for the binary case, we apply the sliding and vanishing axioms of the trace Tr and reunify the two boxes, before concluding.

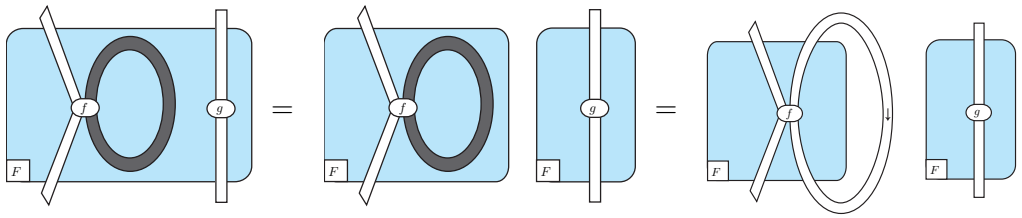


Note that we need the hypothesis that the functor F is *strong* monoidal in order to perform the manipulations for vanishing — while we needed only that it is lax and colax monoidal in the arguments devoted to sliding and tightening.

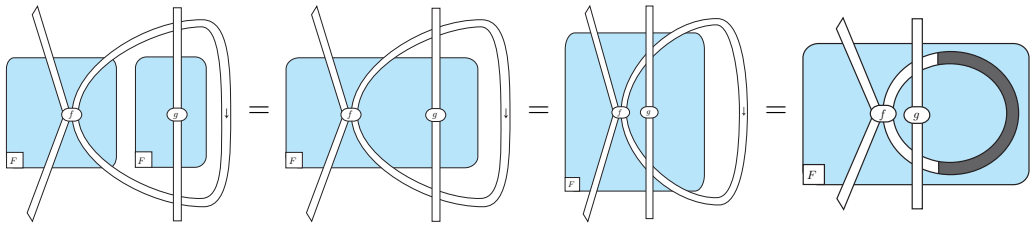
Superposing. We will establish only the first of the two equalities below — since the second one is proved in exactly the same way.



Because the functor F is faithful, this reduces to showing that the two morphisms have the same image in the category $\mathbb{D}$ — which we establish by the series of equalities below. First, we separate the box in two parts, using Equation (9) for the strong monoidal functor F; and apply the definition of tr in one of the two boxes.



Then, after applying the superposing axiom of the trace Tr in the target category $\mathbb{D}$, we merge the two boxes, using again Equation (9) for the strong monoidal functor F; we insert the two braidings inside the box, using the hypothesis that the functor F is balanced; and finally conclude using the definition of tr.



Yanking. The diagrammatic proof follows easily from the hypothesis that the functor F is faithful, and balanced monoidal. The proof is left to the reader as exercise.

From this, we conclude that tr defines a trace in the source category $\mathbb{C}$. The fact that the functor F is traced monoidal follows then immediately from the very definition of tr. This concludes the proof of Proposition 1. □

Application to models of linear logic. In a typical model of linear logic based on a linear adjunction (1) the category $\mathbb{M}$ is a full subcategory of the category of Eilenberg-Moore coalgebras of the comonad $! = L \circ M$ in the category $\mathbb{L}$ — and the functor L is the associated forgetful functor. In that case, Proposition 1 ensures that the category $\mathbb{M}$ is traced when the category $\mathbb{L}$ is traced, and when, moreover, the trace

$$\mathrm{Tr}^{LU}_{LA,LB}(f) \quad : \quad LA \longrightarrow LB \tag{11}$$

of every coalgebraic morphism

$$f \quad : \quad LA \otimes LU \longrightarrow LB \otimes LU \tag{12}$$

is coalgebraic. This is precisely what happens in the relational model of linear logic, where:

- $\mathbb{L}$ is the category *Rel* of sets and relations, with tensor product defined as usual set-theoretic product,
- $\mathbb{M}$ is the co-kleisli category of the comonad ! which transports every set A to the free commutative comonoid $!A$ with finite multisets of elements of A as elements, and multiset union as coproduct. Note that the co-kleisli category $\mathbb{M}$ is understood here as the full subcategory of free coalgebras of the exponential comonad.

This establishes that the category $\mathbb{M}$ has a well-behaved parametric fixpoint operator. A similar method applies to construct well-behaved parametric fixpoint operators in categories of games and strategies [51].

Another application: Masahito Hasegawa observed (private communication) that the category $\mathbb{M}$ is traced whenever it is the co-kleisli category of an idempotent comonad $! = L \circ M$ in a traced monoidal category $\mathbb{L}$. This interesting fact may be deduced from Proposition 1 in the following way.

By definition, a comonad $(!, \epsilon, \delta)$ is *idempotent* when the component of the comultiplication

$$\delta_A \quad : \quad !A \longrightarrow !!A$$

is an isomorphism, for every object A of the underlying category. In that case, the two morphisms $\epsilon_{!A}$ and $!\epsilon_A$ qualify as the inverse of the morphism δ_A, and are thus equal isomorphisms.

Now, suppose that the comonad $! = L \circ M$ generated by a linear adjunction (1) is idempotent; and that the category $\mathbb{L}$ is traced monoidal, with trace noted Tr. For general reasons related to adjunctions, it appears that for every morphism

$$f : A \times U \longrightarrow B \times U \tag{13}$$

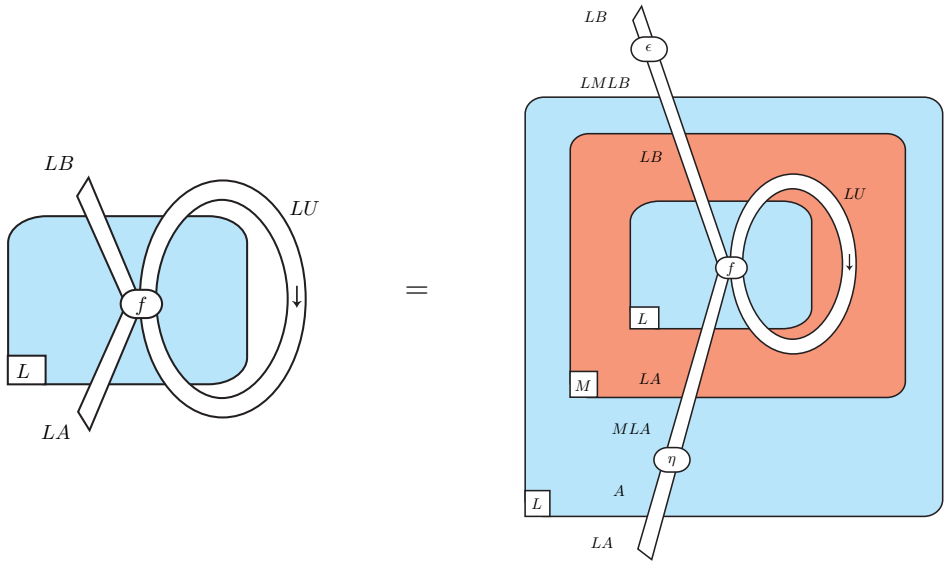
in the category $\mathbb{M}$, the morphism

$$h \quad = \quad \mathrm{Tr}^{LU}_{LA,LB}(m^{-1}_{[A,B]} \circ Lf \circ m_{[A,B]}) \quad : \quad LA \longrightarrow LB$$

is equal in the category $\mathbb{L}$ to the morphism

$$LA \xrightarrow{L\eta} LMLA \xrightarrow{LMh} LMLB \xrightarrow{\epsilon_{LB}} LB. \tag{14}$$

where η denotes the unit of the monad $M \circ L$. The equality is nicely depicted in string diagrams:



Here, Proposition 1 applies, and the category $\mathbb{M}$ is thus traced, whenever the functor L is faithful, and for every morphism f in (13), the morphism defined in (14) is the image $L(g)$ in the category $\mathbb{L}$ of a morphism $g : A \longrightarrow B$ in the category $\mathbb{M}$.

This is precisely what happens when the category $\mathbb{M}$ is defined as the co-kleisli category associated to an idempotent comonad $! = L \circ M$. In that case, indeed, the object B is the free coalgebra $B = MB'$ of an object B' of the category $\mathbb{L}$, and the morphism $\epsilon_{LB} = \epsilon_{!B'}$ is equal to the image $LM\epsilon_{B'} = !\epsilon_{B'}$ of the morphism $M\epsilon_{B'} : MLB \longrightarrow B$. The morphism g is thus defined as:

$$g \quad = \quad A \xrightarrow{\eta} MLA \xrightarrow{Mh} MLB \xrightarrow{M\epsilon_{B'}} B.$$

A nice problem remains open. A few years ago, Ryu Hasegawa [21] constructed a trace related to the Lagrange-Good inversion formula, in a category of analytic functors. This category, which is cartesian, is the co-kleisli category associated to a specific model of linear logic. Interestingly, the diagrammatic account exposed in this tutorial does not seem to apply (at least directly) to Ryu Hasegawa's construction. It would be extremely satisfactory to devise alternative algebraic conditions to cover this important example. We leave this open here.

8 Decomposing the modal box of linear logic

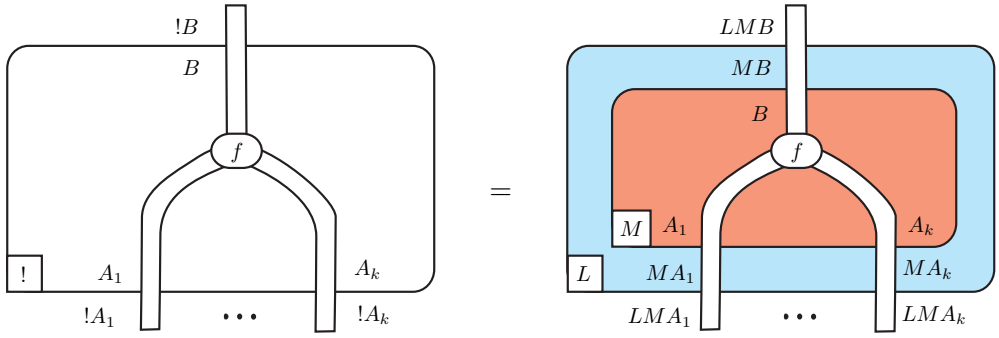
The decomposition $! = L \circ M$ of the exponential modality of linear logic illustrates the general diagrammatic principle that every functorial box separates an inside world (the source category) from an outside world (the target category), each world implementing his own (eg. cartesian, monoidal closed) policy. We take the freedom of considering here a "balanced" version of linear logic, whose categorical model is defined as a balanced monoidal adjunction

$$\mathbb{M} \underset{M}{\overset{L}{\rightleftarrows}} \mathbb{L} \qquad (\perp) \tag{15}$$

between a balanced monoidal category $\mathbb{L}$ and a cartesian category $\mathbb{M}$. Note that in such an adjunction, the functor L is balanced monoidal.

In that setting, the exponential box ! with its auxiliary doors labelled by the formulas $!A_1, ..., !A_k$ and with its principal door labelled by the formula $!B$

is translated as a lax monoidal box M enshrined inside a strong monoidal box L, in the following way:



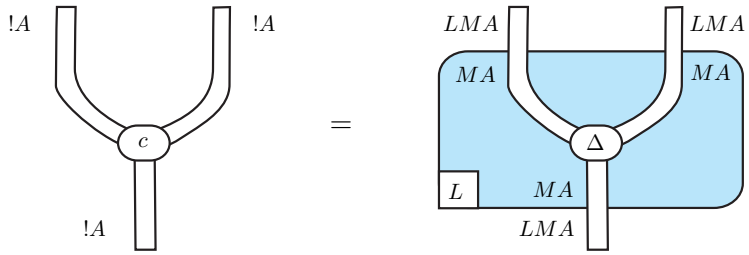
Now, the category $\mathbb{M}$ enshrined "inside" the functorial box L is cartesian, with binary product noted $\times$ here. Hence, every object X of the category $\mathbb{M}$ is equipped with a "diagonal" morphism

$$\Delta_X : X \longrightarrow X \times X$$

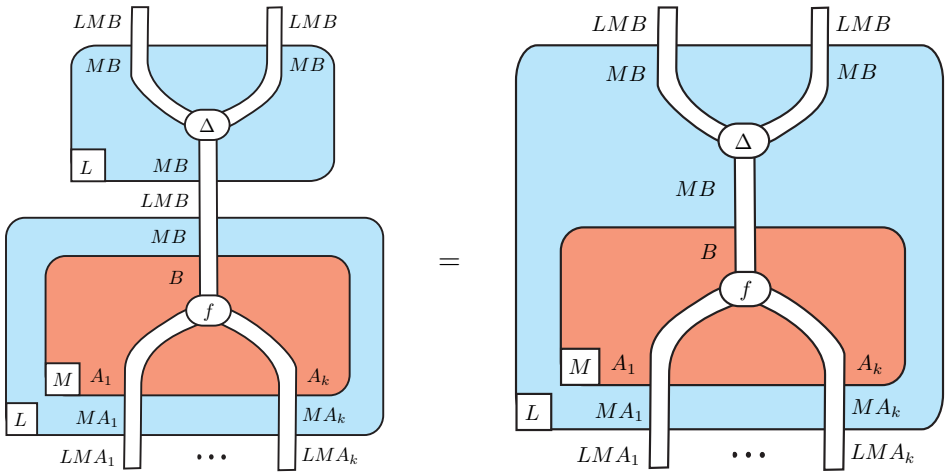
natural in X. In particular, every object A of the category $\mathbb{L}$ induces a diagonal morphism

$$\Delta_{MA} : MA \longrightarrow MA \times MA.$$

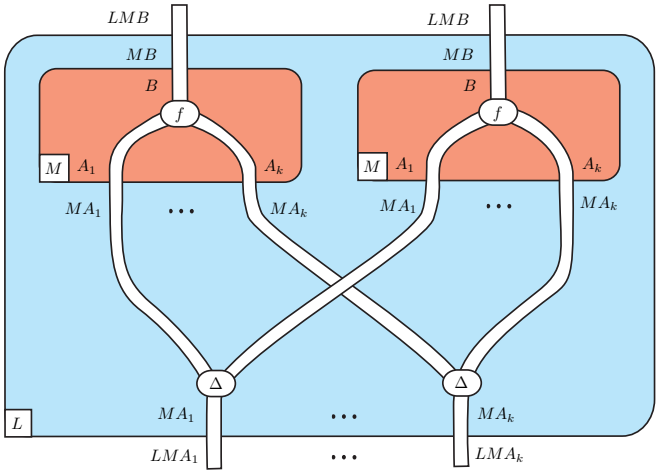
The *contraction* of linear logic is defined as the morphism $L(\Delta_{MA})$ depicted as the diagonal string Δ_{MA} enshrined inside the strong monoidal box L:



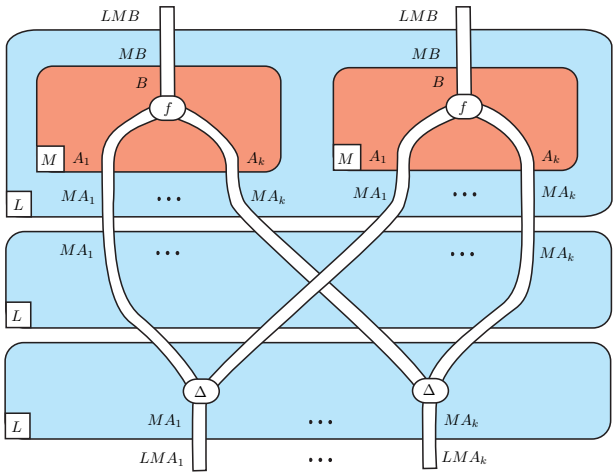
If one translates in string diagrams the usual cut-elimination step of linear logic between a contraction rule and an introduction rule of the exponential box, this decomposes the step in a series of more atomic steps. First, the box L which encapsulates the diagonal Δ_{MA} merges with the box L which encapsulates the content f of the exponential box. This releases the diagonal Δ_{MA} inside the cartesian category $\mathbb{M}$ enshrined in the exponential box.



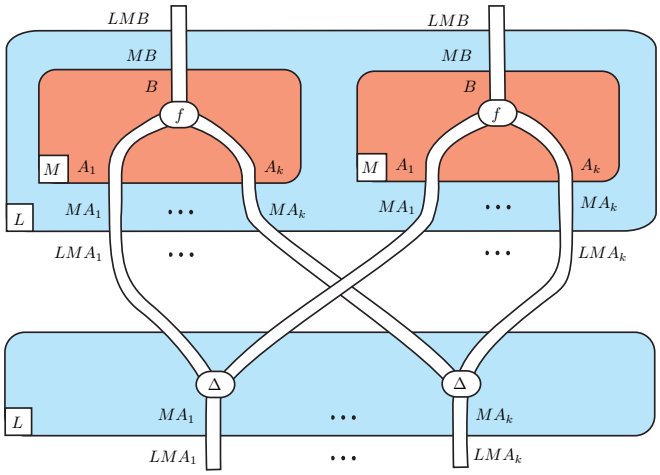
Then, the diagonal Δ_{MA} replicates the content f of the exponential box — or more precisely the morphism f encapsulated by the lax monoidal box M. Note that the duplication step is performed in the cartesian category $\mathbb{M}$ enshrined by the functorial box L, and that the equality follows from the naturality of Δ.



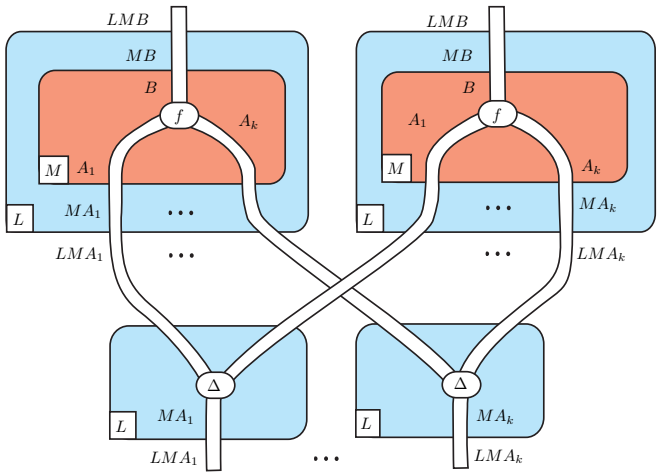
Once the duplication performed, the strong monoidal box is split in three horizontal parts using Equation (8).



The intermediate box may be removed, because the functor L is balanced.



Finally, the remaining monoidal boxes L are split vertically, using Equation (9).



This completes the categorical and diagrammatical transcription of this particular cut-elimination step. The other cut-elimination steps of linear logic involving the exponential box ! are decomposed in a similar fashion.

References

[1] **R. Amadio and P.-L. Curien**. *Domains and Lambda-Calculi.* Cambridge University Press, 1998.

[2] **A. Barber, P. Gardner, M. Hasegawa, G. Plotkin.** From action calculi to linear logic. *Proceedings of Computer Science Logic '97,* Aarhus, Denmark. Volume 1414 of Lecture Notes in Computer Science, Springer Verlag. 1997.

[3] **A. Barber.** *Linear Type Theories, Semantics and Action Calculi.* PhD Thesis of the University of Edinburgh. LFCS Technical Report CS-LFCS-97-371. 1997.

[4] **J. Bénabou.** Introduction to bicategories. *Reports of the Midwest Category Seminar.* Volume 47 of Lecture Notes in Mathematics, Springer Verlag. 1967.

[5] **N. Benton.** A Mixed Linear and Non-Linear Logic: Proofs, Terms and Models. *Proceedings of Computer Science Logic '94,* Kazimierz, Poland. Volume 933 of Lecture Notes in Computer Science, Springer Verlag. June 1995.

[6] **N. Benton, G. Bierman, V. de Paiva, M. Hyland**. *Term assignment for intuitionistic linear logic.* Technical Report 262, Computer Laboratory, University of Cambridge, 1992.

[7] **G. Berry**. Stable models of typed lambda-calculi. *Proceedings of the 5th International Colloquium on Automatas, Languages and Programming,* number 62 in Lecture Notes in Computer Science. Springer Verlag 1978.

[8] **G. Bierman.** On intuitionistic linear logic. *PhD Thesis.* University of Cambridge Computer Laboratory, December 1993.

[9] **G. Bierman.** What is a categorical model of intuitionistic linear logic? *Proceedings of the Second International Conference on Typed Lambda Calculus and Applications.* Volume 902 of Lecture Notes in Computer Science, Springer Verlag. Edinburgh, Scotland, April 1995. Pages 73-93.

[10] **H. Blackwell, M. Kelly, and A. J. Power.** Two dimensional monad theory. *Journal of Pure and Applied Algebra*, 59:1–41, 1989.

[11] **R. Blute, R. Cockett, R. Seely.** The logic of linear functors. *Mathematical Structures in Computer Science*,12 (2002)4 pp 513-539.

[12] **R. Blute, R. Cockett, R. Seely, T. Trimble.** Natural Deduction and Coherence for Weakly Distributive Categories. *Journal of Pure and Applied Algebra,* 113(1996)3, pp 229-296.

[13] **A. Burroni.** Higher Dimensional Word Problem, *Category Theory and Computer Science,* Lecture Notes in Computer Science 530, Springer-Verlag, 1991.

[14] **R. Cockett, R. Seely.** Linearly Distributive Categories. *Journal of Pure and Applied Algebra,* 114(1997)2, pp 133-173.

[15] **R. Cockett, R. Seely.** Linear Distributive Functors. In *The Barrfestschrift, Journal of Pure and Applied Algebra,* Volume 143, Issue 1-3, 10 November 1999.

[16] **B. J. Day and R. Street.** Quantum categories, star autonomy, and quantum groupoids. *Galois Theory, Hopf Algebras, and Semiabelian*

Categories. Fields Institute Communications 43 (American Math. Soc. 2004) 187-226.

[17] **G. Gentzen.** Investigations into logical deduction (1934). An english translation appears in *The Collected Papers of Gerhard Gentzen.* Edited by M. E. Szabo, North-Holland 1969.

[18] **J.-Y. Girard.** Linear logic. *Theoretical Computer Science*, 50: 1-102, 1987.

[19] **J.-Y. Girard.** Linear logic: its syntax and semantics. In *Advances in Linear Logic*, London Mathematical Society Lecture Note Series 222, pp. 1–42, Cambridge University Press, 1995.

[20] **M. Hasegawa.** Recursion from cyclic sharing: traced monoidal categories and models of cyclic lambda-calculi. *Proceeding of the 3rd International Conference on Typed Lambda-Calculi and Applications*, Springer Verlag, Lecture Notes in Computer Science 1210, (1997).

[21] **R. Hasegawa.** Two applications of analytic functors. *Theoretical Computer Science* 272 (2002) 113-175.

[22] **C. Hermida and J. Power**. Fibrational control structures. *Proceedings of CONCUR 1995.* Springer Lecture Notes in Computer Science 962. pp 117–129, 1995.

[23] **M. Hyland and A. Schalk**. Glueing and orthogonality for models of linear logic. *Theoretical Computer Science* 294(1/2): 183-231, 2003.

[24] **G. B. Im and M. Kelly.** A universal property of the convolution monoidal structure, *J. Pure Appl. Algebra* 43, pp. 75-88, 1986.

[25] **A. Joyal.** Remarques sur la théorie des jeux à deux personnes. *Gazette des Sciences Mathématiques du Québec*, volume 1, number 4, pp 46–52, 1977.

[26] **A. Joyal and R. Street.** Braided Tensor Categories, *Advances in Mathematics* 102, 20–78, 1993.

[27] **A. Joyal and R. Street.** The geometry of tensor calculus, I. *Advances in Mathematics* 88, 55–112, 1991.

[28] **A. Joyal, R. Street and D. Verity.** Traced monoidal categories. *Math. Proc. Camb. Phil. Soc.* 119, 447–468, 1996.

[29] **M. Kelly.** Doctrinal adjunction. *Lecture Notes in Math.* 420, pp. 257-280, 1974.

[30] **S. Lack.** Limits for lax morphisms. *Applied Categorical Structures.* 13(3):189-203, 2005.

[31] **Y. Lafont.** *Logiques, catégories et machines.* PhD thesis, Université Paris 7, 1988.

[32] **Y. Lafont**. From Proof Nets to Interaction Nets, In *Advances in Linear Logic*, London Mathematical Society Lecture Note Series 222, pp. 225–247, Cambridge University Press, 1995.

[33] **F. Lamarche** Sequentiality, games and linear logic, Unpublished manuscript. 1992.

[34] **J. Lambek and P. Scott.** Introduction to Higher Order Categorical Logic. Cambridge Studies in Advanced Mathematics Vol. 7. Cambridge University Press, 1986.

[35] **F. W. Lawvere.** Ordinal sums and equational doctrines. *Springer Lecture Notes in Mathematics No. 80*, Springer, Berlin, 1969, pp. 141-155.

[36] **S. Mac Lane.** Categories for the working mathematician. *Graduate Texts in Mathematics* 5. Springer Verlag 2nd edition, 1998.

[37] **M. Maietti, P. Maneggia, V. de Paiva, E. Ritter.** Relating categorical semantics for intuitionistic linear logic. Applied Categorical Structures 13(1): 1-36, 2005.

[38] **P.-A. Melliès.** Typed lambda-calculi with explicit substitutions may not terminate *Proceedings of the Second International Conference on Typed Lambda Calculi and Applications, Edinburgh,* Lecture Notes in Computer Science 902, pp. 328-334, Springer, 1995.

[39] **P.-A. Melliès.** Axiomatic Rewriting 4: a stability theorem in rewriting theory. *Proceedings of Logic in Computer Science 1998*, IEEE Computer Society Press, 1998.

[40] **P.-A. Melliès.** Axiomatic Rewriting Theory I: An axiomatic standardisation theorem. Chapter in the Jan Willem Klop Festschrift, called *Processes, Terms and Cycles: Steps on the Road to Infinity.* Edited by Aart Middeldorp, Vincent van Oostrom, Femke van Raamsdonk and Roel de Vrijer. Lecture Notes in Computer Science 3838, Springer Verlag. 2006.

[41] **P.-A. Melliès.** Categorical semantics of linear logic: a survey. To appear in *Panoramas et Synthèses*, Société Mathématique de France, 2007.

[42] **R. Milner.** Pure bigraphs: structure and dynamics. *Information and Computation*, Volume 204, Number 1, January 2006.

[43] **D. Pavlovic.** Categorical logic of names and abstraction in action calculi. *Mathematical Structures in Computer Science*, 7 (6) (1997) 619–637.

[44] **R. Penrose.** Applications of negative dimensional tensors, in *Combinatorial Mathematics and its Applications*, D. J. A. editor, pp. 221-244, Academic Press, New York, 1971.

[45] **R. Penrose.** Spinors and Space-Time, Vol. 1, pp. 68-71, 423-434, Cambridge University Press, Cambridge, U. K., 1984.

[46] **A. J. Power.** Enriched Lawvere Theories. In *Theory and Applications of Categories*, pp. 83–93, 2000.

[47] **R. Seely**. Linear logic, $*$-autonomous categories and cofree coalgebras. *Applications of categories in logic and computer science*, Contemporary Mathematics, 92, 1989.

[48] **S. Schanuel and R. Street.** The free adjunction. *Cahiers topologie et géométrie différentielle catégoriques* 27 (1986) pp. 81-83.

[49] **R. Street.** Limits indexed by category-valued 2-functors. *J. Pure Appl. Algebra* 8 (1976) 149-181.

[50] **R. Street.** Functorial calculus in monoidal bicategories. *Applied Categorical Structures* 11 (2003) 219-227.

[51] **N. Tabareau.** De l'opérateur de trace dans les jeux de Conway. *Mémoire de Master 2*. Master Parisien de Recherche en Informatique, Université Paris 7, Septembre 2005.