AIX-MARSEILLE UNIVERSITÉ
ED184 – MATHÉMATIQUES ET INFORMATIQUE
UMR 7373
INSTITUT DE MATHÉMATIQUES DE MARSEILLE

Thèse présentée pour obtenir le grade universitaire de docteur en mathématiques

Matteo ACCLAVIO

String diagram rewriting: applications in category and proof theory

Soutenue le 14/12/2016 devant le jury :

Willem HEIJLTJES	University of Bath	Rapporteur
Samuel MIMRAM	École polythecnique	Rapporteur
Vito Michele ABRUSCI	Universitá Roma Tre	Examinateur
Pierre-Louis CURIEN	Université Paris 7	Examinateur
Miriam QUATRINI	Aix-Marseille Université	Examinateur
Luigi SANTOCANALE	Aix-Marseille Université	Examinateur
Lorenzo TORTORA DE FALCO	Universitá Roma Tre	Examinateur
Yves LAFONT	Aix-Marseille Université	Directeur de thèse

Abstract

In the last century, several sciences enriched their syntax in order to model interactions. Not only computer science and quantum physics, but also biology and economics are examples of fields requiring syntax and semantics for concurrency as well as for sequentiality.

String diagrams are suitable for that purpose. In that syntax, we have two compositions: the *parallel* one and the *sequential* one, which may interact by the *interchange rule*. If we consider this rule as an equality, string diagrams are a syntax for *strict monoidal categories*, with a more intuitive graphical representation than traditional algebraic formulas.

In this thesis, we study this 2-dimensional syntax and its semantics. We consider diagram rewriting and we give two applications of those methods:

- a detailed proof of Mac Lane's coherence theorem for symmetric monoidal categories based on convergent diagram rewriting, which is given in arXiv:1606.01722;
- an interpretation of proof derivations by string diagrams for the *MELL* fragment of linear logic, which captures proof equivalence. We get a linear sequentializability test to verify if a diagram corresponds to a proof . This interpretation extends the one for the *MLL* fragment given in arXiv:1606.09016, providing also a cut-elimination result.

Résumé

Dans le dernier siècle, nombreux sciences ont enrichi leur syntaxe pour pouvoir modeler des interactions. Entre eux on peut compter l'informatique, la physique quantique, et aussi la biologie et l'économie : toutes ces sciences sont des exemples de domaines qui ont besoin d'une syntaxe et d'une sémantique soit pour la concurrence que pour la séquentialité.

Les diagrammes des cordes sont bien adapté à cet effet. Dans leur syntaxe on peut retrouver deux compositions : une composition parallèle et une composition séquentielle, qui peuvent interagir à travers une loi d'interchange. Si on considère cette loi comme une égalité, les diagrammes de cordes sont une syntaxe pour les catégories monoidales strictes, avec une représentation graphique plus intuitive que les formules algébriques traditionnelles.

Dans cette thèse, on ètude cette syntaxe de dimension 2 et sa sémantique. On considéré la réécriture des diagrammes et on donne des applications de cet méthode :

- une preuve détaillée du théorème de cohérence de Mac Lane pour les catégories monoidales symétriques basée sur un systéme de réécriture convergent donnée en arXiv:1606.01722;
- une interprétation des dérivations de preuves avec les diagrammes de preuve pour le fragment *MELL* de la logique linéaire, qui capture l'équivalence de preuves. On peut vérifier la séquentialité en temps linéaire, c'est à dire vérifier si un diagramme corresponds à une preuve. Cette interprétation est une extension de celle pour le fragment *MLL* donnée en arXiv:1606.09016 en donnant aussi un résultat de élimination du coupure.

Acknowledgments

I am grateful to my supervisor, Yves Lafont, for his support, both pedagogical and moral. I hope he will not regret the high degree of freedom he gave me in the management of my researches.

I thank Willem Heijltjes and Samuel Mimram to have accepted to be my referees and for all their precious suggestion and remarks on my manuscript. I am also much obliged to all the members of the jury, in particular to Lorenzo Tortora de Falco, who is in charge of the project for the bi-national curriculum in logic at Roma Tre university. This project allowed me to come for the first time in Marseille for the master, opening the way of my actual academic path.

Thanks to all the permanent members of LDP who hosted me patiently and promptly answer to all my scientific doubts: Lionel Vaux, Laurent Regnier, Miryam Quatrini, Emmanuel Beffara and Dimitri Ara.

I would equally like to thank the people of CATHRE project and all the other people with whom I had fruitful exchanges in these years: Michele Abrusci, Beniamino Accattoli, Filippo Bonchi, Pierre-Louis Curien, Stefano Guerrini, Daniel Hirschkoff, Philipe Malbos, Paul-André Melliés, François Metayer, Samuel Mimram, Roberto Maieli, Michele Pagani, Marco Pedicini. I am also particularly grateful to Yves Guiraud for his LaTeX suit *CATEX* [34] for string diagrams which saved me an uncountable amount of time.

A huge thank goes to people with whom I shared these last years of academic life: Anna*, Fra, Eugenia, Michele A*, Etienne, Marc B, Michele B, Fabio, Jovanna, Lucas, Cyrille, Paolo*, Giulione*, Marianna*, Milton*, Andrea*, Elena and surely somebody else who I am now forgetting. A particular mention (*) to who participate to the final rush in redaction and bureaucracy.

Moreover, I want to thank everybody from the outside world who helped me to keep contact with reality: the Pink Puffers, the Funkallisto, the Pompier Poney Club and all the owners of couches where I found hospitality during these years.

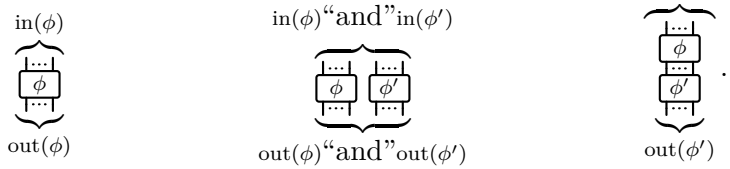
Thank to my family for all the "effort and sacrifices" done to sustain my study aboard.

Contents

Introduction

String diagrams are a graphical representation for some morphisms. This definition could appear vague and unclear, however, it can be considered as the most comprehensive and intuitive definition of this *syntax*... But soft! Back to origins: in 1948 Richard Feynman first introduced his diagrammatic notation in order to represent the complex interaction between sub-atomic particles in quantum physics [25]. The great advantage of his diagrams is their expressiveness: they allow to represent with in a single expression what could be represented by an infinite number of equivalent terms using standard notation. Albeit the seeds of *category theory* was sown by Samuel Eilenberg and Saunders Mac Lane [23] in 1945 and some intuition given by Penrose [73] in 1971 in its work on finite dimension vector spaces, we have to wait 1985 to have a formalization of this syntax for monoidal categories, in the work of André Joyal and Ross Street [44].

The intuitive idea behind string diagrams is quite simple: a diagram represents some kind of transformation, which take some incoming data and produce other out-coming data. These transformations can be composed arranging them in parallel or as a sequence:



(a) A diagram

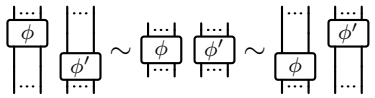
(b) Parallel composition

(c) Sequential composition (if $\text{in}(\phi') = \text{out}(\phi)$)

This "2-dimensional" *syntax* presents some new features about terms construction which leads to new paradigms in rewriting. In particular, there could be some terms which break the dualism of the two compositions and terms having no interaction at all with someof their subterms. Moreover, the relative *rewriting theory* loses some finiteness properties related to confluence [35] even in presence of finite rewriting systems due to the more complex shapes in rewriting overlaps.

The key of the expressiveness of this syntax lies in the so called *interchange*

rule which, following the graphical intuition, allows to move the mutual position of parallel composed diagrams:



.

Whenever we consider this relation as an equality, string diagrams become the fitful syntax for strict monoidal categories and it allows to simplify notations theorems' proofs.

In particular, we give a complete proof of coherence of symmetric monoidal categories via diagram rewriting only [3]. We complete the proof of Lafont's rewriting system confluence given in [52], detailing all its critical pairs solutions. Then, using the same method presented in [36], we explicitly construct the homotopy base of this rewriting system. Finally, we recover the coherence for symmetric monoidal categories by showing the correspondence between this base and the axiomatic coherence conditions.

The original purpose of this thesis was the study of rewriting in this 2-dimensional syntax; the latest researches of the author [4] focused instead on the use of this syntax for a reformulation of Jean-Yves Girard's *linear logic proof nets* syntax [28]. This graphical syntax for linear logic derivations was introduced for the multiplicative fragment and it generates a larger family of terms called *proof structures* generalized by Yves Lafont's *interaction nets* [50]. On one hand, proof nets represent a perfect semantic for equivalent multiplicative proofs; on the other hand, however, they require a *correctness* criterion in order to recognize whenever a proof structure is a proof net, that is, whether it corresponds to some derivations.

In fact, even if Girard's original correctness criterion for multiplicatives and others methods, notably by Danos-Regnier [17] and by Guerrini [31], have linear complexity, their adaptations result ineffective in presence of the multiplicative unit $\perp$ or exponentials. Of course, this syntax can be extended adding some machineries (such as *jumps* for $\perp$ rule [29] and *boxes* for exponential promotion rule) in order to guarantee the correctness. However, even extending the multiplicative fragment in order to include only units, we lose the expressiveness of this semantics along with the correspondence between proof nets and equivalence classes of proofs.

In this thesis we define the new syntax (and its relative semantics) of *linear logic proof diagrams* by means of diagram rewriting systems. Proof diagrams display a graphical representation of proofs, which reminds of proof nets, that keeps more information with respect to the old syntax. Indeed, we are able to control the string *crossings* preventing some incorrect configurations in term construction and keeping linear the complexity of the correction criterion for the *multiplicative and exponential linear logic* fragment. In fact, once the *twisting operators* (i.e. the cells representing crossings) are integrated with interchange rule by means of a proper definition of diagram rewriting rules, we

give a semantics requiring only a control over diagrams inputs and outputs to prove *sequentializability*, i.e. when a diagram represent a proof. In particular, this less coarse semantics differs from proof net syntax in requiring no additional tools such as jumps assignations, making possible to recover the part of proof equivalence we lose in *MLL* proof net with units. Moreover, similar argumentations allow to give a fitful notation for promotion rule (corresponding to proof net boxes) together with a semantics reflecting their interaction with the others exponential rules of *MELL* sequent calculus. Finally, we are able to include in this semantics also the monoidal structure concerning weakening and contraction rules.

The resulting semantics of proof diagrams have some interesting characteristics such as a more sequentialized structure of terms which, in irreducible terms have a layer form. In general, this structure of term allows us to be able to easily spot cells representing splitting rules, simplifying the sequentialization procedure for terms. Furthermore, this feature prevents the recovering of the proof equivalence. In fact, equivalent proofs which differ for the splitting order, as the ones displayed above, are represented by non-equivalent proof diagrams.

$$\dfrac{\dfrac{\begin{array}{cc}\vdots & \vdots\\ \vdash \Gamma, A, B & \vdash \Gamma', C\end{array}}{\vdash \Gamma, \Gamma', (B \otimes_1 C), A}\otimes_1 \qquad \begin{array}{c}\vdots\\ \vdash \Gamma'', D\end{array}}{\vdash \Gamma, \Gamma', \Gamma'', (A \otimes_1 D), (B \otimes_2 C)}\otimes_2 \quad \sim \quad \dfrac{\dfrac{\begin{array}{cc}\vdots & \vdots\\ \vdash \Gamma, A, B & \vdash \Gamma'', D\end{array}}{\vdash \Gamma, \Gamma', (A \otimes_1 D), B}\otimes_2 \qquad \begin{array}{c}\vdots\\ \vdash \Gamma', C\end{array}}{\vdash \Gamma, \Gamma', \Gamma'', (A \otimes_1 D), (B \otimes_2 C)}\otimes_1$$

This forces to introduce in the semantics some rules radically changing the structure of a proof diagram in order to have a sort of standardization for proof derivation based on the order of the axioms with respect to splittings.

In this semantics, free of the so called *commutative cuts*, we introduce some rewriting rules corresponding to *cut-elimination*, proving an analogous result of the cut-elimination theorem. The obtained 2-dimensional syntax of proof diagrams, together with their semantics, perfectly represents the behaviors of proof equivalence and proof semantics of *MELL*. Indeed, we keep the motivations behind the choice of using proof nets syntax giving an intuitive representation of (equivalence classes of) terms in a setting where a proper correctness criterion is not needed and sequentializability of terms can be checked in linear time.

Outline of the thesis

The first chapter contains background material on (word) rewriting and its correlation with algebra. We then recall some notions in category theory in order to give an interpretation of diagram rewriting systems as higher-dimensional rewriting.

The second chapter aims to give formal foundations to graphical representation of string diagrams. For that purpose, we give an alternative construction of string diagram syntax as a semantic of 2-dimensional words. In this context, we show the presence of some peculiar computation behaviors absent in word rewriting leading to new problems in confluence theory. As an application of these methods, we conclude the chapter with an extended formulation of the proof of Mac Lane's coherence theorem for symmetric monoidal categories using string diagram rewriting. In particular, we give the complete proof of the convergence of this system detailing the confluence of its critical pairs, which was never shown in the literature.

In the third chapter we give a formulation of a (diagram) rewriting system, whom terms we call proof diagrams, which mimics *MELL* proof net syntax of linear logic. Due to string diagrams syntax, we need the introduction of certain generators and relations in order to manage sequents as lists of formulas. This allows us to introduce of two non-crossing string in this syntax achieving a linear *sequentializability test* to check if a diagram represent a proof derivation even in presence of units and exponential. While on one hand the more rigid structure of this syntax reflects the focalized structure of proofs, on the other hand this forces to introduce specific relations (ad generators) in order to be able to perform the commutations of inference rules which change the splitting order in proofs. With a proper extension of this model, we finally achieve a semantics for proof diagrams corresponding to the one of equivalent linear logic proofs with a relative cut-elimination theorem.

Finally, the last chapter is devoted to some considerations about results given in the thesis, leaving some open questions and tips for future investigations.

Chapter 1

Backgrounds

"But in our opinion truths of this kind should be drawn from notions rather than from notations."

[Johan Carl Frederich Gauss, about the proof of Wilson's theorem.
In Disquisitiones Arithmeticae (1801) Article 76]

1.1 What is rewriting?

Rewriting is a formalism to represent data systems with discrete local evolutions. With the word *local* we want to remark that transformations will be operated on some portions of the data that are supposed to be, in some way, *linked* to each other. The word *discrete* remarks the discontinuous way in which one observes the evolution of the system "by steps". This makes rewriting a perfect candidate to represent computation as processes transforming data, step by step, by a given local transformation, that is exactly how we naturally think about computation. Let's give an example:

Example 1.1.1. If we observe the following expression

$$2 + 1 + 3 + 7 + 5 + 4 + 4 + 4 + 6 + 3 + 3$$

we could be tempted by computing the sum. In this case, we can scheme different ways to do it: relying on the fact that we assume $+$ to be *associative*, that is

$$(a + b) + c = a + (b + c) \quad \text{for all } a, b, c \in \mathbb{N},$$

we could calculate it by reducing it to a sequence of sums of two natural numbers. For example, we could compute it from the left to the right, from the right to the left or from the inside out, but, in any case, we naturally take some "pieces" of this expression and substitute them with something else — in this case we replace the symbol $+$ and the two adjacent numbers by their sum.

The symbol + links the number on its left to the one on its right, making sum act locally. We are not assuming, in this example, that we substitute non adjacent numbers. Indeed, in order to do this, we need to consider another powerful property called *commutativity* of sum which allows to permute summands.

If we denote by an arrow $\to$ the transition between the steps of our computation, one possible way of computing the sum could be the following: firstly we replace $2+1$ with 3, then $4+4$ with 8 and $7+5$ with 12 and so on

$$\underline{2+1}+3+7+5+4+4+4+6+3+3 \;\to\; 3+3+7+5+4+\underline{4+4}-6+3+3 \;\to$$

$$\to\; 3+3+\underline{7+5}+4+8+6+3+3 \;\to\; 3+3+12+4+8+6+3+3 \quad\to\; \ldots$$

The evolution of this computation is discrete: we do not care about how we transform, for example $2+1$, but just about the result after the transformation $2+1 \to 3$. Ontologically we limit the observations to the two states in which is present the datum $2+1$ and one in which this datum is replaced by 3 ignoring how this new datum has been obtained from the previous one. It could be obtained by counting sheep (discrete steps), drinking glasses of wine (continuous steps) or waiting one unity of time after two other ones (fully continuous).

In this thesis we use some different rewriting systems in an explicit way, but, for any sort of function, computation or, more generally, for any system characterized by a discrete observation of its evolution, we can consider a rewriting system to representing it.

In the next section we introduce some fundamental notions in *rewriting theory* by means of the corresponding notions in *word* (or *string*) rewriting. Even if this formalism is shared by different rewriting system such as *term rewriting*, *Petri nets* and *string diagram rewriting*, our choice is due not only to its utility in this thesis but also for historical reasons.

1.2 Word rewriting

Historically, word rewriting arose in the form of *combinatorial systems* by different authors at the beginning of the 20^{th} century. It was introduced at the end of the 19th century by Walther Von Dyck in order to study groups in terms of generators and relations. This way of studying groups allows to analyze some of its properties without the need of working with the whole set of its elements. For example, in algebraic geometry, in order to compute *fundamental groups* of topological spaces by means of groups amalgams arising from the Seifert-Van Kampen theorem, we use string rewriting systems as groups presentations.

This model was well formalized by Thue in 1914 (by means of the so called *semi-Thue systems* [82]) and used by Post in terms of *Post production*

systems [74] in order to prove what Martin Davis [19] states to be "the first unsolvability proof for a problem from classical mathematics – in this case the word problem for semigroups".

Word rewriting can be considered as the first formal system made to study computations as a discrete dynamic system. In some sense this made it the precursor of Church's lambda calculus (1930), Genzen's sequent calculus (1934) and Turing machines (1936) since it is (historically) the first model in which the object of the study is the computation itself and its potentiality of achieving a certain result. This means, using some modern terminology, that word rewriting was the first mathematical study of languages expressiveness.

Definition 1.2.1 (String (or Word) Rewriting System)**.** Given a set of *letters* or *symbols* Σ called *alphabet* or *signature*, the set Σ^* is the set of *words* on Σ, that is the set of all possible finite concatenations of symbols of the alphabet including the *empty word* 1. A *string (or word) rewriting system* (*SRS* for short) $(\Sigma, \mathcal{R})$ is given by an alphabet Σ and a set of *rewriting rules* $\mathcal{R} \subseteq \Sigma^* \times \Sigma^*$ i.e. a set of ordered pairs of words over Σ. We denote a rewriting rule by $w \to w'$ instead of (w, w') and we say that w is the *premise* or *source* of the rule while w' is its *target* or its *conclusion*. Moreover, we allow each rewriting rule in any context, that is, if $(w, w') \in \mathcal{R}$ then $uwv \to uw'v$ for every $u, v \in \Sigma^*$.

Definition 1.2.2. A *subword* of a word v is a word w such that $v = uwu'$ for some $u, u' \in \Sigma^*$ (u u' can be empty). The *overlap* of two subwords u and w of v is the longest word v' such that $u = u'v'$, $w = v'w'$ and $u'v'w'$ is a subword of v. If w is a subword of v we say that v contains w, moreover if $v = wu$ (resp. $v = uw$) $\exists u \in \Sigma^*$, w it's a prefix (resp. suffix) of v.

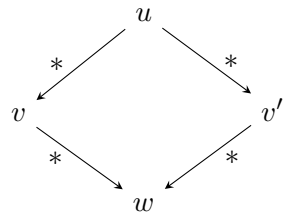
Notation. If $\mathcal{R}$ is a set of rewriting rules, we note $\to_{\mathcal{R}}^*$ (or simply $\to^*$ if there is no ambiguity) the transitive closure of $\mathcal{R}$ and $\leftrightarrow_{\mathcal{R}}^*$ (respectively $\leftrightarrow^*$) its symmetric and transitive closure. Moreover, in order to avoid ambiguities with this latter equivalence relation, if $w, w' \in \Sigma^*$, we write $w = w'$ in order to say that these two words are the same.

Definition 1.2.3 (Reduction chain)**.** If $(\Sigma, \mathcal{R})$ is a rewriting system, we call a *reduction chain* or *reduction path* a sequence $\{a_i\}_{i \in I}$ of words in Σ^* such that for all $i \in I$, $w_i \to w_{i+1}$. The *lenght* of a chain is the cardinality of I. Hence a reduction chain is finite/infinite according to the cardinality of I.

Definition 1.2.4 (Irreducible)**.** If $(\Sigma, \mathcal{R})$ is a rewriting system, a word $w \in \Sigma^*$ is *$\mathcal{R}$-irreducible* (or simply *irreducible* if there is no ambiguity) if there is no reduction chain starting from it. If there are no ambiguities, we denote by $irr(\Sigma)$ the set of $\mathcal{R}$-irreducible forms of the rewriting system $(\Sigma, \mathcal{R})$.

Definition 1.2.5 (Convergent rewriting system)**.** A rewriting system $(\Sigma, \mathcal{R})$ is *noetherian* or *terminating* if there are no infinite reduction chains.

A rewriting sytem is *confluent* or has the *Church-Rosser property* if for every u, v, v' such that $u \to^* v$ and $u \to^* v'$, there is some w such that $v \to^* w$ and $v' \to^* w$.

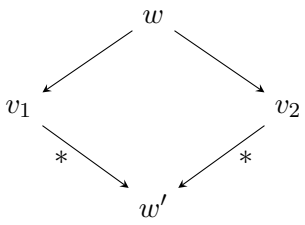


A rewriting system is *convergent* if it is terminating and confluent.

Moreover we say that a rewriting system is *locally confluent* if it is confluent and for any $u, v, v' \in \Sigma^*$ such that $u \to v$ and $u \to v'$, we have $v \to w$ and $v' \to w$ for some $w \in \Sigma^*$.

Definition 1.2.6 (Critical Pair)**.** A *conflict* or *critical pair* (P_1, P_2) between two rewriting rules R_1 and R_2 appears when R_1 and R_2 can be applied to a same word w. In this case one ha two one-step reduction paths $P_1 : w \to v_1$ and $P_2 : w \to v_2$. A *critical peak* is a minimal non-trivial conflict (P_1, P_2) where the sources of R_1 and R_2 are two overlapping subwords of w. For this reason we often note a critical peak by its source w.

A conflict (P_1, P_2) is *solvable* if there exist two rewriting sequences $Q_1 : v_1 \to^* w'$ and $Q_2 : v_2 \to^* w'$. We say that (Q_1P_1, Q_2P_2) is a *solution* of the conflict and we call the following diagram its *confluence diagram*



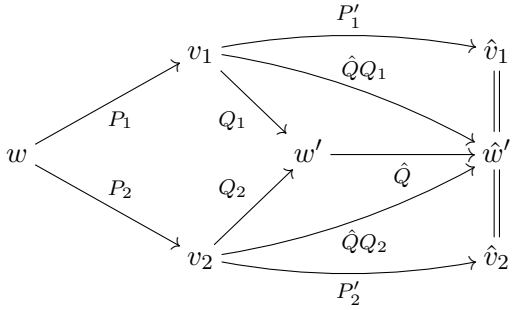
.

Remark 1.2.7. *In word rewriting, a critical peak is a conflict where the corresponding word w contains two subwords v and v' with non-empty overlap such that v and v' are respectively the prefix and the suffix of w*

Proposition 1.2.8 (Newman lemma [70])**.** *A terminating rewriting system is confluent if and only if every critical peak is solvable.*

Proof. The left-to-right implication is trivial by definition of confluent rewriting system and critical peak.

In order to prove the converse we have to prove that any conflict is solvable. Lets consider a conflict (P_1, P_2) and two maximal rewriting paths $P_1'P_1 : w \to^* \hat{v}_1$ and $P_2'P_2 : w \to^* \hat{v}_2$. If (Q_1P_1, Q_2P_2) is a solution of the conflict (P_1, P_2) with $Q_1P_1, Q_2P_2 : w \to w'$ and we have $\hat{v}_1 = w'$ or $\hat{v}_2 = w'$ the proposition is proved. Otherwise, we consider the maximal rewriting paths $\hat{Q}Q_1 : v_1 \to^* \hat{w}'$ and $\hat{Q}Q_2 : v_2 \to^* \hat{w}'$.



By induction over the length of $P_1'P_1$ and $P_2'P_2$, we are able to prove that $\hat{v}_1 = \hat{w}' = \hat{v}_2$. □

Definition 1.2.9 (Normal form)**.** If $(\Sigma, \mathcal{R})$ is a rewriting system and $w \in \Sigma^*$, the word $w' \in \Sigma^*$ is one *irreducible form of* w if it is irreducible and there exists a rewriting path $w \to^* w'$. Moreover, if this irreducible form is unique, we call it the *normal form of* w.

Word problem decidability and Chomsky's hierarchy

Many interesting decidability results are related to word rewriting. In particular, the so called *word problem* is *undecidable*. It consists in determinating if, given a rewriting system $(\Sigma, \mathcal{R})$ and two words $w, w \in \Sigma^*$, $w \leftrightarrow^* w'$, that is there exists an unoriented rewriting path from w to w'. In order to prove this result, we recall some basic definitions in computer science such as *Turing machines* definition. In this section we also recall some results of Noam Chomsky's works in linguistic where word rewriting systems are seen as *formal grammars* that can be classified on the base of expressivity of the language they generate, finally resulting related to some limitation over the form of rewriting rules.

Definition 1.2.10 (Turing Machine)**.** A Turing machine is an abstract machine consisting of:

- An *infinite tape* containing *cells* in which symbols of a fixed *alphabet* are written;
- A *head* which can *read and write* symbols on the tape and move to the left (L) or to the right (R) on it;
- A *set of instructions* for the head depending on the input and the current state.

More formally a Turing machine T is a 5-tuple $(Q, \Sigma, q_0, \perp, \delta)$ where:

- Q is the set of states;
- Σ an alphabet with special symbol $\square$ representing an empty cell (it is not explicitly written in the list of symbols in Σ);
- $q_0 \in Q$ is the *initial state*;
- $\perp \subset Q$ is a set of *final states*;
- $\delta : \Sigma \times Q \setminus \perp \to \Sigma \times Q \times \{L, R\}$ is the *transition function.*

Definition 1.2.11 (Configuration for a Turing machine)**.** Given a Turing machine M, a *configuration* for T is given by the content of its tape, a state and the head position on the tape. An *initial configuration* is a configuration with state q_0, a *final configuration* is a configuration with state $q_j \in \perp$. If a configuration s_{n+1} is *derivable* by s_n we note $s_n \to_M s_{n+1}$ a *transition* of T.

Definition 1.2.12 (Computation)**.** A computation of a Turing machine T is a sequence of configurations $s_0, s_1, \ldots$ with s_0 an initial configuration and such that $s_n \to_M s_{n+1}$ for all $n \geq 0$. A computation terminates if it is of the form $s_0 \to \ldots \to s_n$ with s_n such that there is no configuration s_{n+1} such that $s_n \to s_{n+1}$. Otherwise we say that the computation *diverges.*

Remark 1.2.13. *If $s_\perp$ is a transition containing a final state (a final configuration), there is no s such that $s_\perp \to_M s$, Hence a computation can contain at most one final configuration at the end of a terminating computation.*

Proposition 1.2.14. *Every non deterministic Turing machine $T = (Q, \Sigma, q_0, \perp, \delta)$ can be simulated by a word rewriting system.*

Proof. Let Σ the alphabet of T and $Q = \{q_i\}_{i \in I}$ the states of T. It is possible (see [20]) to code any configuration of a Turing machine T by the word over the alphabet $\Sigma_T = \Sigma \cup Q$ given by the content of its tape (which can be represented as a word in Σ^*) in which the symbol q (corresponding to a state) is placed before the one read by the head. For example the word $a_0 \ldots a_{i-1} q a_i \ldots a_n$ corresponds to the following configuration

				q $\bigtriangledown$				
$\ldots$	a_0	$\ldots$	a_{i-1}	a_i	a_{i+1}	$\ldots$	a_n	$\ldots$

Under this assumption, every transition in δ is interpreted by a rewriting rule:

$$qa_i \to a_i' q' \qquad \text{if} \qquad \delta(a_i, q) = (a_i', q', R)$$

$$a_{i-1} q a_i \to q' a_{i-1} a_i' \qquad \text{if} \qquad \delta(a_i, q) = (a_i', q', L)$$

□

Hence we have the following result:

Theorem 1.2.15. *Word rewriting systems are a Turing-complete computational model.*

Chomsky's hierarchy

In Chomsky's works on linguistic ([15], 1950's), a variant of classical word rewriting systems is proposed in order to present a formal method able to describe the cognitive process involved in the construction of sentences in any natural language. The idea is to build the grammatical structure of an utterance starting from a *start symbol* and then build up a concrete phrase replacing the *non-terminal symbols* (which can be seen as grammatical variables) with *terminal symbols* (corresponding to concrete words). Obviously a formal grammar can be seen as a rewriting system over the alphabet made of grammatical variables and natural language's words and with the rewriting rules consisting of these transformations.

Definition 1.2.16 (Formal grammar)**.** A *(formal) grammar* $G = (S, \Sigma, P, S)$ is given by:

- A finite set N of *non-terminal symbols*;
- A finite set Σ of *terminal symbols* disjoint from N;
- A finite set P of production rules of the form $(N \cup \Sigma)^* \to (N \cup \Sigma)^*$ such that the premise of any rule contains at least one non-terminal symbol in N.
- A special symbol $S \in N$ called *start symbol*.

We call *sentential form* an element of $(N \cup \Sigma)^*$ which can be derived by the start symbol S. It is an element of $\{w \in (N \cup \Sigma)^* | S \to^* w\}$. A sentential form containing no N symbol, that is a sentential form in Σ^*, is called *sentence*. The *language* $L(G)$ is defined as the set of all sentences, i.e. $L(G) = \{w \in \Sigma^* | S \to^* w\}$.

Formal grammars can be classified by the form of their production rules:

Definition 1.2.17 (Chomsky's hierarchy)**.** Formal grammar are classified as follows:

- *type-0 grammars* are formal grammars with no additional restriction over rules. They generate *recursive* languages;
- *type-1 grammars* are formal grammars in which rules are of the form $\alpha A \beta \to \alpha \gamma \beta$ with $A \in N$ and $\alpha, \beta, \gamma \in (N \cup \Sigma^*)$, $\gamma \neq 1$. They generate *context-sensitive* languages;

- *type-2 grammars* are formal grammars in which rules are of the form $A \to \gamma$ with $A \in N$ and $\gamma \in (N \cup \Sigma^*)$, $\gamma \neq 1$. They generate *context-free* languages;

- *type-3 grammars* are formal grammars in which rules are of the form $A \to a$ or $A \to aB$ with $A, B \in N$ and $a \in \Sigma$ or the rule $S \to 1$ if S does not appear as premise in other rules. They generate *regular expressions*.

Word problem and algebra

A *group* is an algebraic structure consisting of a set of elements together with a *binary operation*. This operation is a function which associate to any pair of elements of the group a third one. Moreover, to be a group, four additional axioms have to be satisfied: closure, associativity, identity and invertibility.

A group is one of the simplest algebraic structures and it was the first one studied with this modern algebraic point of view. The concept of group arose from Évariste Galois' studies (1830's) on polynomial equations: he linked their solubility to some particular properties of a group associated to each polynomial. The study of groups was also developed in some other field of math: Felix Klein in 1872 in his *Erlagen program* classifies the new geometries (non-euclidean and projective) discovered in the 19th century considering their groups of symmetries. Moreover, in number theory, in order to prove the Fermat's last problem, this new notion was used to generalize results to class of object with similar numerical properties.

Even though the notion of *monoid* differs from the group one for the absence of the invertibility axiom, it started to be studied later at the beginning of 20th century. By eliminating the notion of *inverse element*, we obtain a structure which can represent the concept of functions composition in computing processes. In fact, even if we know a result and the transformations we have done to obtain it, it could be possible that we can't recover the initial data due to the fact that some of these transformations could be not reversible. For this reason monoids are used for models in which there could be some irreversible processes ans they found applications in theoretical computer science, category theory but also probability and dynamic systems.

An important tool in studying monoids (and groups) are *monoid presentations*. A monoid presentation consists of a set of generators together with a set of relations between certain of their combinations. Once an orientation is fixed for each of these relations, a monoid presentation gives a word rewriting system and vice versa. This allows to study certain infinite objects by means of a finite set of generators and relations. We show that even in finite presented monoids and groups we still have some undecidability problems.

We recall some definitions on monoid theory.

Definition 1.2.18 (Group)**.** A *group* $G = (S, *)$ is given by a set S (support

of G) and a *binary operation* $*$ on S:

$$*_G : S \times S \to S$$

satisfying the following axioms (group axioms):

- (Closure): $\forall g, g' \in G,\ g * g' \in G$;
- (Associativity): $\forall g, g', g'' \in G,\ g * (g' * g'') = (g * g') * g''$;
- (Identity): $\exists e \in G$ such that $\forall g \in G, e * g = g = g * e$;
- (Invertibility): $\forall g \in G\ \exists g^{-1} \in G$ such that $g * g^{-1} = e = g^{-1} * g$;

where $g \in G$ means $g \in S$.

Definition 1.2.19 (Subgroup)**.** If $H \subseteq G$ is a subset of elements of G, H is a *subgroup of* G (noted by $H \leq G$) if $1 \in H$ and H is closed under the operation of G (the axioms are then necessarily satisfied).

Definition 1.2.20 (Quotient Group)**.** A *quotient group* $Q = \frac{G}{\mathcal{R}}$ is a group obtained identifying together elements of a group G by a congruence $\sim$, that is an equivalence relation $\mathcal{R}$ which is compatible with the group operation. The elements of Q (class of equivalence) are usually noted by $[g]_{\mathcal{R}}$, with $g \in G$. Furthermore, the set $N = [1]_{\mathcal{R}}$ of elements which are equivalent to 1_G is a normal subgroup. Vice versa, every N normal subgroup of G induces acongruence on G given by $g \sim g' \Leftrightarrow gg'^{-1} \in N$.

Definition 1.2.21 (Group Homomorphism)**.** Let G, G' to be groups. A *group homomorphism* ϕ is a map $\phi : G \to G'$ such that: $\phi(g *_G g') = \phi(g) *_{G'} \phi(g')$

Definition 1.2.22 (Group Isomorphism)**.** A group isomorphism $\phi : G \to G'$ is a bijective homomorphism.

Definition 1.2.23 (Monoid)**.** A *monoid* $M = (S, *)$ is given by a set S and a *binary operation* $*$ on S:

$$* : S \times S \to S$$

satisfying the following axioms:

- (Closure): $\forall g, g' \in G,\ g * g' \in G$;
- (Associativity): $\forall g, g', g'' \in G,\ g * (g' * g'') = (g * g') * g''$;
- (Identity): $\exists e \in G$ such that $\forall g \in G, e * g = g = g * e$.

Definition 1.2.24 (Submonoid)**.** A subset $N \subseteq M$ is a submonoid of M if it contains the unity and it is closed under the binary operation induced by that one of M (i.e. for every $x, y \in N$, $xy \in N$).

Definition 1.2.25 (Quotient Monoid)**.** If M is a monoid and $\sim$ acongruence on M, the *quotient monoid* $\frac{M}{\sim}$ is the monoid given by the set of equivalence class of M relative to $\sim$ and the operation induced by the one of M.

Definition 1.2.26 (Monoid Homomorphism/Isomorphism)**.** If M and N are two monoids, a map $f : M \to N$ is an *homomorphism* if $f(x *_M y) = f(x) *_N f(y)$ for all $x, y \in M$. An *isomorphism* f is a bijective *homomorphism.*

Definition 1.2.27 (Monoid Presentation)**.** A *monoid presentation* is given by a set of generators Σ and a set of relations $\mathcal{R} \subset \Sigma^* \times \Sigma^*$. We give monoid presentations by means of rewriting system. A presentation is called *finite* if Σ and $\mathcal{R}$ are finite sets.

If M is a monoid, $M = \langle \Sigma, \mathcal{R} \rangle^+$ means that M is equal to the quotient $\frac{\Sigma^*}{\leftrightarrow^*_{\mathcal{R}}}$ of the monoid Σ^* freely generated over Σ where $\leftrightarrow^*_{\mathcal{R}}$ is the congruence generated by $\mathcal{R}$ (the smallest equivalence relation which contains $\mathcal{R}$ and is compatible with the multiplication).

Similarly, a presentation of a group is given by an alphabet Σ and a set of pairs of words on the alphabet $\Sigma \cup \bar{\Sigma}$ where $\bar{\Sigma} = \{\bar{\sigma} | \sigma \in \Sigma\}$. By $G = \langle \Sigma, \mathcal{R} \rangle$ we denote that the group G is isomorph (as monoid) to the monoid $M_G = \langle \Sigma \cup \bar{\Sigma}, \mathcal{R} \cup \mathcal{I}_\Sigma \rangle^+$ where $\mathcal{I}_\Sigma = \{(\sigma\bar{\sigma}, 1), (\bar{\sigma}\sigma, 1)\}_{\sigma \in \Sigma}$.

Example 1.2.28. The group $\mathbb{Z} \simeq \langle b | \emptyset \rangle =: F_1$ has a *canonical presentation* $\langle b \rangle := \langle b, \emptyset \rangle$ as a group and a *canonical presentation* $\langle \{b, \bar{b}\}, \mathcal{R}_b = \{(\bar{b}b, 1), (b\bar{b}, 1)\} \rangle^+$ as a monoid. If $w = b\bar{b}, w' = \bar{b}b$ then $ww' = b\bar{b}^2b = 1$.

Definition 1.2.29 (Word problem for monoids)**.** Given a presentation $(\Sigma, \mathcal{R})$ of the monoid M, its word problem consists to answer to the following question:

$$\textit{Given } v, w \in \Sigma^* \textit{ do we have } v \leftrightarrow^*_{\mathcal{R}} w \textit{?}$$

The word problem can be defined in the same way for monoids and groups which admit a finite presentation.

By the Proposition 1.2.14 we are able to prove the following

Theorem 1.2.30 (Post-Markov ([75], [63],(1947))**.** *There exists a finite semigroup with undecidable word problem.*

Proof. It is enough to consider the monoid $M_T = \langle \Sigma, \mathcal{R} \rangle^+$ where $(\Sigma, \mathcal{R})$ is a word rewriting system coding a Turing machine T. In such a monoid, if we consider two words $w, w_\perp$ that correspond to two different configurations of T with $w_\perp$ a final state, verifying whether $w \leftrightarrow^* w_\perp$ corresponds (at least) to verify whether the computation starting form the configuration w ends. This implies to answer to an instance of the halting problem for T, which is undecidable. □

The following example is given by Tseitin in [85].

Example 1.2.31. The semigroup $(\{a, b, c, d\}, \mathcal{R})$ where $\mathcal{R}$ are the relations:

$$ac = ca, \qquad ad = da, \qquad bd = db, \qquad ce = eca, \qquad dc = edb, \qquad cca = ccae$$

has unsolvable word problem.

Due to the presence of invertible elements, the proof of this theorem can not be extended to word problem for groups. Its undecidability was proved independently in 1950' by Novikov and Boone.

Theorem 1.2.32 (Novikov-Boone)**.** *There exists a group with undecidable word problem.*

Besides the original proofs, some reformulations of them in terms of rewriting are given by Bokut' [9], Aandrea-Cohen [1] and Lafont [54]. The difference between them are highlighted in [2].

The permutation group

In this section we recall the definition of *permutations* and the algebraic structure induced by their composition. Moreover we give some additional definitions and tools useful to prove some confluence results in section 3.

Definition 1.2.33 (Permutation)**.** A *permutation* σ is a bijection of a set S into itself. There are two different notations for permutations over a finite set:

- Cauchy's *two-lines notation:* the first row is the list of elements in S with, and for each one its image below it in the second row:

$$\begin{pmatrix} x_1 & x_2 & \dots & x_n \\ \sigma(x_1) & \sigma(x_2) & \dots & \sigma(x_n) \end{pmatrix}$$

- *Disjoint cycles notation:* it is a list of disjoint lists of elements of S called *cycles* of the form

$$(x = \sigma^{n_x}(x), \sigma(x), \sigma^2(x), \dots, \sigma^k(x)).$$

  The natural integer $k \in \mathbb{N}$ is called *order of x in σ*. This notation often omits to write cycles of order 1.

A *transposition* is a permutation over a set of two elements. A *cycle* in S_n is a permutation of order n.

Definition 1.2.34 (Permutation group)**.** If $n \in \mathbb{N}$, the *permutation group over a set of n elements* S_n is the group of permutations where the operation is given by their composition $\circ$ as function.

Definition 1.2.35 (Erasing a string)**.** If $\sigma \in S_{n+1}$, we define $Er_k(\_) : S_{n+1} \to S_n$ as follows:

$$Er_k(\sigma) = \begin{cases} Er_k(\sigma)(i) = \sigma(i) & \text{if } i < k, \sigma(i) < \sigma(k) \\ Er_k(\sigma)(i) = \sigma(i) - 1 & \text{if } i < k, \sigma(i) > \sigma(k) \\ Er_k(\sigma)(i-1) = \sigma(i) & \text{if } i > k, \sigma(i) < \sigma(k) \\ Er_k(\sigma)(i-1) = \sigma(i) - 1 & \text{if } i > k, \sigma(i) < \sigma(k) \end{cases}$$

We write $Er_{\{k_1,k_2\}}(\sigma) = Er_{k_j}(Er_{k_i}(\sigma))$ with $k_i = min\{k_1, k_2\}$ and $k_j = max\{k_1, k_2\}$.

1.3 Some category theory

In this section we recall some definition in category theory in order to fix notation for the future chapters.

Definition 1.3.1. A *category* $\mathcal{C}$ is given by a class of *objects* $ob(\mathcal{C})$ and a class of *morphisms* or *arrows* $hom(\mathcal{C})$ such that:

- for any pair of objects a and b it is defined the class $hom_{\mathcal{C}}(a, b)$ of morphisms from a to b;
- for every three objects a,b and c it is defined a binary operation $\circ : hom_{\mathcal{C}}(a, b) \times hom_{\mathcal{C}}(b, c) \to hom_{\mathcal{C}}(a, c)$ called *morphisms composition*. We note the composition of $f : a \to b$ and $g : b \to c$ as $g \circ f$. This composition satisfies the following axioms
  - *associativity*: for all $f : a \to b$, $g : b \to c$ and $h : c \to d$, $h \circ (g \circ f) = (h \circ g) \circ f$
  - *identity*: any object a admits an *identity morphism* $\mathbf{id}_a : a \to a$ such that for any morphism $f : x \to a$ and every morphism $g : a \to x$, we have $\mathbf{id}_a \circ f = f$ and $g \circ \mathbf{id}_a = g$.

In some sense, a category can be seen as a structure which mix the notions of monoid, which can be seen as a category with just one object, and of partial order, which can be seen as a category with just one morphism between each pair of objects. Moreover we can also interpret a category by means of simplicial complex with objects as 0-simplices and morphisms as 1-simplices.

Definition 1.3.2. A *functor* $F : A \to B$ is a morphism between two categories A and B. It is given by a function $F_o : ob(A) \to ob(B)$ which associates to any object $a \in ob(A)$ an object $F(a) \in ob(B)$ and a function over the arrows such that for any $f : a \to a'$ in $hom_A(a, a')$ it associates an arrow $F(f) : F(a) \to F(a')$ in $hom_B(F(a), F(a'))$ such that:

- for all a object of A, $F(\mathbf{id}_a) = \mathbf{id}_{F(a)}$;

- for all morphism $f : a \to a'$, $g : a' \to a''$ in A, $F(g \circ f) = F(g) \circ F(f)$;

Definition 1.3.3 (Category isomorphism)**.** A *category isomorphism* $F : A \to B$ is an invertible functor.

Definition 1.3.4 (Natural transformation)**.** If $S, T : C \to B$ are two functors, a *natural transformation* $\tau : S \to T$ is a function which associate to any object c of C an arrow $\tau_c : S(c) \to T(c)$ of B such that for each $f : c \to c'$ in C the following diagram commutes:

$$\begin{array}{ccc} S(c) & \xrightarrow{\tau(c)} & T(c) \\ {\scriptstyle S(f)}\downarrow & & \downarrow{\scriptstyle T(f)} \\ S(c') & \xrightarrow[\tau(c')]{} & T(c') \end{array}$$

.

Monoidal categories

In this work we focus principally on a particular family of categories called *monoidal categories*. These are particular categories with an additional algebraic structure over their objects: monoidal categories are monoids in the category of categories with the cartesian products [61].

Definition 1.3.5. A *monoidal category* is a category $\mathcal{C}$ equipped with a (bi)functor $\square : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ and an object e of $\mathcal{C}$ (that can be seen as a functor $e : * \to \mathcal{C}$) together with three families of natural isomorphisms

$$\alpha = \alpha_{xyz} : (x \square y) \square z \to x \square (y \square z),$$

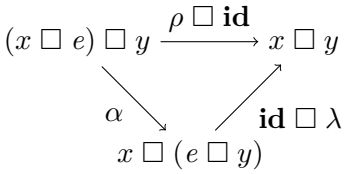
$$\lambda = \lambda_x : e \square x \to x, \qquad \rho = \rho_x : x \square e \to x,$$

such that the following diagrams commute:

- *pentagonal identity*:

$$\begin{array}{ccccc} & & (x \square (y \square z)) \square t & \xrightarrow{\alpha} & x \square ((y \square z) \square t) \\ & {\scriptstyle \alpha \square \mathbf{id}}\nearrow & & & \searrow{\scriptstyle \mathbf{id} \square \alpha} \\ ((x \square y) \square z) \square t & & & & x \square (y \square (z \square t)) \\ & {\scriptstyle \alpha}\searrow & & \nearrow{\scriptstyle \alpha} & \\ & & (x \square y) \square (z \square t) & & \end{array}$$

- *triangular identity*:

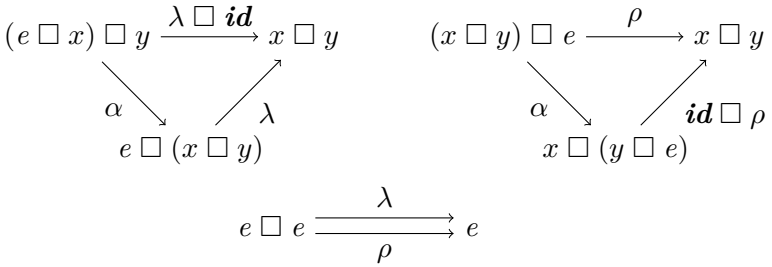


Theorem 1.3.6 (Coherence theorem for monoidal categories, Mac Lane [61])**.**

Every diagram made of α, λ, ρ in an arbitrary monoidal category $\mathcal{C}$ commutes.

We note that the pentagonal and triangular identities correspond to the confluence of two critical peaks for the rewriting system given by α, λ and ρ. In fact, there are three other critical peaks which are handled by the following lemma, which is proved in Section 2.5.

Lemma 1.3.7 (Kelly [48])**.**
The commutativity of the following diagrams is derivable from the pentagonal and the triangular identities:

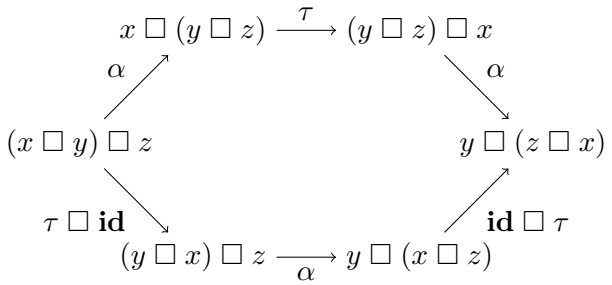


Definition 1.3.8 (Symmetric monoidal category)**.**
A *symmetric monoidal category* is a monoidal category $\mathcal{C}$ equipped with an extra family of natural isomorphisms called braidings

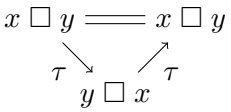
$$\tau = \tau_{xy} : x \square y \to y \square x$$

such that the following diagrams commute:

- the *hexagonal identity*:



- *involutivity* of τ:



Example 1.3.9. Any cartesian category, for instance the category of sets, has a structure of symmetric monoidal category, where $\square$ is the cartesian product $\times$ and $\tau : x \times y \to y \times x$ is the exchange of components.

In particular, we call *product category* (*PRO* for short) a monoidal category with objects in one-to-one correspondence with natural numbers and product the integer sum. A *PROs* is a symmetric *PRO*.

Definition 1.3.10. A diagram is *linear*[1] if every variable appears at most once on each vertex.

Theorem 1.3.11 (Coherence of symmetric monoidal categories, [62])**.** *In an arbitrary symmetric monoidal category, every linear diagram made of λ, ρ, α and τ is commutative.*

Remark 1.3.12. *The linearity of diagrams is necessary for the statement of this theorem since in a symmetric monoidal category, we have diagrams such as*

$$x \square x \underset{\tau}{\overset{\boldsymbol{id}}{\rightrightarrows}} x \square x$$

made of parallel arrows which are not equal in general.

In order to give a categorical higher-dimensional point of view on monoidal categories, we use the *cellular syntax*: categories are 1-*cells*, functors 2-*cells*, natural transformations 3-*cells*, diagrams are pairs of parallel 3-cells (same source, same target) and commutative diagrams are 4-*cells*. The border of a

[1] In [61], Mac Lane calls this a *formal* diagram

4-cell is the pair of parallel 3-cells that defines its diagram. In this syntax, coherence corresponds to the existence of a 4-cell whose border is a given pair of parallel 3-cells for any such pair (corresponding to linear diagrams).

This cellular syntax comes from what we can consider as the extension of rewriting system constructions: a word rewriting system can express with a simpler object a more complex monoid with infinite interaction between its elements, similarly a category can be generated by a graph.

Definition 1.3.13 (Category generated by a graph)**.** If $G = (V, A)$ is a direct (small) graph with vertex set V and arrows A, the (small) free category generated $G^* = \mathcal{C}_G$ by the graph G has object the set V and, for each $a, b \in V$, $hom_{\mathcal{C}}(a, b)$ the set of paths from a to b.

Definition 1.3.14 (n-Graph)**.** An *ω-Graph* is givem by a diagram of sets

$$G_0 \underset{t_0}{\overset{s_0}{\leftleftarrows}} G_1 \underset{t_1}{\overset{s_1}{\leftleftarrows}} \cdots \underset{t_2}{\overset{s_2}{\leftleftarrows}} G_n \underset{t_1}{\overset{s_1}{\leftleftarrows}} G_{n+1} \underset{t_2}{\overset{s_2}{\leftleftarrows}} \cdots$$

such that, for every $n \in \mathbb{N}$ the following equations hold:

$$s_n s_{n+1} = s_n t_{n+1} \qquad t_n s_{n+1} = t_n t_{n+1}.$$

The elements of G_n are called *n-cells.* An *n-Graph* is defined in a similar way by a finite sequence

$$G_0 \underset{t_0}{\overset{s_0}{\leftleftarrows}} G_1 \underset{t_1}{\overset{s_1}{\leftleftarrows}} \cdots \underset{t_2}{\overset{s_2}{\leftleftarrows}} G_n.$$

Definition 1.3.15 (2-Category)**.** The mean idea in building 2-category is to extend the notion of category to the *Hom-sets* of C such that for every $X, Y \in C$, $Hom(X, Y)$ has a category's structure. A 2-category is a 2-graph.

$$C_0 \underset{t_1}{\overset{s_1}{\leftleftarrows}} C_1 \underset{t_2}{\overset{s_2}{\leftleftarrows}} C_2$$

satisfying the following properties:

- the graph $C_0 \underset{t_1}{\overset{s_1}{\leftleftarrows}} C_1$ has a structure of category:

  $$X \mapsto X \overset{id_X}{\to} X, \qquad X \overset{f}{\to} Y \overset{g}{\to} Z \mapsto X \overset{gf}{\to} Z$$

  for all $X, Y, Z \in C_0, f, g \in C_1$;

- the graph $C_1 \underset{t_2}{\overset{s_2}{\leftleftarrows}} C_2$ has a structure of category:

$$X \xrightarrow{f} Y \mapsto X \underset{f}{\overset{f}{\downarrow \overrightarrow{id}_f}} X, \qquad X \underset{f'}{\overset{f}{\underset{\lambda}{\overset{\mu}{\rightrightarrows}}}} Y \mapsto X \underset{f'}{\overset{f}{\downarrow \overrightarrow{\lambda\mu}}} Y$$

  for all $f, f' \in C_1,\ \lambda, \mu \in C_2$;

- the graph $C_0 \underset{t_0t_1}{\overset{s_0s_1}{\leftleftarrows}} C_2$ has a structure of category:

$$X \mapsto X \underset{id_X}{\overset{id_X}{\downarrow \overrightarrow{Id}_X}} X, \qquad X \underset{f'}{\overset{f}{\downarrow \overrightarrow{\lambda}}} Y \underset{g'}{\overset{g'}{\downarrow \overrightarrow{\mu}}} Z \mapsto X \underset{g'f'}{\overset{gf}{\downarrow \overrightarrow{\mu * \lambda}}} Z$$

  for all $X, Y \in C_0, \lambda, \mu \in C_2$;

- and the compatibility conditions:

$$Id_X = id_{id_X}, \qquad \mu'\mu * \lambda'\lambda = (\mu' * \lambda')(\mu * \lambda)$$

  for all $X \in C_0$, and $X \underset{\rightarrow}{\overset{\rightarrow}{\underset{\downarrow\lambda'}{\overset{\downarrow\lambda}{\rightrightarrows}}}} Y \underset{\rightarrow}{\overset{\rightarrow}{\underset{\downarrow\mu'}{\overset{\downarrow\mu}{\rightrightarrows}}}} Z$.

In particular, this last condition explixitally means that there is an exchange rule $(id_{g'} * \lambda)(\mu * id_f) = \mu * \lambda = (\mu * id_{f'})(id_g * \lambda)$ for all $X \underset{f}{\overset{f'}{\downarrow \overrightarrow{\lambda}}} Y \underset{g}{\overset{g'}{\downarrow \overrightarrow{\mu}}} Z$.

Definition 1.3.16 (ω-Category)**.** If C is an ω-graph we define, for $0 \leq i < j$, a graph C_{ij}:

$$C_i \underset{t_{ij}}{\overset{s_{ij}}{\leftleftarrows}} C_j$$

with $s_{ij} = s_i s_{i+1} \dots s_{j-1}$ and $t_{ij} = t_i t_{i+1} \dots t_{j-1}$ and, for $0 \leq i < j < k$, a 2-graph C_{ijk}:

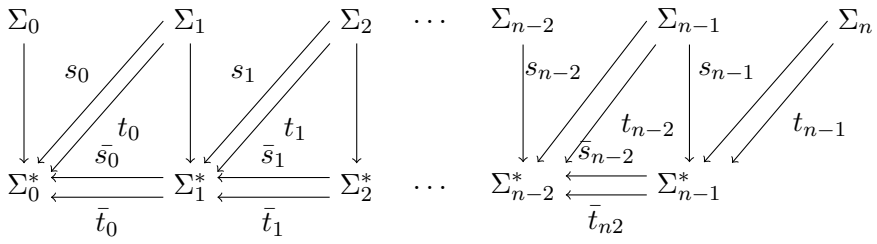
$$C_i \underset{t_{ij}}{\overset{s_{ij}}{\leftleftarrows}} C_j \underset{t_{jk}}{\overset{s_{jk}}{\leftleftarrows}} C_k$$

An ω-graph has the structure of ω-category if every C_{ijk} is a 2-category for all $0 \leq i < j < k$ (with the above defined compositions and identities). We denote by $*_k$ the sequential composition of k-cells.

Polygraphs [12] (or *computads* [80]) are the presentation of higher dimension categories by means of generator and relations.

Definition 1.3.17 (Polygraph)**.** The category $\mathbf{Pol}_0$ of 0-polygraph is the category **Set**, a *0-cell* of a 0-polygraph is one of its elements.

An *n-polygraph* $\Sigma = (\Sigma_{n-1}, \Sigma)$ is given by an $(n-1)$-polygraph Σ_{n-1} and a family Σ of n-spheres on the free $(n-1)$-category generated by Σ_{n-1}. An n-cell is an element of Σ_n and, for all $k < n$, a k-cell of Σ is a k-cell of the polygraph Σ_{n-1}. An n-polygraph is finite it has a finite number of cells in every dimension.



An (n,k)-polygraph is an n-polygraph where the cells of dimension greater than k are invertible, that is for each p-cell f with $k < p \leq n$ there exist a p-cell f^{-1} such that

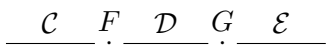
$$f *_p f^{-1} = \mathbf{id}_{s_{p-1}(f)} \qquad \text{and} \qquad f^{-1} *_p f = \mathbf{id}_{t_{p-1}(f)}$$

If we consider polygraphs as presentations for higher-dimensional categories, the relation between an (n,k)-polygraph and the relative n-polygraph reminds the one between a group and a monoid presentation given by the same rewriting system. Hence, even if they are given by the same set of rules, the presentation of the group implicitly considers a rewriting system enriched by an additional set of generators (a formal inverses for each generator) and an additional set of relations (the ones which relate each generator to its inverse). In the same way, the (n,k)-polygraph can be consider as its relative n-polygraph enriched by a formal inverse for each p-cell with $k < p \leq n$ and the relative q-cells with $p < q$ representing the equalities with identities. For a more formal construction of (n,k)-polygraph we remind [37].

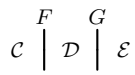
1.4 String diagrams

String diagrams (or simply *diagrams*) are a way of notating natural transformations and functors (for a gentle introduction to the topic, refer to Chapter 2). The main idea of this representation is to graphic a functor $F : \mathcal{C} \to \mathcal{D}$ as

and a sequential composition $\mathcal{C} \overset{F}{\to} \mathcal{D} \overset{G}{\to} \mathcal{E}$ as

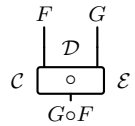


Expanding this notation to a bidimensional representation, sequential composition will be represent as



i.e. the objects (in this case categories in **Cat**) are connected components of plane and functors their border. A natural transformation ϕ between two functon $F, G : \mathcal{C} \to \mathcal{D}$ will be plotted as follow

while the natural transformation $\circ : \mathcal{C} \overset{F}{\to} \mathcal{D} \overset{G}{\to} \mathcal{E} \Rightarrow \mathcal{C} \overset{G \circ F}{\to} \mathcal{E}$:



In general, string diagrams can be used to represent 2-arrows in any 2-category: connected components of the plane are labeled by objects, the *strings* (which separate plane portions) represents arrows, and *gates* (which connect strings) represents 2-arrows.

Example 1.4.1. String diagrams allow to easy represent natural transformations of monads.

Definition 1.4.2 (Monad)**.** A monad $\mathcal{C}$ is a category equipped with an endofunctor $T : \mathcal{C} \to \mathcal{C}$ and two natural transformation ϵ and η such that:

- $\eta : id_{\mathcal{C}} \to T$
- $\mu : T \circ T \to T$

satisfying the coherence conditions

$$\mu \circ T\mu = \mu \circ \mu T \qquad \text{and} \qquad \mu \circ T\eta = \mu \circ \eta T = id_T$$

that is the following diagrams commute

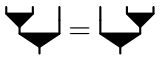
$$\begin{array}{ccc} T^3 & \xrightarrow{\mu T} & T^2 \\ {\scriptstyle \mu T}\downarrow & & \downarrow{\scriptstyle \mu} \\ T & \xrightarrow{\mu} & T^2 \end{array} \qquad \begin{array}{ccccc} T & \xrightarrow{T\eta} & T^2 & \xleftarrow{T\eta} & T \\ & {\scriptstyle id_T}\searrow & \downarrow{\scriptstyle \mu} & \swarrow{\scriptstyle id_T} & \\ & & T & & \end{array} .$$

Since the unique functor T we'll study is an endo-functor on $\mathcal{C}$ we can omit to label the portion of plane (always labeled by $\mathcal{C}$) and the strings (which suppose to be all labeled by T). The two natural transformation will be represented by

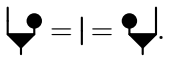
$\mu =$ $\qquad \eta =$

The coherence condition of η and μ are represented by the following diagrammatic equations:

- $\mu \circ T\mu = \mu \circ \mu T$:



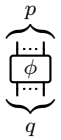
- $\mu \circ T\eta = id_T = \mu \circ \eta T$:



Monochrome String Diagrams

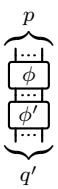
We now recall some basic notions in string diagram rewriting by considering the *monochrome string diagrams* settings, where there are no labels on backgrounds or strings. For an introduction to string diagrams, see J. Baez's notes [6].

Given p and q natural numbers, a diagram $\phi : p \Rightarrow q$ with p *inputs* and q *outputs* is pictured as follows:



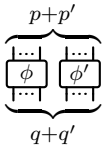
Diagrams may be composed in two different ways. If $\phi : p \Rightarrow q$ and $\phi' : p' \Rightarrow q'$ are diagrams, we define:

- *sequential* composition: if $q = p'$, the diagram $\phi' \circ \phi : p \Rightarrow q'$ corresponds to usual composition of maps:



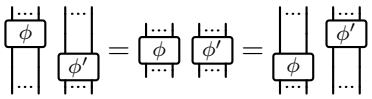
This composition is associative with unit $\mathbf{id}_p : p \Rightarrow p$ for each $p \in \mathbb{N}$. In other words, we have $\phi \circ \mathbf{id}_p = \phi = \mathbf{id}_q \circ \phi$. The *identity diagram* id_p is pictured as follows:

- *parallel* composition: the diagram $\phi * \phi' : p + p' \Rightarrow q + q'$ is pictured as follows:



This composition is associative with unit $\mathbf{id}_0 : 0 \Rightarrow 0$. In other words, we have $\mathbf{id}_0 * \phi = \phi = \phi * \mathbf{id}_0$. This $\mathbf{id}_0$ is called the *empty diagram*.

Our two compositions satisfy the *interchange rule*: if $\phi : p \Rightarrow q$ and $\phi' : p' \Rightarrow q'$, so $(\mathbf{id}_q * \phi') \circ (\phi * \mathbf{id}_{p'}) = \phi * \phi' = (\phi * \mathbf{id}_{q'}) \circ (\mathbf{id}_p * \phi')$ that corresponds to the following picture:



Monochrome string diagrams can be interpreted as morphisms in a **PRO**, that is a strict monoidal category whose objects are natural numbers and whose product on objects is addition. To be coherent with the cellular notation we use in next sections, diagrams represent 2-arrows in the $\mathbf{2} - \mathbf{PRO}$ obtained by suspension of a regular **PRO** (see [35]).

Definition 1.4.3 (Signature)**.** A *signature* $\mathcal{S}$ is a finite set of *atomic diagrams* (or *gates type*). Given a signature, a diagram $\phi : p \Rightarrow q$ is a morphism in the **PRO** $\mathcal{S}^*$ freely generated by $\mathcal{S}$, i.e. by the two compositions and identities. A *gate* is an occurrence of an atomic diagram, we note $g : \alpha$ if g is an occurrence of $\alpha \in \mathcal{S}$.

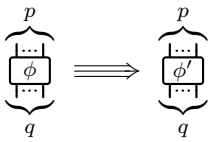
Definition 1.4.4. We say that ϕ is a *subdiagram* of ϕ' whenever there exist $\psi_u, \psi_d \in \mathcal{S}^*$ and $k, k' \in \mathbb{N}$ such that $\phi' = \psi_d \circ (\mathbf{id}_\Gamma * \phi * \mathbf{id}_\Delta) \circ \psi_u$.

Notation. Given $\phi \in \mathcal{S}^*$ and $\mathcal{S}' \subseteq \mathcal{S}$, we write $|\phi|_{\mathcal{S}'}$ the number of gates in ϕ with gate type $\alpha \in \mathcal{S}'$.

Definition 1.4.5. We call *horizontal* a diagram ϕ generated by parallel composition (and identities) only in $\mathcal{S}^*$. It is *elementary* if $|\phi|_{\mathcal{S}} = 1$.

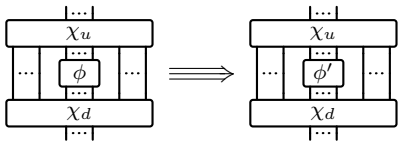
Diagram rewriting

Definition 1.4.6 (Diagram Rewriting System)**.** A *diagram rewriting system* is a couple $(\mathcal{S}, \mathcal{R})$ given by a signature $\mathcal{S}$ and a set $\mathcal{R}$ of rewriting rules of the form



where $\phi, \phi' : p \Rightarrow q$ are diagrams in $\mathcal{S}^*$.

Definition 1.4.7. We allow each rewriting rules under any context, that is, if $\phi \Rrightarrow \phi'$ in $\mathcal{R}$ then, for every $\chi_u, \chi_d \in \mathcal{S}^*$,



.

We say that ψ *reduces*, or *rewrites*, to ψ' (denoted $\psi \overset{*}{\Rrightarrow} \psi'$) if there is a *rewriting sequence* $P : \psi = \psi_0 \Rrightarrow \psi_1 \Rrightarrow \ldots \Rrightarrow \psi_n = \psi'$.

We here recall some classical notions in rewriting:

- A diagram ϕ is *irreducible* if there is no ϕ' such that $\phi \Rrightarrow \phi'$;
- A rewriting system *terminates* if there is no infinite rewriting sequence;
- A rewriting system is *confluent* if for all ϕ_1, ϕ_2 and ϕ such that $\phi \Rrightarrow \phi_1$ and $\phi \Rrightarrow \phi_2$, there exists ϕ' such that $\phi_1 \overset{*}{\Rrightarrow} \phi'$ and $\phi_2 \overset{*}{\Rrightarrow} \phi'$;
- A rewriting system is *convergent* if both properties hold.

Some observation on notations

In this thesis we work on diagram rewriting systems which can also be presented polygraphs. For this reason we use both notation and definitions: if $(\mathcal{S}, \mathcal{R})$ is a diagram rewriting system and $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2, \Sigma_3)$ a 3-polygraph we establish the following correspondences:

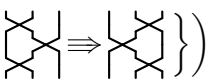
- diagrams backgrounds are labeled by 0-cells in Σ_0;
- diagrams strings are labeled by 1-cells in Σ_1;
- diagrams gates are labeled by 2-cells in Σ_2, this is, there is a one-to-one correspondence between $\mathcal{S}$ (the set of gate type) and Σ_2;
- a diagram ϕ is a (composed) 2-cell of Σ, we note $\phi \in \Sigma$;

- rewriting rules in $\mathcal{R}$ are the elementary 3-cells in Σ_3, the existence of a rewriting path corresponds to the existence of a composed 3-cell in Σ_3;
- a solvable critical pair give correspond to a pair of parallel (i.e. same source, same target) 3-cells and so we can define a 4-cell in a proper 4-polygraph containing Σ for such pair.
- If Σ is an n-polygraph, we note $\langle\Sigma\rangle$ the set of $(n-1)$-cells quotiented by the set of n-cells.

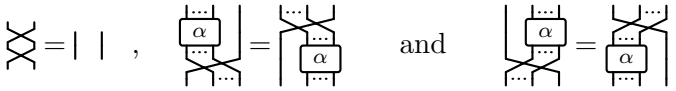
Twisting Polygraph

In this section we introduce a notion of polygraph which generalizes polygraphic presentations of symmetric monoidal categories.

Definition 1.4.8 (Symmetric polygraph)**.** We call the *polygraph of permutation* the following monochrome 3-polygraph:

$\mathfrak{S} = \Big(\Sigma_0 = \{\square\}, \Sigma_1 = \{|\}, \Sigma_2 = \{\times\}, \Sigma_3 = \Big\{$  $\Big\}\Big)$

We call *symmetric* a 3-polygraph Σ with one 0-cell, one 1-cell (i.e. $\Sigma_1 = \{|\}$), containing one 2-cell $\times \in \Sigma_2$ and such that the following holds



in the 2-category Σ^*.

Theorem 1.4.9 (Convergence of $\mathfrak{S}$)**.** *The polygraph $\mathfrak{S}$ is convergent.*

Proof. As in [52], in order to prove termination we interprete every diagram $\phi : n \to m \in \mathfrak{S}^*$ with a monotone function $[\phi] : \mathbb{N}^n \to \mathbb{N}^m$. These have a well founded order induced by product order on $\mathbb{N}^p$:

$$f, g : \mathbb{N}^{*p} \to \mathbb{N}^{*p} \text{ then } f \geq g \text{ iff } f(\bar{x}) \geq g(\bar{x}) \text{ for all } \bar{x} \in \mathbb{N}^{*p}.$$

We interprete the gate $\times$ by the function $[\times](x, y) \to (y, x + y)$. This allow as to associate to any 3-cell $\phi \Rrightarrow \psi$ two monotone maps $[\phi]$ and $[\psi]$ such that $[\phi] > [\psi]$:

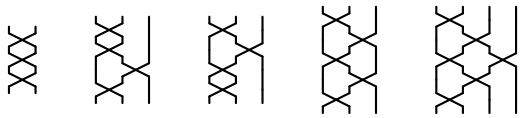
$$\Big[\Big]\Big](x, y) = (2x + y, x + y) > (x, y) = \Big[|\ |\Big](x, y),$$

$$\Big[\Big]\Big](x, y, z) = (2x + y + z, x + y, x) > (x + y + z, x + y, x) = \Big[\Big]\Big](x, y, z)$$

By the compatibility of the order with sequential and parallel composition, this suffice to prove that, for any couple of diagrams, $[\phi] > [\psi]$ holds if $\phi \to^* \psi$.

Since there exists no infinite decreasing suite of monotone maps on positive integers, infinite reduction paths can not exist.

In order to prove convergence, by Prop. 1.2.8, it suffices to check the confluence of the following 5 critical peaks [2]:



□

Each diagram in $\mathfrak{S}$ can be interpreted as a permutation in the *group of permutations over n elements* S_n with product $\circ$ defined as their function composition. On the other hand, each $\sigma \in S_n$ corresponds to some diagrams in $\mathfrak{S}$. In particular, we interpret the diagram $\mathbf{id}_{k-1} * \times * \mathbf{id}_{n-(k+1)} : n \to n$ as the transposition $(k, k+1) \in S_n$.

Notation. We call *left ladder over n elements* a diagram of the form

$$Lad_1^l = | : 1 \Rightarrow 1 \quad \text{and} \quad Lad_n^l = \text{} : n \Rightarrow n,$$

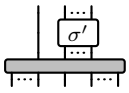
corresponding to the permutation $Lad_n^l = (1, n, n-1, \ldots, 2) \in S_n$. In a similar way, a *right ladder over n elements*

$$Lad_1^r = | : 1 \Rightarrow 1 \quad \text{and} \quad Lad_n^r = \text{} : n \Rightarrow n$$

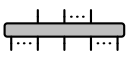
corresponds to the permutation $Lad_n^r = (n, 1, 2, \ldots, n-1) \in S_n$.

Proposition 1.4.10. *For any permutation $\sigma \in S_n$ there is a unique diagram in normal form $\sigma : n \Rightarrow n \in \mathfrak{S}$ corresponding to σ. We call it the* canonical diagram of σ.

Proof. We define $\mathfrak{S}_1 = \{|\}$ and $\mathfrak{S}_{n+1}$ the set of diagrams in $\mathfrak{S}$ of the form:



$= \sigma : n+1 \Rightarrow n+1$

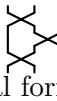
with  $\in \mathfrak{S}_n$ and  =  $\left|\cdots\right| = Lad_k^l * id_{(n+1-k)}$. We have $|\mathfrak{S}_n| = n!$ since $|\mathfrak{S}_1| = 1$ and $|\mathfrak{S}_{n+1}| = (n+1)|\mathfrak{S}_n|$ on account of $n+1 = |\{Lad_k^l\}_{1 \le k \le n+1}| = |\{Lad_k^l * \mathbf{id}_{(n+1-k)}\}_{1 \le k \le n+1}|$.

To exhibit a one-to-one correspondence between S_{n+1} and $\mathfrak{S}_{n+1}$, for any $\sigma \in S_{n+1}$ we define $Er(\sigma) \in S_n$ the permutation

$$Er(\sigma) = \begin{cases} i \to \sigma(i+1) & if \quad \sigma(i+1) < \sigma(1) \\ i \to \sigma(i+1) + 1 & if \quad \sigma(1) < \sigma(i+1) \end{cases}.$$

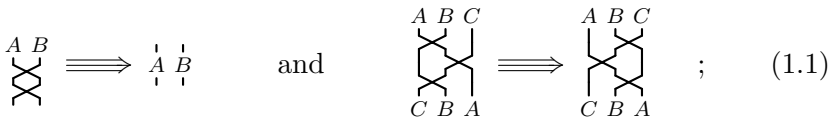
[2] see Section 2.4 for a proper definition of critical peak in string diagram rewriting

and $\sigma = (Lad^l_k * \mathbf{id}_{(n+1-\sigma(1))}) \circ (\mathbf{id}_1 * Er(\sigma))$.

Any element in $\mathfrak{S}_n$ contains no subdiagram of the form  nor  meaning that it is irreducible and so, by the confluence of $\mathfrak{S}$, in normal form. □

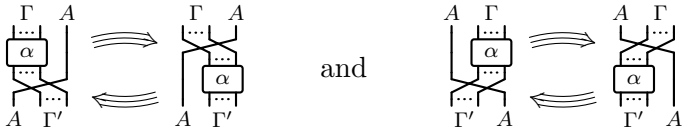
Definition 1.4.11 (Twisting polygraph)**.** A *twisting polygraph* is a 3-polygraph Σ with one 0-cell equipped with a set $T_\Sigma \subseteq \Sigma_1$ called *twisting family* such that for each $A, B \in T_\Sigma$ there is a *twisting operator* $T_{A,B} : A * B \Rightarrow B * A \in \Sigma_2$ and Σ_3 includes the following families $T_\mathcal{R}$ of *twisting relations*:

- For all $A, B, C \in T_\Sigma$



; (1.1)

- For all $\alpha : \Gamma \to \Gamma' \in \Sigma_2$, $\alpha \neq T_{A,B}$ with $\Gamma, \Gamma' \in T^*_\Sigma$, $A \in T_\Sigma$, at least one of the two possible orientation of the following rewriting rules is in Σ_3.



. (1.2)

Moreover, if ϕ, ψ are *twisting diagrams* (i.e. diagrams made only of twisting operators) $\phi \overset{*}{\underset{\mathcal{R}_\Sigma}{\Rrightarrow}} \psi$ iff $\phi \overset{*}{\underset{\mathcal{R}_T}{\Rrightarrow}} \psi$ where $\mathcal{R}_T$ is the set given by rewriting rules of (1.1). A *total-twisting polygraphy* is a twisting polygraph with $T_\Sigma = \Sigma_1$.

The idea behind twisting polygraphs is to present diagram rewriting systems where, in equivalence classes modulo rewriting, the crossings of strings labeled by the twisting family are not taken into account. In fact, the family of relations (1.1) says that these crossings are involutive and satisfy Yang-Baxter equation [43] for braidings, while relations in (1.2) allow gates to "cross" a string in case of fitting labels.

We interpret a twisting diagram $\sigma : \Gamma \Rightarrow \sigma(\Gamma)$ as the permutations in $S_{|\Gamma|}$ acting over the order of occurrence of 1-cells in the word $\Gamma \in T^*_\Sigma$. For this reason, as in $\mathfrak{S}$, we define left ladders, right ladders and the standard diagrams $\hat{\phi}^\Gamma_\sigma : \Gamma \to \sigma(\Gamma)$ (or simply σ) with source and target in T^*_Σ. In conformity with the twisting polygraph restrictions over Σ_3, we can prove the uniqueness of σ as in Proposition 1.4.10.

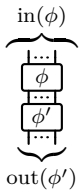
Chapter 2

Diagrammatic 2-dimensional syntax

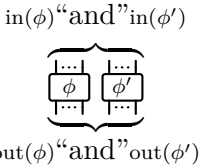
"I have always considered drawing not as an exercise of particular dexterity [...] but as a means deliberately simplified so as to give simplicity and spontaneity to the expression, which should speak without clumsiness, directly to the mind of the spectator."

[H. Matisse, Le Point "Revue artistique et littéraire" (1939)]

As seen in Section1.4, string diagrams are a way to denote certain transformations. Their syntax includes a sequential and a a parallel composition, namely:

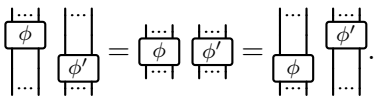


(a) Sequential composition



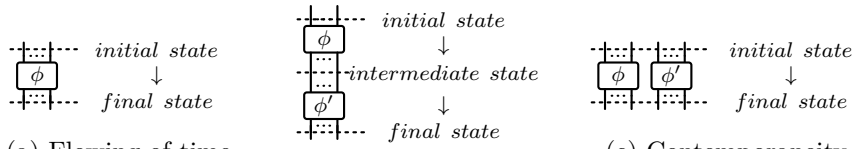
(b) Parallel composition

These compositions interact by the interchange rule



In order to give solid intuitions in to these three fundamental concepts of this syntax, we borrow the concepts of sequentiality and contemporaneity form relativity theory. If we consider the flow of time like an horizontal line sliding down-wise over diagrams, a diagram can be considered as a transformation

operating on some observables of a system represented by its inputs and its outputs (Fig. 2.2a).



(a) Flowing of time (b) Sequentiality (c) Contemporaneity

Figure 2.2

Hence, a sequential composition is a succession of transformations where, at each step, all observables are taken into account in order to perform some modifications. Each observable in the final state of the system (output) depends on all transformations on the initial observables (inputs). Whereas, a parallel composition should be interpreted as multiple transformations modifying independents observables.

Moreover, this independence has to be considered as "contemporaneity" with respect to our relativistic point of view: two events are considered contemporary if they can be observed at the same time with respect to a frame of reference. For example, any observation we can measure on the event horizon should be considered contemporary even if we are measuring an event happened some seconds after the big bang. Similarly, by interchange rule, the relative position of parallel diagrams does not matter as long as we consider the final outputs. This means that we can bend the sliding horizontal line of time until we do not need to consider simultaneously some observables, either to perform new transformations (sequential composition) or to record the final state (outputs).

It is not a coincidence that string diagrams appeared for the first time in Richard Feynman's notes [25] as syntax for describing the behavior of subatomic particles in *quantum field theory*. Without entering into detail, the idea behind Feynman diagrams is to represent particles as strings and their interactions by points in a two dimensional space there the sliding line of time swipe from left to right [30]. As Feynman recalled later [65]:

"Each diagram signified a mathematical expression [...] I would like to make little pictures of all that was going on; these were physical pictures involving the mathematical terms."

Even if parallel composition is represented vertically and sequential composition horizontally, we keep the interchange rule. This allows to change mutual position of points and consequently to represent with the same diagram many equivalent equations, which describe the same configuration of interactions observed by different temporal point of view. On the other hand, in Feynman diagrams we allow string crossing and string turn-backs. On the ontological

point of view, they are more a graphical (by means of graph) syntax than a diagrammatic one.

The introduction of the *category theory* by Eilenberg and Mac Lane (1945, [24]) and in particular the notion of *monoidal category* (1963, [62]), gives some new mathematical backgrounds also to quantum physics. In particular, in Roger Penrose paper [73] from 1971 we have a formalism for morphisms in a particular monoidal category used to represent subatomic particles interaction. By the way, this formalism was inspired from Feynman diagrams more than category theory syntax.

Finally in 1991 in their paper on braided monoidal categories [44], André Joyal and Ross Street, formalized the syntax of string diagram presented in the previous chapter. Once defined, this higher dimensional syntax was progressively taken into account in representation of *algebraic theories* due to their categorical interpretation given by Lawvere [57].

In this chapter we want to formalize some notions of this diagrammatic 2-dimensional syntax. In particular we give formal definitions to some intuition we have in using the graphical representation of string diagrams.

Then we underline some particular features of this language which do not appear in the "classical" mathematic language where terms are, in some sense, one dimensional. In particular, as we show, some of them give new paradigms of interaction between terms and subterms which cause the weakening of some finitary properties such as ruling out an extension of the Squier theorem for monoids based on word rewriting ([78], [79]) in the more general framework of string diagram rewriting. Moreover we show how term equivalence and to recognize subterms become non-trivial in this paradigm.

We conclude the chapter with a proof of Mac Lane's coherence for symmetric monoidal categories based on diagram rewriting only. We detail the proof of the confluence of the 68 critical peaks of this system given in [52]. Then, using the same homotopical methods presented by Guiraud and Malbos in [36] for their non-constuctive proof of the theorem, we give an homotopy base for this rewriting system and we use it in order to give a strategy of proof for the theorem.

In order to have a more manageable syntax for the presentation of our new notions, in this chapter we work in monochrome string diagrams paradigm, that is, with neither background nor string labels.

2.1 Formal diagrams

In this section we give an alternative construction of string diagrams as purely 2-dimensional words we call *formal diagrams*. In this formalism, string diagrams are equivalent classes of formal diagrams under a set of equivalences corresponding to interchange rules. We give a normalization procedure for

formal diagrams in order to have a normal form showing how we can use this procedure in order to decide formal diagrams equivalence.

Definition 2.1.1 (Diagram signature)**.** A *diagram signature* (or simply *signature*) is a set $\mathcal{S}$ equipped with two maps $\mathrm{in}, \mathrm{out} : \mathcal{S} \to \mathbb{N}$. An element $\alpha \in \mathcal{S}$ is called a *gate type*, we denote $\alpha : p \to q$ when $p = \mathrm{in}(\alpha)$ and $q = \mathrm{out}(\alpha)$.

Definition 2.1.2 (Parallel Diagram)**.** A *parallel diagram* over the signature $\mathcal{S}$ is a list

$$d_1 * \cdots * d_n$$

where $d_i \in \mathcal{S}_{\mathbf{id}} = \mathcal{S} \cup \{\mathbf{id}_1\}$. Let $H_{\mathcal{S}}$ be the set of parallel diagrams over $\mathcal{S}$ and we say that the symbol $*$ represents the *parallel composition* of parallel diagrams.

We use the symbol $\mathbf{id}_0$ for the empty list and the symbol $\mathbf{id}_k$ for the list $\underbrace{\mathbf{id}_1 * \cdots * \mathbf{id}_1}_{k}$. A *gate g of type α* (noted $g : \alpha$) is an occurrence of a symbol $\alpha \in \mathcal{S}$ while an *atomic diagram (in $\mathcal{S}$)* is a gate of $\mathcal{S}$ or a list $\mathbf{id}_k$ for some $k \in \mathbb{N}$.

We extend the definitions of the two maps $\mathrm{in}, \mathrm{out}$ to any element $\mathfrak{h} = d_1 * \cdots * d_n \in H_{\mathcal{S}}$ as follows:

$$\mathrm{in}(\mathfrak{h}) = \sum_{i=1}^{n} \mathrm{in}(d_i) \qquad\qquad \mathrm{out}(\mathfrak{h}) = \sum_{i=1}^{n} \mathrm{out}(d_i)$$

where, by definition, $\mathrm{in}(\mathbf{id}_1) = \mathrm{out}(\mathbf{id}_1) = 1$.

With the help of a second (partial) composition depending on these two functions, we are able to give a notion of *formal diagram*: a particular kind of 2-dimensional word. The *sequential composition* $\mathfrak{h}_1 * \mathfrak{h}_2$ is defined if $out(\mathfrak{h}_1) = in(\mathfrak{h}_2)$. Whenever it is defined, we consider it associative.

Definition 2.1.3 (Formal Diagram)**.** A *formal diagram* (over the signature $\mathcal{S}$) is a (well-defined) *sequential composition* $F = \mathfrak{h}_1 \circ \cdots \circ \mathfrak{h}_k$ of parallel diagrams. We denote by $Form_{\mathcal{S}}$ the set of formal diagrams over $\mathcal{S}$.

To extend the parallel composition $*$ of parallel diagrams to formal diagrams, we want it to be compatible with sequential definition, that is, to satisfy the following law:

$$(\mathfrak{h}_1 \circ \mathfrak{h}_2) * (\mathfrak{h}'_1 \circ \mathfrak{h}'_2) = (\mathfrak{h}_1 * \mathfrak{h}'_1) \circ (\mathfrak{h}_2 * \mathfrak{h}'_2).$$

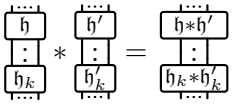
For this purpose, we define the function *height of a formal diagram*

$$\begin{array}{rccc} \mathrm{hgt} : & Form_{\mathcal{S}} & \to & \mathbb{N} \\ & F = \mathfrak{h}_1 \circ \cdots \circ \mathfrak{h}_k & \to & \mathrm{hgt}(F) = k \end{array}$$

in order to give the following definition.

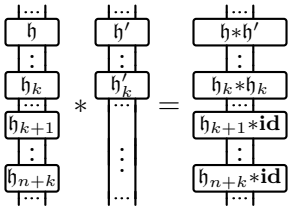
Definition 2.1.4 (Parallel composition extended to formal diagrams)**.** Given two formal diagrams $F = \mathfrak{h}_1 \circ \cdots \circ \mathfrak{h}_k, F' = \mathfrak{h}'_1 \circ \cdots \circ \mathfrak{h}'_{k'} \in H_{\mathcal{S}}$, their *parallel composition* $F * F'$ is defined:

- if $\mathrm{hgt}(F) = \mathrm{hgt}(F') = k$ then $F * F' = (\mathfrak{h} * \mathfrak{h}') \circ \cdots \circ (\mathfrak{h}_k * \mathfrak{h}'_k)$;



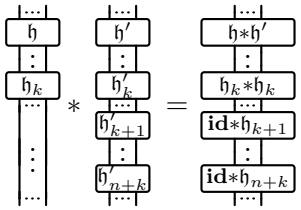
- if $\mathrm{hgt}(F) = k$ and $\mathrm{hgt}(F') = k + n$ then

  $F * F' = (\mathfrak{h}_1 * \mathfrak{h}'_1) \circ \cdots \circ (\mathfrak{h}_k * \mathfrak{h}'_k) \circ (\mathbf{id}_{\mathrm{out}(F)} * \mathfrak{h}'_{k+1}) \circ \cdots \circ (\mathbf{id}_{\mathrm{out}(F)} * \mathfrak{h}'_{k+n})$;



- if $\mathrm{hgt}(F') = k'$ and $\mathrm{hgt}(F) = k' + n$ then

  $F * F' = (\mathfrak{h}_1 * \mathfrak{h}'_1) \circ \cdots \circ (\mathfrak{h}_k * \mathfrak{h}'_k) \circ (\mathfrak{h}_{k'+1} * \mathbf{id}_{\mathrm{out}(F')}) \circ \cdots \circ (\mathfrak{h}_{k'+n} * \mathbf{id}_{\mathrm{out}(F')})$.



This coincides with the parallel composition in $H_{\mathcal{S}}$ since every parallel diagram can be seen as a formal diagram of height 1.

Definition 2.1.5. If $F \in Form$, we denote by $|F|$ the number of gates of F. If $\mathcal{S}' \subseteq \mathcal{S}$, we denote by $|F|_{\mathcal{S}'}$ the number of gates in F of type $\alpha \in \mathcal{S}'$.

To proceed, we need to introduce some notion on formal diagrams.

Definition 2.1.6 (Formal subdiagram)**.** If $F \in Form_{\mathcal{S}}$ is a formal diagram, we say that F' is a *formal subdiagram* of F (noted $F' \subseteq F$) if there exist $F_u, F_d, F_l, F_r \in Form_{\mathcal{S}}$ (the letters u, d, l, r stand for up, down, left and right) such that:

$$F = F_u \circ (F_l * F' * F_r) \circ F_d.$$

We say that a formal subdiagram is horizontal if $F_u = F_d = \mathbf{id}_0$ and vertical if $F_l = F_r = \mathbf{id}_0$.

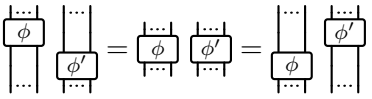
Diagrams and formal diagrams

In order to recover the semantics of diagrams, we have to quotient the set of formal diagrams by the equivalence relation $\simeq_{exc}$ generated by

$$(\alpha * \mathbf{id}_{in(\beta)}) \circ (\mathbf{id}_{out(\alpha)} * \beta) \simeq_{exc} \alpha * \beta \simeq_{exc} (\mathbf{id}_{in(\alpha)} * \beta) \circ (\alpha * \mathbf{id}_{out(\beta)})$$

in a compatible way with the two compositions $\circ$ and $*$, that is, if $F \simeq_{exc} F'$, for all $G \in Form_{\mathcal{S}}$ then $F * G \simeq_{exc} F' * G$, $G * F \simeq_{exc} G * F'$ and $F \circ G \simeq_{exc} F' \circ G$, $G \circ F \simeq_{exc} G \circ F'$ (when they are defined). Moreover, by compatibility with units, if $F \in Form_{\mathcal{S}}$ then $F \circ \mathbf{id}_{\mathrm{in}(F)} = F = \mathbf{id}_{\mathrm{out}(F)} \circ F$

This relation is called the *interchange rule* due to its diagrammatic representation



Remark 2.1.7. *The interchange rule can be derived by the neutrality of* $\boldsymbol{id}_k$*'s for the sequential composition. In fact if we assume* $\boldsymbol{id}_{\mathrm{in}(\mathfrak{h})} \circ \mathfrak{h} = \mathfrak{h} = \mathfrak{h} \circ \boldsymbol{id}_{\mathrm{out}(\mathfrak{h})}$*, then* $(\boldsymbol{id}_{\mathrm{in}(\alpha)} * \beta) \circ (\alpha * \boldsymbol{id}_{\mathrm{out}(\beta)}) = (\boldsymbol{id}_{\mathrm{in}(\alpha)} \circ \alpha) * \beta = \alpha * \beta = \alpha * (\boldsymbol{id}_{\mathrm{in}(\beta)} \circ \beta) = (\alpha * \boldsymbol{id}_{\mathrm{in}(\beta)}) \circ (\boldsymbol{id}_{\mathrm{out}(\alpha)} * \beta)$ *by definition of parallel composition.*

The equivalence relation, $\simeq_{exc}$ leads to a decision problem on formal diagrams:

Definition 2.1.8 (Diagrams equivalence problem)**.** The *equivalence problem for formal diagrams* consists to answer the following question:

Given two formal diagrams F and F', are they equivalent modulo $\simeq_{exc}$?

In order to answer to this question we define a family of formal diagrams and we show it is in one-to-one correspondence with the equivalence classes in some *non-degenerate* signatures.

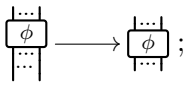
Definition 2.1.9 (Degenerate signature)**.** A signature $\mathcal{S}$ is *degenerate* if there $\alpha \in \mathcal{S}$ with $\mathrm{in}(\alpha) = 0$ and $\beta \in \mathcal{S}$ with $\mathrm{out}(\beta) = 0$.

Definition 2.1.10 (Squeezed form)**.** A *squeezed form* is a formal diagram F containing no horizontal formal subdiagram of the form $\mathfrak{h} \circ \mathbf{id}_{out(\mathfrak{h})}$ or $(\mathfrak{h}'_x * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}''_x) \circ (\mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1})$ with $\mathrm{out}(\mathfrak{h}'_x) = \mathrm{in}(\mathfrak{h}_{x+1})'$ and $\mathrm{out}(\mathfrak{h}'_x) = \mathrm{in}(\mathfrak{h}''_{x+1})$.

The following procedure associate to any formal diagram a unique squeezed form:

Definition 2.1.11 (Squeezing procedure)**.** If $F = \mathfrak{h}_1 \circ \mathfrak{h}_2 \circ \cdots \circ \mathfrak{h}_m$ where $\mathfrak{h}_i = d_{i,1} * \cdots * d_{i,n_i}$, $d_{i,j} \in \mathcal{S} \cup \{\mathbf{id}_k\}_{k \in \mathbb{N}}$, the *squeezed form of* F is formal diagram $Sqz(F)$ defined by iterating, as long as possible, on F the following transformations:

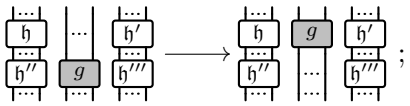
- if $\mathfrak{h}_i = \mathbf{id}_k$ and $i > 1$ then $\mathfrak{h}_{i-1} \circ \mathfrak{h}_i \to_s \mathfrak{h}_{i-1}$. This correspond to the following move on string diagrams:



- if $\mathfrak{h}_x = \mathfrak{h} * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}'$ and $\mathfrak{h}_{x+1} = \mathfrak{h}'' * \alpha * \mathfrak{h}'''$ for some $1 \leq x < m$, $\alpha \in \mathcal{S}$, $\mathfrak{h}, \mathfrak{h}', \mathfrak{h}'', \mathfrak{h}''' \in Form_{\mathcal{S}}$ such that $\mathrm{out}(\mathfrak{h}) = \mathrm{in}(\mathfrak{h}'')$, then

$$\mathfrak{h}_x \circ \mathfrak{h}_{x+1} \to_s (\mathfrak{h} * \alpha * \mathfrak{h}') \circ (\mathfrak{h}'' * \mathbf{id}_{\mathrm{out}(\alpha)} * \mathfrak{h}''').$$

  This correspond to the following move on string diagrams:

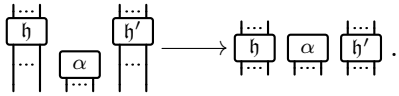


These transformations respectively correspond to the neutrality of $\mathbf{id}_i$'s for (defined) sequential post-compositions and to the application of the interchange rule moving a gate "upwards", i.e. the neutrality of $\mathbf{id}_k$'s in sequential composition.

In particular, for all $\mathfrak{h}, \mathfrak{h}' \in \mathcal{S}^{\circ}_{\mathbf{id}}$, if $\alpha \in \mathcal{S}$ such that $\mathrm{in}(\alpha) = 0$ and $k' + k'' = \mathrm{out}(\mathfrak{h} \circ \mathfrak{h}')$ we have

$$(\mathfrak{h} \circ \mathfrak{h}') \circ (\mathbf{id}_{k'} \circ \alpha \circ \mathbf{id}_{k''}) \to_s (\mathfrak{h} \circ \alpha \circ \mathfrak{h}').$$

This corresponds to the following move on string diagrams:



This may generate conflicts in case of $\beta \in \mathcal{S}$ such that $\mathrm{out}(\beta) = 0$ since

$$\begin{array}{ccccc} (\alpha * \mathbf{id}_0) \circ \beta & = \alpha \circ \beta = & (\mathbf{id}_0 * \alpha) \circ \beta \\ \downarrow s & & \downarrow s \\ \alpha * \beta & & \beta * \alpha \end{array}.$$

In order to prove the existence of a squeezed form for a given formal diagram we need a proof that the squeezing procedure ends. For this purpose, we study these transformations as rewriting rules over (a particular family of strings over) the alphabet $\Sigma_{\mathcal{S}} = \mathcal{S} \cup \{\mathbf{id}_1, *, \circ\}$. If we denote

$$\mathcal{S}^*_{\mathbf{id}} = \{\text{word of the form } d_1 * \cdots * d_k | d_i \in \mathcal{S} \cup \{\mathbf{id}_1\}\}$$

and we keep this notation with $\mathbf{id}_k = \mathbf{id}_1 * \ldots \mathbf{id}_1$ and the definitions of the two functions $\mathrm{in}, \mathrm{out} : Form_{\mathcal{S}} \to \mathbb{N}$, the above relations correspond to the following set $\mathcal{R}_{\mathcal{S}}$ of rewriting rules:

i) for all $\mathfrak{h} \in \mathcal{S}^*_{\mathbf{id}}$,

$$\circ\mathfrak{h} \circ \mathbf{id}_{out(\mathfrak{h})}\circ \to \circ\mathfrak{h}\circ;$$

ii) for all $\alpha \in \mathcal{S}$ such that $\mathrm{in}(\alpha) \neq 0$ and for all $\mathfrak{h}'_x, \mathfrak{h}''_x, \mathfrak{h}'_{x+1}, \mathfrak{h}''_{x+1} \in \mathcal{S}^*_{\mathbf{id}}$ such that $\mathrm{out}(\mathfrak{h}'_x) = \mathrm{in}(\mathfrak{h}'_{x+1})$ and $\mathrm{out}(\mathfrak{h}''_x) = \mathrm{in}(\mathfrak{h}''_{x+1})$,

$$\circ\mathfrak{h}'_x * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1}\circ \to \circ\mathfrak{h}'_x * \alpha * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \mathbf{id}_{\mathrm{out}(\alpha)} * \mathfrak{h}''_{x+1}\circ;$$

iii) for all $\alpha \in \mathcal{S}$ such that $\mathrm{in}(\alpha) = 0$ and for all $\mathfrak{h}'_x, \mathfrak{h}''_x, \mathfrak{h}'_{x+1}, \mathfrak{h}''_{x+1} \in \mathcal{S}^*_{\mathbf{id}}$ such that $\mathrm{out}(\mathfrak{h}'_x) = \mathrm{in}(\mathfrak{h}'_{x+1})$ and $\mathrm{out}(\mathfrak{h}''_x) = \mathrm{in}(\mathfrak{h}''_{x+1})$,

$$\circ\mathfrak{h}'_x * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1}\circ \to \circ\mathfrak{h}'_x * \alpha * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \mathbf{id}_{\mathrm{out}(\alpha)} * \mathfrak{h}''_{x+1}\circ$$

where $\mathfrak{h}'_x \neq \mathfrak{g} * \mathfrak{z}$ for some $\mathfrak{z} \in \mathcal{S}^*_{\mathbf{id}}$ with $\mathrm{out}(\mathfrak{z}) = 0$.

Lemma 2.1.12 (Tetris Lemma)**.** *The rewriting system* $(\Sigma_{\mathcal{S}}, \mathcal{R}_{\mathcal{S}})$ *terminate.*

Proof. We associate to any string $s \in \Sigma_{\mathcal{S}}$ the ordinal:

$$\mathrm{ord}(s) = L_0(s) + 2L_1(s) + 2^2L_2(s) + \cdots + 2^{\|s\|_\circ}L_{\|s\|_\circ}(s)$$

where the L_i's are the maps associating to a string $s \in \Sigma_{\mathcal{S}}$ the number of symbols in $\mathcal{S}$ occurring after the i-th and before the $(i+1)$-th occurrence of a symbol $\circ$ plus 1. This is, $L_i(s)$ corresponds to $1 + \|\mathfrak{h}_i\|_{\mathcal{S}}$ (the number of gates in $\mathfrak{h}_i$ plus 1) whenever $F = \underset{i}{*}\mathfrak{h}_i \in Form_{\mathcal{S}}$ and $\circ F\circ \in \Sigma_{\mathcal{S}}$.

For any rewriting rule $s \to s' \in \Sigma_{\mathcal{S}}$, $\mathrm{ord}(s) > \mathrm{ord}(s')$. Indeed:

- for the rules in i), we observe that:

$$\mathrm{ord}(\circ\mathfrak{h} \circ \mathbf{id}_{\mathrm{out}(\mathfrak{h})}\circ) = \mathrm{ord}(\circ\mathfrak{h}\circ) + 2^{2+\|\mathfrak{h}\|_\circ} > \mathrm{ord}(\circ\mathfrak{h}\circ);$$

- for any $k \in \mathbb{N}$, $\mathfrak{h}'_x, \mathfrak{h}''_x, \mathfrak{h}'_{x+1}, \mathfrak{h}''_{x+1} \in \mathcal{S}^*_{\mathbf{id}}$ we have

$$\mathrm{ord}(\circ w_x\circ\mathfrak{h}'_{x+1}*\mathbf{id}_k*\mathfrak{h}''_{x+1}) = \mathrm{ord}(\circ w_x\circ w_{x+1}\circ) = \mathrm{ord}(\circ\mathfrak{h}'_x*\mathbf{id}_k\circ\mathfrak{h}''_x\circ w_{x+1}\circ)$$

if $w_x = \mathfrak{h}'_x * \mathfrak{h}''_x$ and $w_{x+1} = \mathfrak{h}'_{x+1} * \mathfrak{h}''_{x+1}$. So, for any $s \to s'$ in the sets ii) and iii), we have

$$\mathrm{ord}(s) = 2\|w_x\|_{\mathcal{S}} + 4(\|w_{x+1} + 1) < \mathrm{ord}(s') = 2(\|w_x\|_{\mathcal{S}} + 1) + 4\|w_{x+1}\|_{\mathcal{S}}$$

It follows that there can not be an infinite reduction chain since any reduction path induces a decreasing chain of ordinals — which are well-ordered. □

In a degenerate signature, the coexistence of some $\alpha \in \mathcal{S}$ with $\mathrm{in}(\alpha) = 0$ together with some $\beta \in \mathcal{S}$ with $\mathrm{out}(\beta) = 0$ generates conflicts in the corresponding rewriting system $(\Sigma_{\mathcal{S}}, \mathcal{R})$. In fact, for any $\mathfrak{h}'_x, \mathfrak{h}''_x, \mathfrak{h}'_{x+1}, \mathfrak{h}''_{x+1} \in \mathcal{S}^*_{\mathbf{id}}$ such that $\mathrm{out}(\mathfrak{h}'_x) = \mathrm{in}(\mathfrak{h}'_{x+1})$ and $\mathrm{out}(\mathfrak{h}''_x) = \mathrm{in}(\mathfrak{h}''_{x+1})$, both rules

$$\circ\mathfrak{h}'_x * \beta * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1}\circ \to \circ\mathfrak{h}'_x * \beta * \alpha * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \mathbf{id}_{\mathrm{out}(\alpha)} * \mathfrak{h}''_{x+1}\circ$$

and

$$\circ\mathfrak{h}'_x * \beta * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1}\circ \to \circ\mathfrak{h}'_x * \alpha * \beta * \mathfrak{h}''_x \circ \mathfrak{h}'_{x+1} * \mathbf{id}_{\mathrm{out}(\alpha)} * \mathfrak{h}''_{x+1}\circ$$

are in iv). The reason why we isolate this set of rules is due to the fact that it arises only in case of degenerate signatures, causing the divergence of the system.

Lemma 2.1.13. *If $\mathcal{S}$ is non-degenerate then the rewriting system $(\Sigma_{\mathcal{S}}, \mathcal{R})$ is confluent.*

Proof. As remarked, a non-degenerate signature gives a rewriting system $(\Sigma_{\mathcal{S}}, \mathcal{R})$ where no conflict generate by two rules in iv) is defined, then the rewriting system has the following classes of local confluent critical peaks:

- $\circ\mathfrak{h} \circ \mathbf{id}_{out(\mathfrak{h})} \circ \mathbf{id}_{out(\mathfrak{h})}\circ$;
- $\circ\mathfrak{h} \circ \mathbf{id}_{out(\mathfrak{h})} \circ \mathbf{id}_k * \beta * \mathbf{id}_{(out(\mathfrak{h})-k)}\circ$;
- $\circ(\mathfrak{h}'_x * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}''_x) \circ (\mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1}) \circ \mathbf{id}_{\mathrm{out}(\mathfrak{h}_{x+1})}\circ$;
- $\circ(\mathfrak{h}'_x * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}''_x * \mathbf{id}_{\mathrm{in}(\beta)} * \mathfrak{h}'''_x) \circ (\mathfrak{h}'_{x+1} * \alpha * \mathfrak{h}''_{x+1} * \beta * \mathfrak{h}'''_{x+1})\circ$;
- $\circ(\mathfrak{h}'_x * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}''_x) \circ (\mathfrak{h}'_{x+1} * \alpha * \beta * \mathfrak{h}''_{x+1})\circ$;
- $\circ(\mathfrak{h}'_x * \mathbf{id}_{\mathrm{in}(\alpha)} * \mathfrak{h}''_x) \circ (\mathfrak{h}'_{x+1} * \beta\alpha * *\mathfrak{h}''_{x+1})\circ$.

for all $\mathfrak{h}, \mathfrak{h}'_x, \mathfrak{h}''_x, \mathfrak{h}'''_x, \mathfrak{h}'_{x+1}, \mathfrak{h}''_{x+1}, \mathfrak{h}'''_{x+1} \in \mathcal{S}^*_{\mathbf{id}}$, $\alpha, \beta \in \mathcal{S}$ with $\mathrm{in}(\beta) = 0$.

The local confluence of all critical peaks permits, by Proposition 1.2.8, to prove the convergence of $(\Sigma_{\mathcal{S}}, \mathcal{R})$. □

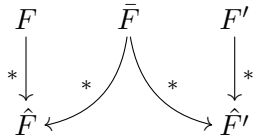
This yields the following theorem.

Theorem 2.1.14 (Decidability of the equivalence problem)**.** *The equivalence problem for formal diagrams is decidable.*

Proof. The rules in $(\Sigma_{\mathcal{S}}, \mathcal{R})$ correspond to the neutrality of $\mathbf{id}_k$'s diagrams for sequential composition. By Remark 2.1.7, this is equivalent to the equivalence under the interchange rule of $\mathcal{R}_{\mathcal{S}}$-equivalent formal diagrams, so in order to test if $F \simeq_{exc} F'$ we should prove that $F \leftrightarrow^*_{\mathcal{R}_{\mathcal{S}}} F'$, i.e. we have to solve a word problem on $(\Sigma_{\mathcal{S}}, \mathcal{R})$.

If the signature is non-degenerate, the confluence of $(\Sigma_{\mathcal{S}}, \mathcal{R})$ allows to assert the equivalence of two formal diagrams F and F' if and only if they have the same $\mathcal{R}_{\mathcal{S}}$-normal form.

In the degenerate case, we solve the equivalence problem for two element F, F' exhibiting an element $\bar{F}$ such that $\bar{F} \to^* \hat{F}$ and $\bar{F} \to^* \hat{F}'$ where $\hat{F}$ and $\hat{F}'$ are respectively squeezed forms of F and F':



If we bipartite $\mathcal{R}_{\mathcal{S}}$ in $\mathcal{R}_c = \{\text{rules in } i), \text{ and } ii)\}$ and $\mathcal{R}_d = \{\text{rules in } iii)\}$ we note that there are no critical pairs between rules in $\mathcal{R}_c$ and $\mathcal{R}_d$. This means that any critical pair between a rule in $\mathcal{R}_c$ and $\mathcal{R}_d$ is locally confluent. By this fact we deduce that if H is a common ancestor of any irreducible formal diagram in $[F]$ and $H \to^*_{\mathcal{R}_c} \bar{F}$ then $\bar{F} \to \hat{F}$ for any $\hat{F} \in [F]$.

As remarked, the non-convergent conflicts of the system arise from conflicts between rules in $iv)$, namely in presence of formal diagrams of the form $G \circ G'$ with $\text{in}(G) = 0$ and $\text{out}(G') = 0$ which can be rewritten as both $G' * G$ and $G * G'$.

Let $\bar{F}$ be the minimal ancestor of $\hat{F}$ such that $\bar{F}$ contains no subdiagrams of such form.This is the diagram obtained by substituting in $\hat{F}$ all subdiagrams of the form $G' * G$ or $G * G'$ with $G \circ G'$, whenever $in(G') = out(G) = 0$. By definition $\bar{F} \to \hat{F}$ and, as remarked, it reduces to any formal diagram $\hat{F}'$ obtained by $\hat{F}$ pemutating G and G' in some subdiagrams $G \otimes G'$ or $G' \otimes G$ of $\hat{F}$ with $in(G) = out(G) = 0$. It turns out that those are all the irreducible formal diagrams equivalent to $\hat{F}$.

By these facts, in order to answer equivalence problem for two formal diagrams F, F', it suffice to compute $\bar{F}$ and test if one of its irreducible form $\hat{F}'$ is equal to one of irreducible form of $\bar{F}$. □

Of course, the decidability of equivalence problem for formal diagram can be proved using a method similar to the one introduced by Guiraud in his thesis [32] using the categorical construction of string diagram together with the fact that equivalent formal diagrams correspond to the same string diagram.

Our proof setting is motivated by the fact that we want to keep this result as pure rewriting of 2-dimensional words, that is more manageable setting used in computer program literature (for example [34]) to code string diagrams. This result finally results independent form the interpretation of formal diagram by means of string diagram in order to work.

Definition 2.1.15 (Expanded form)**.** A formal diagram $F \in Form_{\mathcal{S}}$ is an *expanded form* if it is of the form:

$$F = \bigcirc_{1 \le i \le h(F)} (\mathbf{id}_{i_1} * \alpha_i * \mathbf{id}_{i_2})$$

with $\alpha_i \in \mathcal{S}$ or $F = \mathbf{id}_0$.

Proposition 2.1.16. *Any formal diagram admits a $\simeq_{exc}$-equivalent expanded form.*

Proof. Let $F \in Form_{\mathcal{S}}$, by induction on the number of gates:

- if $F = F_{exp} \circ F'$ and F_{exp} is an expanded form, it suffices to find an expanded F'_{exp} equivalent to $F' = (\mathbf{id}_j * \mathfrak{h}) \circ F''$. Then $F_{exp} \circ F'_{exp}$ will be the formal diagram we are looking for.
- if $F = (\mathbf{id}_j * \mathfrak{h}) \circ F_d$ with $\mathfrak{h} = \alpha * \mathbf{id}_{j_2} * \mathfrak{h}'$, the formal diagram

$$F' = (\mathbf{id}_j * \alpha * \mathbf{id}_{j'_2}) \circ (\mathbf{id}_{j+out(\alpha)+j_2} * \mathfrak{h}'') \circ F_d$$

  is $\simeq_{exc}$-equivalent to F and the number of gates in $\mathfrak{h}''$ is less than in the one in $\mathfrak{h}'$.

□

By definition, a formal diagram can admit different $\simeq_{exc}$-equivalent expanded forms. Unless specified, we refer to an expanded form F_{exp} of a given formal diagram F as the one obtained by the above procedure.

With the definition of formal diagram and the exchange rule, we can finally recover what we have previously introduced as *diagram over a signature*.

Definition 2.1.17 (Diagram, alternative definition)**.** A *diagram* or *string diagram* (over the signature $\mathcal{S}$) is an equivalence class of formal diagrams modulo the equivalence relation $\simeq_{exc}$.

Firstly, we want prove the equivalence between this definition and the standard one of string diagram.

Theorem 2.1.18. *Given a non-degenerate signature $\mathcal{S}$, any digram ϕ (over $\mathcal{S}$) represents an equivalence class $[F]_{\simeq_{exc}}$ of formal diagrams (over $\mathcal{S}$). Furthermore there is a one-to-one correspondence between diagrams and formal diagrams in squeezed form.*

Proof. To prove it we give a bijection between these two sets. By induction on the number of gates in a diagram $\phi \in \mathcal{S}^*$, we define $[\![\cdot]\!]^{-1} : \mathcal{S}^* \to \frac{Form_{\mathcal{S}}}{\simeq_{exc}}$ as follows:

- if $\phi = \emptyset$ then $[\![\phi]\!]^{-1} = [\mathbf{id}_0]_{\simeq_{exc}}$;
- if $\phi = \mathbf{id}_i$ then $[\![\phi]\!]^{-1} = [\mathbf{id}_i]_{\simeq_{exc}}$;

- if ϕ contains at least a gate then

$$[\![\phi]\!]^{-1} = [\mathbf{id}_j * \alpha * \mathbf{id}_{in(\phi')}]_{\simeq_{exc}} \circ [\![id_{j+out(\alpha)} * \phi']\!]^{-1} \circ [\![\phi'']\!]^{-1},$$

  where α is the upper-leftmost gate in ϕ, that is $\phi = \mathbf{id}_i \circ \mathbf{id}_j * \alpha * \phi' \circ \phi''$ for some $i, j \in \mathbb{N}$ and $\phi', \phi'' \in \mathcal{S}^*$ with number of gates less than that of ϕ.

By the compatibility of $\simeq_{exc}$ with the two operations $\circ$ and $*$, we have found the equivalence class $[\![\phi]\!]^{-1} = [\bigcirc_{1\leq i\leq h}(\mathbf{id}_{i_1} * \alpha_i * \mathbf{id}_{i_2})]_{\simeq_{exc}}$.

In the same way, given the equivalence class $[F]_{\simeq_{exc}}$ of a formal diagram F with expanded form $F_{exp} = \bigcirc_i(\mathbf{id}_{i_1} * \alpha_i * \mathbf{id}_{i_2})$, we define

$$[\![F]\!] = \bigcirc_i [\![\mathbf{id}_{i_1} * \alpha_i * \mathbf{id}_{i_2}]\!]$$

since $[F]_{\simeq_{exc}} = [F']_{\simeq_{exc}}$.

Moreover in a non-degenerate signature, by Lemma 2.1.12, each representative of the same equivalence class $\phi = [\![F]\!]$ has the same squeezed form. This induces a one-to-one correspondence between formal diagrams in squeezed form and diagrams. □

Definition 2.1.19 (Lifting of a diagram)**.** If $\phi \in \mathcal{S}^*$ is a diagram and $\mathcal{S}$ a non-degenerate signature, the *lifting of* ϕ is the unique squeezed form F in the equivalence class $[\![\phi]\!]$. We denote $\tilde{\phi} = F$ the lifting of ϕ.

Now that we have proven the equivalence of these two definitions, to make uniform the notation we adopt the following convention: we note the atomic diagrams with the letters d (if they are gates we also use the letter g), we denote $d : \alpha$ with $\alpha \in \mathcal{S}_{\mathbf{id}}$ when d is of type α i.e. when d is an occurrence of the symbol α. If not necessary to specify it, we note $d : \mathbf{id}$ for $d \in \{id_i\}_{i\in\mathbb{N}}$.

Topology on diagrams

With this new formalism of formal diagrams, we are able to define some topological concepts on diagrams coming from some notion which we like to borrow from graph theory. Intuitively, we consider diagrams as a sort of circuits where information flows in the strings up-to-down . In these circuits, gates operate some elementary transformations on their inputs returning some outputs so that any diagram $\phi : p \Rightarrow q$ represents a process with p inputs and q outputs. Their parallel and sequential compositions represent respectively the parallel and sequential execution. Identity diagrams are represented by parallel wires in which information has no transformation.

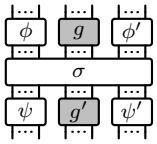
In this section a fixed signature $\mathcal{S}$ is always supposed to be given.

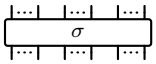
Definition 2.1.20 (Free port)**.** Given a diagram $\phi : p \Rightarrow q$, a *free port* f of ϕ is one of its inputs or outputs, which we enumerate from left to right.

Definition 2.1.21 (Communicating gates)**.** Two atomic diagrams d, d' are *subsequential* in $\phi = [\![\mathfrak{h}_1 * \cdots * \mathfrak{h}_n]\!]$ if there is an $1 \leq i \leq n$ such that $\mathfrak{h}_i = \mathfrak{h}_{i,1} \otimes d \otimes \mathfrak{h}_{i,2}$, $\mathfrak{h}_{i+1} = \mathfrak{h}_{i+1,1} \otimes d' \otimes \mathfrak{h}_{i+1,2}$ and $out(\mathfrak{h}_{i,1}) \leq in(\mathfrak{h}_{i+1,1}) < out(\mathfrak{h}_{i,1} \otimes d)$. This means that one of the output port of d is linked by a wire with an input port of d'.

Two gates g, g' are *communicating* (denoted $\downarrow_{g'}^{g}$) if there exists a sequence of subsequential atomic diagrams $g = d_0, d_1, \ldots, d_n = g'$ such that $d_i : \mathbf{id}$ for all $0 < i < n$.

In a twisting polygraph, two gates d, d' which are not twisting operators are *twisting-communicating* if the diagram contains a subdiagram of the form



where  $: n \Rightarrow n$ is a twisting diagram corresponding to the permutation $\sigma \in S_n$ with $n = \text{out}(\phi) + \text{out}(g) + \text{out}(\phi') = \text{in}(\psi) + \text{in}(g') + \text{in}(\psi')$ such that

$$\exists i, j \text{ whit } \text{out}(\phi) < i < n - \text{out}(\phi') \text{ , } \text{in}(\psi) < j < n - \text{in}(\psi') \text{ and } \sigma(i) = j.$$

The communication condition can be graphically interpreted by the existence of a string connecting an output port of g and an input port of g' possibly crossing gates of twisting operators.

Definition 2.1.22 (Port)**.** A *port* (*input* or *output* port) of a gate $g : \alpha$ of ϕ is the corresponding free port of a diagram ϕ_α consisting of a single gate of type α. A port of a gate g of ϕ is free if

- $\tilde{\phi} = \mathfrak{h} \otimes g \otimes \mathfrak{h} * F$ or $\tilde{\phi} = (\mathfrak{h} \otimes d \otimes \mathfrak{h}') * \tilde{\phi}'$, $d : \mathbf{id}$ and $\downarrow_{g}^{d}$;
- $\tilde{\phi} = F * (\mathfrak{h} \otimes g \otimes \mathfrak{h})$ or $\tilde{\phi} = \tilde{\phi}' * (\mathfrak{h} \otimes d \otimes \mathfrak{h}')$, $d : \mathbf{id}$ and $\downarrow_{d}^{g}$;

for some $F, F' \in Form_{\mathcal{S}}$, $\mathfrak{h}, \mathfrak{h}' \in H_{\mathcal{S}}$.

Definition 2.1.23 (Gates path)**.** A *gate path* $P : g_1 \rightsquigarrow g_n$ in a diagram is given by a sequence of gates $g_1, \ldots, g_n$ such that $\downarrow_{g_{i+1}}^{g_i}$ for all $1 \leq i \leq n-1$. In a twisting polygraph, a gate path is made of twisting-communicating non-twisting gates.

An *undirected* gate path $P : g \leftrightsquigarrow g'$ is a sequence of gates $g = g_1, \ldots, g_n = g'$ such that $\downarrow_{g_{i+1}}^{g_i}$ or $\downarrow_{g_i}^{g_{i+1}}$ for all $1 \leq i < n$.

This allows to define connectedness in diagrams:

Definition 2.1.24 (Connection)**.** If Σ is a polygraph, a diagram $\phi \in \Sigma$ is *connected* if for all gates $g, g' \in \phi$, g, g' not twisting operators, there is an undirected gates path $P : g \leftrightsquigarrow g'$.

Some gates in diagrams will play a particular role in further discussion:

Definition 2.1.25 (Left/Right/Central external gate)**.** A gate g of ϕ is *left (right) external* in ϕ if there is an expanded form $F_{exp} \simeq_{exc} \tilde{\phi}$ of the form $F_{exp} = F' * (g \otimes \mathbf{id}_k) * F''$ (resp. $F_{exp} = F' * g * F''$) for some $F', F'' \in \mathcal{S}^*$, $k \in \mathbb{N}$. It is central if both properties hold, i.e. $F_{exp} = F' * (g) * F''$

A gate g of a diagram ϕ is *internal* if it is not external, that is, no input or output port is free in ϕ.

The notion of subdiagram can be recovered by the notion of formal subdiagram:

Definition 2.1.26 (Subdiagram)**.** A diagram $\phi' \in \mathcal{S}^*$is a *subdiagram* of ϕ (noted $\phi' \subseteq \phi$) if there are $F, F' \in Form_{\mathcal{S}}$ such that $\phi' = [\![F']\!]$, $\phi = [\![F]\!]$ and $F' \subseteq F$.

These definitions correspond to the following intuition: if we consider a diagram ϕ as a sort of electronic circuit, we should think a subdiagram as a piece of this circuit which is able to "work", that is, each gate $g \in G$ has its ports either free or connecting it to another gate $g' \in G$.

Definition 2.1.27 (Subdiagram generated by a set of gates)**.** If ϕ is a diagram and G a subset of its gates, the smallest subdiagram generated by G is the subdiagram ϕ_G containing all gates in G.

It can be found as $[\![F_G]\!]^{-1}$ where F_G is the biggest formal subdiagram of F containing all gates in G.

Subdiagram decidability] To check if ψ is a subdiagram of ϕ is decidable.

Proof. Let G_ϕ and G_ψ be the set of gates of ϕ and ψ respectively. If $G_\psi \not\subseteq G_\phi$ then $\psi \not\subseteq \phi$, otherwise, we consider all subdiagrams of ϕ generated by its subset of gates in a one-to-one correspondence with G_ψ conform with gate types. Then it suffice to prove if any of such diagrams is equivalent to ψ, which is decidable by Theorem 2.1.14. □

This definition allows to define the intersection of two subdiagrams.

Definition 2.1.28 (Subdiagrams intersection)**.** If $\phi', \phi'' \subseteq \phi$ we define $\phi' \cap \phi''$ as the biggest subdiagram $\psi \subseteq \phi$ such that $\psi \subseteq \phi'$ and $\psi'' \subseteq \phi''$.

Definition 2.1.29 (Border)**.** The *border* of a diagram ϕ is the set $\partial(\phi)$ of gates with free ports.

Definition 2.1.30 (Expandable intersection)**.** Two subdiagrams ϕ' and ϕ'' of a diagram ϕ with non-trivial intersection have *expandable intersection* if there are communicating in ϕ, $g' \in \partial(\phi')$ and $g'' \in \partial(\phi'')$ such that $g', g'' \notin \phi' \cap \phi''$.

For example if we consider the diagram  and its subdiagrams  and  , these have an expandable intersection.

We define some peculiar subset of gates in a diagram:

Definition 2.1.31 (Cone)**.** Given a gate g in a diagram ϕ we define:

- the upper cone of g $\overline{\partial}_\phi(g) = \{g' \in \phi | \exists \text{ a gate path } P : g' \rightsquigarrow g\}$;
- $\underline{\partial}_\phi(g) = \{g' \in \phi | \exists \text{ a gate path } P : g \rightsquigarrow g'\}$;
- the cone of g in ϕ is the set of gates $\underline{\overline{\partial}}_\phi(g) := \overline{\partial}_\phi(g) \cup \overline{\partial}_\phi(g)$.

Similarly, if $\phi' \subseteq \phi$, we define its cone:

- the upper cone of ϕ' $\overline{\partial}_\phi(\phi') = \{g \in \phi | g \notin \phi',\ \exists g' \in \phi' \text{ such that } g \in \overline{\partial}_\phi(g')\}$;
- $\underline{\partial}_\phi(\phi') = \{g \in \phi | g \notin \phi',\ \exists g' \in \phi' \text{ such that } g \in \underline{\partial}_\phi(g')\}$;
- the cone of ϕ' in ϕ is $\underline{\overline{\partial}}(\phi')_\phi = \overline{\partial}(\phi')_\phi \cup \overline{\partial}(\phi')_\phi$.

In a twisting polygraph, a *twisting-cone* is given by the set of non-twisting gates connected by gates paths with respect to twisting-communication.

In particular, the border is the part of the diagram that interact with others diagrams in sequential composition.

Definition 2.1.32 (Stripe)**.** If f is a free port of a diagram ϕ and g_f the unique gate such that $f \in g_f$, then the following are defined:

- $\overline{\partial}_\phi(f) = \begin{cases} \overline{\partial}_\phi(g_f) & \text{if } f \text{ is an output} \\ \emptyset & \text{if } f \text{ is an input} \end{cases}$;
- $\underline{\partial}_\phi(f) = \begin{cases} \emptyset & \text{if } f \text{ is an output} \\ \underline{\partial}_\phi(g_f) & \text{if } f \text{ is an input} \end{cases}$;

If F is an initial or final segment of the set of inputs (or output) free ports, that is, a set of form $\{1, 2, \ldots, k\}$ or $\{k', k'+1, \ldots, n\}$ where $n = in(\phi)$ (resp. $n = out(\phi)$), we define:

- $\underline{\partial}_\phi(F) = \{g \in \phi | g \in \underline{\partial}_\phi(g_f) \text{ exists } f \in F\}$;
- $\overline{\partial}_\phi(F) = \{g \in \phi | g \in \overline{\partial}_\phi(g_f) \text{ exists } f \in F\}$

The *left* (n, m)*-stripe of* ϕ is the subdiagram of ϕ generated by the set of gates

$$\{g \in \phi | g \notin \underline{\partial}_\phi(F_n) \cap \overline{\partial}_\phi(F_m)\}$$

where F_n is the set of the last $in(\phi) - n$ input ports and F_m the set of the last $out(\phi) - m$ ports.

The *right* (n, m)*-stripe of* ϕ is defined in the same way but with F_n and F_m respectively the set of the first $in(\phi) - n$ input ports and the set of $out(\phi) - m$ ports.

2.2 Diagram term rewriting

In this section we introduce the notion of *diagrammatic variables* and a relative *diagrammatic term substitution*. The main idea is to consider the *signature of diagrammatic variables*

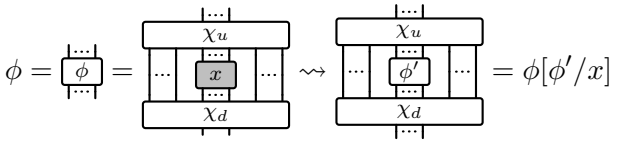
$$\Xi = \{\chi_{i,j} = \boxed{\chi_{ij}} : i \Rightarrow j | i, j \in \mathbb{N}\}$$

and include it in a signature:

Definition 2.2.1 (Extended signature)**.** If $\mathcal{S}$ is a signature, we call $\Xi_\mathcal{S} = \mathcal{S} \cup \Xi$ the *extended signature of* $\mathcal{S}$.

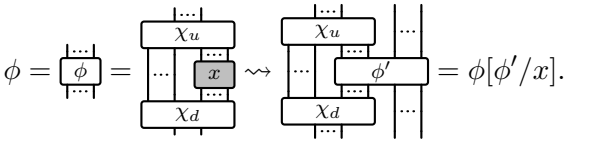
Then we define a *variable substitution for diagrams*.

Definition 2.2.2. If $\phi, \phi' \in \Xi_\mathcal{S}$ are two diagrams such that ϕ contains a gate $x : \chi_{i,j} \in \Xi$ and $\phi' : i \to j$, we define the *substitution of* x *by* ϕ' *in* ϕ as the diagram $\phi[\phi'/x]$ replacing the occurrence of x in ϕ by ϕ':



In the string diagram formalism, we are able to define a substitution which happens to be not well-typed in the classical terms of rewriting.

Definition 2.2.3 (External substitution)**.** If $\phi, \phi' \in \Xi_\mathcal{S}$ are two diagrams such that ϕ contains a right-external gate $x : \chi_{i,j} \in \Xi$ and $\phi' : i' \to j'$ with $i \leq i', j \leq j'$, we define the *right-external substitution of* x *by* ϕ' *in* ϕ as the following diagram:



In a similar way, if x is left-external we define the *left-external substitution of* x *by* ϕ' *in* ϕ as $\phi[\phi'/x]$.

Definition 2.2.4 (Closed term (diagram))**.** If $\mathcal{S}$ is a signature, a diagram $\phi \in \Xi^*_{\mathcal{S}}$ is a *closed term* if $|\phi|_\Xi = 0$, otherwise it is an *open term*. We call it *semi-closed term* if the following conditions are satisfied:

- no gate of type $\chi \in \Xi$ has all its input or all its output ports free;
- if $X \subset \phi$ and $X \in \Xi^*$, then $|X| = 1$.

A semi-closed term can be intuitively seen as a diagram with non adjacent "holes" labeled by diagrammatic variables. The introduction of the external substitution gives a model for diagram term rewriting in which we do not need the diagram which we are going to substitute to fit with the type of variable. We recall Samuel Mimram [69] where a similar result for external substitution is given by extending diagrammatic semantics to compact closed and rotative categories.

Definition 2.2.5 (Partial sub-diagram)**.** If $\mathcal{S}$ is a signature, a *partial sub-diagram of* ϕ with $\phi \in \mathcal{S}^*$ is a semi-closed term in $\phi' \in \Xi_{\mathcal{S}}$ with $|\phi|_\Xi = k$ such that there are $\phi_1, \ldots, \phi_k \in \mathcal{S}^*$ such that $\phi = \phi'[\phi_1/x_1, \ldots, \phi_k/x_k]$.

2.3 2-Dimensional grammars

As shown in the previous section, a string diagram can be seen as a term in a language generated by certain formal grammar. In this section we underline that those grammars are characterized by the fact that there exist certain term patterns are forbidden while, on the other hand, some others give a new form of interaction between subterms.

We are not able to give a well-ordered classification with respect to inclusion, also because some characteristics of this classification are independent. Consequently, in the next section we show how these features can influence the confluence property of rewriting.

Definition 2.3.1 (1-Factor)**.** A signature $\mathcal{S}$ has the 1-*factor* of size k if there exist $\phi \in \mathcal{S}^*$, $\phi : k \to k$ such that there exists a family of diagrams $\{\phi^{\circ n} = \overset{n}{\underset{i=1}{\bigcirc}}\phi\}$ with $\phi \neq \mathbf{id}$.

This feature is natural in word syntax, in fact if a is a symbol of an alphabet, we can always iterate the symbol in order to express a word of length $n \in \mathbb{N}$ by successive occurrence of this letter. On the other hand, such feature is not present, for example, in the *signature of binary trees* $\{$  $\}$, where all non-identity diagrams are of type $n \to m$ with $n > m$.

Another important feature that diagrammatic syntax can present is what we call *dark energy*.

Definition 2.3.2 (0-Factor)**.** A signature $\mathcal{S}$ has the 0-*factor* if there exist at least an element $\phi \in \mathcal{S}^*$ such that $\phi : 0 \to 0$.

As remarked in [77], the existence of such elements create some particular paradigms in the language: it breaks the duality of the two compositions causing problems in diagram equivalence problem complexity. In fact, if ϕ and ϕ' are of type $0 \to 0$, then

$$\phi * \phi' = \phi \circ \phi' = \phi' * \phi = \phi \circ \phi.$$

Moreover remark first that the existence of a such term implies the existence of an infinity of terms of such type: it suffices to consider the set of $\overset{k}{\underset{i=1}{*}} \phi$ with $k \in \mathbb{N}$.

The choice of name them dark energy is due to the fact that, following the interpretation of diagrams as transformations of inputs to outputs observable data, such diagrams operate no transformations such as dark energy in quantum physics can not be observed.

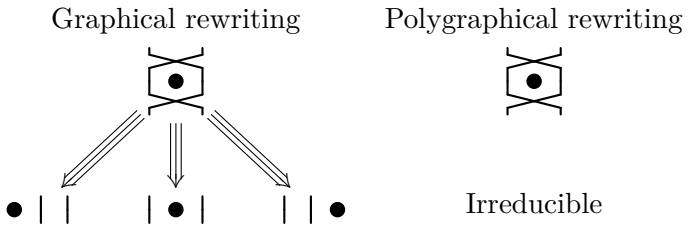
One of the particularity of the existence of such terms is related to the existence of another factor:

Definition 2.3.3 (2-Factor)**.** A signature $\mathcal{S}$ has a 2-*factor* if there exists a $\phi \in \mathcal{S}^*$ such that there exist two gate paths with same source and same target. We call a connected portion of background contained between two such strings a *cage*.

The presence of cages and dark energies give us two possible ways of rewriting diagrams: on one side we have a rewriting inspired by graph rewriting (we call it graphical) and on the other hand we have the rewriting induced by polygraphic formulation of string diagram rewriting systems (we call it polygraphical).

The graphical interpretation considers diagrams as a sort of planar directed graphs. In case of dark energy inside a cage there are no limitation in rewriting. We only have to choose where dark energies are placed if a cage get splitted or opened. It could also happen that it does not matter where we place it since it could move freely in the diagram causing no interaction with the rest of it. On the other hand, a polygraphical rewriting is blocked in case of no specific rules to manage dark energy in a cage: if some dark energy appear in a cage of a diagram either this whole term (the diagram with its dark energy) is a premise of a rewriting rule rewriting is stacked.

Let consider the following example: $\mathcal{S} = \{\times : 2 \to 2, \bullet : 0 \to 0\}$ with a unique rewriting rule $\times \Longrightarrow \mid \mid$. In this case we have the following possibilities:



Due to the categorical interpretation we give to string diagrams, in this text we consider polygraphical rewriting in order to see diagrams rewriting as a formalism for higher dimensional category theory.

Definition 2.3.4 (Irreducible Factor)**.** A diagram rewriting system has *irreducible* 0*-factor* if there exists an irreducible term $\phi \in \mathcal{S}^*$ with $\phi : 0 \to 0$. It has *irreducible* 1*-factor* if there exists an irreducible non-identity term $\phi \in \mathcal{S}^*$ such that the family $\{\phi^{\circ n} = \overset{n}{\underset{i=1}{\circ}} \phi\}$ is made of irreducible terms. It has *irreducible* 2*-factor* if there exists an irreducible term $\phi \in \mathcal{S}^*$ which has a 2-factor.

Some examples are given in Appendix C.

2.4 Confluence problems in rewriting

Once specified our syntactical and computational settings (polygraphical rewriting), we want to show some computational properties relates to the presence of factors. In particular, we give some necessary conditions in order to have an extension of Squier theorem for monoids based to word rewriting to more general algebraic theories presented by diagram rewriting.

In this section, we assume to work in terminating rewriting systems so that we focus only on the verification of confluence. The termination of a rewriting system can be proved by some methods e.g. by that of Yves Lafont in [52] or, more in general, by means of *reduction orders* as proposed by Yves Guiraud in his thesis [32]. In particular, we show how some combinations of factors can rule out in diagram rewriting the finiteness of some homotopy properties, generalizing some cases observed by Yves Guiraud and Philipe Malbos in [35].

Definition 2.4.1 (Archetype of conflict)**.** If $(\mathcal{S}, \mathcal{R})$ is a diagram rewriting system, an *archetype of conflict (A,B)* (or *archetype* for short) is a minimal semi-closed term $\phi \in \Xi_{\mathcal{S}}$ with two subdigrams $s(A), s(B) \subseteq \phi$ with no expandable intersection such that if $g : \chi \in \Xi$, g has no free ports. An archetype is closed if it is a closed term.

The condition to have no expandable intersection minimize the number of archetypes. In fact, without this condition we obtain an additional one for any diagram substitution of a gate $g : \chi_{n,n} \in Xi$ with the corresponding identity $\mathbf{id}_n$.

Proposition 2.4.2. *In a finite rewriting system $(\mathcal{S}, \mathcal{R})$ there exists a finite number of archetypes.*

Proof. The finiteness of the number of rewriting rules implies the finiteness of possible intersections. By definition, an archetype is minimal for each intersection. □

Definition 2.4.3 (Local irreducibility)**.** If $(\mathcal{S}, \mathcal{R})$ is a rewriting system, $\phi' \subseteq \phi$ and $\mathcal{R}_\phi$ the set of rewriting rules applicable on ϕ, then ϕ is *locally irreducible on ϕ'* if $s(A) \cap \phi' = \emptyset$ for all $A \in \mathcal{R}_\phi$. We say that ϕ it is *left-(n, m)-irreducible* if it is on its left-(n, m) stripe. We define in an analogous way the notion of *right-(n, m)-irreducible*.

We can now give an alternative definition of critical peak in a diagram rewriting system:

Definition 2.4.4 (Critical peak)**.** A *critical peak* of a rewriting system $(\mathcal{S}, \mathcal{R})$ is given by a pair of rewriting rules (R_1, R_2) such that $s(R_1) \cap s(R_2)$ is a non-empty diagram. In particular, a critical peak can be represented by a diagram ϕ such that:

- $\phi = s(R_1) \circ \psi$;
- $\phi = \psi' \circ s(R_2)$;
- $s(R_1) \cap s(R_2) \neq \emptyset$
- $\phi = \phi_{G_{R_1} \cup G_{R_2}}$ where G_{R_1} and G_{R_2} are respectively the set of gates of $s(R_1)$ and $s(R_2)$;
- No rewriting rules different from R_1 either R_2 can be applied to ϕ.

We are also able to characterize the diagrams representing a critical peak as follows:

Proposition 2.4.5. *If ϕ is a diagram representing a critical peak, then it is a closed term obtained by term substitution of from archetype of conflict ψ:*

$$\phi = \psi[X_1/x_1, \dots X_n/x_n, Y_1/y_1, \dots Y_k/y_k, Z_1/z_1, \dots Z_k/z_h]$$

where $x_1, \dots, x_n$ are internal gates, $y_1, \dots, y_k$ are left-external gates and $z_1, \dots, z_h$ are right-external gates of type in Ξ while $X_1, \dots X_n$, $Y_1, \dots Y_k$, $Z_1, \dots Z_k$ are irreducible, right-$(in(y_i), out(y_i))$ irreducible and left-$(in(z_i), out(z_i))$ irreducible for all i respectively.

Definition 2.4.6 (Global conflict)**.** In a rewriting system $(\mathcal{S}, \mathcal{R})$, a *global conflict* (or *indexed critical pair* [35]) is a family of critical peaks obtained by a term substitution from the same non-closed archetype of conflict. A *reduced global conflict* is its subfamily obtained by a substitution with irreducible terms. A global conflict is solvable if all of its elements are solvable.

Theorem 2.4.7 (Confluence)**.** *A rewriting system $(\mathcal{S}, \mathcal{R})$ is confluent if and only if all of its critical peaks are confluent.*

Corollary 2.4.8. *A rewriting system $(\mathcal{S}, \mathcal{R})$ is confluent if all its archetype of conflicts are confluent.*

Corollary 2.4.9. *It is possible to extend Squier theorem to all convergent rewriting systems $(\mathcal{S}, \mathcal{R})$ whose archetypes are all closed.*

Proof. Since the number of archetypes of a rewriting system is finite and they are all closed, the number of critical peak is finite. Then it suffices to give the construction of the homotopy basis as in the original formulation of the theorem. □

In particular, here we can note where factors play a role in diagram rewriting confluence.

Proposition 2.4.10. *If $\Sigma = (\mathcal{S}, \mathcal{R})$ is a a rewriting system, then:*

1. *If there is a non-closed archetype and Σ has irreducible 0-factor then Σ has an infinite number of non-confluent critical peaks;*

2. *If there is an external-open archetype ϕ_Ξ and Σ has irreducible 1-factor of size k, if $g : \chi \in \Xi$ is an external gate in ϕ_Ξ with $\mathrm{in}(g), \mathrm{out}(g) \leq k$ then Σ has an infinite number of critical peaks.*

Proof. 1. If an archetype is not closed, than any diagrammatic term substitution creates at least one cage. This means that in this cage can be present some dark energy and the global conflict results unsolvable.

2. The existence of the irreducible 1-factor $\{\phi_n\}_{n\in\mathbb{N}}$ implies, by external substitution, that there is an infinite family of critical peaks $\{\phi_\Xi[\phi_n/g]\}_{n\in\mathbb{N}}$. □

For this reason we define our last factor

Definition 2.4.11 (3-Factor)**.** A rewriting system $(\mathcal{S}, \mathcal{R})$ has a 3-*factor* if it has non-closed archetypes of conflicts.

Some examples of these notions are given in Appendix C.

2.5 The coherence of symmetric monoidal categories

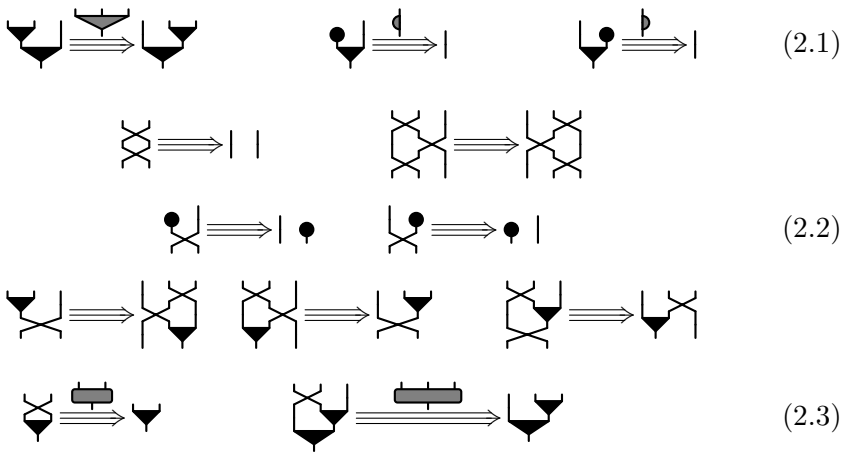
In this section we show an example of the tools presented in the previous sections. We focus on the convergence proof of two diagram rewriting systems $\mathfrak{M}$ and $\mathfrak{F}$ that presents functors and natural isomorphisms of respectively

monoidal category and symmetric monoidal categories and we prove their confluence. In particular, in order to prove the confluence of $\mathfrak{F}$ stated by Lafont in [52], we check its 68 critical peaks. Then, using the method presented in [36], we give a new constructive proof of the coherence theorems using rewriting: since the confluence of our rewriting systems which present functors and natural transformations of these theories is proved, we extend these polygraphs by a set of unoriented 4-cells. This allow us to construct a 4-cell for each pair of unoriented parallel 3-cells corresponding to linear diagram in the theory of symmetric monoidal categories. This method leads to a refinement of Kelly's Lemma 1.3.7 which tell us that the five coherence conditions given are sufficient. Finally we are able to prove the theorems establishing a correspondence between 4-cells and commutative or trivial diagrams.

The coherence theorem for monoidal categories can be interpreted as an equivalence between terms differing on parenthesis dispositions only. As suggested by Huet in [41] (see also Mellie's notes [66]), the natural transformations of this theory can be interpreted as rewriting rules and coherence theorem as the confluence of this system. The main difference between the proof given in this Section and the one given by Mac Lane in [62] concerns the formalism of symmetries in the proof. In the original Mac Lane's proof they are some actions external to the syntax acting on the position of variables inside terms while in our proof they are integrated in the syntax. Moreover, the choice of Lafont's rewriting system instead of Guiraud-Malbos' one suggest that, in this representation of symmetries, the structure they generate can not be separated from the monoidal structure due to the interaction with the bifunctor of the product if we want to achieve the uniqueness of normal forms.

Definition 2.5.1. Let $\mathfrak{F}$ be the rewriting system defined on the signature given by the following gates

and with the following rules:



(2.1)

(2.2)

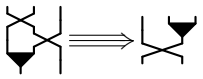
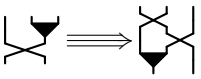
(2.3)

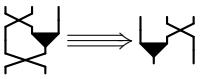
By restriction, we define the rewriting system $\mathfrak{M}$ as the one given by the signature $\{$  $,$  $\}$ and the set of rules (2.1).

Remark 2.5.2. *The set of rule can be derived by the following set of equivalencies:*

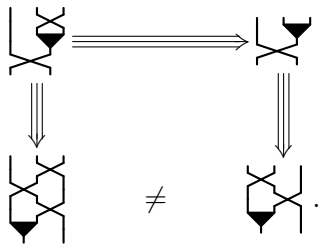
We have divided this set of rules into three subsets: the subset (2.1) consisting of monoidal rules *that are the same rules as $\mathfrak{M}$ which correspond to associativity and left and right unitor, the* structural rules *(2.2) to manage resources and their interaction and the subset (2.3) consisting of rules*  *, corresponding to the natural transformation τ, and*  *which is added to our rewriting system to obtain its confluence.*

Remark 2.5.3. *Lafont's and Guiraud-Malbos' presentations of polygraphs representing symmetric monoidal categories differ in the orientations of the rules 2.2. In particular, the two systems differ for the orientation of*

Lafont:  *Guiraud-Malbos:* 

and the presence in Lafont's system of the rule  *needed to recover confluence.*

Guiraud-Malbos system is not confluent; more precisely, the following conflict is not solvable:



On the other hand, confluence is guaranteed in confluence diagrams concerning the string diagrams representing terms in symmetric monoidal categories, i.e. the ones with just 1 *output. Moreover their method is not constructive and, using the existence of a Tietze-equivalent [13] confluent presentation (the one given by Lafont), they are still able to prove the coherence theorem.*

This leads an important observation:

Remark 2.5.4. *In a theory including a monoidal category, the structure of symmetries can not be separated form the one of product (and unit) whenever they interact without occurring in confluence problems.*

In this particular case, the non-solvability of the exhibited conflict is due to the impossibility of applying the symmetry of the product to the leftmost irreducible diagram because of the presence of other symmetries.

Here, for completeness, we recall the proof of termination for the polygraph $\mathfrak{F}$ given in [52].

Proposition 2.5.5. *The rewriting system $\mathfrak{F}$ is terminating.*

Proof. To prove the termination we use the method given in [] Where any diagram $\phi : p \to q$ in $\mathfrak{F}$ is interpreted as a monotone function $[\phi] : \mathbb{N}^{*p} \Rightarrow \mathbb{N}^{*q}$ where $\mathbb{N}^* = \mathbb{N} \setminus \{0\}$. We give a (product) order on $\mathbb{N}^{*p}$ defined as follow:

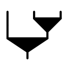
$$(x_1, \ldots, x_p) \geq (y_1, \ldots, y_p) \Leftrightarrow x_1 \geq y_1, \ldots, x_p \geq y_p$$

that we use to define a well founded order on monotone functions: if $f, g : \mathbb{N}^{*p} \to \mathbb{N}^{*p}$ then $f \geq g$ iff $f(x_1, \ldots, x_p) \geq g(x_1, \ldots, x_p)$ for all $(x_1, \ldots x_p) \in \mathbb{N}^{*p}$. Both diagram compositions (sequential and parallel) are compatible with this order: if $\phi, \psi : p \to q$ and $[\phi] \geq [\psi]$ so $[\phi * \chi] \geq [\psi * \chi]$ and $[\chi * \phi] \geq [\chi * \psi]$ for any χ since $\geq$ is a product order and for any $\chi' : q \to q'$ and $\chi'' : p' \to p$, $[\phi \circ \chi'] \geq [\psi \circ \chi']$ and $[\chi'' \circ \phi] \geq [\chi'' \circ \psi]$ since we are considering only monotone maps on positive integers.

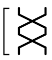
Each gate of the signature can be interpreted as follows:

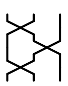
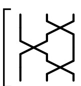
[] $(x, y) \to (y, x + y)$ [] $(x, y) \to 2x + y$ [] $(\emptyset) \to 1.$

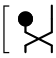
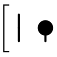
This allow as to associate to any rule $\phi \to \psi$ two monotone maps $[\phi]$ and $[\psi]$ such that $[\phi] > [\psi]$:

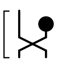
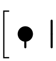
[] $(x, y, z) = 4x + 2y + z > 2x + 2y + z =$ [] $(x, y, z),$

[] $(x) = x + 1 > x = [\,|\,](x),$ [] $(x) = x + 1 > x = [\,|\,](x),$

[] $(x, y) = (2x + y, x + y) > (x, y) = [\,|\ \ |\,](x, y),$

[] $(x, y, z) = (2x + y + z, x + y, x) > (x + y + z, x + y, x) =$ [],

[] $(x) = (x + 1, 1) > (x, 1) =$ [] $(x),$

[] $(x) = (x + 1, x) > (1, x) =$ [] $(x),$

$$\left[\;\right](x,y,z) = (2x+y+z, 2x+y) > (x+y+z, 2x+y) = \left[\;\right](x,y,z),$$

$$\left[\;\right](x,y,z) = (3x+2y+z, x) > (x+2y+z, x) = \left[\;\right](x,y,z),$$

$$\left[\;\right](x,y,z) = (3x+y+z, x+y) > (2x+y+z, y) = \left[\;\right](x,y,z),$$

$$\left[\;\right](x,y) = 3x+2y > 2x+y = \left[\;\right](x,y),$$

$$\left[\;\right](x,y,z) = 4x+2y+z > 2x+2y+z = \left[\;\right](x,y,z).$$

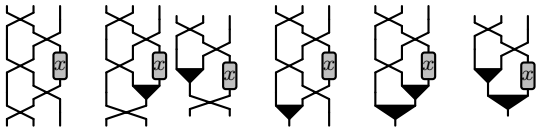
By the compatibility of the order with sequential and parallel composition, this suffice to prove that, for any couple of diagrams, $[\phi] > [\psi]$ holds if $\phi \to^* \psi$. Since there exists no infinite decreasing suite of monotone maps on positive integers, infinite reduction paths can not exist. □

By restriction on the sets of 2 and 3-cells we prove also the termination for $\mathfrak{M}$.

Corollary 2.5.6. *The rewriting system $\mathfrak{M}$ is terminating.*

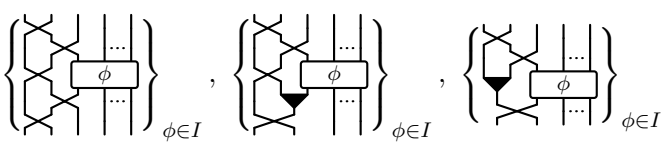
In the original proof of $\mathfrak{F}$ confluence given by Yves Lafont in [52], it is given a method to recognize the 68 critical peaks of this system but is not shown their confluence. Here we detail both these aspects of the proof: we argument the individuation of the critical peaks and we constructively exhibit their solutions. In order to give some practical examples of some notion introduced in the previous section we give a slightly different way express the original method.

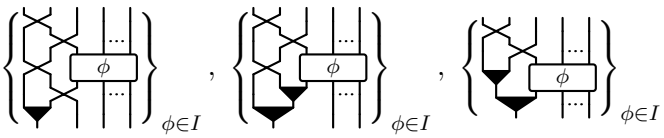
The rewriting system $\mathfrak{F}$ has the following archetypes of conflict:

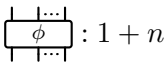


and so

Proposition 2.5.7 ($\mathfrak{F}$'s global conflicts)**.** *In $\mathfrak{F}$, there are six global conflicts*

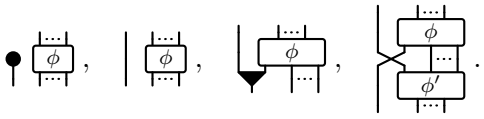




where $I = \left\{ \phi \in \mathfrak{F}^* \mid \phi = \right.$  $\left. : 1+n \Rightarrow 1+m \right\}$.

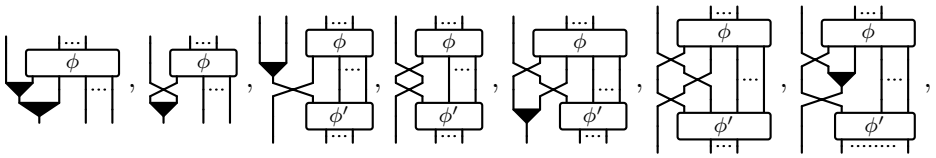
In order to prove $\mathfrak{F}$ (and $\mathfrak{M}$) confluence we need to prove some additional properties:

Lemma 2.5.8. *An irreducible* 2*-cells of* $\mathfrak{F}$ *have one of the following four class of shapes:*



Proof. In order to prove the lemma, we observe how the rewriting rules act on a 2-cell belonging in one of the above class: a 2-cell of the first two classes can be rewritten only in a 2-cell of the same class, a 2-cell of the third one can be rewritten in one of the first three classes while a 2-cell of the fourth in any of these classes.

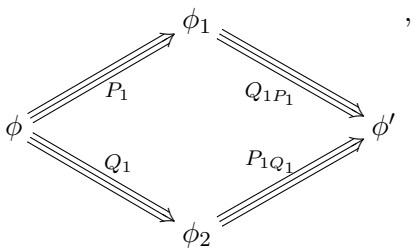
Moreover, these are the only possible configurations where the number of gates of a 2-cell we cross following the left-most path going from the left-most input to the left-most output is less than two. If this condition is not satisfy, we note that a 2-cell can not be irreducible. In fact, by induction over the number of gates in a 2-cell, the possible shapes with irreducible context are the following:



which result all reducible. □

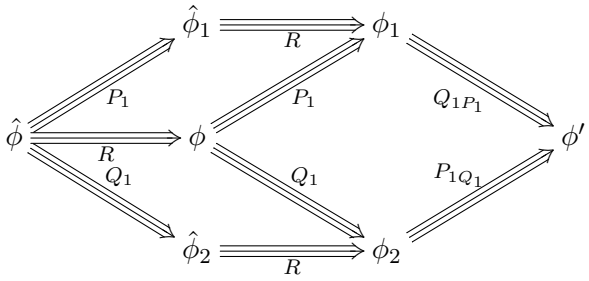
Lemma 2.5.9 (Global conflict solution)**.** *In a confluent rewriting system* $(\mathcal{S}, \mathcal{R})$ *all critical peaks in a global conflict are confluent if and only if all reduced global conflicts are.*

Proof. The left-to-right implication is trivial. In order to prove the right-to-left implication it suffice to remark that every 2-cell $\hat{\phi}$ associated to a of a critical peak in a global conflict reduces, by the convergence, to a reduced global conflict ϕ. If the latter have admit the following confluence diagram:



,

then the confluence diagram of $\hat{\phi}$ is the following:



.

□

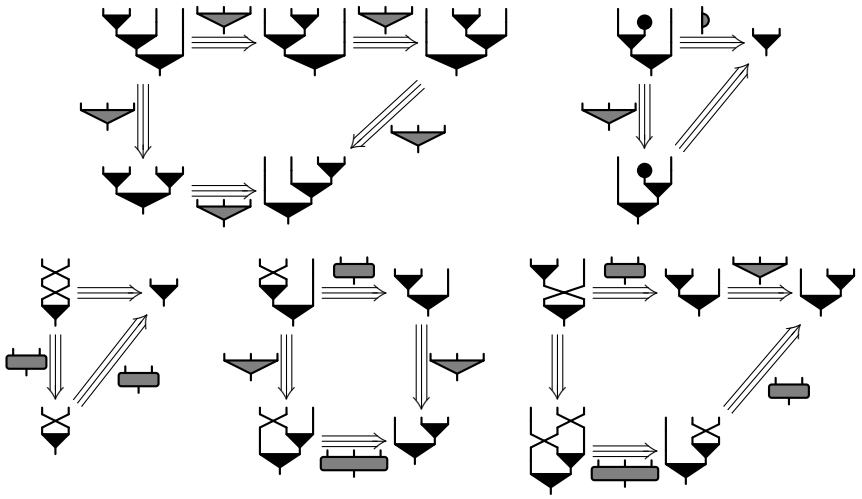
Proposition 2.5.10. *The rewriting system $\mathfrak{F}$ is convergent.*

Proof. The rewriting system is terminating so it suffice to prove the local confluence in order to have convergence. After Proposition 2.5.7 and Lemma2.5.8, by Lemma 2.5.9, the local convergence is proved verifying the following 68 critical peaks. We will handle them in batches:

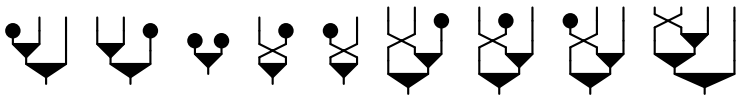
- 5 non-trivial *base* peaks

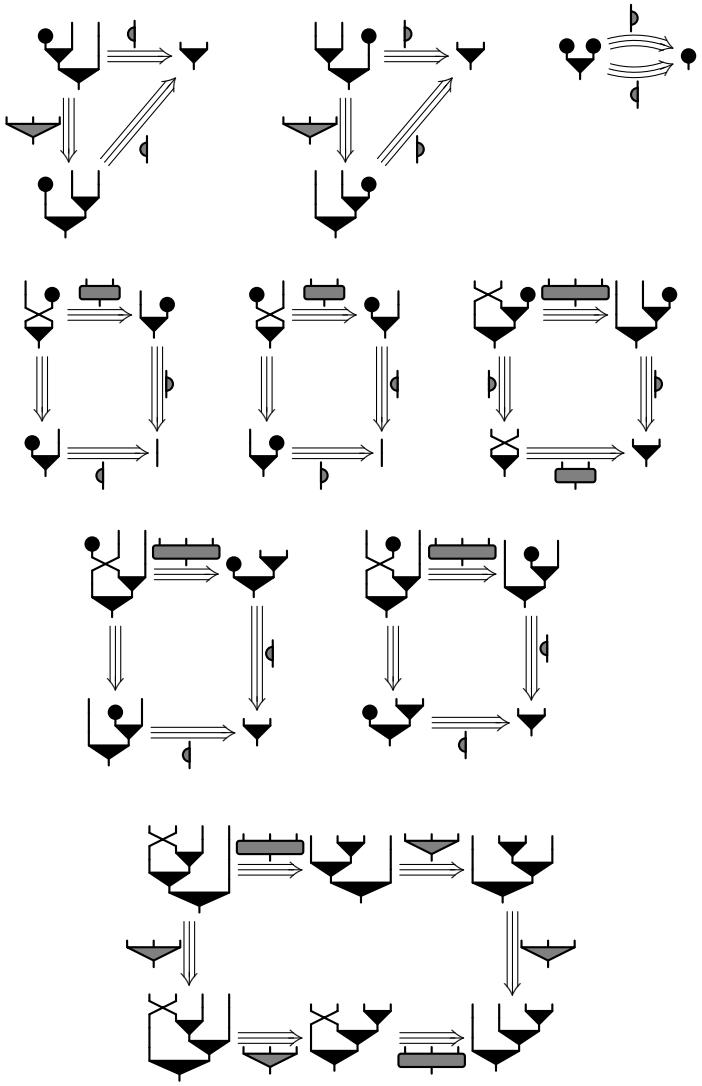
  these correspond to coherence conditions except the last two which define the natural transformation needed for confluence;



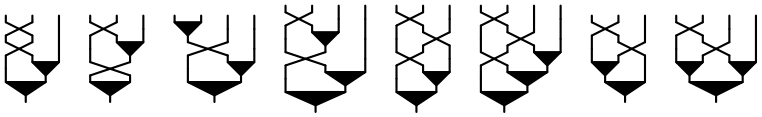
- 9 non-trivial *Kelly*-peaks:



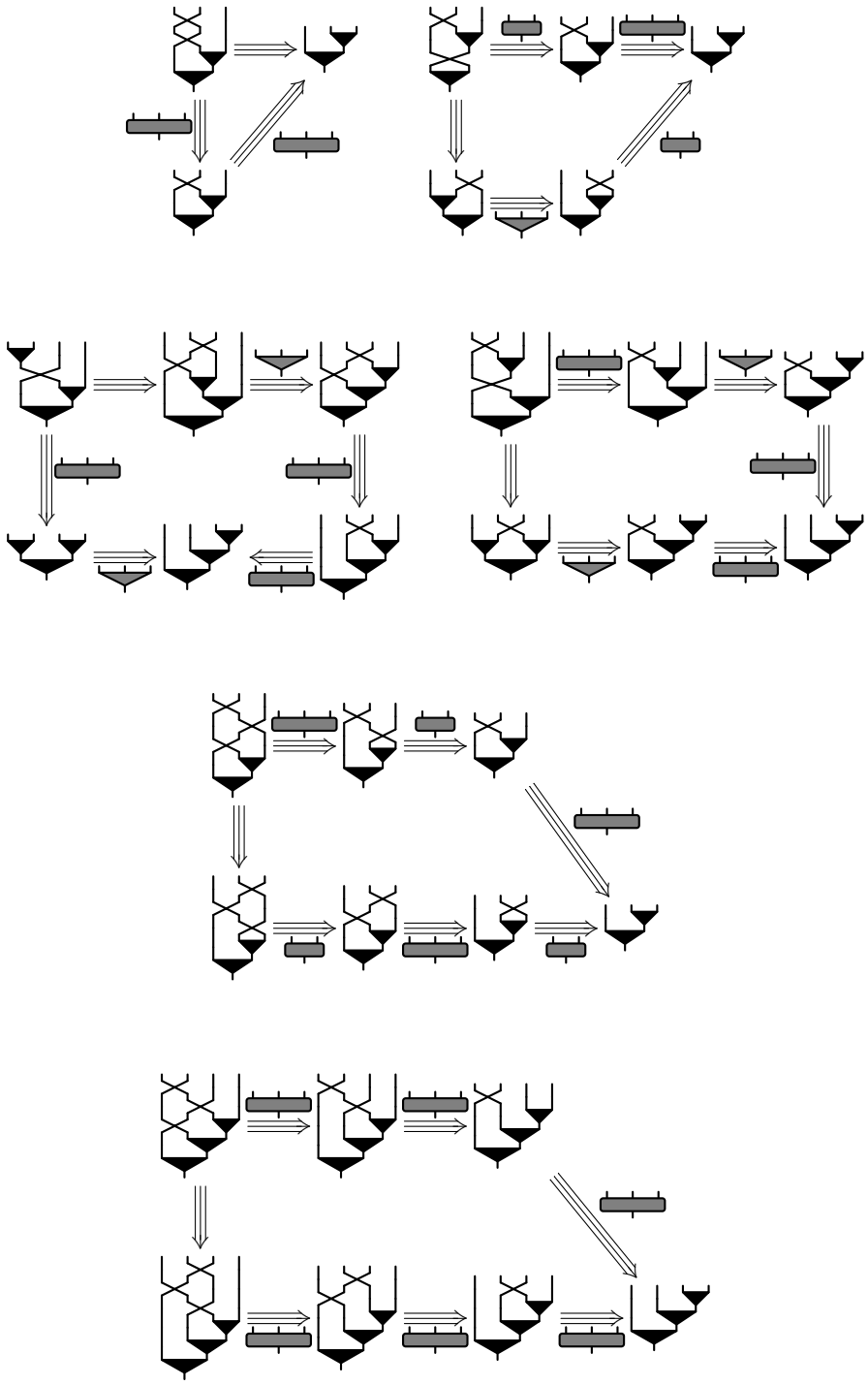
each conflict is generated by two non-structural rules and the confluence of diagrams is given by non-structural rules;

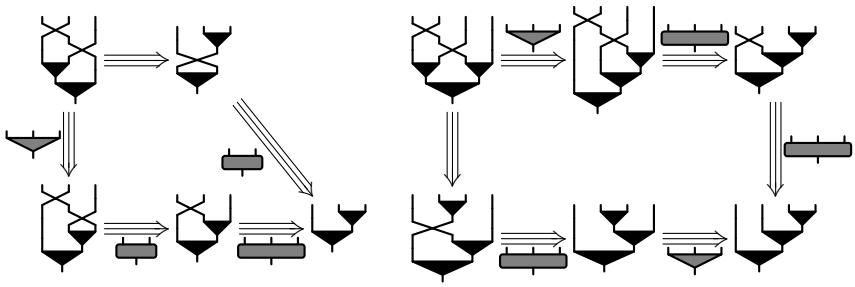


- 8 non-trivial *weak-Kelly*-peaks:

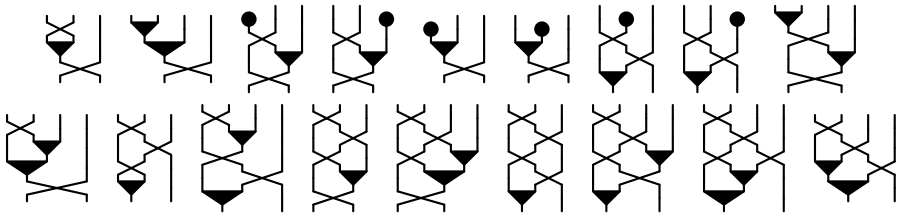


each conflict is generated by a structural rules and a non-structural rules, the confluence of diagrams are given by non-structural rules;

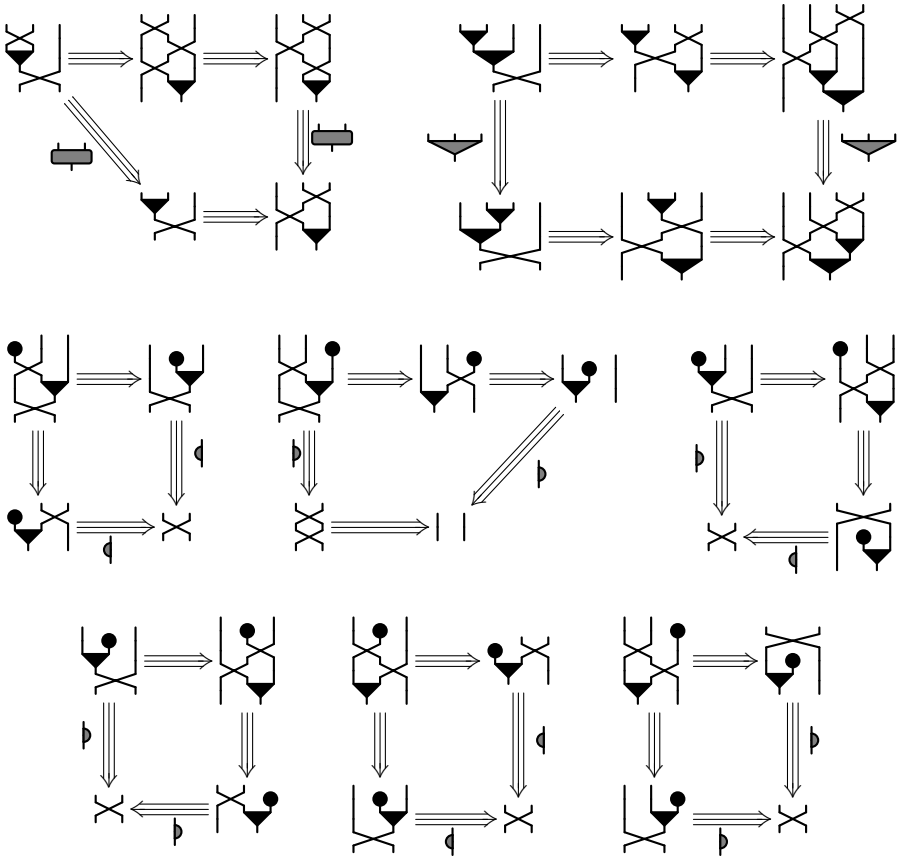


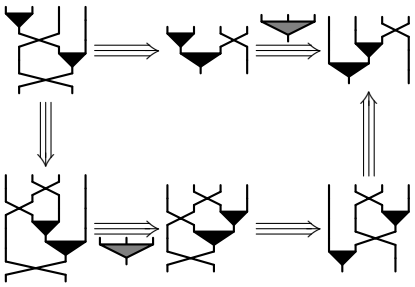


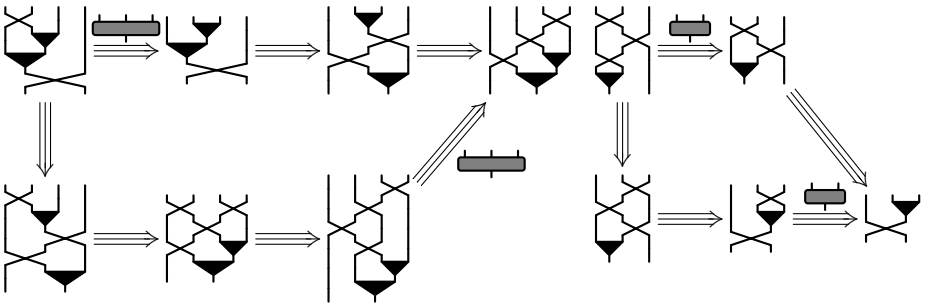
- 18 *simply trivial* peaks:

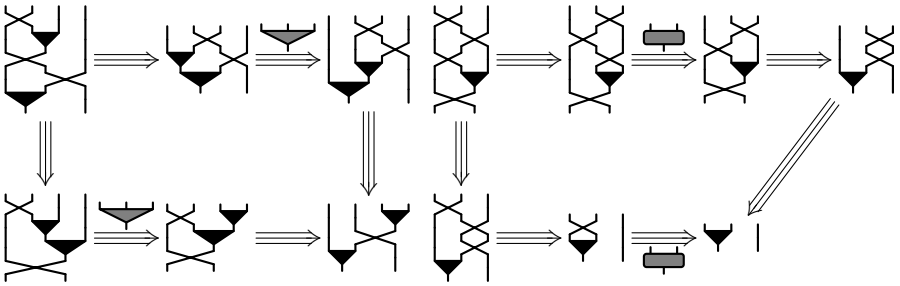


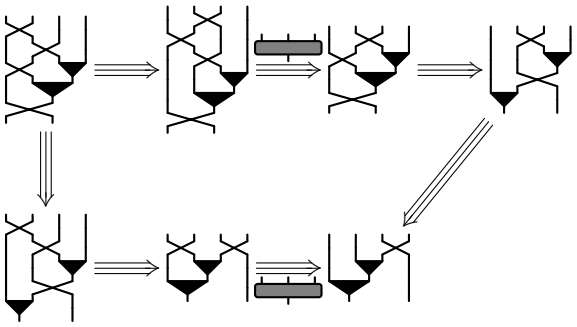
the confluence diagrams are made of parallel 3-cells consisting of the same non-structural rule modulo structural ones;

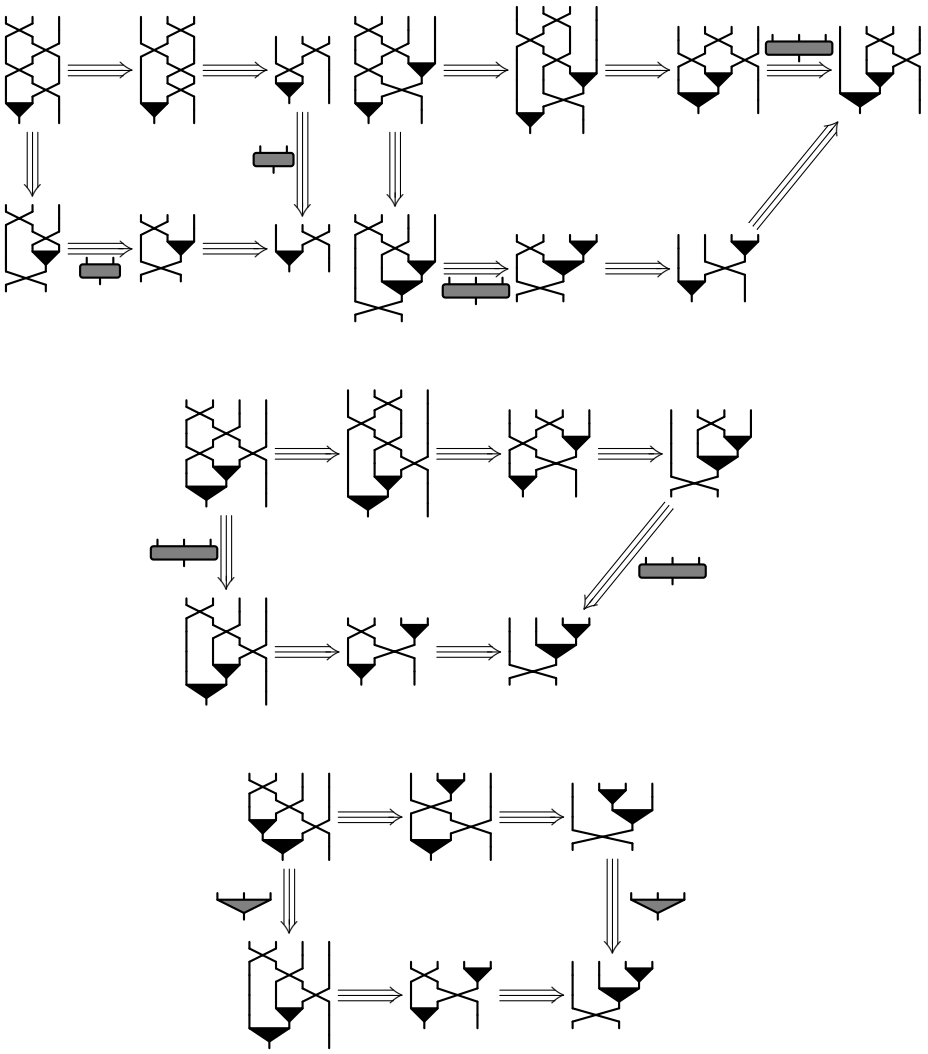




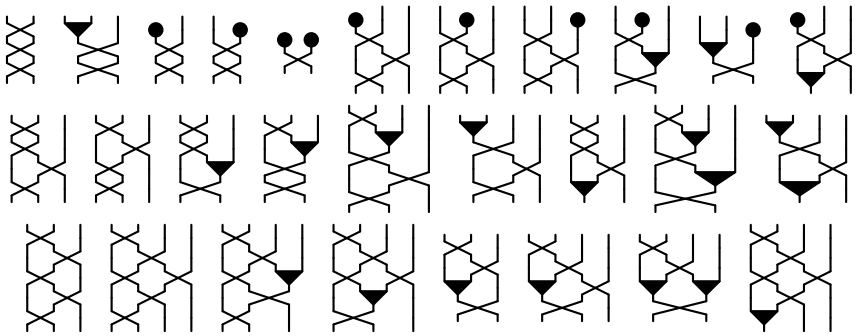




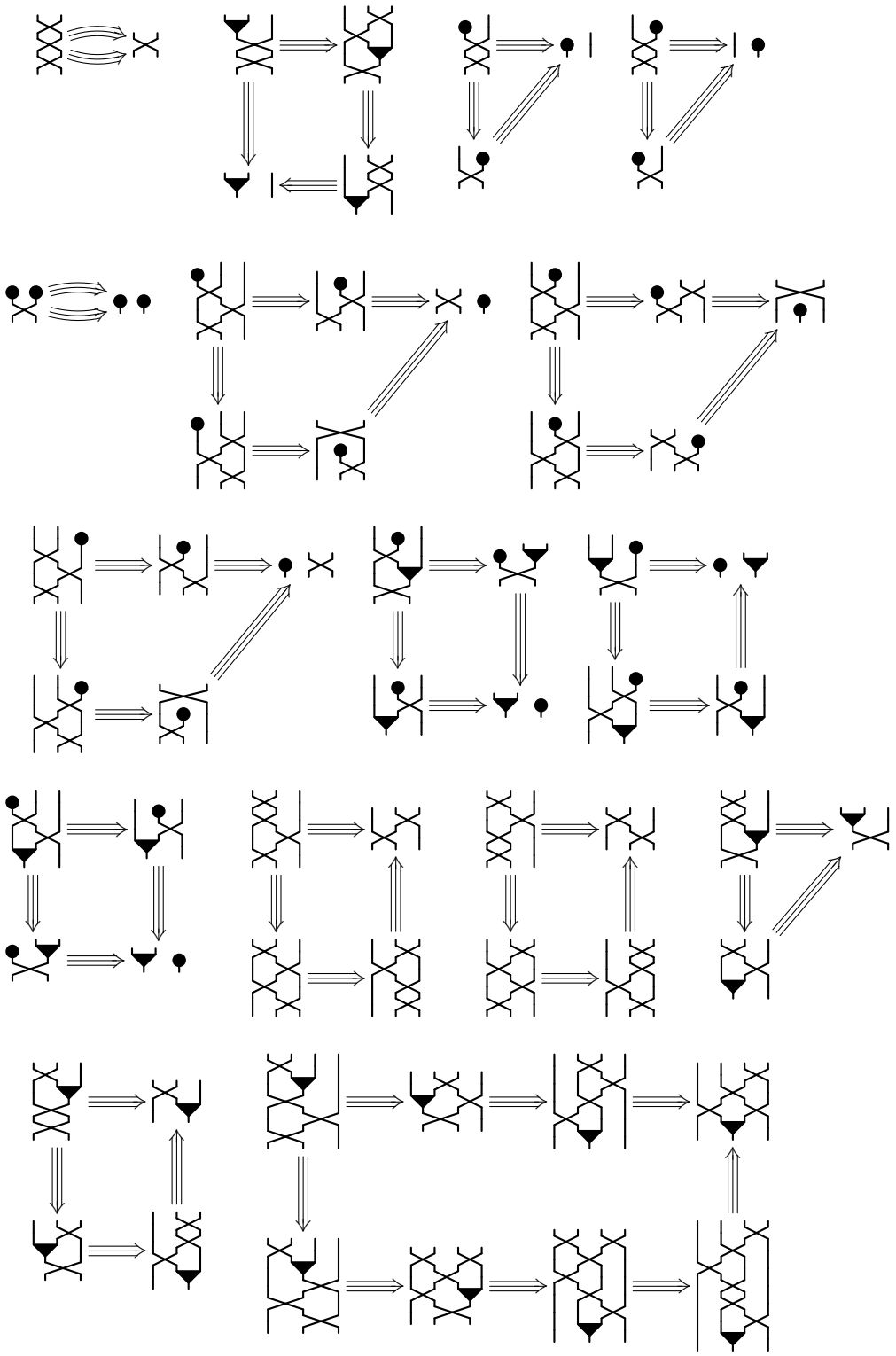


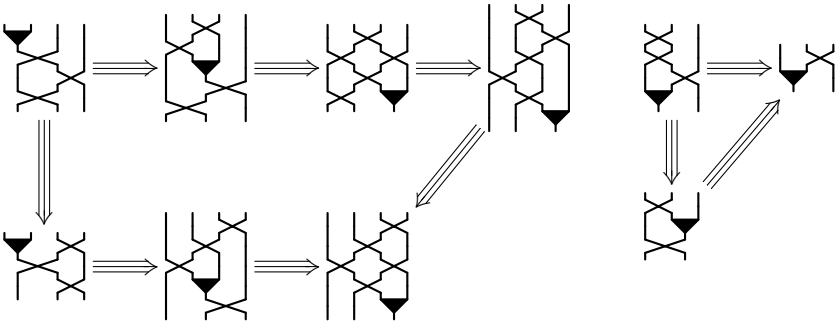


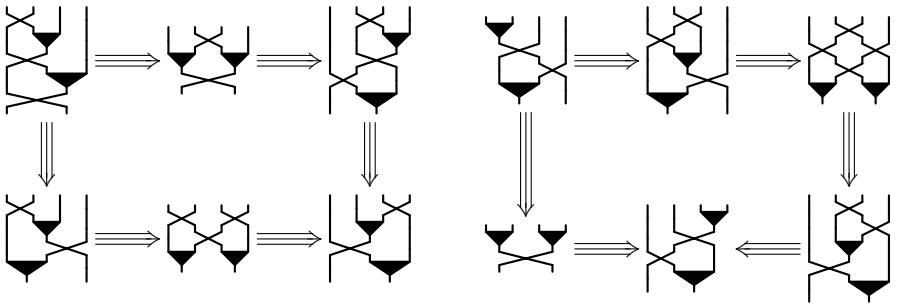
- 28 *strongly trivial* peaks:

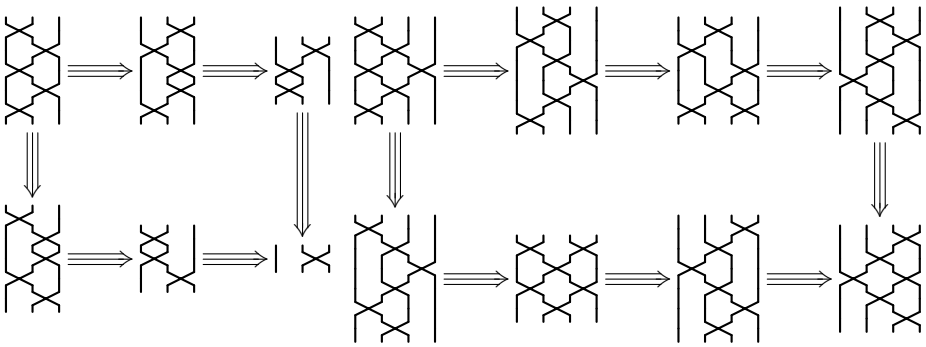


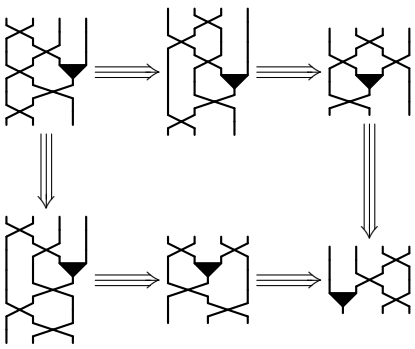
all the rules in confluence diagrams are made of structural rules:

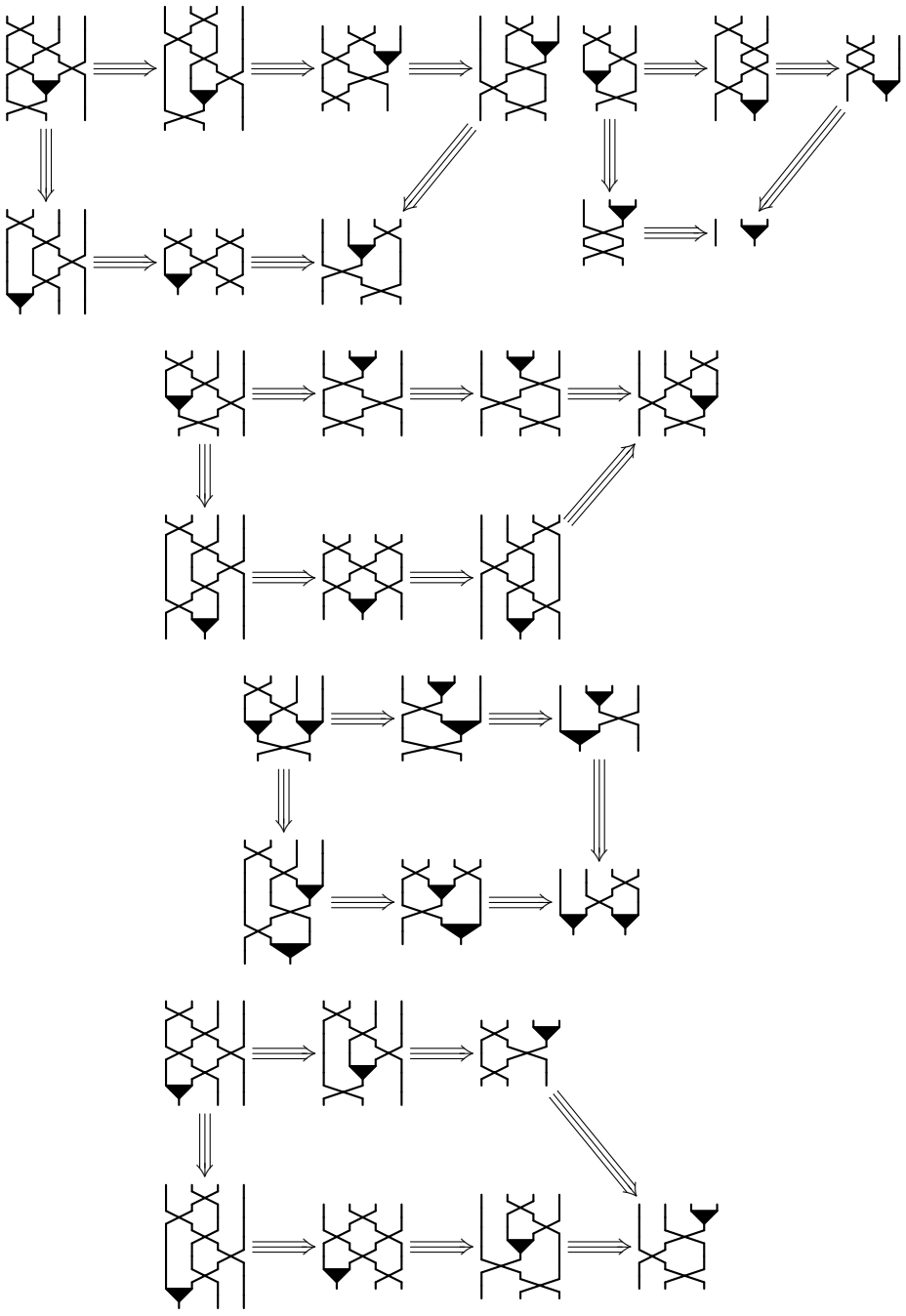












□

It follows:

Theorem 2.5.11. *The rewriting systems $\mathfrak{F}$ is convergent.*

Proof. The confluence of $\mathfrak{F}$ follows Propositions 2.5.10 and 2.5.5. □

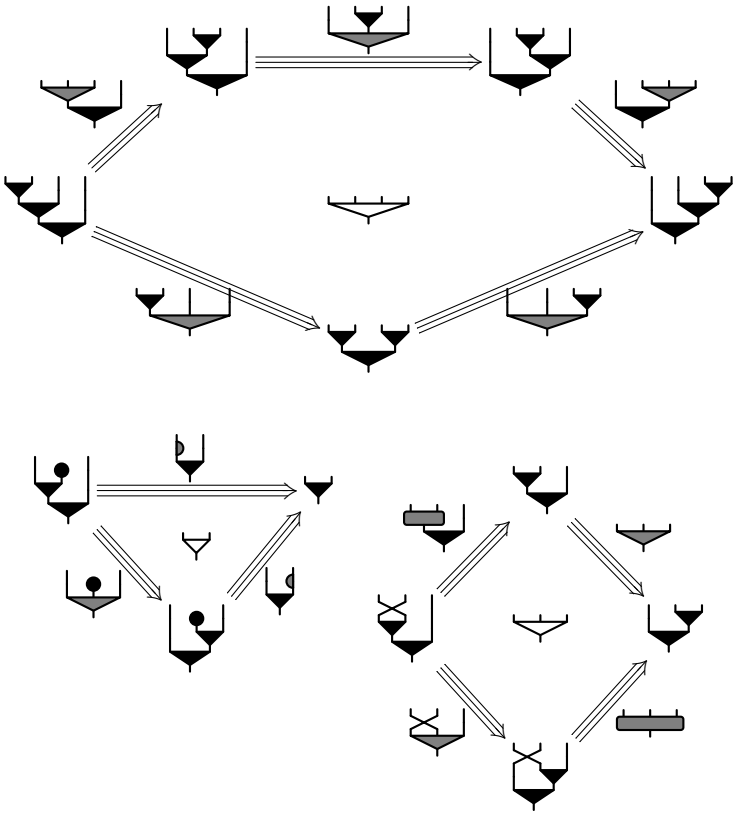
Corollary 2.5.12. *The rewriting systems $\mathfrak{M}$ is convergent.*

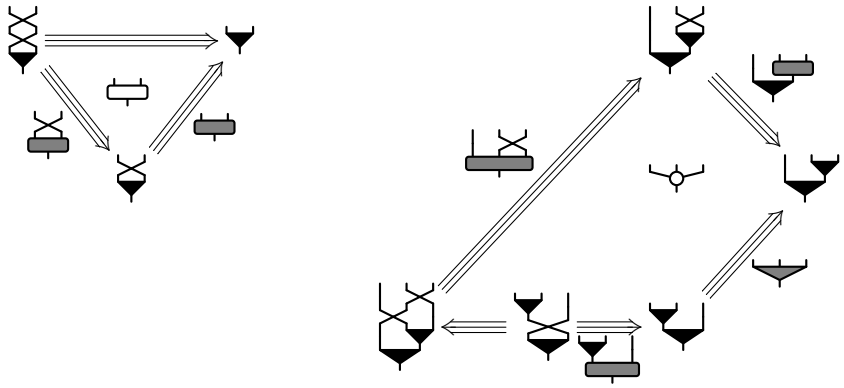
Proof. The confluence of $\mathfrak{M}$ can be evinced by Proposition 2.5.10 since the solutions given for its critical peaks

are made only of 3-cells in $\mathfrak{M}$. This property, together with termination proved in Corollary 2.5.6, permits to prove the statement. □

In order to use the confluence of these rewriting systems, we need to extend these polygraph by a set of 4-cells. Since these cells represent identities, the natural setting for this extension is the one of (n, k)-*polygraph*. We define some non-oriented 4-cells representing identities corresponding to some coherence conditions.

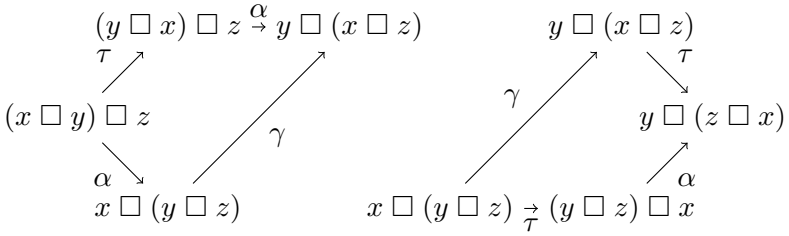
Definition 2.5.13. In $\mathfrak{F}$, we define the following 4-cells with border the solution of the five *base* critical peaks:





We denote $\mathfrak{F}^+$ the $(4,3)$-polygraph $\mathfrak{F}$ enriched by these non-oriented 4-cells.

Remark 2.5.14. *The rule*  *corresponds to an extra (superfluous) natural isomorphism that we add to symmetric monoidal category theory. It is defined by the commutativity of one of the two following diagrams*

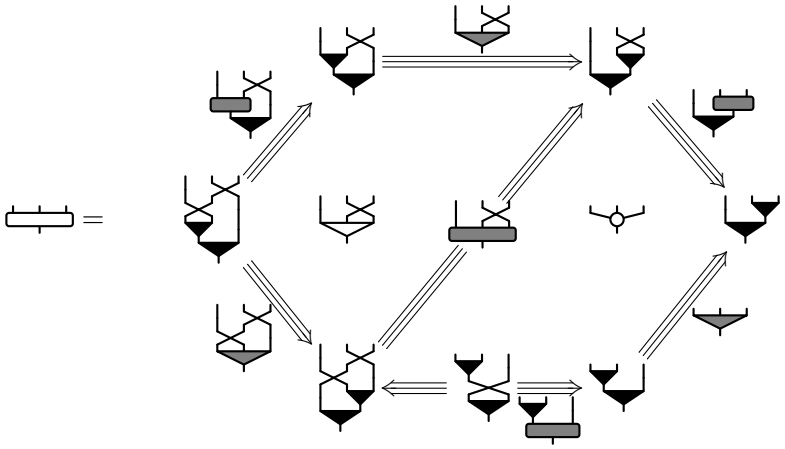


Since the hexagonal identity is defined in the theory and it represents the commutativity of the diagram obtained merging the two above diagrams along the side γ, if we chose the first one as definition of $\gamma = \alpha\tau\alpha^{-1}$, the commutativity of the second one trivially follows.

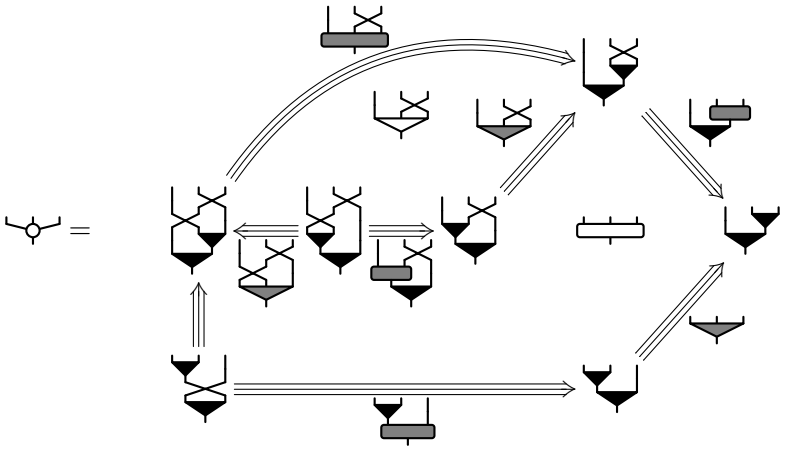
On the other hand, in Mac Lane's definition of symmetric monoidal categories the natural transformation corresponding to the 3-cell  is not defined and together with the two coherence conditions relative to  and  . At their place we have the hexagonal identity which should correspond to a 4-cell  . The border of this latter can not be represented by a pair of parallel 3-cells of our $(4,3)$-polygraph as shown in the following Lemma.

Lemma 2.5.15. *The 4-cells*  *and*  *allow one to define a 4-cell*  *with border made of non-parallel 3-cells interpretable as the hexagonal identity:*

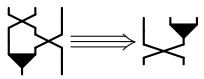
Proof.



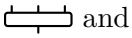
and so



□

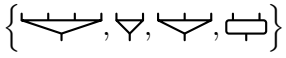
By this fact, we should extend our construction to $(4,2)$-polygraphs as the cases studied in [36] and [60] in order to be able to define this cell as atomic. On the other hand, with this construction we weaken our setting more than how we really need. With a more accurate observation, the only 3-cell we ask to be invertible are the one corresponding to twisting relations (see Definition 1.4.11) which correspond to trivial identities whenever we interpret 2-cells by functors. In the particular case given above, we ask the invertibility of the 3-cell  to be able to recover a good orientation of the border of the 4-cell  .

This means that, once we give the 4-cell , if we give the 4-cell then a 4-cell with the same border of can be derived. In our rewriting

system we choose to keep the two 4-cells  and  instead of  and  for some reasons related to the prove of coherence theorem.

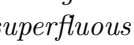
This construction reminds Kelly's lemma for monoidal categories [48] which assert that just three of the five coherence conditions originally given by Mac Lane, corresponding the rewriting system conflicts, suffice to prove coherence. In the same way, we can prove that the five coherence conditions given in symmetric monoidal categories suffice. To prove this result, we construct a 4-cell for any Kelly and weak-Kelly critical peaks (corresponding to non-trivial conflicts in the theory) using only the 4-cells defined by the coherence conditions together with the 4-cells which have a trivial interpretation in the theory.

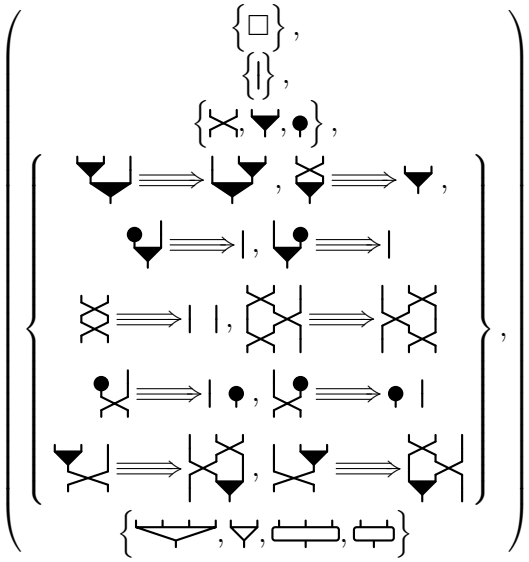
Proposition 2.5.16 (Refinement of Kelly's lemma for symmetric monodal category)**.**
For all Kelly-peaks *and* weak-Kelly-peaks*, a* 4*-cell can be defined from set of* 4*-cells*



plus the set of 4*-cells with border a solution of a trivial or strongly-trivial critical peak (we note them with the symbol* $\odot$*).*

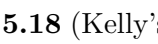
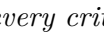
Proof. See Appendix B for the proof. □

Remark 2.5.17. *In this proof the* 4*-cell*  *corresponding to the hexagonal coherence condition is not needed. On the other hand, we use the* 4*-cell*  *which can be derived by an extended higher-dimensional Knuth-Bendix completion [39] when we introduce the superfluous* 3*-cell*  *in the* $(4,2)$*-polygraph*



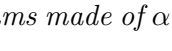
Tietze-equivalent [83] to $\mathfrak{F}$ *by Remark 2.5.2.*

From this Proposition we can recover an alternative proof of the original Kelly's Lemma 1.3.7.

Corollary 2.5.18 (Kelly's lemma)**.** *The 4-cells*  *and*  *suffice to define a 4-cell for every critical peak of* $\mathfrak{M}$*:*

Using the $(4,3)$-polygraph given by $\mathfrak{M}$ enriched with the two non-oriented 4-cells  and  , we are able to prove the original Mac Lane theorem.

Theorem 2.5.19 (Coherence theorem for monoidal categories, Mac Lane [61])**.**
Every diagrams made of α, λ, ρ *in an arbitrary monoidal category* $\mathcal{C}$ *commutes.*

Proof. We observe that all 2-cells of $\mathfrak{M}$ with one output can be interpreted by functors in a monoidal category. Moreover, the rewriting system $\mathfrak{M}$ has exactly one 3-cell for each natural transformation in a monoidal category, which guarantees a one-to-one correspondence with pairs of parallel 3-cells with target and source a one output 2-cells. The two coherence conditions defined by the triangular and the pentagonal identities correspond to the 4-cells  and  and, by Cor.2.5.18, they suffice to generate all 4-cells with border any critical peak. Moreover, the convergence of $\mathfrak{M}$ (Prop.2.5.12) allows one to find (using Lemma 2.5.9) a 4-cell with border any pair of parallel 3-cells just using the aforementioned 4-cells.

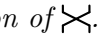
Finally, the commutativity of each diagram in the monoidal category theory follows by the correspondence between confluence diagrams and parallel 3-cells and, in particular, between commutative diagrams and 4-cells. □

This theorem can be interpreted as follows: any object of a monoidal category a can be transformed by means of natural transformation to an object of the form

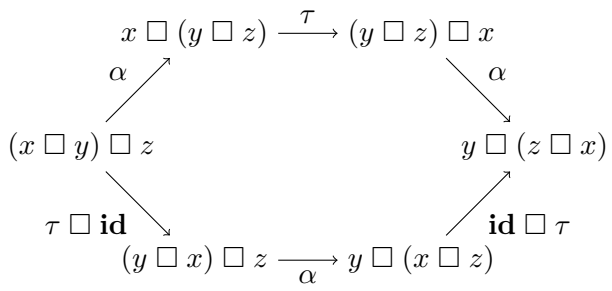
$$(x_1 \square (\dots \square (x_{n-2} \square (x_{n-1} \square x_n)) \dots)$$

where $x_i \neq e$ for all $i = 1, \dots, n$, that is where we associate to the right. In fact, rules move a gate of type  to the right moving the corresponding parenthesis positions to the right and the rules which erase an occurrence of a gate  together with the relative gate correspond to the elimination of an occurrence of e and the relative parenthesis.

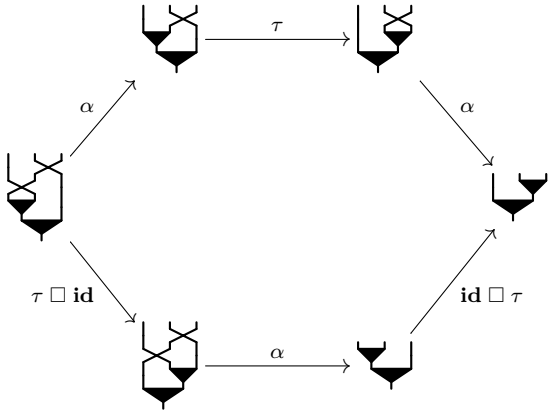
In order to apply a similar method to prove coherence for symmetric monoidal categories, we need the following lemma in order to prove that we can represent a confluence diagram of this theory by some 3-cells in $\mathfrak{F}$:

Lemma 2.5.20. *For every confluence diagram in a symmetric monoidal category, there is a pair of non-oriented* 3*-cells representing it modulo the standard interpretation of*  *.*

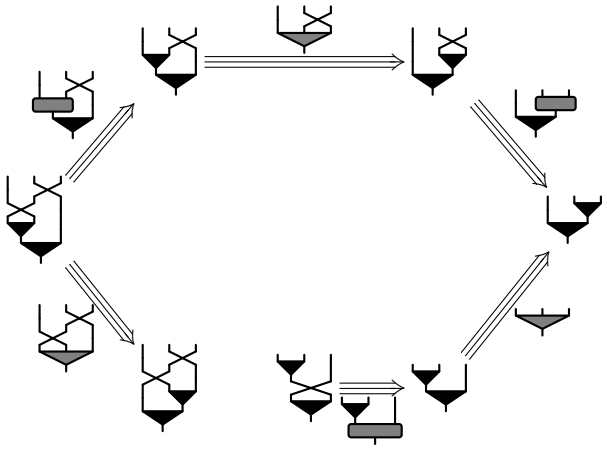
Proof. Given a confluent diagram we want to prove its commutativity, this pair of can be found in the following way, as example we show this algorithm works on the hexagon identity:



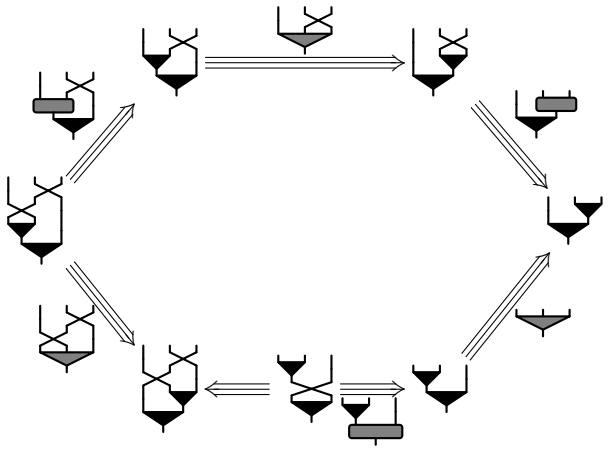
- for all vertexes of the diagram, we take a 2-cell which is irreducible modulo the rewriting rules (2.2) of $\mathfrak{F}$ which represents the corresponding object;



- for all arrows of the diagram, we draw the corresponding 3-cell (if α, if ρ, if λ, if τ) adding, when needed, their source or target 2-cells;



- if some 2-cells has been added, then there exist a rewriting path made of 3-cells in (2.2) from these latter to the one given in the first step since both can be interpreted by the same object in the confluence diagram;



□

If we consider the $(4,3)$-polygraph $\mathfrak{F}^+$ as a $(4,2)$-polygraph , we are able to give a proof of Mac Lane theorem for symmetric monoidal categories.

Theorem 2.5.21 (Coherence of symmetric monoidal categories, [62])**.** *In an arbitrary symmetric monoidal category, every linear diagram made of λ, ρ, α and τ is commutative.*

Remark 2.5.22. *The linearity of diagrams is necessary for the statement of this theorem since in a symmetric monoidal category, we have diagrams such as*

$$x \,\square\, x \underset{\tau}{\overset{\boldsymbol{id}}{\rightrightarrows}} x \,\square\, x$$

made of parallel arrows which are not equal in general.

Proof. As in the previous theorem, in order to prove the commutativity of all linear diagrams of symmetric monoidal category theory, we want to use the fact that for every pair of parallel 3-cells of the rewriting system $\mathfrak{F}$ a 4-cell in the $(4,2)$-polygraph $\mathfrak{F}^+$ having such pair as border is defined from the set of atomic 4-cells corresponding to the coherence conditions.

On the other hand, the previous method have to be adapted since, in this case, not every pair of parallel 3-cells in $\mathfrak{F}$ naturally corresponds to a linear diagram of the theory even if their source and target 2-cells have one input. This incongruence is due to the fact that the 2-cell $\times$ does not correspond to any natural isomorphism in the symmetric monoidal category

theory. Its introduction is due to the fact that, in general, in order to represent an *algebraic theory* through diagrams [49], we need some operators to manage resources: *duplication*, *erasing* and *permutation* (in this particular case we only consider linear diagrams, so we need no duplication nor erasing). We give a *standard interpretation* of  in the sense that we consider an equality between the pairs (x, y) and (y, x). This lead the identification of the interpretation of some diagrams corresponding to the same term and, consequently, the trivial interpretation (by an identity) of a 3-cell between them. By this fact, if a pair of parallel 3-cell is solution of a semi-trivial or trivial peak, the 4-cell associated to it corresponds to a trivial commutative diagram (no natural isomorphisms or two times the same one). This is reason why, in order to simplify the notation, we have chosen to note all such 4-cells with the same symbol  .

Since $\mathfrak{F}$ is confluent (Propositions 2.5.11), once given a set of 4-cells with border the solutions of all critical peaks, one can find a 4-cell with border each pair of parallel 3-cells by lemma 2.5.9. Moreover, by Proposition 2.5.16 this set of 4-cells can be restrict to the set

plus the set of 4-cells we denote with , i.e. with border a solution of a trivial or strongly-trivial critical peak. These cells correspond respectively to pentagonal, triangular and τ-involutivity coherence conditions, the one defining the natural transformation γ plus some trivial coherence conditions.

Even if some linear diagram can not be represented by a pair of parallel 3-cells in the $(4, 3)$-polygraph $\mathfrak{F}^+$ by Lemma 2.5.20, we can always found a pair of parallel 3-cells in $\mathfrak{F}^+$ if we consider it as a $(4, 2)$-polygraph such that, by the standard interpretation of , represent the given diagram. Any such diagram, can be decomposed whenever a critical pair appear in order to define a 4-cell, as done for the decomposition of the hexagon identity given in Lemma 2.5.15.

The commutativity of each linear diagram of symmetric monoidal category theory follows by the correspondence between 4-cells  ,  ,  ,  ,  and commutative (or trivial) diagrams in this theory.

□

This theorem have the same interpretation of the one for monoidal categories: any object of a symmetric monoidal category a can be transformed by means of natural transformation to an object of the form

$$(x_1 \square (\dots \square (x_{n-2} \square (x_{n-1} \square x_n)) \dots)$$

where $x_i \neq e$ for all $i = 1, \dots, n$, that is where we associate to the right.

The rules concerning the monoidal structure only act on the shape of therms in the same way. Moreover, the other rules do not change it since they

perform modifications over the shape of symmetries and they move gates of type to the right or gates of type  downward (until they reach a gate of type to be canceled).

Chapter 3

String diagrams for linear logic

"We have taken the word, the sentence, logic and number as the foundation stones of our civilisation, forcing our brains to use limiting modes of expression which we assume are the only correct ones."

[Tony Buzan. *The mind map book* (1993)]

In this chapter we give a new diagrammatic syntax for linear logic proofs by means of *proof diagrams*. As the primal scope of proof net is to give a 2-dimensional representation of proofs which helps reasoning thanks to the fact that they represent equivalence classes of proofs (see Appendix A). Moreover we equip this syntax with an intuitive semantics, in order to be able to represent with a diagram a class of equivalent proofs.

Jean-Yves Girard's *proof nets* are the first graphical syntax for proof theory introduced for linear logic multiplicative sequent calculus [28]. Their underline syntax, formalized by Yves Lafont's *interaction nets*, comes from graph theory and reminds by their semantical properties the use of Roger Penrose's diagrams for equation representation. On one hand proof nets allow to represent sequent calculus derivations allowing to represent by the same net equivalent proofs. On the other hand, this syntax requires *correctness criteria* in order to check if a term correspond to a correct proof. In particular, in order to accommodate multiplicative units, we should add some additional linking in the nets which rules out the correspondence between proof nets and equivalence classes of proofs. The same problem rises in the representation of exponential inference rules together with the necessity of introducing *boxes* in order to verify well-typing of *promotion rule*.

In order to familiarize with the syntax and semantic of proof diagrams and to underline the differences with proof nets, we firstly give some polygraphs that reproduce the graphical notation of proof structures with an explicit notation for string crossings. Although proof diagrams representation reminds

proof net one, these have a more sequentialized structure. Due to the interchange rule together with the semantic of *twisting relations*, this semantics is immediately compatible with all the rule commutations captured by the equivalence relation over proofs induced by proof nets.

Then we extend this syntax with two *non-twisting* string types which can be interpreted as a 2-dimensional denotation for parenthesis. In some sense, this concept is already used in proof nets boxes notation where a portion of a net is isolated; in proof diagrams this become part of the syntax making possible to represent correct application of inference rules only. This means that, when control strings are included in proof diagrams syntax, we are not able to represent terms corresponding to non-sequentializable proof structures.

The definition of some diagram rewriting rules allows us to recover a semantic including any rules commutation which does not change splitting order, these are all commutation different from the following:

$$\dfrac{\dfrac{\vdots \quad \vdots}{\vdash \Gamma, A, B \qquad \vdash \Gamma', C}\otimes_1 \qquad \vdots}{\vdash \Gamma, \Gamma', (B \otimes_1 C), A \qquad \vdash \Gamma'', D}\otimes_2 \quad \sim \quad \dfrac{\dfrac{\vdots \quad \vdots}{\vdash \Gamma, A, B \qquad \vdash \Gamma'', D}\otimes_2 \qquad \vdots}{\vdash \Gamma, \Gamma', (A \otimes_1 D), B \qquad \vdash \Gamma', C}\otimes_1$$
$$\vdash \Gamma, \Gamma', \Gamma'', (A \otimes_1 D), (B \otimes_2 C) \qquad\qquad \vdash \Gamma, \Gamma', \Gamma'', (A \otimes_1 D), (B \otimes_2 C)$$

and

$$\dfrac{\vdots \qquad \dfrac{\vdots \quad \vdots}{\vdash \Gamma', C \qquad \vdash \Gamma, A, B}\otimes_1}{\vdash \Gamma'', D \qquad \vdash \Gamma, \Gamma', (C \otimes_1 B), A}\otimes_2 \quad \sim \quad \dfrac{\vdots \qquad \dfrac{\vdots \quad \vdots}{\vdash \Gamma'', D \qquad \vdash \Gamma, A, B}\otimes_2}{\vdash \Gamma', C \qquad \vdash \Gamma, \Gamma', (D \otimes_1 A), B}\otimes_1$$
$$\vdash \Gamma, \Gamma', \Gamma'', (D \otimes_1 A), (C \otimes_2 B) \qquad\qquad \vdash \Gamma, \Gamma', \Gamma'', (D \otimes_1 A), (C \otimes_2 B)$$

ruling out the cut-elimination theorem due to the impossibility of removing commutative cuts.

In order to have a model respecting the whole semantics of equivalent proofs, we finally extend our model in order to include some rewriting rules corresponding to a normalization procedure on proof derivation trees. This eliminates commutative cuts by re-ordering the axioms and 1 positions according to the order of splitting rules. In this model we define a cut-elimination procedure proving the cut-elimination theorem for proof diagrams.

3.1 Proof diagrams for *MELL*

In this section we give two particular 3-polygraphs for *MELL* and its subfragments, i.e. string diagrams representing linear logic derivations that we call *proof diagrams*.

In describing these polygraphs, we give them by means of some class of cells which are indexed by the linear logic formulas of the sequent calculus

we want to represent. We start each construction of one of these two polygraphs from the set of cells we need to represent proof of the multiplicative fragment of linear logic. Then we extend it to the one representing the multiplicative fragment with units and, finally, to the polygraph representing the multiplicative and exponential fragment (with unit).

In our construction, each extension is given by the sets of cells representing the new rules of the fragment and their interactions. However we have to keep in mind the presence of an abuse of notation motivated by the choice of avoiding heavy notations or redundant definitions: even if we say that we include some sets of cells of the previous polygraph, this is not sound. In fact, in order to give a complete definition, we should extend the sets of formulas indexing these sets of cells to the set of formulas of the fragment we consider.

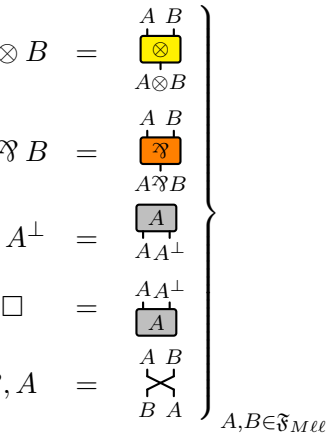
Notation. In order to unify sequent and 1-cell composition notations, we replace the $*$ symbol of parallel composition with a comma.

Notation. When not needed we will not explicitly specify the type indexes of a gate type, for example denoting $g : \otimes$ instead of $g : \otimes_{A,B}$.

Definition 3.1.1. The 3-polygraph Σ_{MLL} is the *polygraph of multiplicative linear logic with cut-elimination*. It is given by the following sets of cells:

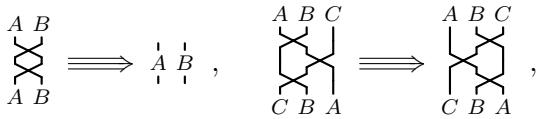
- $\Sigma_0^M = \{ \Box \}$;
- $\Sigma_1^M = \mathfrak{F}_{M\ell\ell}$;
- $\Sigma_2^M =$

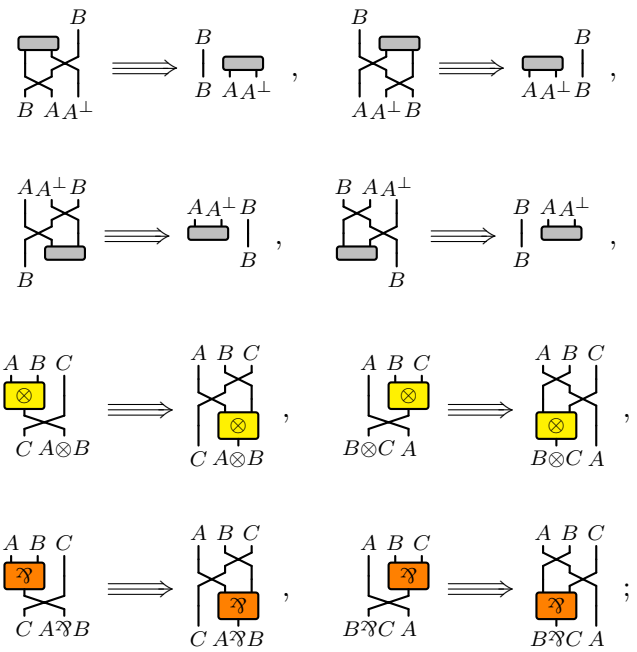
$$\left\{ \begin{array}{lccccc} \otimes_{A,B} : & A, B & \Rightarrow & A \otimes B & = & \\ \parr_{A,B} : & A, B & \Rightarrow & A \parr B & = & \\ Ax_A : & \Box & \Rightarrow & A, A^\perp & = & \\ Cut_A : & A, A^\perp & \Rightarrow & \Box & = & \\ T_{A,B} : & A, B & \Rightarrow & B, A & = & \end{array} \right\}_{A,B \in \mathfrak{F}_{M\ell\ell}}$$



If there is no ambiguity we note  and  instead of  and  ;

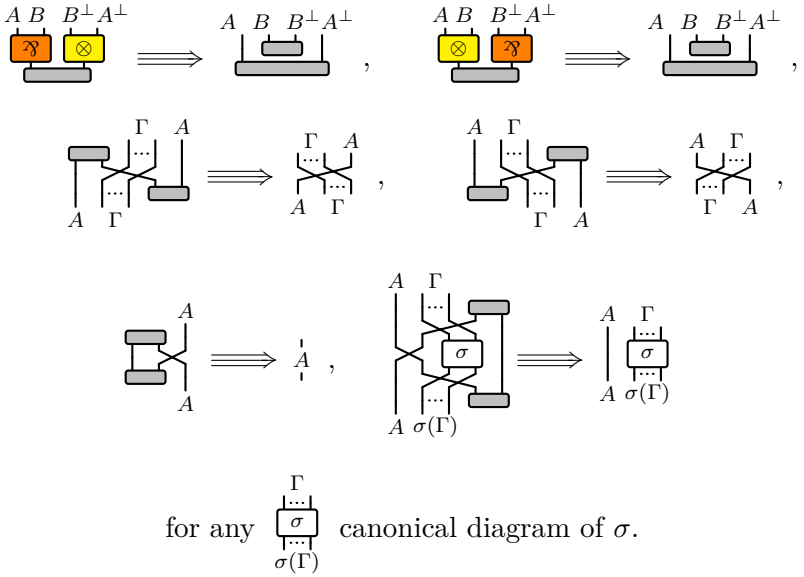
- $\Sigma_3^M = \Sigma_{Twist}^M \cup \Sigma_{Cut}^M$ where:
  - Σ_{Twist}^M is given by the following twisting relations, for all $A, B, C \in \mathfrak{F}_{M\ell\ell}$:





together with two rules representing the involution $A^{\perp\perp} = A$

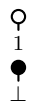
– Σ^M_{Cut} is the set of rules for the cut elimination:

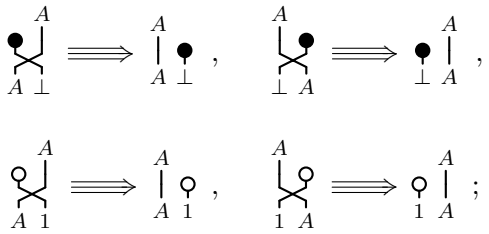


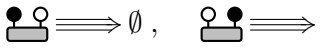
for any canonical diagram of σ.

The twisting relations are the rule we need to introduce in order to make the polygraph twisting, that is to make gates able to cross strings. Their orientation is different from the one given in for the polygraph $\mathfrak{F}$ since, even if $\otimes$ and $\parr$ interact with twisting operators ,they are not commutative () and this avoids the confluence problem we occur in $\mathfrak{F}$. Rules in Σ^M_{Cut} are the rules for cut-elimination. The first two rules reminds the usual rules we have in proof net cut-elimination, the others all corresponds to the cut-elimination with Ax. The reason why we have a wide number of rules is due to the explicit notation for crossing which have to take into account all the possible configurations of irreducible twisting diagrams. Moreover, this set is complete with respect to all possible shapes of irreducible diagrams modulo twisting relations rewriting.

Definition 3.1.2. The *polygraph of multiplicative linear logic with units and cut-elimination* Σ_{MLL_u} is given by the following sets of cells:

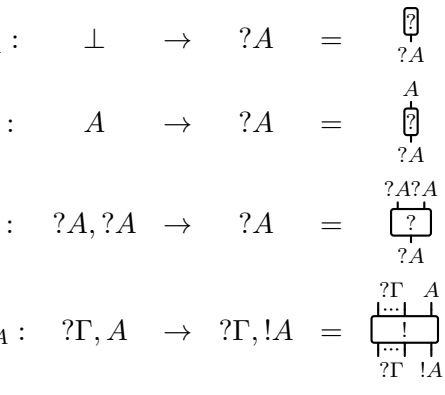
- $\Sigma^u_0 = \{ \Box \}$;
- $\Sigma^u_1 = \mathfrak{F}_{M\ell\ell_u}$;
- $\Sigma^u_2 = \left\{ \begin{array}{lllll} 1: & \Box & \Rightarrow & 1 & = \\ \bot: & \Box & \Rightarrow & \bot & = \end{array} \right.$  $\left. \begin{array}{l} \\ \\ \end{array} \right\} \cup \Sigma^M_2$;
- $\Sigma^u_3 = \Sigma^u_{Twist} \cup \Sigma^u_{Cut}$ where:
  - Σ^c_{Twist} is Σ^M_{Twist} along with the following twisting relations for all $A \in \mathfrak{F}_{M\ell\ell_u}$:



  - Σ^u_{Cut} is Σ^M_{Cut} along with the following rules for cut elimination:  .

- $\Sigma_2^{MELL} = \Sigma_3^c \cup \Sigma_3^{ELL}$ where:

$$\Sigma_3^{ELL} = \left\{ \begin{array}{lcccl} ?\mathbf{W}_A : & \bot & \to & ?A & = \\ ?\mathbf{D}_A : & A & \to & ?A & = \\ ?\mathbf{C}_A : & ?A, ?A & \to & ?A & = \\ !\mathbf{P}_{?\Gamma,A} : & ?\Gamma, A & \to & ?\Gamma, !A & = \end{array} \right\}_{A \in \mathfrak{F}_{Mell}, \Gamma \in \mathfrak{F}_{Mell}^*}$$

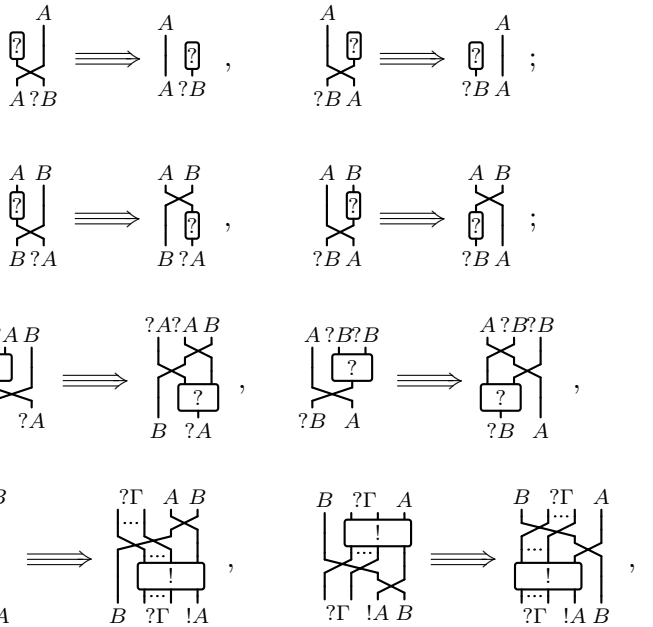


- $\Sigma_3^{MELL} = \Sigma_{Twist}^{MELL} \cup \Sigma_{Cut}^{MELLu}$ where

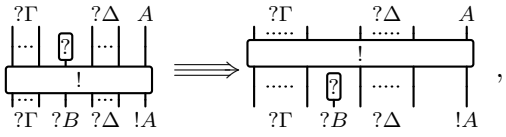
$$\Sigma_{Twist}^{MELL} = \Sigma_{Twist}^c \cup \Sigma_{Twist}^{ELL} \cup \Sigma_{Box}^{ELL} \cup \Sigma_{mon}^{ELL}$$

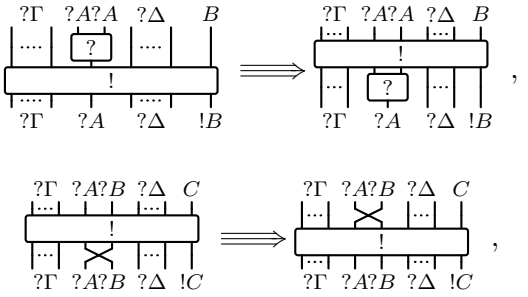
and $\Sigma_{Cut}^{MELL} = \Sigma_{Cut}^c \cup \Sigma_{Cut}^{ELL}$ with:

  - Σ_{Twist}^{ELL} is the set of twisting relations:

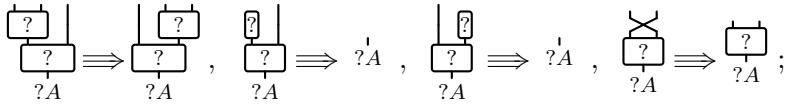


  - Σ_{Box}^{ELL} is the set of relation for boxes:



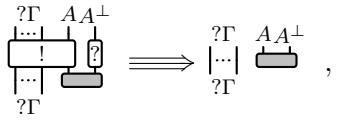


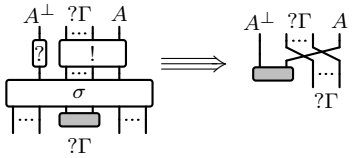
– Σ^{ELL}_{mon} is the set of relation for the monoidal structure concerning contraction and weakening:



– Σ^{ELL}_{Cut} is the set of rules for the cut elimination:

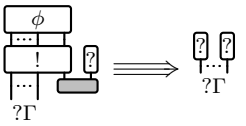
∗ Box vs Dereliction:

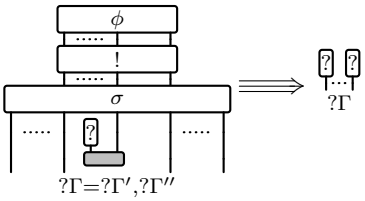




for all $\sigma \in S_{|?\Gamma|+2}$ such that $\sigma(|\Gamma + 2|) = \sigma(1) + 1$,

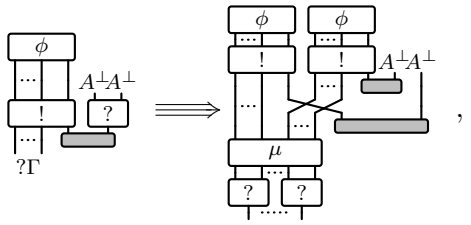
∗ Box vs Weakening



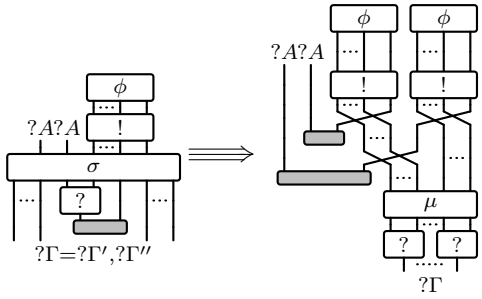


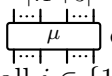
for all $\sigma \in S_{|?\Gamma|+1}$ such that $\sigma^{-1}(|?\Gamma' + 1|) = |?\Gamma| + 1$,

∗ Box vs Contraction

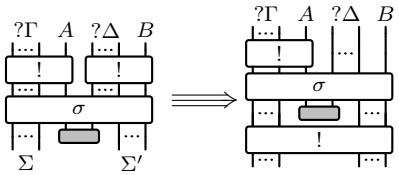


,

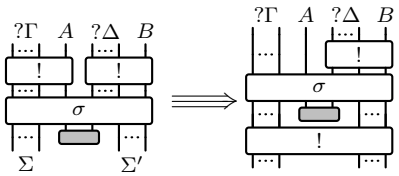


for any $\sigma \in S_{|?\Gamma+3|}$ with $\sigma(2) = \sigma(1) + 1$ and $\sigma(|?\Gamma| + 3) = \sigma(2)+1$ and  canonical diagram of $\mu \in S_{2n}$ with $\mu(i) = \mu(i+n)$ for all $i \in \{1, \ldots, n\}$, and $\phi \in \Sigma_{MELLc}$;

∗ Box vs Box



if $\sigma^{-1}(|\Sigma|+1) = |?\Gamma, A|$ and $|?\Gamma, A| \leq \sigma^{-1}(|\Sigma|+2) \leq |?\Gamma, A, ?\Delta|$,



if $\sigma^{-1}(|\Sigma| + 2) = |?\Gamma, A, ?\Delta, B|$ and $\sigma^{-1}(|\Sigma| + 1) \leq |?\Gamma|$.

In this polygraph, we introduced some 2-cells $!\mathbf{P}$ in order to represent boxes: these should be considered as the border of a box and, as in the extension of Mazza [64] proof nets, they can be considered as cells with multiple active ports .

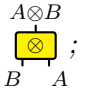
As for Σ^M, twisting relations represent the gate crossings of strings. Moreover, we introduce the set Σ^{ELL}_{Box} representing the interaction of twisting operators, weakening and contraction gates with the boxes. The set Σ^{ELL}_{mon} gives the symmetric monoidal structure concerning contraction and weakening rules. Finally, the set Σ^{ELL}_{Cut} gives all the possible configurations of cut-elimination taking into account the structure of twisting diagrams.

Remark 3.1.4 (Monoidal structure of contraction and weakening)**.** *The rules in Σ^{ELL}_{mon} gives the structure of commutative comonad of !. Any diagram made only of gates of type ?**W** and ?**C** can be considered as a multiple weakening or a multi-contraction with unordered inputs and one output (all labeled by the same formula ?A). Moreover, the last rule gives the commutative structure of contraction.*

Remark 3.1.5. *The polygraphs Σ_{MLL} , Σ_{MLLu} and Σ_{MELL} are total-twisting. This is we can represent any string crossings.*

This means that, in these polygraphs, we have the same interaction between gates and crossing we find in linear logic proof nets. On the other hand, this syntax has some important differences with respect of proof nets:

Remark 3.1.6. *The 2-cells in Σ_{MELL} are similar to MELL proof structures. We remark the following important differences:*

- *Ax and Cut rules are not wirings but cells;*
- *Wiring may cross only by means of twisting operators, that is, only if both wires are labeled by colors of the twisting family;*
- *Diagrams have a top-to-bottom orientation, that is, there are not diagrams like* , *nor upside-down tensor like* 

;
- *Twisting relations and rewriting rules concerning cut elimination have the same semantical status;*
- *There are no boxes but just some 2-cells representing their border.*

Proof diagrams, as proof nets, can be used to represent linear logic derivations. Moreover, we can characterize the structure of input and outputs of proof diagrams corresponding to derivations.

Theorem 3.1.7 (Interpretation of proofs in Σ_{MELL})**.** *For any derivation $d(\Gamma)$ of $\vdash \Gamma$ in MELL there is a proof diagram $\phi_{d(\Gamma)} : \square \Rightarrow \Gamma \in \Sigma_{MELL}$.*

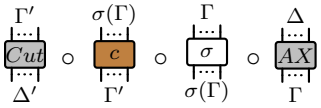
Proof. Let $d(\Gamma)$ be a derivation in *MELL* of $\vdash \Gamma$. First we observe that, if there is a diagram $\phi : \Delta \Rightarrow \Gamma$ then there is a diagram $\phi^\sigma = \hat{\phi}_\sigma \circ \phi : \Delta \Rightarrow \sigma(\Gamma)$ for any permutation $\sigma \in S_{|\Gamma|}$. By this fact we can proceed by induction on the number of inference rules appearing in $d(\Gamma)$:

- If just one inference rule occurs in $d(\Gamma)$, it must be an Ax rule, $\Gamma = A, A^\perp$ and $\phi_{d(\Gamma)} = Ax_A : \Box \Rightarrow A, A^\perp$;
- If $n+1$ inference rules occur in $d(\Gamma)$, then we consider the last one and we distinguish two cases depending on its arity (see Rem. A.0.2):
  - If it is unary and $\Gamma = \Gamma', \perp$, then, by inductive hypothesis, there is a diagram $\phi_{\Gamma'} : \Box \Rightarrow \Gamma'$ and $\phi_\Gamma = \phi_{\Gamma'}, \perp$;
  - Similarly if $\Gamma = \Gamma', ?A$, by inductive hypothesis, there is a diagram $\phi_{\Gamma'} : \Box \Rightarrow \Gamma'$ and $\phi_\Gamma = \phi_{\Gamma'}, ?A$;
  - If it is unary and $\Gamma = \Gamma', A \parr B$, then, by inductive hypothesis, there is a diagram $\phi_{d(\Gamma',A,B)} : \Box \Rightarrow \Gamma', A, B$ of the derivation $d(\Gamma', A, B)$ with n inference rules. Therefore
$$\phi_{d(\Gamma)} = (\mathbf{id}_{\Gamma'}, \parr_{A,B}) \circ \phi_{d(\Gamma',A,B)} : \Box \Rightarrow \Gamma;$$
  - Similarly if the last rule is a unary ?**C** with $\Gamma = \Gamma', ?A$, then, by inductive hypothesis there is a diagram $\phi_{\Gamma',?A,?A} : \Box \Rightarrow \Gamma', ?A, ?A$ and
$$\phi_\Gamma = (\mathbf{id}_{\Gamma'}, C_{?A}) \circ \phi_\Gamma : \Box \Rightarrow \Gamma;$$
  - Similarly if the last rule is a unary ?**D** with $\Gamma = \Gamma', ?A$, ten, by inductive hypothesis there is a diagram $\phi_{\Gamma',A} : \Box \to \Gamma', A$ and
$$\phi_\Gamma = (\mathbf{id}_{?\Gamma}, D_{?A}) \circ \phi_{\Gamma',A} : \Box \Rightarrow \Gamma;$$
  - If it is a unary !**S**, $\Gamma = ?\Gamma', !A$, then, by inductive hypothesis, there is a diagram $\phi_{\Gamma',A} : \Box \Rightarrow \Gamma', A$ and
$$\phi_\Gamma = P_{?\Gamma',A} \circ \phi_{?\Gamma',A} : \Box \Rightarrow \Gamma;$$
  - If it is binary and $\Gamma = \Delta, A \otimes B, \Delta'$, then, by inductive hypothesis, there are two diagrams $\phi_{d(\Delta,A)} : \Box \Rightarrow \Delta, A$ and $\phi_{d(B,\Delta')} : \Box \Rightarrow B, \Delta'$ relative to the two derivations $d(\Delta, A)$ and $d(B, \Delta')$ with at most n inference rules. Therefore
$$\phi_{d(\Gamma)} = (\mathbf{id}_\Delta, \otimes_{A,B}, \mathbf{id}_{\Delta'}) \circ (\phi_{d(\Delta,A)}, \phi_{d(B,\Delta')}) : \Box \Rightarrow \Gamma$$
  - Similarly, if it is binary and $\Gamma = \Delta, Cut(A, A^\perp), \Delta'$, then
$$\phi_{d(\Gamma)} = (\mathbf{id}_\Delta, cut_A, \mathbf{id}_{\Delta'}) \circ (\phi_{d(\Delta,A)}, \phi_{d(A^\perp,\Delta')}) : \Box \Rightarrow \Gamma.$$

□

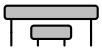
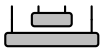
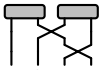
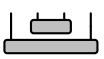
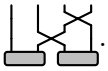
Even if we do not give the more general case for diagrams in Σ_{MELL}, it is possible to characterize the irreducible diagrams in the polygraph Σ_{MLLu}

Proposition 3.1.8. *If $\phi \in \Sigma_{MLLu}$ then there is a diagram $\bar{\phi} \in \Sigma_{MLLu}$ such that ϕ and $\bar{\phi}$ are equivalent modulo the equivalence relation $\simeq_{Tw}$ generated by Σ^c_{Twist} and $\bar{\phi}$ in the form:*



where:

-  *is an horizontal diagram over the signature $\{Ax_A\}_{A\in\mathfrak{F}_{M\ell\ell_u}}$;*
-  $= \hat{\phi}_\sigma$ *for some $\sigma \in S_{|\Gamma|}$;*
-  *is a diagram over the signature $\{\otimes_{A,B}, \parr_{A,B}, \bot\}_{A,B\in\mathfrak{F}_{M\ell\ell_u}}$;*
-  *is an horizontal diagram over the signature $\{Cut_A\}_{A\in\mathfrak{F}_{M\ell\ell_u}}$.*

Proof. The equivalence relation $\simeq_{Tw}$ permits to avoid configurations as  or  since  $\simeq_{Tw}$  and  $\simeq_{Tw}$  .

Together with the natural orientation of twisting relation for $\parr, \otimes$ and $\bot$, the interchange rule and the unicity of $\hat{\phi}_\sigma$ proved in Proposition 1.4.10, this allow to find a diagram $\bar{\phi}$ with the required properties. □

As intuition suggests, we can use this syntax in order to represent proof nets

Proposition 3.1.9. *We can associate a proof structure P_ϕ to any proof diagram $\phi \in \Sigma_{MELL}$.*

Proof. Let $\phi \in \Sigma_{MELL}$, the associate proof net C_ϕ is the one given by:

- the set of free ports is the set of free ports of ϕ;
- the set of cells is the set of gates in ϕ which are not twisting operators;
- the label $l(c)$ of a cell $c \in C_\phi$ is the gate type of the corresponding gate g;
- the set of wires is the set pair of twisting-communicating gates in ϕ;
- the set $\partial(w)$ defined by the associate pair of gates;
- for every gate $g : !P$ is defined the box B_g given by the set of gates $g' \in \phi$ such that $g' \rightsquigarrow g$ with respect of twisting communication.

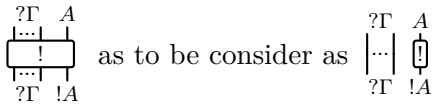
□

In particular, the semantics of Σ_{MELL} results to be the same of *MELL* linear logic proof nets.

Theorem 3.1.10 (Proof net correspondence in Σ_{MELL})**.** *There is a one-to-one correspondence between the set of proof diagrams in Σ_{MELL} modulo the equivalence relation $\simeq_{Tw}$ generated by Σ^{MELL}_{Twist} and the set of multiplicative proof structures with constants.*

Proof. It suffices to remark that twisting relations do not change (twisitng) communicating relations between gates corresponding to adjacencies in the corresponding proof net.

Since we want to prove this result to Σ_{MELL} where contraction and weakening cells are free to move inside or outside certain boxes, we have to consider any $!P_{?\Gamma,A}$ gate present in a gate path as an identity over all its $?\Gamma$ inputs and outputs ports as below:



Then, we define a (twisting) communication relation between all gates in a proof diagrams ϕ.

The proof net P_ϕ is the one defined by this adjacency relation over the set of cells in one-to-one correspondence with non-twisting gates of ϕ. The boxes of P_ϕ are the upper cones $\overline{\partial}(g)$ of the gates $g : !\mathbf{P}$, $g \in \phi$.

On the other hand, we can consider proof diagrams as some particular representations in the plane of the corresponding proof structure graphs. If two proof diagrams can be interpreted by the same proof structure this means that the twisting adjacency relations for these two diagrams are the same and so their graphs are isomorph. It can be easily checked that in order to show that the transformations in the 2-dimensional graph representation corresponding to twisting relations suffices whenever want to keep the orientation of ingoing (inputs) and outgoing (outputs) edges on nodes in a representation with no loop on wires. □

If we note $[P]_{Tw}$ the class of $\simeq_{Tw}$-equivalent diagrams in Σ_{MELL} corresponding to the proof net P, we formulate the following

Corollary 3.1.11 (Cut-elimination step correspondence)**.** *If P, P' are two proof nets, then*

$$P \rightsquigarrow_{cut} P' \Leftrightarrow \exists \phi, \phi' \in \Sigma_{MELLc},\ \phi \in [P]_{Tw},\ \phi' \in [P']_{Tw} \text{ such that } \phi \underset{\Sigma^{MELLc}_{Cut}}{\Longrightarrow} \phi' .$$

Proof. It follows the definition of P_ϕ, $P_{\phi'} = P'$ and $[P]_{Tw}$. □

This leads to a cut-elimination theorem for Σ_{MELL}:

Theorem 3.1.12 (Cut elimination in Σ_{MELL})**.** *If $\phi : \Box \Rightarrow \Gamma \in \Sigma_{MELL}$ is irreducible and it corresponds to a correct proof net, then in ϕ there is no gate $g : Cut$.*

3.2 Proof diagrams with control for *MELL*

In order to have a correctness criterion for proof diagrams, we enrich the set of string labels with two new non-twisting colors L (left) and R (right) and re-define some 2-cells.

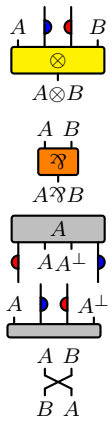
The idea is to use the latter to internalize the notion of well-paranthesization in a setting where a proof derivation can be seen as a sequence of operations over lists of sequents: unary derivation rules act on single sequents (as in the case of $\parr$), binary ones act on two sequents (as in the case of $\otimes$ and Cut) and the 0-ary one, that is Ax, generates a new sequent.

In order to integrate the the control strings with the cellular syntax, we need to make interact cells with these strings any time they interact with sequents parenthesization: Ax and 1 generate new sequents, so they should have no inputs and outputs of the form $L, A, A^\perp, R$ and $L, 1, R$ respectively while $\otimes$ and Cut have to be with inputs of the form A, R, L, B. We underline that unary promotion rule acts on a whole sequent (not locally) and for this reason the relative 2-cells, even if the rule is unary, have a structure interacting with control strings. For this reason, cells representing promotion rules will be with inputs of the form $L, ?\Gamma, A, R$ and output Ł, $?\Gamma, !A, R$.

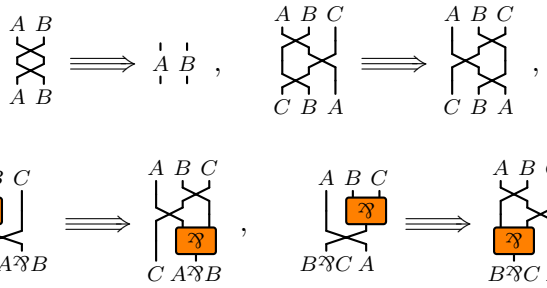
Definition 3.2.1. The *control polygraph of multiplicative linear logic* $\tilde{\mathfrak{M}}$ is given by the following sets of cells:

- $\tilde{\mathfrak{M}}_0 = \{ \Box \}$;
- $\tilde{\mathfrak{M}}_1 = \mathfrak{F}_{M\ell\ell} \cup \{L = \text{◖}, R = \text{◗}\}$;
- $\tilde{\mathfrak{M}}_2 =$

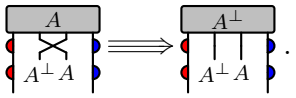
$$\left\{ \begin{array}{lccccc} \otimes_{A,B} : & A,R,L,B & \Rightarrow & A \otimes B & = & \\ \parr_{A,B} : & A,B & \Rightarrow & A \parr B & = & \\ Ax_A : & \Box & \Rightarrow & L,A,A^\perp,R & = & \\ Cut_A : & A,R,L,A^\perp & \Rightarrow & \Box & = & \\ T_{A,B} : & A,B & \rightarrow & B,A & = & \end{array} \right\}_{A,B \in \mathfrak{F}_{M\ell\ell}} ;$$



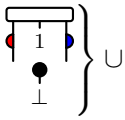
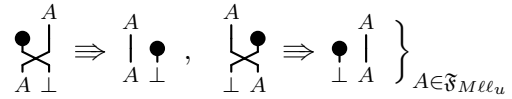
- $\tilde{\mathfrak{M}}_3 = \tilde{\mathfrak{M}}_{Twist}$ is given by the following twisting relations:



together with one rule representing the involution $A^{\perp\perp} = A$:



Definition 3.2.2. The *control polygraph of multiplicative linear logic with constants* $\tilde{\mathfrak{L}}$ is given by

- $\tilde{\mathfrak{L}}_0 = \{ \square \}$;
- $\tilde{\mathfrak{L}}_1 = \mathfrak{F}_{M\ell\ell_u} \cup \{L = \blacktriangleleft, R = \blacktriangleright\}$;
- $\tilde{\mathfrak{L}}_2 = \left\{ \begin{array}{lllll} 1: & \square & \Rightarrow & L, 1, R & = \\ \perp: & \square & \Rightarrow & \perp & = \end{array} \right.$  $\left. \begin{array}{l} \\ \\ \end{array} \right\} \cup \tilde{\mathfrak{M}}_2$;
- $\tilde{\mathfrak{L}}_3 = \tilde{\mathfrak{L}}_{Twist} =$  $\cup\, \tilde{\mathfrak{M}}_{Twist}$.

Remark 3.2.3. *The polygraphs* $\tilde{\mathfrak{L}}$ *(resp.* $\tilde{\mathfrak{M}}$*) is twisting with twisting family* $\mathfrak{F}_{M\ell\ell_u}$ *(resp.* $\mathfrak{F}_{M\ell\ell}$*) .*

This means that in these polygraph we can represent string crossing between all non-crossing strings, recovering the possibility to reorder sequents (vieved as lists).

The introduction of control string allows to prove a correspondence between certain proof diagrams and linear logic derivations. In fact, the well parenthesization guarantees that binary rules are applied to two formulas appertaining different sequents while unary to formulas appertaining to the same sequent. This characteristic can not be formalized in linear logic proof net syntax since string crossings have not a formal statute: they are just graphical representations of adjacency relation between cells.

Moreover, the control strings guarantee also the sequentializability of proof diagrams even in presence of $\perp$ gates. In fact, in proof nets need jumps in

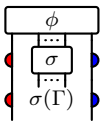
order to prove correctness and these have no restrictions on their assignation. In proof diagrams, any $\bot$ gate can be associated only to certain Ax and 1 gates leafs of the derivation tree branching which it belongs. This branching is limited by the $\otimes$ or Cut gate of which the considered $\bot$ is a subformula of its principal formulas.

Theorem 3.2.4 (Proof diagrams correspondence in $\tilde{\mathfrak{U}}$)**.**

$$\vdash_{MLL_u} \Gamma \Leftrightarrow \exists \phi \in \tilde{\mathfrak{U}} \textit{ such that } \phi : \square \Rightarrow L, \Gamma, R.$$

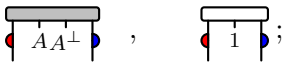
Proof. To prove the left-to-right implication $\Rightarrow$, as in Teor. 3.1.7, we remark that, if there is a diagram $\phi : \square \Rightarrow L, \Gamma, R$ with Γ sequent in MLL_u, then there is a diagram

$$\phi^\sigma = (\mathbf{id}_L, \hat{\phi}_\sigma, \mathbf{id}_R) \circ \phi : \square \Rightarrow L, \sigma(\Gamma), R$$



for any permutation $\sigma \in S_{|\Gamma|}$. Then we proceed by induction on the number of inference rules in a derivation $d(\Gamma)$ in MLL_u:

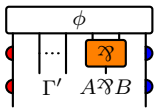
- If just one inference rule occurs $d(\Gamma)$, then it is an Ax or 1 and $\Gamma = A, A^\bot$ and $\phi_{d(\Gamma)} = Ax_A : \square \Rightarrow L, A, A^\bot, R$ or respectively $\Gamma = 1$ and $\phi_{d(\Gamma)} = 1 : \square \Rightarrow L, 1, R$:



- If $n + 1$ inference rules appear, then we consider the last one and we distinguish two cases in base of its arity:

  - If it is unary it could be a $\parr$ or $\bot$. In the first case, $\Gamma = \Gamma', A \parr B$, then, by inductive hypothesis, there is a diagram $\phi_{d(\Gamma', A, B)} : \square \Rightarrow L, \Gamma', A, B, R$ of the derivation $d(\Gamma', A, B)$ and

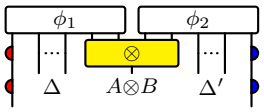
$$\phi_{d(\Gamma)} = (\mathbf{id}_{L,\Gamma'}, \parr_{A,B}, \mathbf{id}_R) \circ \phi_{d(\Gamma', A, B)} : \square \Rightarrow L, \Gamma, R$$



    Similarly if it is a $\bot$, $\Gamma = \Gamma', \bot$, then, by inductive hypothesis, there is a diagram $\phi_{\Gamma'} : \square \Rightarrow L, \Gamma', R$ and $\phi_\Gamma = (L, \bot, \mathbf{id}_{\Gamma'}, R) \circ \phi_{\Gamma'}$;

– If it is binary and $\Gamma = \Delta, A \otimes B, \Delta'$, then, by inductive hypothesis, there are two diagrams $\phi_1 = \phi_{d(\Delta,A)} : \square \Rightarrow L, \Delta, A, R$ and $\phi_2 = \phi_{d(B,\Delta')} : \square \Rightarrow L, B, \Delta', R$ relative to the two derivations $d(\Delta, A)$ and $d(B, \Delta')$ with at most n inference rules. Therefore

$$\phi_{d(\Gamma)} = (\mathbf{id}_{L,\Delta}, \otimes_{A,B}, \mathbf{id}_{\Delta',R}) \circ (\phi_{d(\Delta,A)}, \phi_{d(B,\Delta')}) : \square \Rightarrow L, \Gamma, R$$



– Similarly, if it is binary and $\Gamma = \Delta, Cut(A, A^\perp), \Delta'$, then

$$\phi_{d(\Gamma)} = (\mathbf{id}_{L,\Delta}, Cut_{A^\perp}, \mathbf{id}_{\Delta',R}) \circ (\phi_{d(\Delta,A)}, \phi_{d(A^\perp,\Delta')}) : \square \Rightarrow L, \Gamma, R.$$

In order to prove sequentialization, i.e. the right-to-left implication $\Leftarrow$, we proceed by induction on the number $|\phi|_{\mathcal{S}}$ of gates in ϕ:

- If $|\phi|_{\tilde{\mathfrak{U}}} = 0$ then $\phi : \mathbf{id}_\Gamma : \Gamma \Rightarrow \Gamma$. By hypothesis ϕ has no input (i.e. $s_2(\phi) = \square$) so it is the identity diagram over the empty string, this is the empty diagram $\mathbf{id}_0 : \square \Rightarrow \square$ which it is not sequentializable since $t_2(\phi) = \square \neq L, R$;

- If $|\phi|_{\tilde{\mathfrak{U}}} = 1$ then ϕ is an elementary diagram. The elementary diagrams with source $\square$ and target L, Γ, R with $\Gamma \in \mathfrak{F}_{M\ell\ell_u}{}^*$ are atomic made of a unique 2-cell of type $Ax_A : \square \Rightarrow L, A, A^\perp, R$ for some $A \in \mathfrak{F}_{M\ell\ell_u}$ or $1 : 0 \to L, 1, R$. The associated sequents $\vdash A, A^\perp$ and $\vdash 1$ are derivable in MLL_u;

- Otherwise there is 2-cell of type $\alpha : \Gamma' \Rightarrow \alpha(\Gamma') \in \tilde{\mathfrak{U}}_2$ and $\Gamma = \Delta, \alpha(\Gamma'), \Delta'$. In this case $\phi = (\mathbf{id}_{L,\Delta}, \alpha, \mathbf{id}_{\Delta,R}) \circ \phi'$ where $\phi' : \square \Rightarrow L, \Delta, \Gamma', \Delta', R$. We have the following cases:

  – If $\alpha = T_{A,B}$, $\Gamma' = A, B$ and $\alpha(\Gamma') = B, A$. The diagram ϕ' is sequentializable by inductive hypothesis since $|\phi|_{\tilde{\mathfrak{U}}} = |\phi'|_{\tilde{\mathfrak{U}}} + 1$;
  – Similarly if $\alpha = \parr_{A,B}$, $\Gamma' = A, B$ and $\alpha(\Gamma') = A \parr B$;
  – Similarly if $\alpha = \perp$, $\Gamma' = \emptyset$ and $\alpha(\Gamma') = \perp$;
  – If $\alpha = \otimes_{A,B}$ so $\Gamma' = A, R, L, B$, $\alpha(\Gamma') = A \otimes B$ and

  $$\phi' : \square \Rightarrow L, \Delta, A, R, L, B, \Delta', R.$$

  This diagram is a parallel composition $\phi' = \phi'_l, \phi'_r$ with

  $$\phi'_l : \square \Rightarrow L, \Delta, A, R \quad \text{and} \quad \phi'_r : \square \Rightarrow L, B, \Delta', R$$

  of two diagrams which satisfy inductive hypothesis since $|\phi|_{\tilde{\mathfrak{U}}} = |\phi'_l|_{\tilde{\mathfrak{U}}} + |\phi'_r|_{\tilde{\mathfrak{U}}} + 1$;

– Similarly if $\alpha = Cut_A$ with $B = A^{\perp}$ we have $\Gamma' = A, R, L, A^{\perp}$ and $\alpha(\Gamma') = \emptyset$.

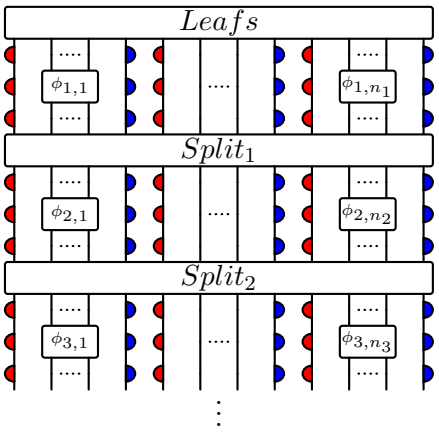
□

This theorem assure that proof diagrams can be used to represent any proof derivation (with explicit exchange rules) in MLL_u. In particular, we have a bijection between proofs in MLL_u with explicit exchange rules and proof diagrams with no input and output of the form L, Γ, R.

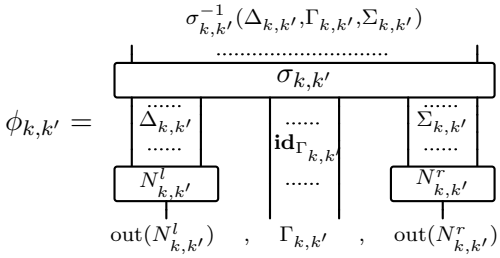
Layers structure of irreducible proof diagrams in $\tilde{\mathfrak{U}}$

By the interchange law, is possible to rearrange the position of some gates in a proof digram, in particular we can move upward and downward the leafs of derivation and parallel applications of inference rules, i.e. rules acting on independent sets of formulas.

Then, the structure of irreducible multiplicative proof diagrams in $\tilde{\mathfrak{U}}$ is characterized by a layer structure. An irreducible multiplicative proof diagram has the following form:



where each $\phi_{k,k'} : \sigma^{-1}_{k,k'}(\Delta_{k,k'}, \Gamma_{k,k'}, \Sigma_{k,k'}) \Rightarrow \mathrm{out}(N^l_{k,k'}), \mathbf{id}_{\Gamma_{k,k'}}, \mathrm{out}(N^r_{k,k'})$ is a diagram of shape



satisfying the following conditions:

- $\Delta_{k,k'}, \Gamma_{k,k'}, \Sigma_{k,k'}$ could be empty sequents;
- $N^l_{k,k'}, N^r_{k,k'}$ are empty diagrams (and so $\Delta_{k,k'}, \Sigma_{k,k'}$ are empty sequents) if respectively $k' = 1$ and $k' = n_i$ for every i;
- $Leafs$ is an elementary diagram over the signature $\{Ax_F, 1\}_{F \in \mathfrak{F}_{M\ell\ell_u}}$:

$$Ax : \square \Rightarrow \langle, \Gamma_1, \rangle , \ \dots \ , \ \langle, \Gamma_{n_0}, \rangle$$

  with $\Gamma_i = F_i, F_i^\perp$ for some $F_i \in \mathfrak{F}_{M\ell\ell_u}$ or $\Gamma_i = 1$;
- $\sigma_{k,k'}$ is a twisting diagram;
- $Split_i$ is an elementary diagram over the signature $\{\otimes, Cut\}$, we enumerate its gates $\alpha_1, \dots, \alpha_{s_i}$ from left to right so that

$$Split_i = \langle, \mathbf{id}_{W_1}, \alpha_1, \mathbf{id}_{W_2}, \dots, \mathbf{id}_{W_1}, \alpha_{s_i}, \mathbf{id}_{W_{s_i}} \rangle$$

  where $\mathbf{id}_{W_j} = \mathbf{id}_{\Theta_{j,1}}, \rangle, \dots \langle, \mathbf{id}_{\Theta_{j,m_j}}$ for some $\Theta_{j,k} \in \mathfrak{F}_{M\ell\ell_u}{}^*$ and $m_j \in \mathbb{N}$ such that

$$\sum_{j=1}^{s_i} m_j = n_i - s_i.$$

With respect of this notation, we can observe the following property:

Proposition 3.2.5 (Layers in $\tilde{\mathfrak{U}}$)**.** *An irreducible multiplicative proof diagram in $\tilde{\mathfrak{U}}$ has a* standard layer form*:*

$$Layer_\ell \circ \dots \circ Layer_1 \circ Leafs : \square \Rightarrow L, \Gamma, R$$

where $Layer_i = Pos_i \circ Neg_i \circ Twist_i$ with

- *$Leafs$ is an elementary diagram over the signature $\{Ax_F, 1\}_{F \in \mathfrak{F}_{M\ell\ell_u}}$:*

$$Leafs : \square \Rightarrow \langle, \Gamma_1, \rangle , \ \dots \ , \ \langle, \Gamma_{n_0}, \rangle$$

  with $\Gamma_i = F_i, F_i^\perp$ for some $F_i \in \mathfrak{F}_{M\ell\ell_u}$ or $\Gamma_i = 1$;
- *$Twist_i$ is a* block twisting diagram*:*

$$Twist_i = \langle, \sigma_{i,1}, \rangle , \ \dots \ , \ \langle, \sigma_{i,n_i}, \rangle$$

  in particular we have that $Twist_1$ is an identity;
- *$Neg_i = Block^-_{i,1}, \dots, Block^-_{i,n_i}$ is the i^{th}* negative layer *with*

$$Block^-_{i,k} = \langle, N^l_{i,k}, \boldsymbol{id}_{\Gamma_i,k}, N^r_{i,k}, \rangle$$

- $Pos_i = Block^+_{i,1}, \ldots, Block^+_{i,n_i}$ *is the* i^{th} positive layer *of the form*

  $Block^+_{i,k} = \boldsymbol{id}_{W^1_{i,k}}, \boldsymbol{id}_{W^2_{i,k}}, \ldots \boldsymbol{id}_{W^t_{i,k}}, \vdash, \boldsymbol{id}_{\Delta_{i,k}}, \alpha_i, \boldsymbol{id}_{\Sigma_{i,j}}, \boldsymbol{id}_{W^{t+1}_{i,k}}, \ldots, \boldsymbol{id}_{W^{m_{i,k}}_{i,k}}$

  where $\boldsymbol{id}_{W^j_{i,k}} = \vdash, \Gamma^j_{i,k}, \dashv$ *for some sequent* $\Gamma^j_{i,k} \in \mathfrak{F}_{M\ell\ell_u}{}^*$;

This layer structure of an irreducible proof diagram ϕ corresponds to a particular arrangement of inference rules application in the relative derivation $d(\phi)$: each positive rule ($\otimes$ or Cut) is preceded by all and only inference rules needed to produce their active formula. This is due to the fact that twisting relations concerning $\parr$ and $\bot$ gates moves downward and, together with interchange rules, allows to perform the corresponding inference rule latest.

In order to achieve a focalized structure ([5], [56]) of proof diagrams we should performs not only the essential *asynchronous* inference rules. We claim that reversing the direction of twisting relations concerning $\parr$ and $\bot$ gates (so moving these gates upward in a diagram) we should be able to achieve an irreducible form corresponding to a focalized proof.

3.3 Proof diagrams in MELL

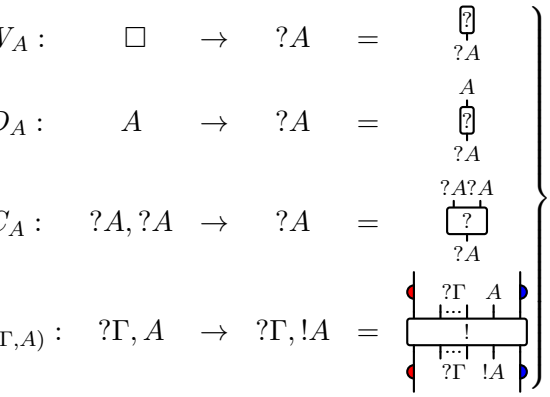
In this section we extend the polygraph $\tilde{\mathfrak{U}}$ in order to accommodate the 2-cells for exponential rules.

We remind the reader that unary promotion rule acts is a context-sensitive rule: even if it acts only on the formula it promotes, this rule require some condition on the whole sequent on which it is applied. Then, in order to define the 2-cells representing promotions, we have to make it properly interact with control strings. A cell representing promotion rules needs to control all the sequent on which it is applied, then its inputs have to be of the form $L, ?\Gamma, A, R$ and its output of the form $L, ?\Gamma, !A, R$.

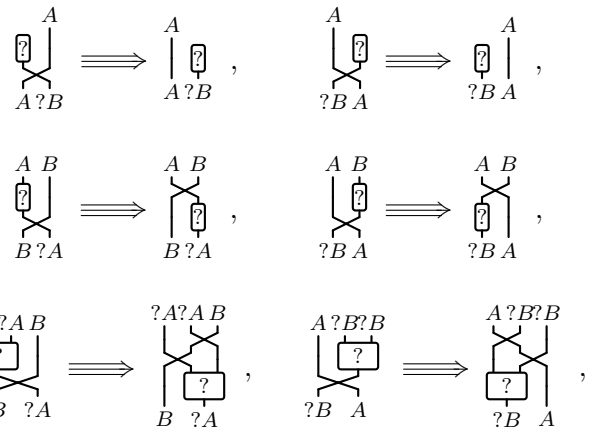
Definition 3.3.1. The *control polygraph of multiplicative exponential linear logic* $\tilde{\mathfrak{E}}$ is given by the following sets of cells:

- $\tilde{\mathfrak{E}}_0 = \{ \Box \}$;
- $\tilde{\mathfrak{E}}_1 = \mathfrak{F}_{Me\ell\ell} \cup \{L = \vdash, R = \dashv\}$;
- $\tilde{\mathfrak{E}}_2 = \tilde{\mathfrak{U}}_2 \cup \mathfrak{E}xp$ where

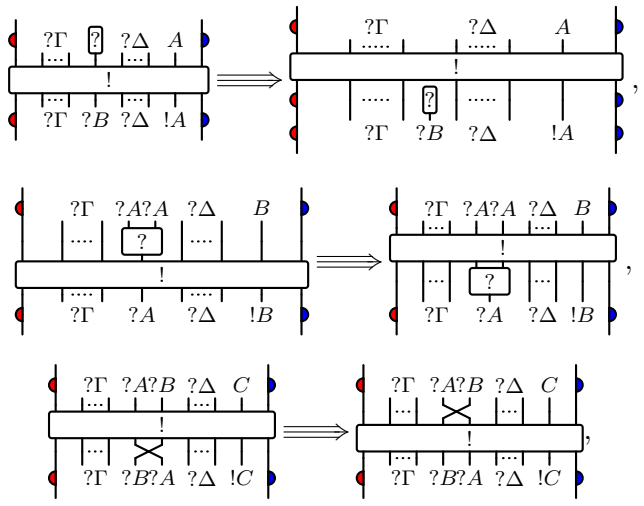
$$\mathfrak{E}xp = \left\{ \begin{array}{llllll} ?W_A : & \square & \to & ?A & = & \\ ?D_A : & A & \to & ?A & = & \\ ?C_A : & ?A, ?A & \to & ?A & = & \\ !P_{(?\Gamma,A)} : & ?\Gamma, A & \to & ?\Gamma, !A & = & \end{array} \right\}_{A \in \mathfrak{F}_{Mell}, \Gamma \in \mathfrak{F}_{Mell}^*} ;$$



- $\tilde{\mathfrak{E}}_3 = \tilde{\mathfrak{E}}_{Twist} \cup \tilde{\mathfrak{E}}_{Box} \cup \tilde{\mathfrak{E}}_{mon}$ where:
  - $\tilde{\mathfrak{E}}_{Twist}$ is the set $\tilde{\mathfrak{U}}_{Twist}$ together with the following twisting relations:

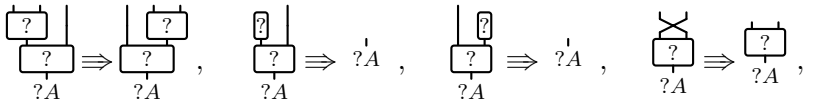


    for all $A, B \in \mathfrak{F}_{Mell}$;
  - $\tilde{\mathfrak{E}}_{Box}$ is the set of relation for boxes:



for all $A, B, C \in \mathfrak{F}_{Mell}$ and $\Gamma, \Delta \in \mathfrak{F}_{Mell}{}^*$;

– $\tilde{\mathfrak{E}}_{mon}$ is the set of relation for the !-*comonad*, that is the symmetric monoidal structure concerning contraction and weakening:



for all $A \in \mathfrak{F}_{Mell}$.

Any gate $g : !\mathbf{P}$ can be interpreted as the border of a *MELL* proof net box. In particular, the content of this box is the subdiagram $\overline{\partial}(g)$. The set of rewriting rules of this polygraph consists of the set of twisting relation plus two sets $\mathfrak{E}_{Box}$ and $\tilde{\mathfrak{E}}_{mon}$. The set $\tilde{\mathfrak{E}}_{Box}$ gives the interactions between the promotion rule with contraction, weakening and, of course, twisting operators. In this polygraph we move twisting operators upward while we move contraction and weakening outside a box. In order to give the semantics of commutative monoid for contraction and weakening, we define the set $\tilde{\mathfrak{E}}_{mon}$ reminding the set of rewriting rules of $\mathfrak{F}$ we have defined in Section 2.5 for the symmetric monoidal categories. In particular, they represent respectively the associativity, left and right unitor, the commutativity of contraction rule.

Remark 3.3.2 (Monoidal structure of contraction and weakening)**.** *The rules in $\tilde{\mathfrak{E}}_{mon}$ gives to the diagrams made of ?W and ?C gates a structure of monoidal category reminding the polygraph $\mathfrak{F}$. By this fact, anytime a subdiagram made only of these gates occurs, we can consider it as multiple parallel weakening or a unique multi-contraction gates with n inputs and one output all labeled by the same formula ?A.*

Remark 3.3.3. *The polygraph $\tilde{\mathfrak{E}}$ is twisting with twisting family $\mathfrak{F}_{Mell}$.*

This means that we can represent any crossing of strings labeled by *MELL* formulas and these crossings have the behavior we attend with respect of their interaction with cells connected with no control strings.

Moreover, we are able to prove the termination in this polygraph. This means that in this polygraph we have irreducible proof diagrams we can use as representative of their equivalence classes.

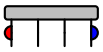
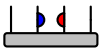
Theorem 3.3.4 (Termination of $\tilde{\mathfrak{E}}$)**.** *The polygraph $\tilde{\mathfrak{E}}$ is terminating.*

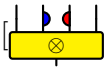
Proof. Similarly to the proof of Proposition 2.5.5, in order to prove termination, we define a *termination order* [33] associating to any proof diagram ϕ a triple $[\![\phi]\!] = ([\phi], [\phi]_C, \|\phi\|^X_{Box})$ given by two monotone functions $[\phi], [\phi]_C : \mathbb{N}^{*p} \Rightarrow \mathbb{N}^{*q}$ and the integer

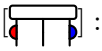
$$\|\phi\|^X_{Box} = \sum_{g:!\mathbf{P}}^{g\in\phi} |\underline{\partial}(g)|_{T_{A,B}}.$$

counting the number of twisting operator in the lower cone of a gate of type $!\mathbf{P}$. Then, by the order over function induced by the product order on $\mathbb{N}^{*p}$, we give a lexicographic order over these pairs.

We associate the following functions to each gate of the signature $\tilde{\mathfrak{E}}$:

$[$  $] : \emptyset \to (1,1,1,1)$, $[$  $] : (z_1, x, y, z_2) \to \emptyset$,

$[$  $] : (x, y) \to x + y + 1$, $[$  $](x, z_1, z_2, y) \to x + y + 1$,

$[$ $] : (x, y) \to (y, x + y)$, $[$  $] : (\emptyset) \to 1$, $[$  $] : (\emptyset) \to (1,1,1)$,

$[$  $] : (x, y) \to x + y + 1$, $[$  $] : (x) \to x + 1$, $[$ $] : (\emptyset) \to 1$,

$[$  $] : (z_1, x_1, \ldots x_n, z_2) \to (z_1, 2x_1, \ldots 2x_n, z_2)$.

The function associated by $[-]_C$ differs from the one associated by $[-]$ only in the following case:

$$[\,?\,]_C : (x, y) \to 2x + y \ ,$$

In particular, this allows us to associate to any rule $\phi \Rrightarrow \phi'$ two monotone maps $[\phi]$ and $[\phi']$ such that $[\phi] > [\phi']$ (and then $[\![\phi]\!] > [\![\phi']\!]$), in fact:

$$\left[\,\right](x, y) = (2x + y, x + y) > (x, y) = \left[\,\right](x, y),$$

$$\left[\,\right](x, y, z) = (2x+y+z, x+y, x) > (x+y+z, x+y, x) = \left[\,\right](x, y, z),$$

$$\left[\,\right]\emptyset = (0, 2, 1, 0) > (0, 1, 1, 0) = \left[\,\right]\emptyset,$$

$$\left[\,\right](x, y, z) = (x+y+z+1, x+y+1) > (x+y+z, x+y+1) = \left[\,\right](x, y, z),$$

$$\left[\,\right](x, y, z) = (x+y+z+1, x+y+1) > (x+y+z, x+y+1) = \left[\,\right](x, y, z),$$

$$\left[\,\right](x, y, z) = (y+z+1, x+y+z+1) > (y+z+1, x+y) = \left[\,\right](x, y, z),$$

$$\left[\,\right](x, y, z) = (y+z+1, x+y+z+1) > (y+z+1, x+y) = \left[\,\right](x, y, z),$$

$$\left[\,\right](x) = (x + 2, 1) > (x, 1) = \left[\,\right](x),$$

$$\left[\,\right](x) = (x + 2, x) > (1, x) = \left[\,\right](x),$$

$$\Big[\ \boxed{?}\ \Big](x) = (x+2,1) > (x,1) = \Big[\ |\ \boxed{?}\ \Big](x),$$

$$\Big[\ \boxed{?}\ \Big](x) = (x+2,x) > (1,x) = \Big[\ \boxed{?}\ |\ \Big](x),$$

$$\Big[\ \boxed{?}\ \boxed{!}\ \Big](z_1,x_1,\dots,x_n,z_2) = (z_1,2x_1,\dots,2x_k,2,2x_{k+1}\dots,2x_n,z_2) >$$

$$> (z_1,2x_1,\dots,2x_k,1,2x_{k+1}\dots,2x_n,z_2) = \Big[\ \boxed{!}\ \boxed{?}\ \Big](z_1,x_1,\dots,x_n,z_2),$$

$$\Big[\ \boxed{?}\ \boxed{!}\ \Big](z_1,x_1,\dots,x_n,z_2) = (z_1,2x_1,\dots,2(x_k+x_{k+1}+1),\dots,2x_n,z_2) >$$

$$> (z_1,2x_1,\dots,2x_k{+}2x_{k+1}{+}1\dots,2x_n,z_2) = \Big[\ \boxed{!}\ \boxed{?}\ \Big](z_1,x_1,\dots,x_n,z_2),$$

$$\Big[\ \boxed{?}\ \boxed{?}\ \Big](x) = x+2 > x = \Big[\ |\ \Big](x), \qquad \Big[\ \boxed{?}\ \boxed{?}\ \Big](x) = x+2 > x = \Big[\ |\ \Big](x),$$

$$\Big[\ \boxed{?}\ \Big](x,y) = 2x+y+1 > x+y+1 = \Big[\ \boxed{?}\ \Big](x,y),$$

$$\Big[\ \boxed{?}\ \boxed{?}\ \Big](x,y,z) = 2x+y+z+2 > x+y+z+2 = \Big[\ \boxed{?}\ \boxed{?}\ \Big](x,y,z).$$

The rules $\boxed{?}\ \boxed{?} \Rrightarrow \boxed{?}$ and $\boxed{!} \Rrightarrow \boxed{!}$ does not satisfy this inequality, in fact

$$\Big[\ \boxed{?}\ \boxed{?}\ \Big](x,y,z) = x+y+z+2 = x+y+z+2 = \Big[\ \boxed{?}\ \boxed{?}\ \Big](x,y,z)\ ,$$

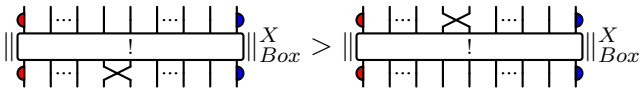
$$\Big[\ \boxed{!}\ \Big](z_1,x_1,\dots,x_n,z_2) = (z_1,2x_1,\dots,2(x_k+x_{k+1}),2x_k,\dots,2^{x_n},z_2) =$$

$$= (z_1,2x_1,\dots,2x_k{+}2x_{k+1},2x_k,\dots,2x_n,z_2) = \Big[\ \boxed{!}\ \Big](z_1,x_1,\dots,x_n,z_2).$$

On the other hand, we have $[\![\phi]\!] > [\![\phi']\!]$ also in these cases since

$$\Big[\ \boxed{?}\ \boxed{?}\ \Big]_C(x,y,z) = 4x+2y+z > 2x+2y+z = \Big[\ \boxed{?}\ \boxed{?}\ \Big]_C(x,y,z)\ .$$

and



By the compatibility of the order with sequential and parallel composition, this suffice to prove that, for any couple of diagrams, $[\![\phi]\!] > [\![\phi']\!]$ holds if $\phi \overset{*}{\Rightarrow} \psi$. Since there exists no infinite decreasing suite of monotone maps on positive integers, infinite reduction paths can not exist. □

We can extend the proof diagrams correspondence proved for $\tilde{\mathfrak{U}}$ in Theorem 3.2.4 to $\tilde{\mathfrak{E}}$. In this polygraph we observe the good properties of correctness for $?W$ gates for similar argumentations of $\perp$ gates in $\tilde{\mathfrak{U}}$ while $?\mathbf{C}$ and $?\mathbf{D}$ gates gives no correctness problems as $?\mathbf{C}$ and $?\mathbf{D}$ do in *MELL* proof nets. Moreover, the shape of $!\mathbf{P}$ cells guarantee, the correctness the application of a promotion rule. In fact, any gate $g : !\mathbf{P}$ determines a subdiagram $\overline{\partial}(g) : \square \Rightarrow L, ?\Gamma, A, R$ which is correct by induction over their number of $!\mathbf{P}$ gates. We can identify any such upper cone $\overline{\partial}(g)$ with the content of a box in a *MELL* proof net. Also in this case, the presence of control strings prevent the crossing of boxes and that a string cross a box border.

Theorem 3.3.5 (Proof diagrams correspondence in $\tilde{\mathfrak{E}}$)**.**

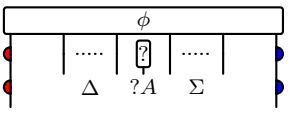
$$\vdash_{MELL} \Gamma \Leftrightarrow \exists \phi \in \tilde{\mathfrak{E}} \textit{ such that } \phi : \square \Rightarrow L, \Gamma, R.$$

Proof. In order to prove the theorem, we have to extend proof of Theorem 3.2.4 with the new cases relative to exponential inference rules and the relative 2-cells.

To prove the left-to-right implication $\Rightarrow$ we have to consider the cases when the last of inference rules of a derivation $d(\Gamma)$ in *MELL* is an exponential rules:

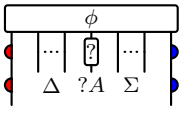
- If it is an unary $?W$, then $\Gamma = \Delta, ?A, \Sigma$ and

$$\phi_\Gamma = (\mathbf{id}_\Delta, ?W_A, \mathbf{id}_\Sigma) \circ \phi_{(\Delta,\Sigma)} =$$



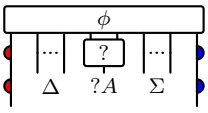
- If it is an unary $?D$, then $\Gamma = \Delta, ?A, \Sigma$ and

$$\phi_\Gamma = (\mathbf{id}_\Delta, ?D_A, \mathbf{id}_\Sigma) \circ \phi_{(\Delta,A,\Sigma)} =$$



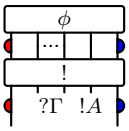
- If it is an unary $?C$, then $\Gamma = \Delta, ?A, \Sigma$ and

$$\phi_\Gamma = (\mathbf{id}_\Delta, ?C_A, \mathbf{id}_\Sigma) \circ \phi_{(\Delta,?A,?A,\Sigma)} =$$



- If it is an unary $!P$, then $\Gamma = ?\Gamma', !A$ and

$$\phi_\Gamma = (\mathbf{id}_{?\Gamma'}, !P_{(?\Gamma',A)}) \circ \phi_{(\Gamma',A)} =$$



In order to prove sequentialization, i.e. the right-to-left implication $\Leftarrow$, we have have the following additional cases when it exists a 2-cell of type $\alpha : \Gamma' \Rightarrow \alpha(\Gamma') \in \tilde{\mathfrak{E}}_2$ and $\Gamma = \Delta, \alpha(\Gamma'), \Delta'$:

- If $\alpha = ?W_A$, $\Gamma' = \emptyset$ and $\alpha(\Gamma') = ?A$;
- If $\alpha = ?D_A$, $\Gamma' = A$ and $\alpha(\Gamma') = ?A$;
- If $\alpha = ?C_A$, $\Gamma' = ?A, ?A$ and $\alpha(\Gamma') = ?A$;

and then $\phi = (\mathbf{id}_{L,\Delta}, \alpha, \mathbf{id}_{\Delta,R}) \circ \phi'$ where $\phi' : \square \Rightarrow L, \Delta, \Gamma', \Delta', R$ exists by inductive hypothesis on the number of gates.

On the other hand, if $\alpha = !P_{(?\Gamma,A)}$, $\alpha : L, ?\Gamma, A, R \Rightarrow L, ?\Gamma, !A, R$ and $\phi = !P_{(?\Gamma,!A)} \circ \phi'$ where $\phi' : \square \Rightarrow L, ?\Gamma, A, R$ exists by inductive hypothesis on the number of gates.

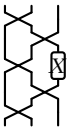
□

Corollary 3.3.6 (Linearity of sequentializability in $\tilde{\mathfrak{E}}$)**.** *It is possible to check if a proof diagram in $\tilde{\mathfrak{E}}$ is sequentializable in linear time.*

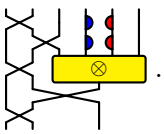
Proof. It suffice to verify if $\mathrm{in}(\phi) = \square$ and $\mathrm{out}(\phi)$ is a string over $\mathfrak{F}_{Me\ell\ell} \cup \{(,)\}$ of the form L, Γ, R with $\Gamma \in \mathfrak{F}_{Me\ell\ell}{}^*$. □

This complexity result we are able to give strongly depends on the introduction of control strings. Indeed, the construction of terms is have to satisfies the well-typing of diagrams with respect of control (corresponding to well-parenthesization) that guarantees the impossibility of configuration such as binary inference rules applied to two formulas of the same branching of a derivation tree or unary inference rules applied on two formulas from two different branching.

However, the direct management of string crossing generate a wider family of critical pairs in rewriting. Some of these critical peaks are not solvable such as the ones in the global conflict generated by the archetype of conflict



for example



This rules out a confluence for this polygraph. We can assert that the lose of the existence of a unique canonical normal-form representative is the price we pay in order to have linear correctness criterion.

This result should not be surprising after Remark 2.5.4. In fact $\tilde{\mathfrak{E}}$ contains the monoidal structure of the !-comonad creating some conflicts with the same shape of the one given in Remark 2.5.2.

As shown in Section 3.2 for the polygraph $\tilde{\mathfrak{U}}$, irreducible $\tilde{\mathfrak{E}}$ proof diagrams have, by interchange law, a layer structure. Then, an irreducible proof diagram in $\tilde{\mathfrak{E}}$ can be interpreted as a focalized *MELL* proof (with explicit exchanges).

As in $\tilde{\mathfrak{U}}$, irreducible proof diagrams in $\tilde{\mathfrak{E}}$ are arranged by layers.

Proposition 3.3.7 (Layers in $\tilde{\mathfrak{E}}$)**.** *Irreducible proof diagrams in* $\tilde{\mathfrak{E}}$ *have a* standard layer form*:*

$$Layer_\ell \circ \cdots \circ Layer_1 \circ Ax : \square \Rightarrow L, \Gamma, R$$

where $Layer_i = Box_i \circ Pos_i \circ Neg_i \circ Twist_i$ *with*

- Ax *is an elementary diagram over the signature* $\{Ax_F, 1\}_{F \in \mathfrak{F}_{Mell_u}}$*:*

  $$Ax : \square \Rightarrow \blacklozenge, \Gamma_1, \blacklozenge \ , \ \ldots \ , \ \blacklozenge, \Gamma_{n_0}, \blacklozenge$$

  with $\Gamma_i = F_i, F_i^\perp$ *for some* $F_i \in \mathfrak{F}_{Mell_u}$ *or* $\Gamma_i = 1$*;*

- $Twist_i$ *is a* block twisting diagram*:*

  $$Twist_i = \blacklozenge, \sigma_{i,1}, \blacklozenge \ , \ \ldots \ , \ \blacklozenge, \sigma_{i,n_i}, \blacklozenge$$

  in particular we have that $Twist_1$ *is an identity;*

- Neg_i *is the* i^{th} negative layer*, a diagram over the signature* $\{\parr, \bot, ?C, ?W, ?D\}$*;*

- Pos_i *is the* i^{th} positive layer*, an horizontal diagram over the signature* $\{\otimes, Cut\}$*;*

- Box_i *is the* i^{th} boxes layer*, an horizontal diagram over the signature* $\{!P\}$*;*

3.4 Towards a new syntax for proofs

In the previous section, we have presented proof diagrams, a particular class of string diagrams suitable for interpreting linear logic proof derivations. In particular, such a setting exhibits an internal correction criterion and, as we have shown, a correspondence between *MELL* (with or without constants, one-sided) sequent calculus proof derivations and proof diagrams. Moreover,

the sequentiability of a proof diagram, i.e. whether it corresponds to a proof in sequent calculus, can be verified in linear time.

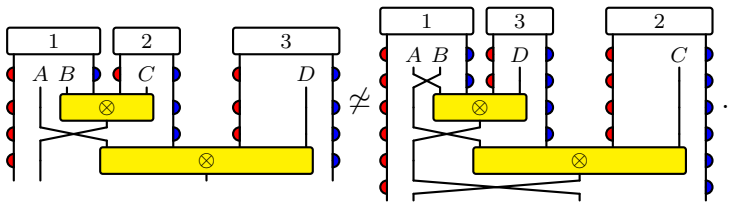
Our results raise an important question about the quotient set over proofs introduced by proof diagrams, and how it relates to that performed by proof nets. For this, let $\sim_D$ be the equivalence relation over proof derivations induced by proof diagrams equivalence in $\langle\tilde{\mathfrak{E}}\rangle$, $\sim_N$ one induced by proof net quotient and $\sim$ the standard proof equivalence (see Appendix A).

One the one hand, on multiplicative fragment, $\sim_D$ captures all commutations of reversible inference rules $\parr$ and $\bot$ by the interchange rule and twisting relations that make the induced quotient less coarse than $\sim_N$ since in proof net syntax the assignation of jumps for $\bot$ cells is mandatory for correctness. Moreover, $\sim_D$ also captures ?**W**, ?**C**, ?**D** interactions with string crossings and boxes interactions with weakening, duplication and, of course, twisting operators. On the other hand, proof diagrams are not able to identify all binary commutation. In fact, for $\otimes$ and Cut, $\sim_D$ equates only permutations of the following kind

$$\dfrac{\dfrac{\begin{matrix}1\\\vdots\\\vdash \Sigma, A\end{matrix}\qquad\begin{matrix}2\\\vdots\\\vdash B,\Gamma,C\end{matrix}}{\vdash \Sigma,\alpha(A,B),\Gamma,C}\,\alpha\qquad\begin{matrix}3\\\vdots\\\vdash D,\Delta\end{matrix}}{\vdash \Sigma,\alpha(A,B),\Gamma,\beta(C,D),\Delta}\,\beta \quad\sim\quad \dfrac{\begin{matrix}1\\\vdots\\\vdash \Sigma, A\end{matrix}\qquad\dfrac{\begin{matrix}2\\\vdots\\\vdash B,\Gamma,C\end{matrix}\qquad\begin{matrix}3\\\vdots\\\vdash D,\Delta\end{matrix}}{\vdash A,\Gamma,\beta(C,D),\Delta}\,\beta}{\vdash \Sigma,\alpha(A,B),\Gamma,\beta(C,D),\Delta}\,\alpha$$

where $\alpha, \beta \in \{\otimes, Cut\}$, that is, when $\otimes$ or Cut permutations that do not change the order of the leafs in a derivation tree. It follows that proof nets equivalence is coarser, in some sence, than proof diagrams one on the multiplicative fragment with no unit.

For a concrete example, consider a provable linear logic sequent of the form $\vdash B \otimes C, A \otimes D$: this latter exhibits two different derivations that correspond to the following two non-equivalent proof diagrams



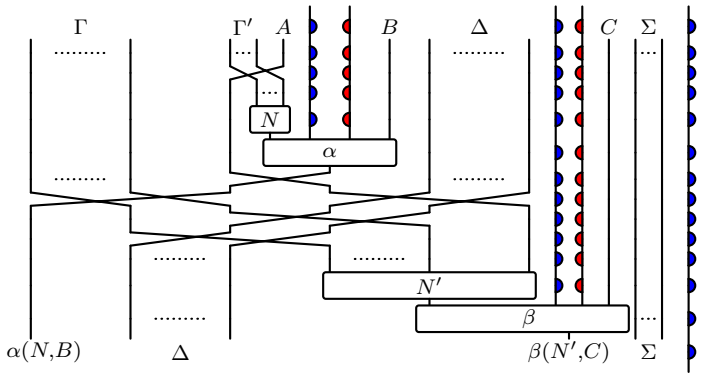
On the other hand, the two proof derivations have the same proof net.

In order to achieve a semantics for proof diagrams including the equivalence described above, we should be able to perform some transformation on diagrams corresponding to leafs permutation in the derivation trees. These transformation are forbidden in $\tilde{\mathfrak{E}}$ since the presence of control strings prevent the application of any twisting relation. In exchange, the possibility of doing such such transformation is crucial in order to prove linear logic

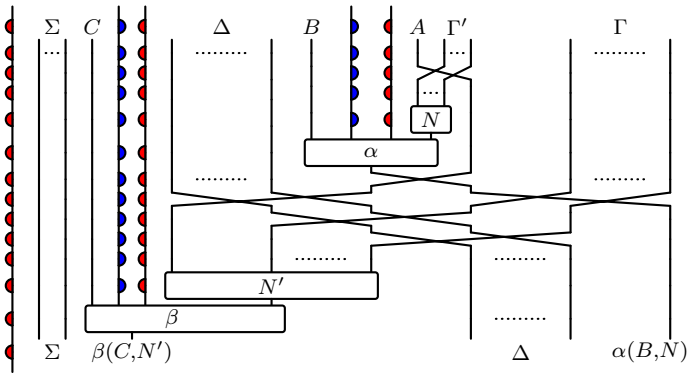
cut-elimination. In fact, some occurrences of cut rules can be applied to non-active formulas, requiring the application of the rule of *commutative cut* in cut-elimination procedure, that is a permutation between two branching in a derivation tree.

In particular, the syntax of proof diagrams allows to observe the "tangle" nature in formulas management underlying these configurations thanks to the informations given by the structure of string crossing. Now we want to give a procedure to *untangle* proofs, this is a standardization procedure in order to recover the whole proof equivalence. For this scope, we need to give a specific order for axioms in a proof diagram in order to avoid configurations presenting what we call a *crossing split*.

Definition 3.4.1 (Crossing split)**.** If $\phi \in \tilde{\mathfrak{E}}$ is an irreducible proof diagram, we consider its layer form, we says it has a *crossing split* if it contain a subdiagram of the form



or



where α, β are splitting cell (i.e. gates of type $\otimes$ or Cut). This is a subdiagram with two splitting gates of type $\otimes$ or Cut where the leftmost (resp. rightmost) split gate g has in its upper cone $\overline{\partial}(g)$ one free port such that the successive (resp. previous) of type R (L).

In other words, we have a crossing split every time we have two applications of binary rule (Cut or $\otimes$) where one of the active formula of the second rule is

the last (resp. first) formula of the first (resp. second) sequent on which the first binary rule is applied.

These correspond to derivations equivalent of one with the following form:

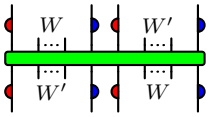
$$\dfrac{\dfrac{\dfrac{\overset{1}{\vdots}}{\dfrac{\vdash \Gamma, \Gamma', A}{\vdash \Gamma, A, N(\Gamma')}\, N} \qquad \overset{2}{\vdots}}{\vdash \Gamma, \alpha(N(\Gamma'), B), \Delta, A}\, \alpha(N(\Gamma'), B) \qquad \overset{3}{\vdots}}{\vdash \Gamma, \alpha(N(\Gamma'), B), \Delta, \beta(A, C), \Sigma}\, \beta(A, C)$$

or

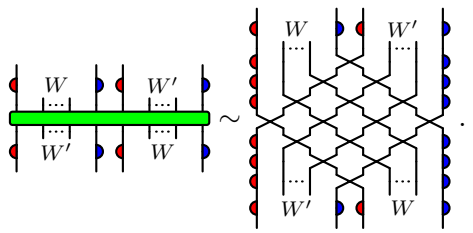
$$\dfrac{\dfrac{\overset{1}{\vdots}}{\vdash \Sigma, C} \qquad \dfrac{\overset{2}{\vdots}}{\vdash \Delta, B} \qquad \dfrac{\overset{3}{\vdots}}{\dfrac{\vdash \Gamma, \Gamma', A}{\vdash N(\Gamma'), A, \Gamma}\, N}}{\vdash A, \Delta, \alpha(B, N(\Gamma')), \Gamma}\, \alpha(B, N(\Gamma'))}{\vdash \Sigma, \beta(C, A), \Delta, \alpha(B, N(\Gamma')), \Gamma}\, \beta(C, A)$$

Recovering proof equivalence for proof diagrams

In this section we extend control polygraph rewriting in order to be able to permute the position of certain Ax and 1 gates in a diagram and their relative sub-derivation. Even if this procedure can be given by means of "big steps", this would imply the definition of rewriting rules with open terms. In order to keep rewriting local, we introduce some new gates representing branching crossing and some rewriting rules able to use them to cross derivation tree branching. This will be done by means of some gates of shape



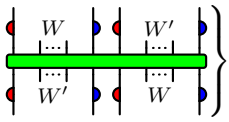
which can be seen as some "big twist" crossing a two blocks of strings of the form L, W, R and L, W', R where $W, W' \in (\mathfrak{F}_{Mell} \cup \{(,)\})^*$, intuitively:



We show that this rewriting system terminates, this procedure of crossing branching is convergent and that we keep our complexity result for proof diagram correction.

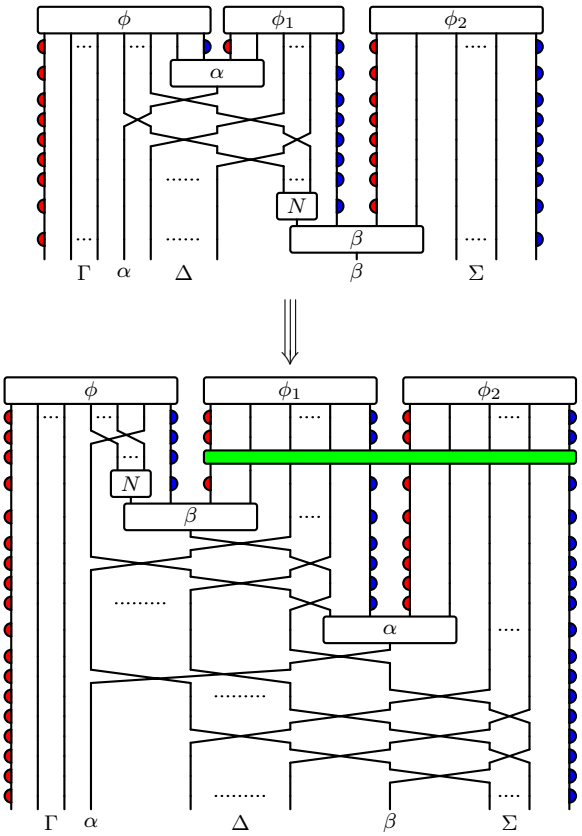
Definition 3.4.2 (Polygraph of diagrammatic *MELL* proof nets)**.** The *polygraph of diagrammatic MELL proof nets* is the polygraph obtained extended the polygraph $\tilde{\mathfrak{E}}$ with the following cells:

- $\mathfrak{E}_0 = \tilde{\mathfrak{E}}_0$;
- $\mathfrak{E}_1 = \tilde{\mathfrak{E}}_1$;
- $\mathfrak{E}_2 = \tilde{\mathfrak{E}}_2 \cup \mathfrak{B}ig$ where

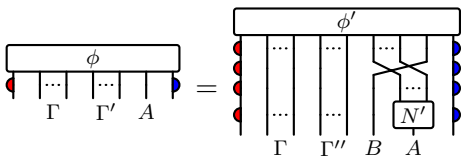
$$\mathfrak{B}ig = \left\{ B_{W,W} = \right.$$  $$\left. \right\}_{W,W' \in (\mathfrak{F}_{Mell} \cup \{L,R\})^*} ;$$

- $\mathfrak{E}_3 = \tilde{\mathfrak{E}}_3 \cup \mathfrak{E}_{Big}$ where $\mathfrak{E}_{Big}$ is made of the following 3-cells for all $\Gamma, \Delta \in \mathfrak{F}_{Mell}{}^*$,  $\in \tilde{\mathfrak{E}}_2$:

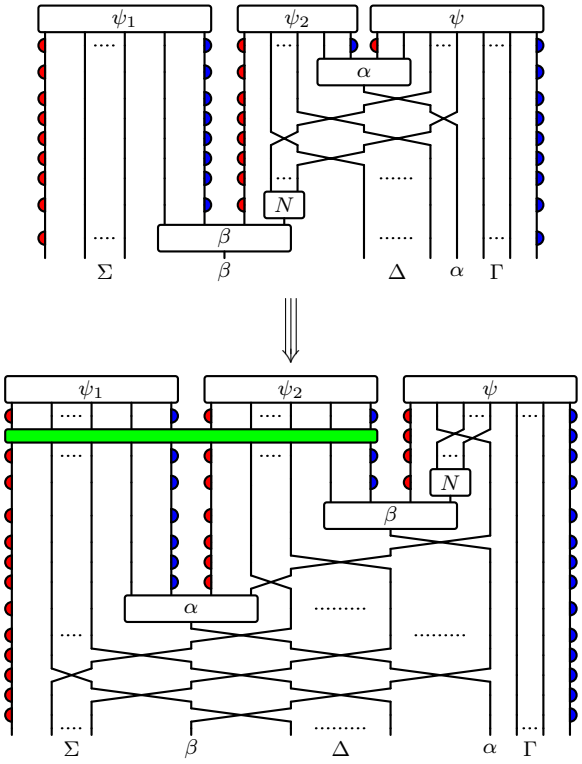
  – $\mathfrak{B}$-introduction: for any $\alpha, \beta \in \{Cut, \otimes\}$ and $\phi, \phi_1, \phi_2, \psi, \psi_1, \psi_2$ irreducible in $\mathfrak{E}$ we define



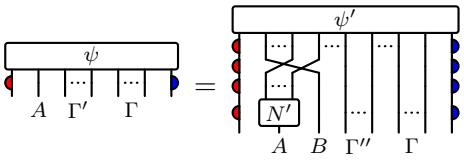
where, if $\Gamma' = \Gamma'', B$, ϕ is of the form



with $N, N' \in \{\between, \parr, \bot, ?\mathbf{D}, ?\mathbf{W}, ?\mathbf{C}\}^*$,

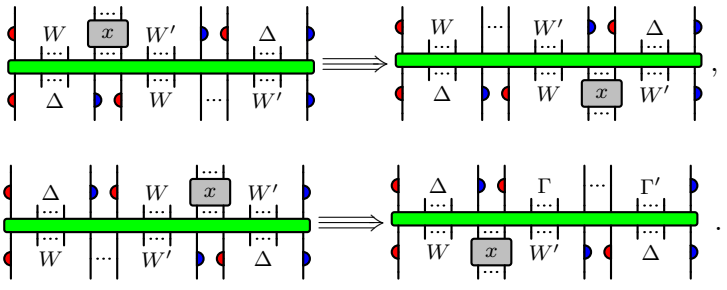


where, if $\Gamma' = B, \Gamma''$, ψ is of the form



with $N, N' \in \{\invamp, \parr, \bot, ?\mathbf{D}, ?\mathbf{W}, ?\mathbf{C}\}^*$;

– The untangle relations: for any  $\in \tilde{\mathfrak{E}}_2$

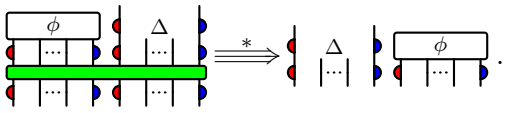


Under the interpretation of B-gates as some kind of big twists crossing two derivation branching, the $\mathfrak{B}$-introduction rules act on a crossing split in

order to eliminate it in an irreducible diagram in $\tilde{\mathfrak{E}}$. In order to do this, a $\mathfrak{B}$-introduction rule operate on the three branching concerning the crossing split: it untangle the string twistings on the principal branch (the one in common of both splitting cells) and it crosses the other two branchings in the derivation tree by the introduction of a B-gate. Additionally, the untangle relations moves gates from the top to the bottom of a B-gate, with respect of the intuition that a B-gate represent a sort of big twisting operator, progressively rewriting the starting diagram into another one where the order of Ax and 1, preventing the presence of the crossing split that induced the $\mathfrak{B}$-introduction.

We observe some important properties for rewriting rules in $\mathfrak{E}_3$:

Proposition 3.4.3. *If $\phi \in \mathfrak{E}$ is a proof diagram $\phi : \square \Rightarrow L, \Gamma, R$, then there is a 3-cell generated by the set of of untangle relations*

$Un_B(\phi) :$ 

.

Proof. By induction over the number of gates in ϕ, it suffice to show that each untangle relation decrease the number of gates of the upper cone $\bar{\partial}(g)$ of a B-gate. □

Proposition 3.4.4 (Untangle procedure is confluent)**.** *If $\phi \in \tilde{\mathfrak{E}}$ is an irreducible proof diagram $\phi : \square \Rightarrow L, \Gamma, R$, then there is a unique diagram $\hat{\phi} \in \tilde{\mathfrak{E}}$ with no crossing split such that $\phi \underset{\tilde{\mathfrak{E}}_{Big}}{\Rrightarrow} \hat{\phi}$.*

Proof. Let $\rightarrow_B$ be the rewriting over the proof diagrams diagrams in $\mathfrak{E}$ generated by 3-cells in $\mathfrak{B}ig$. We note that a diagram with no crossing split is irreducible for this rewriting since it contains no B-gates and no B-gates can be introduced. For any proof diagram, the number of crossing split is finite, then also the number of $\mathfrak{B}$-introduction rules present in any rewriting path is finite. By induction on the number of gates in a diagram ϕ and the Proposition 3.4.3, it results that this rewriting terminates. In order to prove convergence, it suffices to remark that untangle relations critical pairs are strongly confluent. Moreover $\mathfrak{B}$-introduction rules generate no non-trivial conflicts. □

Corollary 3.4.5 (Untangle sequence)**.** *Any path with irreducible target in $\rightarrow_B$ can be written as a sequence of* untangle sequences*, these are rewriting path made of a single $\mathfrak{B}$-introduction followed by untangle relations we need to erase this cell.*

Corollary 3.4.6. *Conflicts between 3-cells in $\tilde{\mathfrak{E}}_3$ and $\mathfrak{E}_{Big}$ are confluent and the corresponding rewriting rules commute.*

This lead the following theorem about the termination of rewriting in $\mathfrak{E}$.

Corollary 3.4.7 (Termination in $\mathfrak{E}$)**.** *The polygraph $\mathfrak{E}$ is terminating.*

Proof. By Theorem 3.3.4 we know that $\tilde{\mathfrak{E}}$ terminates and by Proposition 3.4.4 the rewriting of untangle procedure too. The result follows by the possibility of commutation of these two rewritings remarked in Corollary 3.4.6 and by the fact that rules in $\mathfrak{E}_{Big}$ rewrites on irreducible proof diagram modulo $\tilde{\mathfrak{E}}_3$. □

A whole untangle sequence on a proof diagram corresponds to the elimination of a non-commutative cut commuting the order of branching in the derivation tree.

We extend the Theorem 3.3.5 to proof diagrams in $\mathfrak{E}$. This leads the extension of Corollary 3.3.6 of the linear complexity of sequentializability of a proof diagram in $\mathfrak{E}$.

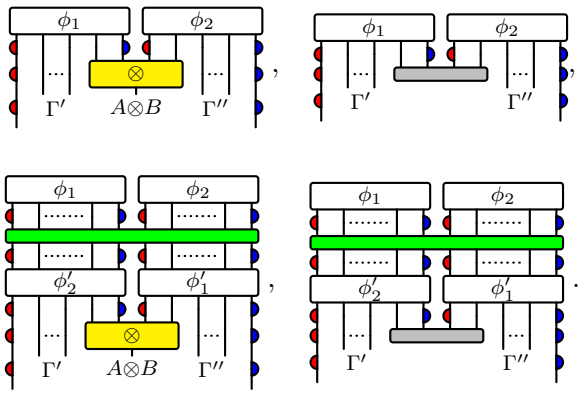
Theorem 3.4.8 (Diagrammatic proof net correspondence in $\mathfrak{E}$)**.**

$$\vdash_{MELL} \Gamma \Leftrightarrow \exists \phi \in \mathfrak{E} \text{ such that } \phi : \square \Rightarrow L, \Gamma, R.$$

Proof. In order to prove this theorem, we have to modify the proof of the right-to-left implication in Theorem 3.3.5 in order to include the cases of 2-cells in $\mathfrak{B}ig$. We observe that a proof diagram $\phi : \square \Rightarrow L, \Gamma, R$ contains a gate of type $B \in \mathfrak{B}ig$ iff

$$\phi = (\mathbf{id}_{L,\Gamma'}, \alpha, \mathbf{id}_{\Gamma'',R}) \circ B \circ (\phi', \phi'')$$

with α a binary rule in $\{Cut, \otimes\}$ and $\Gamma = \Gamma', \Gamma''$. So, in case of a binary rule α we have the following four cases:



the first two cases are handled by the same strategy. In order to prove sequentialization for the two new cases it suffices to remark that the idea behind the B-gates are twisting, so even if the diagram $(\phi'_2, \phi'_1) \circ B \circ (\phi_1, \phi_2) : \square \Rightarrow L, \Gamma'', R, L, \Gamma', R$ is not sequentializable, the diagrams $\phi'_1 \circ \phi_1 : \square \Rightarrow L, \Gamma', R$ and $\phi'_2 \circ \phi_2 : \square \Rightarrow L, \Gamma'', R$ are. □

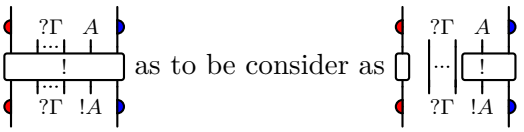
By the termination of $\mathfrak{E}$ we deduce that each proof diagram corresponds to a focalized linear logic derivation with explicit exchanges (before positive rules) and without crossing split. That is, the rewriting of $\mathfrak{E}$ induce a standardization procedure over the splitting order in linear logic derivations.

The polygraph $\mathfrak{E}$ is the candidate for a new semantic of proof net. By the twisting relations we are able to make proof diagrams gates interact with crossings (twisting operators) as cells are free to move with respect of crossings in proof net representations. Nevertheless, the presence of the non-twisting control strings require the machinery of B-gates and untangle sequences in order to recover this interaction in some particular cases concerning the splitting rules.

Moreover, the control strings plays two other important roles in units and box correctness. On one hand, the position of a $\bot$ or ?$\mathbf{W}$ with respect of control strings assigns plays the same role of jump assignation in proof net correctness but, on the other hand, this position is not so abridging as a jump since it links the cell not to a leak of a derivation tree but to one of its branchings, and consequently to any leaf of such branching. Furthermore, the structure of $!\mathbf{P}$ gates assures that its upper cone is a proof diagram $\phi : \square \Rightarrow L, ?\Gamma, A, R$, that is a proof diagram corresponding to a linear logic derivation. This implies the correctness criterion on boxes for *MELL* proof net.

Theorem 3.4.9 (Proof net correspondence in $\mathfrak{E}$)**.** *There is a one-to-one correspondence between the set of proof diagrams in $\mathfrak{E}$ modulo modulo the equivalence relation $\simeq_{Tw}$ generated by $\tilde{\mathfrak{E}}_{Twist} \cup \mathfrak{E}_{Big}$ and the set of multiplicative proof structures with constants.*

Proof. Without losing generality, by Proposition 3.4.3, we consider proof diagrams with no B-gates. As in Proposition 3.1.10, it suffices to remark that twisting relations do not change (twisting) communicating relations between gates which correspond to adjacencies in the associated proof net. Then, if we consider any $!P_{?\Gamma,A}$ gate as below



we define a (twisting) communication relation between all gates in a proof diagrams ϕ.

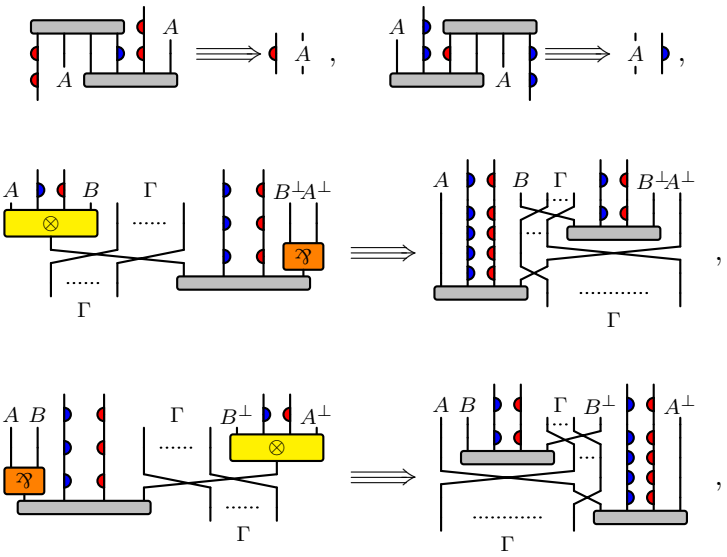
The proof net P_ϕ is the one defined by this adjacency relation over the set of cells in one-to-one correspondence with non-twisting gates of ϕ. The boxes of P_ϕ are the upper cones $\overline{\partial}(g)$ of the gates $g : !\mathbf{P}$, $g \in \phi$. $\square$

3.5 Cut elimination for proof diagrams

In this section we give the set of 3-cells corresponding to cut-elimination. We note that the absence of crossed split avoids the presence of the so called *non-commutative cuts*.

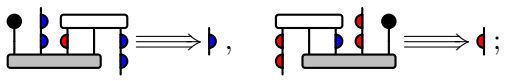
Definition 3.5.1 (Cut-elimination 3-cells)**.** We define the following sets of 3-cells:

- $\tilde{\mathfrak{M}}_{Cut}$ is made of the following 3-cells:

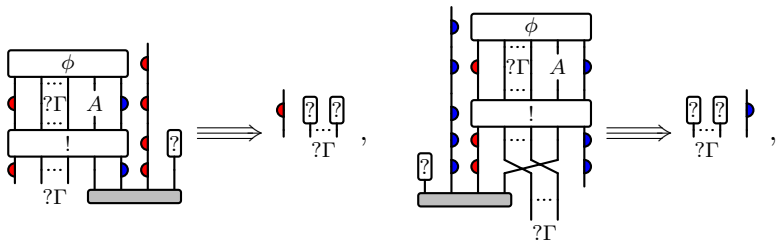


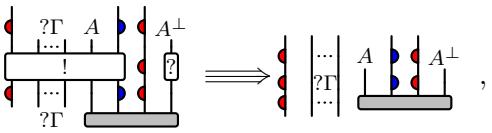
  for all $A, B \in \mathfrak{F}_{Mell}$, $\Gamma \in \mathfrak{F}_{Mell}{}^{*}$;

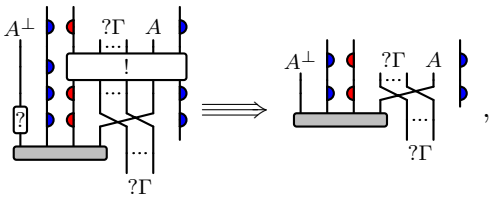
- $\tilde{\mathfrak{U}}_{Cut}$ is made of the following 3-cells:

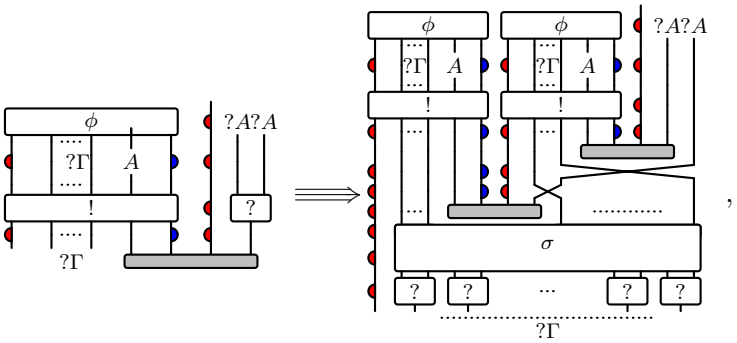


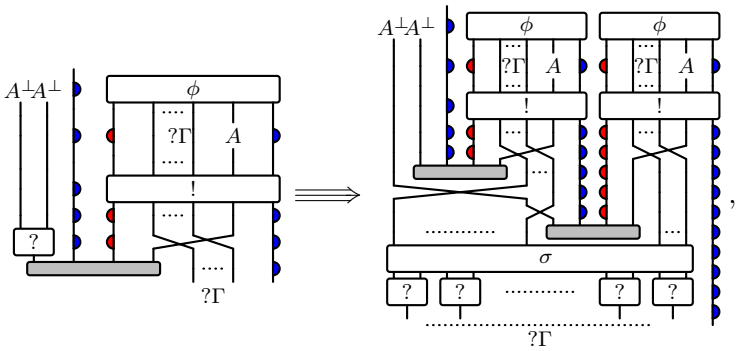
- $\tilde{\mathfrak{E}}_{Cut}$ is made of the following 3-cells:

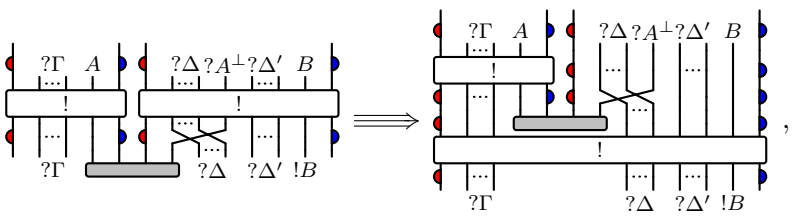


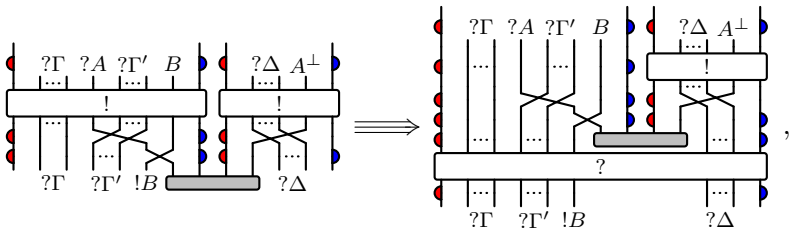

?Γ
A
A⊥
!
?
?Γ
?Γ
A
A⊥


A⊥
?Γ
A
!
?
?Γ
A⊥
?Γ
A
?Γ


ϕ
?Γ
A
!
?A?A
?
?Γ
ϕ
?Γ
A
!
ϕ
?Γ
A
!
?A?A
σ
?
?
?
?
?Γ


A⊥A⊥
ϕ
?Γ
A
!
?
?Γ
A⊥A⊥
ϕ
?Γ
A
!
ϕ
?Γ
A
!
σ
?
?
?
?
?Γ


?Γ
A
!
?Δ ?A⊥ ?Δ′
B
!
?Γ
?Δ
?Δ′
!B
?Γ
A
?Δ ?A⊥ ?Δ′
B
!
?Γ
?Δ
?Δ′
!B



for all $A, B \in \mathfrak{F}_{Mell}$, $\Gamma, \Gamma', \Delta, \Delta' \in \mathfrak{F}_{Mell}{}^*$, $\phi \in \tilde{\mathfrak{E}}$ irreducible, $\sigma \in S_{2n}$ such that $\sigma(i) + 1 = \sigma(n + i)$.

We can now define the semantics of linear logic by means of polygraphs:

Definition 3.5.2. We define the polygraph of diagrammatic *MELL* proof nets $\mathfrak{E}_{Cut}$ the one given by $\mathfrak{E}$ plus the set of 3-cells $\tilde{\mathfrak{M}}_{Cut} \cup \tilde{\mathfrak{U}}_{Cut} \cup \tilde{\mathfrak{E}}_{Cut}$. By its restrictions over the 2-cell sets we can define the polygraph of diagrammatic *MLL* and MLL_u proof nets $\mathfrak{M}_{Cut}$ and $\mathfrak{U}_{Cut}$.

In these polygraphs, as in linear logic sequent calculus, it is possible to prove a cut-elimination theorem following their termination.

Theorem 3.5.3 (Termination of $\mathfrak{E}_{Cut}$)**.** *The polygraph $\mathfrak{E}_{Cut}$ is terminating.*

Proof. The termination of $\mathfrak{E}_{Cut}$ follows from the one of $\mathfrak{E}$ together extending the termination order $[\![\phi]\!] = ([\phi], [\phi]_C, \|\phi\|^X_{Box})$ given for $\tilde{\mathfrak{E}}$ to $[\![\phi]\!]_{Cut} = (\|\phi\|_{Cut}, [\![\phi]\!])$ where

$$\|\phi\|_{Cut} = \sum_{g:Cut}^{g\in\phi} \|g\|_{Cut}$$

is the sum of the weights of Cut-gates. This weights is defined in the same way of the one in the cut-elimination proof for linear logic sequent calculus (See Theorem A.0.3 in Appendix A). □

Theorem 3.5.4 (Cut-elimination in $\mathfrak{E}_{Cut}$)**.** *If $\phi \in \mathfrak{E}_{Cut}$ is irreducible, then $|\phi|_{\{Cut\}} = 0$.*

Proof. If $\phi \in \mathfrak{E}_{Cut}$ is irreducible, then it is irreducible also in $\mathfrak{E}$. This means that in ϕ there are no crossing split and then any Cut-gate belongs in a subdiagram ψ of ϕ of the shape of a source $s(R)$ of a 3-cell $R \in \tilde{\mathfrak{E}}_{Cut}$. □

Another proof of this theorem follows the correspondence between gates in a proof diagram and cells in the associate proof net and the cut-elimination theorem for *MELL* proof nets. In fact, it suffice to give an interpretation of how the three sets of rules $\tilde{\mathfrak{E}}_3$, $\mathfrak{E}_{Cut}$ and $\mathfrak{B}ig$ act on the proof net corresponding to the proof diagram: twisting relation in $\tilde{\mathfrak{E}}_3$ corresponds to no action or, rules in $\tilde{\mathfrak{E}}_{Box}$ to take of some cell from out of a box, the set of rule $\tilde{\mathfrak{E}}_{mon}$ give the monoidal structure over weakening and contraction, each rule in $\mathfrak{E}_{Cut}$

correspond to a cut-elimination step and rules in $\mathfrak{Big}$ correspond to no action since even if 2-cells in $\mathfrak{Big}$ corresponds to no possible label for interaction nets cells, their interpretation is wire crossing.

Chapter 4

Conclusions

"A mathematician is a blind man in a dark room looking for a black hat which isn't there."

[C. Darwin]

In Chapter 2 we have formalized the structure of string diagrams and their semantics. In particular, it is given a construction of these terms in term of 2-dimensional word rewriting. This allows us to formalize some notions such as gates adjacency, gates paths and connection reflecting the intuition given by their graphical representation and their adaptation into the case of twisting polygraphs. Moreover, in this formalism we can also define some topological notions such as the one of internal and external gates.

With the introduction of the diagrammatic variables, we also formalize the diagrammatic term substitution. Together with the notion of external gates, this allows well-define some non-well-typed term substitutions giving some new paradigms for the notion of context which allow to recover some finiteness properties for finite rewriting systems.

Future study of confluence in sting diagram syntax, should be take into account of these tools which leads to some open questions about their application.

In Chapter 3 is defined a new semantic of proof diagrams for the linear logic proof nets. The principal innovation with respect of proof net is the control of string crossings which allow the definition of the control strings $\blacktriangleleft, \blacktriangleright$. On one hand, this guarantees the introduction of an internal correctness criterion which prevent the formulation of incorrect terms because they reproduce, in some sense, a good parethesization in derivation underlying the splitting order. On the other hand, this rule out from the semantic of $\tilde{\mathfrak{M}}$, $\tilde{\mathfrak{U}}$ and $\tilde{\mathfrak{E}}$ the commutativity of $\otimes$ and *Cut* when they are not in parallel. As we shown, in order to recover the full semantic of linear logic and a one-to-one correspondence between equivalence classes of proofs and 2-cells, we have to extend these models to the polygraphs $\mathfrak{M}, \mathfrak{U}$ and $\mathfrak{E}$. As we remarked, in

this model the tool of jumps is not more required: crucial in our settings is the fact that $\perp$ gates have a specific position in diagrams, that one can interpret as a jump assignment. For example, given a $\perp$ gate, we can point its jump to the unique gate of type Ax or 1 connected to the left-nearest L string. In particular, this means that equivalent proof diagrams in $(\tilde{\mathfrak{U}})^*$ may correspond to different jump assignments on the same proof net and this justify the fact that the complexity of correction is still linear. Indeed, proof diagrams exhibit a local sequentialization criterion which is ruled out in proof nets by complexity arguments (P. Lincoln and T. Winkler [59] , W. Heijltjes [40]), due to the number of jumps to check.

Moreover, the 3-cells in $\tilde{\mathfrak{E}}_{Box}$ guarantee the good semantic of boxes with the possibility of taking out a box a **!W** or a **!C**. We underline that, even in this case, control strings prevent guarantee the well-definition of boxes since the diagram given by the upper cone $\overline{\partial}(g)$ of a gate of type $!P$ is a correct proof diagram (corresponding to a branch of a derivation tree). We also observe that $\mathfrak{E}_{mon}$ gives some rules which allow us to give a standard form for multiple contraction/weakening over the same formula.

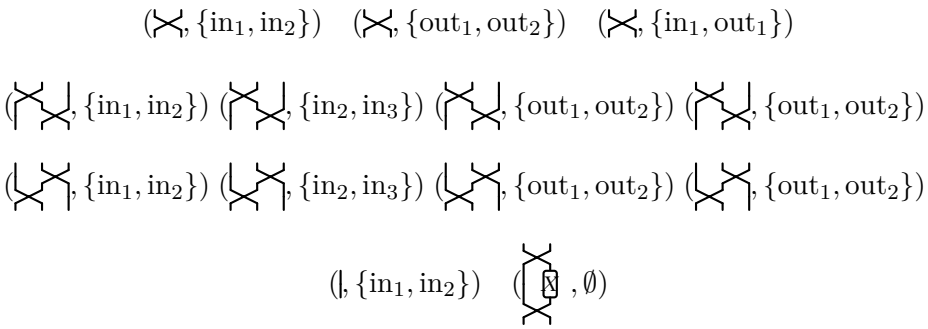
The definition of 3-cells for cut-elimination, complete this model with a representation of proof net with linear correctness criterion for the studied fragment and a semantics less coaster then the one of proof nets coinciding with the semantics of equivalent linear logic *MELL* proofs.

4.1 Open questions on 2-dimensional grammars

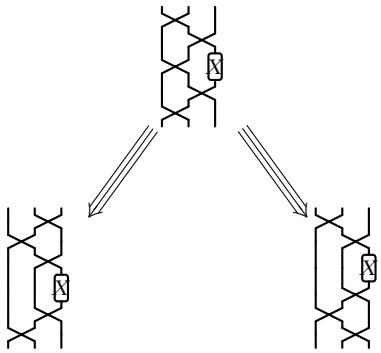
Determination of global conflicts

One possible direction of study is to decompose rewriting rules premises in order to define *minimal active diagrams* (MAD for short), these are irreducible partial diagrams together with a set of their free port such that, properly composing with another such diagrams give a reducible diagram. Once define the set of a rewriting system's MAD, we can use them in order to restrict the number of elements in a global conflict.

Example 4.1.1. In $\mathfrak{S}$ we have the following MAD:



and a unique archetype of conflict



which is solvable only if we give a proper substitution of the 2-cell  . In fact, observing MAD we note that rewriting can be possible only in case of the substitution with a | or  , which corresponds to the two critical pairs

of the generated global conflict.

On syntax' dimension

A possible investigation comes out from the study of string diagrams related to a notion of *dimension* of a syntax. In fact, diagrams can not properly be considered in general as a 2-dimensional syntax since gates with a number of input and outputs different from 1 reduce the degrees of liberty in the construction of terms. For example, in binary trees generated by {  }, each node gives not only the information of its inputs and its output, but also an order over them. Moreover, these information rule out some possible configuration for terms such as, in this case, the absence of 1-factor.

Question 1. Is it possible to define a notion of dimension for diagrammatic grammars?

In particular we are interested if factors have any correlation to the property of *finite derivation type* for string diagrams and the relative application to an extension of Squier Theorem [78].

4.2 Open questions on proof diagrams model

On proof diagram semantics

We have defined the set of cut-elimination rules $\tilde{\mathfrak{E}}_{Cut}$ over the set of diagrams in $\mathfrak{E}$. This choice is due to the fact that, in cut-elimination procedure, we are

forced to apply non-commutative cuts in order to arrange derivation in suitable configurations. As we have shown, these non-commutative cuts corresponds to untangle procedures. If we consider the proof equivalence induced by $\tilde{\mathfrak{E}}$ over linear logic proof, this is not able to equate some configurations of $\otimes$ and Cut, morally leading out the cut-elimination result. Considering this observation, the following question arises:

Question 2. If we define $\tilde{\mathfrak{E}}_{Cut}$ the polygraph $\tilde{\mathfrak{E}}$ extended with 3-cells in $\mathfrak{E}_{Cut}$, what does it represents the induced semantics over linear logic proofs?

On confluence

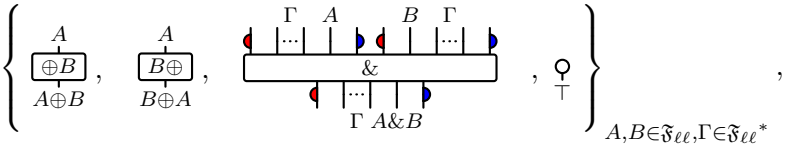
The presence of some non-convergent critical peaks rules out the confluence of the polygraphs presenting proof diagrams. This is due to some unsolvable critical pairs concerning twisting operators.

On the other hand, since proof diagrams propose is to represent equivalent linear logic derivations, we can naturally consider some equivalences over proof diagrams concerning the representations and the positions of the twisting operators, i.e representing the omission of exchange rules in linear logic sequent calculus.

Question 3. Is it possible to define an equivalence relation over proof diagrams in order to achieve a confluence modulo ?

Additives

The use of proof diagrams for linear logic proof net can be extended in order to include additives and define a polygraph $\mathfrak{LL}$ representing the whole linear logic sequent calculus. If we note $\mathfrak{F}_{\ell\ell}$ the set of formulas of linear logic, the idea is to define the some new gate types



the relative twisting relations for the $\oplus$-gates and some relations for &-gates, similar in shape with rules il $\tilde{\mathfrak{E}}_{Box}$ for boxes, in order to recover the proof equivalence in this semantics. Of course, also for this fragment, we should adapt the rules for the introduction of B-gates with a set of 3-cells $\mathfrak{L}_{Big}$ in order to accommodate the new binary operator &. In this polygraph, we can finally describe the 3-cells for cut-elimination rules.

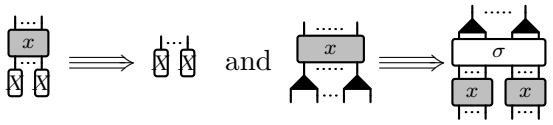
Question 4. How many shape of 3-cells do we need?

Question 5. Do we need the introduction of auxiliary 2-cells (such as B-gates in $\mathfrak{E}$) in order to recover the proof semantics?

For this model we attend the linearity of the correctness criterion such as the one for $\mathfrak{E}$ due to the restrictions in terms construction due to the presence of control strings.

Box management

Some of rules involving the gates representing boxes imply the duplication and the erasing of a whole subdiagram representing the content of a box. Of course, as we have done with the introduction of B-gates for the elimination of crossing splits, we can extend our polygraph in order to decompose these "big steps" into "small steps" achieving a local rewriting. In order to manage also resources *duplication* and *erasing*, we should need the introduction of the gates  (for the duplication) and  (for the erasing) together with some rewriting rules such as



where $\sigma \in S_{2n}$ with

$$\sigma(i) = \begin{cases} i & \text{if } i \text{ odd} \\ n+i & \text{if } i \text{ even} \end{cases}$$

in order to make these gates behave as we wish. Of course, the shapes targets of some $\mathfrak{E}_{Cut}$ 3-cells should be redefined. In particular, in rewriting some intermediate steps are present during the duplication or erasing of a box, giving some diagrams corresponding (not directly) to no proof in linear logic sequent calculus.

Question 6. What does it represent a proof diagrams where these gates occurs?

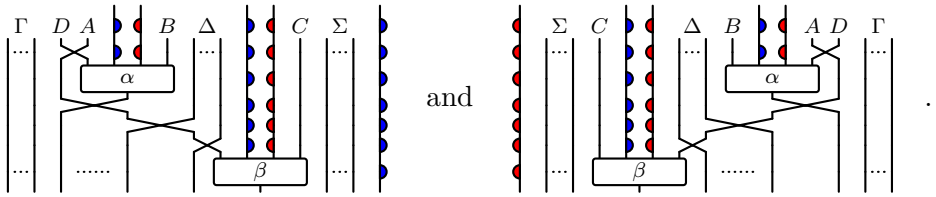
Moreover, the presence of these gates for resources management require a definition of a new correctness criterion for proof diagrams .

Question 7. Can we define a (linear) correctness criterion on these diagrams?

On untangle sequences

Of course the premises of $\mathfrak{B}$-introduction rules require the verification of wide irreducible subdiagrams in order to be applied. On the other hand we note

that any of these premise admit an $\sim_{Tw}$-equivalent subdiagram of the form



Question 8. Is it possible to give a rewriting system equivalent to $\mathfrak{E}$ with more smaller or non-irreducible premises for $\mathfrak{B}$-introduction rules similarly to the the ones listed above?

Appendix A

Linear Logic backgrounds

Jean-Yves Girard's *linear Logic* [28] is a substructural logic which refines classical logic by introducing exponentials, thus permitting to control applications of weakening and contraction.

In the present thesis we focus on the multipicative exponential fragment of linear logic's sequent calculus with constants (*MELL*). We first recall the standard inference rules:

Structural Rules	Identity or Axiom $\dfrac{}{\vdash A, A^{\perp}}\, Ax$	Cut $\dfrac{\vdash \Sigma, A \qquad \vdash \Gamma, A^{\perp}}{\vdash \Sigma, \Gamma}\, Cut$
Multiplicative Rules	Tensor $\dfrac{\vdash \Sigma, A \qquad \vdash B, \Gamma}{\vdash \Sigma, (A \otimes B), \Gamma}\, \otimes$	Par $\dfrac{\vdash \Sigma, A, B}{\vdash \Sigma, A \parr B}\, \parr$
Exponential Rules	Weakening $\dfrac{\vdash \Sigma}{\vdash \Sigma, ?A}\, ?\mathbf{W}$ Dereliction $\dfrac{\vdash \Sigma, A}{\vdash \Sigma, ?A}\, ?\mathbf{D}$	Contraction $\dfrac{\vdash \Sigma, ?A, ?A}{\vdash \Sigma, ?A}\, ?\mathbf{C}$ Of course / Bang $\dfrac{\vdash ?\Sigma, A}{\vdash ?\Sigma, !A}\, !\mathbf{P}$

We also consider the usually omitted exchange rule:

$$\frac{\vdash A_1, \ldots, A_k}{\vdash A_{\sigma(1)}, \ldots, A_{\sigma(k)}}\, \sigma \in S_k$$

We recall that the sequent calculus for the multiplicative fragment of of linear logic (MLL) is composed only of the inference rules $Ax, Cut, \otimes, \parr$. In order to represent multiplicative constants, we need to add to MLL the inference rules of $\bot$ and 1, thus respectively obtaining the multiplicative fragment with constants MLL_u and the multiplicative exponential fragment with constants $MELL$.

	Bottom	1
Constants	$\dfrac{\vdash \Sigma}{\vdash \Sigma, \bot}\ \bot$	$\dfrac{}{\vdash 1}\ 1$

Remark A.0.1 (On Negation)**.** *We assume that negation is involutive, i.e.* $A^{\bot\bot} = A$ *and that the De-Morgan laws apply with respect to* $\parr$ *and* $\otimes$*, i.e.* $(A \heartsuit B)^\bot = B^\bot \heartsuit^\bot A^\bot$ *for any formulas* A, B *where* $\heartsuit = \parr$ *and* $\heartsuit^\bot = \otimes$ *or vice versa* $\heartsuit = \otimes$ *and* $\heartsuit^\bot = \parr$*. Moreover* $1^\bot = \bot$*,* $(!A)^\bot = ?A^\bot$ *and* $(?A)^\bot = !A^\bot$*.*

Remark A.0.2 (On Rules)**.** *In this model we interpret all inference rules as operations with specific arities over the set of sequents:* Ax *and* 1 *are* 0*-ary,* $\parr$*,* $\bot$*,* $?\mathbf{W}$*,* $?\mathbf{C}$*,* $?\mathbf{D}$ *and* $!\mathbf{P}$ *are unary while* $\otimes$ *and* Cut *are binary.*

Notation. We indicate with $\mathfrak{F}_{M\ell\ell}$, $\mathfrak{F}_{M\ell\ell_u}$, $\mathfrak{F}_{Me\ell\ell}$ the set of formulas respectively in MLL, MLL_u and $MELL$.

Cut Elimination

As in classical sequent calculus, also linear logic has a *cut-elimination* theorem:

Theorem A.0.3. *If a sequent* $\vdash \Gamma$ *is provable in MELL so there is a prove of* Γ *without Cut rule.*

Proof. In order to prove the theorem, we give a procedure to eliminate Cut rules form derivation without proving its termination.

We say that (an occurrence of) a formula is *active* in a rule if it is present in the premise of the rule but not in the conclusion, and *principal* whether it is active in the preceding rule of the derivation.

We distinguish three cases in the cut-elimination proof, depending whether the two active formulas occurring in the Cut rule are principal or not:

- If both formulas active in the cut are not principal, it will be possible to simply switch the position of the Cut rule in the derivation tree;
- If at least one of the formulas active in the Cut is not principal, it will be possible to switch the position of the Cut (also call *cummutative cut*):
  - In the case of binary rules:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, A\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Sigma', C, B\end{array}}{\vdash \Sigma, \Sigma', C, (A \odot B)}\odot \qquad \begin{array}{c}\vdots\\ \vdash \Gamma, C^{\perp}\end{array}}{\vdash \Sigma, \Sigma', (A \odot B)} Cut$$

$$\Downarrow$$

$$\dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma', C, B\end{array} \qquad \begin{array}{c}\vdots\\ \vdash \Gamma, C^{\perp}\end{array}}{\vdash \Sigma', \Gamma, B} Cut}{\vdash \Sigma, \Sigma', \Gamma, (A \odot B)}\odot$$

where

$$A \odot B = \begin{cases} A \otimes B & \text{if } \odot = \otimes \\ \emptyset & \text{if } \odot = Cut \end{cases} .$$

In fact, we never consider the case of $\odot = Cut$ but we note it in order to have an example of what an untangle procedure does (see Section 3.4).

– In case of unary rules:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, C\end{array}}{\vdash \odot(\Sigma), C}\odot \qquad \begin{array}{c}\vdots\\ \vdash \Gamma, C^{\perp}\end{array}}{\vdash \odot(\Sigma), \Gamma} Cut$$

$$\Downarrow$$

$$\dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, C\end{array} \qquad \begin{array}{c}\vdots\\ \vdash \Gamma, C^{\perp}\end{array}}{\dfrac{\vdash \Sigma}{\vdash \odot(\Sigma)}\odot} Cut$$

where

$$\odot(\Sigma) = \begin{cases} \Sigma, \perp & \text{if } \odot = \perp \\ \Sigma', A \parr B & \text{if } \odot = \parr \text{ and } \Sigma = \Sigma', A, B \\ \Sigma, ?A & \text{if } \odot = ?\mathbf{W} \\ \Sigma', ?A & \text{if } \odot = ?\mathbf{C} \text{ and } \Sigma = \Sigma', ?A, ?A \\ \Sigma', ?A & \text{if } \odot = ?\mathbf{D} \text{ and } \Sigma = \Sigma', ?A \end{cases} .$$

- If both formulas active in the occurrence of Cut are principal, the Cut is eliminated as follows (we need to consider all the possible combinations of rules):

  – $\otimes$ vs $\parr$

$$\dfrac{\dfrac{\vdots}{\vdash \Sigma, A} \quad \dfrac{\vdots}{\vdash B, \Sigma'}}{\vdash \Sigma, A \otimes B, \Sigma'}\otimes \quad \dfrac{\vdots}{\vdash \Gamma, A^\perp, B^\perp}\parr}{\vdash \Sigma, \Gamma, \Sigma'}Cut$$

$$\Downarrow$$

$$\dfrac{\vdots}{\vdash \Sigma, A} \quad \dfrac{\dfrac{\vdots}{\vdash B, \Sigma'} \quad \dfrac{\vdots}{\vdash \Gamma, A^\perp, B^\perp}}{\vdash \Gamma, A^\perp, \Sigma'}Cut}{\vdash \Sigma, \Gamma, \Sigma'}Cut$$

  – $\perp$ vs 1

$$\dfrac{\dfrac{}{\vdash 1}1 \quad \dfrac{\dfrac{\vdots}{\vdash \Gamma}}{\vdash \Gamma, \perp}\perp}{\vdash \Gamma}Cut$$

$$\Downarrow$$

$$\vdots$$
$$\vdash \Gamma$$

  – ?**W** vs !**P**

$$\dfrac{\dfrac{\vdots}{\vdash \Sigma}}{\vdash \Sigma, ?A}?\mathbf{W} \quad \dfrac{\vdots}{\vdash ?\Gamma, A^\perp}\,!\mathbf{P}}{\vdash \Sigma, ?\Gamma}Cut$$

$$\Downarrow$$

$$\dfrac{\dfrac{\vdots}{\vdash \Sigma}\,?\mathbf{W}}{\dfrac{\vdots}{\vdash \Sigma, ?\Gamma}\,?\mathbf{W}}$$

– $?\mathbf{D}$ vs $!\mathbf{P}$

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Sigma, A}}{\vdash \Sigma, ?A}\,?\mathbf{D} \qquad \dfrac{\dfrac{\vdots}{\vdash ?\Gamma, A^{\perp}}}{\vdash ?\Gamma, !A^{\perp}}\,!\mathbf{P}}{\vdash \Sigma, ?\Gamma}\,Cut$$

$$\Downarrow\Downarrow$$

$$\dfrac{\dfrac{\vdots}{\vdash \Sigma, A} \qquad \dfrac{\vdots}{\vdash ?\Gamma, A^{\perp}}}{\vdash \Sigma, ?\Gamma}\,Cut$$

– $?\mathbf{C}$ vs $!\mathbf{P}$

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Sigma, ?A, ?A}}{\vdash \Sigma, ?A}\,?\mathbf{C} \qquad \dfrac{\dfrac{\vdots}{\vdash ?\Gamma, A^{\perp}}}{\vdash ?\Gamma, !A^{\perp}}\,!\mathbf{P}}{\vdash \Sigma, ?\Gamma}\,Cut$$

$$\Downarrow\Downarrow$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Sigma, ?A, ?A} \qquad \dfrac{\dfrac{\vdots}{\vdash ?\Gamma, A^{\perp}}}{\vdash ?\Gamma, !A^{\perp}}\,!\mathbf{P}}{\vdash \Sigma, ?\Gamma, ?A}\,Cut \qquad \dfrac{\dfrac{\vdots}{\vdash ?\Gamma, A^{\perp}}}{\vdash ?\Gamma, !A^{\perp}}\,!\mathbf{P}}{\vdash \Sigma, ?\Gamma, ?\Gamma}\,Cut}{\vdash \Sigma, ?\Gamma}\,?\mathbf{C}}{\dfrac{\vdots}{\vdash \Sigma, ?\Gamma}\,?\mathbf{C}}$$

– $!\mathbf{P}$ vs $!\mathbf{P}$

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash ?\Sigma, ?B, A}}{\vdash ?\Sigma, ?B, !A}\,\mathbf{!P} \qquad \dfrac{\dfrac{\vdots}{\vdash ?\Gamma, B^{\perp}}}{\vdash ?\Gamma, !B^{\perp}}\,\mathbf{!P}}{\vdash ?\Sigma, ?\Gamma, !A}\,Cut$$

$$\Downarrow$$

$$\dfrac{\dfrac{\vdots}{\vdash ?\Sigma, ?B, A} \qquad \dfrac{\dfrac{\vdots}{\vdash ?\Gamma, B^{\perp}}}{\vdash ?\Gamma, !B^{\perp}}\,\mathbf{!P}}{\dfrac{\vdash ?\Sigma, ?\Gamma, A}{\vdash ?\Sigma, ?\Gamma, !A}\,\mathbf{!P}}\,Cut$$

□

The proof of termination of this procedure is given in [28] giving a weight *to any occurrence of Cut inference rules in a derivation. The termination order for this reduction is given by the sum of these weight. In fact, each cut-elimination step reduces the weight of Cuts in a derivation.*

Proof equivalence

In linear logic we consider the equivalence over proof generated by the following relations over derivations in sequent calculus:

- If $\odot_1, \odot_2 \in \{\otimes, Cut\}$:

$$\dfrac{\dfrac{\vdots}{\vdash \Gamma, A} \quad \dfrac{\vdots}{\vdash \Sigma, B, C}\,\odot_1}{\vdash \Gamma, \Sigma, (A \odot_1 B), C} \quad \dfrac{\vdots}{\vdash \Delta, D}\,\odot_2}{\vdash \Gamma, \Sigma, \Delta, (A \odot_1 B), (C \odot_2 D)} \quad \sim \quad \dfrac{\dfrac{\vdots}{\vdash \Gamma, A} \quad \dfrac{\dfrac{\vdots}{\vdash \Sigma, B, C} \quad \dfrac{\vdots}{\vdash \Delta, D}\,\odot_2}{\vdash \Sigma, \Delta, B, (C \odot_2 D)}\,\odot_1}{\vdash \Gamma, \Sigma, \Delta, (A \odot_1 B), (C \odot_2 D)}$$

- If $\odot_1 \in \{\otimes, Cut\}$, $\odot_2 \in \{\parr, \perp, ?\mathbf{W}, ?\mathbf{C}, ?\mathbf{D}, !P\}$:

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, A} \quad \dfrac{\vdots}{\vdash B, \Delta}\,\odot_1}{\vdash \Gamma, (A \odot_1 B), \Delta}\,\odot_2}{\vdash \Gamma, (A \odot_1 B), \odot_2(\Delta)} \quad \sim \quad \dfrac{\dfrac{\vdots}{\vdash \Gamma, A} \quad \dfrac{\vdots}{\vdash B, \Delta}\,\odot_2}{\vdash \Gamma, B, \odot_2(\Delta)}\,\odot_1}{\vdash \Gamma, (A \odot_1 B), \odot_2(\Delta)}$$

where

$$\odot_2(\Delta) = \begin{cases} \Delta, \bot & \text{if } \odot_2 = \bot \\ \Delta', A \parr B & \text{if } \odot_2 = \parr \text{ and } \Delta = \Delta', A, B \\ \Delta, ?A & \text{if } \odot_2 = ?\mathbf{W} \\ \Delta', ?A & \text{if } \odot_1 = ?\mathbf{C} \text{ and } \Delta = \Delta', ?A, ?A \\ \Delta', ?A & \text{if } \odot_2 = ?\mathbf{D} \text{ and } \Delta = \Delta', A \\ ?\Delta', !A & \text{if } \odot_2 = ?\mathbf{D} \text{ and } \Delta = ?\Delta', A \end{cases}.$$

- If $\odot_1, \odot_2 \in \{\parr, \bot, ?\mathbf{W}, ?\mathbf{C}, ?\mathbf{D}\}$:

$$\dfrac{\dfrac{\vdots}{\dfrac{\vdash \Gamma, \Delta}{\vdash \odot_1(\Gamma), \Delta}\odot_1}}{\vdash \odot_1(\Gamma), \odot_2(\Delta)}\odot_2 \quad \sim \quad \dfrac{\dfrac{\vdots}{\dfrac{\vdash \Gamma, \Delta}{\vdash \Gamma, \odot_2(\Delta)}\odot_2}}{\vdash \odot_1(\Gamma), \odot_2(\Delta)}\odot_1$$

where

$$\odot_1(\Gamma) = \begin{cases} \Gamma, \bot & \text{if } \odot_1 = \bot \\ \Gamma', A \parr B & \text{if } \odot_1 = \parr \text{ and } \Gamma = \Gamma', A, B \\ \Gamma, ?A & \text{if } \odot_1 = ?\mathbf{W} \\ \Gamma', ?A & \text{if } \odot_1 = ?\mathbf{C} \text{ and } \Gamma = \Gamma', ?A, ?A \\ \Gamma', ?A & \text{if } \odot_1 = ?\mathbf{D} \text{ and } \Gamma = \Gamma', A \\ ?\Gamma', !A & \text{if } \odot_2 = ?\mathbf{D} \text{ and } \Gamma = ?\Gamma', A \end{cases}.$$

and

$$\odot_2(\Delta) = \begin{cases} \Delta, \bot & \text{if } \odot_2 = \bot \\ \Delta', A \parr B & \text{if } \odot_2 = \parr \text{ and } \Delta = \Delta', A, B \\ \Delta, ?A & \text{if } \odot_2 = ?\mathbf{W} \\ \Delta', ?A & \text{if } \odot_1 = ?\mathbf{C} \text{ and } \Delta = \Delta', ?A, ?A \\ \Delta', ?A & \text{if } \odot_2 = ?\mathbf{D} \text{ and } \Delta = \Delta', A \\ ?\Delta', !A & \text{if } \odot_2 = ?\mathbf{D} \text{ and } \Delta = ?\Delta', A \end{cases}.$$

- If $\odot = ?\mathbf{W}$ or $?\mathbf{C}$:

$$\dfrac{\dfrac{\vdots}{\dfrac{\vdash ?\Gamma, A}{\vdash \odot(?\Gamma), A}\odot}}{\vdash \odot(?\Gamma), !A}\,!P \quad \sim \quad \dfrac{\dfrac{\vdots}{\dfrac{\vdash ?\Gamma, A}{\vdash ?\Gamma, !A}\,!P}}{\vdash \odot(?\Gamma), !A}\odot_1$$

where

$$\odot(?\Gamma) = \begin{cases} ?\Gamma, ?A & \text{if } \odot = ?\mathbf{W} \\ ?\Gamma', ?A & \text{if } \odot = ?\mathbf{C} \text{ and } \Gamma = \Gamma', ?A, ?A \end{cases}.$$

- The monoidal structure concerning $?\mathbf{C}$ and $?\mathbf{W}$:

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, ?A, ?A, ?A}}{\vdash \Gamma, ?\mathbf{C}(?A, ?A), ?A}\,?\mathbf{C}}{\vdash \Gamma, ?A}\,?\mathbf{C} \quad \sim \quad \dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, ?A, ?A, ?A}}{\vdash \Gamma, ?A, ?\mathbf{C}(?A, ?A)}\,?\mathbf{C}}{\vdash \Gamma, ?A}\,?\mathbf{C}$$

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, ?A}}{\vdash \Gamma, ?A, ?A}\,?\mathbf{W}}{\vdash \Gamma, ?A}\,?\mathbf{C} \quad \sim \quad \dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, ?A}}{\vdash \Gamma, ?A, ?A}\,?\mathbf{W}}{\vdash \Gamma, ?A}\,?\mathbf{C}$$

We note that, in addition to the omission of the exchange rules, we consider that permutation over derivation tree's branching has no effect and so, we have to consider all the symmetric variants of the above relations. For instance, for the first relation we need to consider as well the following variant (which is the one discussed in Section 3.4):

$$\dfrac{\dfrac{\vdots \quad \vdots}{\dfrac{\vdash \Gamma, A, B \qquad \vdash \Gamma', C}{\vdash \Gamma, \Gamma', (B \odot_1 C), A}\,\odot_1 \qquad \vdots}{\vdash \Gamma, \Gamma', \Gamma'', (A \odot_1 D), (B \odot_2 C)}\,\odot_2 \quad \sim \quad \dfrac{\dfrac{\vdots \quad \vdots}{\dfrac{\vdash \Gamma, A, B \qquad \vdash \Gamma'', D}{\vdash \Gamma, \Gamma', (A \odot_1 D), B}\,\odot_2 \qquad \vdots}{\vdash \Gamma, \Gamma', \Gamma'', (A \odot_1 D), (B \odot_2 C)}\,\odot_1$$

Proof nets

Proof nets were introduced by J.-Y.Girard in [29] in order to represents proof of multiplicative linear logic [28]. Any proof net represents an equivalence class of logical derivations. In order to give proof nets graphical representation we need to define a larger class of diagrams called *proof structurs*.

Definition A.0.4 (MLL proof structure)**.** A MLL *proof structure* is a graph N with labeling on vertices and edges satisfying the following rules:

- if a vertex v is labeled by Ax, so $deg(v) = 2$ and the two edges are labeled by X and $X^{\perp}$;
- if a vertex v is labeled by $\otimes$, so $deg(v) = 3$ and the three edges are labeled by X and Y and $X \otimes Y$;

- if a vertex v is labeled by $\parr$, so $deg(v) = 3$ and the three edges are labeled by X and Y and $X \parr Y$;
- if a vertex v is labeled by Cut so $deg(v) = 2$, the two edges are labeled by X and $X^{\perp}$;
- if a vertex v is labeled by X so it is a leaf (this is $deg(v) = 1$) and the edge is labeled by X;

where X, Y are MLL formuas.

Interaction nets [50] are born as generalization of Girard's linear logic's proof structures. They are given by graph labeled by a set $\mathcal{S}$ of *symbols* each one with an *arity* $n \geq 0$.

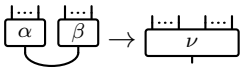
Definition A.0.5 (Interaction net)**.** An *interaction net* is given by:

- a finite set of free ports X;
- a finite set of cells C;
- a label $l(c)$ for each $c \in C$ (which defines the number of its active and non-active ports);
- a finite set of wires W;
- a set $\partial(w)$ of 0 or 2 ports for each $w \in W$.

where a *port* is either an element of X or a pair (c, i) with $c \in C$ and i ranking from 0 to the arity of $\ell(c)$. A *wiring* is a net without cells and cyclic wires.

The basic intuition is that an interaction net is a graphs with labeled vertices of degree ≥ 0: some vertices (called cells) are labeled by a set of simbols and they have always a *principal port* and n (equal to the arity of the symbol in labeling) auxiliary ports, while free ports are always leaves. The edges are called *wires* and connects the two ports (free ports, auxiliary ports and principal ports) of $\partial(w) \neq \emptyset$. Moreover we assume the existence of wires (when $\partial(w) = \emptyset$) connectin no ports called *cyclic wires*.

If α is a symbol of arity n, an α-cell can be viewed as a net with $n+1$ free ports. In general every subset of $N' \subseteq C \cup X$ can be seen as a net $\langle N' \rangle$: it will be the graph vith vertex $N' \cup X'$ where X' is a set of free ports in one-to-one correspondence with the set of ports $\delta(N) = \{v \in C \cup X \setminus N' \text{ such that } \exists w \in W, \partial(w) \cap N' \neq \emptyset \text{ and } v \in \partial(w)\}$. Fixed a finite alphabet $\mathcal{S}$ of symobls $\alpha_1, \ldots, \alpha_m$ with respective arities $n_1, \ldots, n_m$ an *interaction rule* is a a graph rewriting rules in the form



with $\alpha, \beta \in \mathcal{S}$, α and β have respectively n_i and n_j free ports and ν is a net with $n_i + n_j$ free ports.

Interaction rules are performed on a net substituting its subnets N' of two cells such that $\langle N' \rangle$ is a premise of an interaction rule.

Definition A.0.6. An *interaction system* is a set of interaction rules which can be applied without any ambiguity: if a rule is given for α_i, α_j, than no other one is given for α_i, α_j (or for α_j, α_i).

In particular an interaction system over an alphabet of m symbols is necessarily finite with at most $\frac{m(m+1)}{2}$ rules.

Proposition A.0.7 (Local confluence)**.** *If a net μ reduces in one step to ν and ν' with $\nu \neq \nu'$, then ν and ν' reduce in one step to a common net ξ.*

Proof. Since interaction rules are applied only on cells connected through their principal ports, the two rules act on two disjoint parts of the net so they can be applied independently. □

Definition A.0.8 ($MELLc$ proof structure)**.** A $MLLc$ proof structure is an interaction net over the alphabet $\{Ax, \otimes, \parr, Cut, \bot, 1, \bot\}$, with respectively 2, 3, 3, 2, 1 and 1 ports.

Here we do not recall boxes definition for $MELLc$ proof nets, since many variant of them are present in literature (for examples see [72], [22], [64]). In order to give an intuition of these structure, we define the additional symbols ?**W**, ?**C**, ?**D** and !**P** (with respectively 1, 3, 2 and 2 ports) in the alphabet and we associate to any cell of type !**P** its *box*, this is a subset of net's cells.

Definition A.0.9 (Proof net)**.** A *proof net* is a correct proof structure.

Following Girard's original correction criterion for MLL proof nets [28] and its extention to $MLLc$ [29], Danos and Regnier [17], have introduced a method which ensures correctness by means of graph acyclicity after *swichings* assignation on $\otimes$ cells. Then, Guerrini reformulated the correction criterion by means of graph contractability [31]. For correctness in $MELL$ we remind Tortora de Falco thesis as well [84].

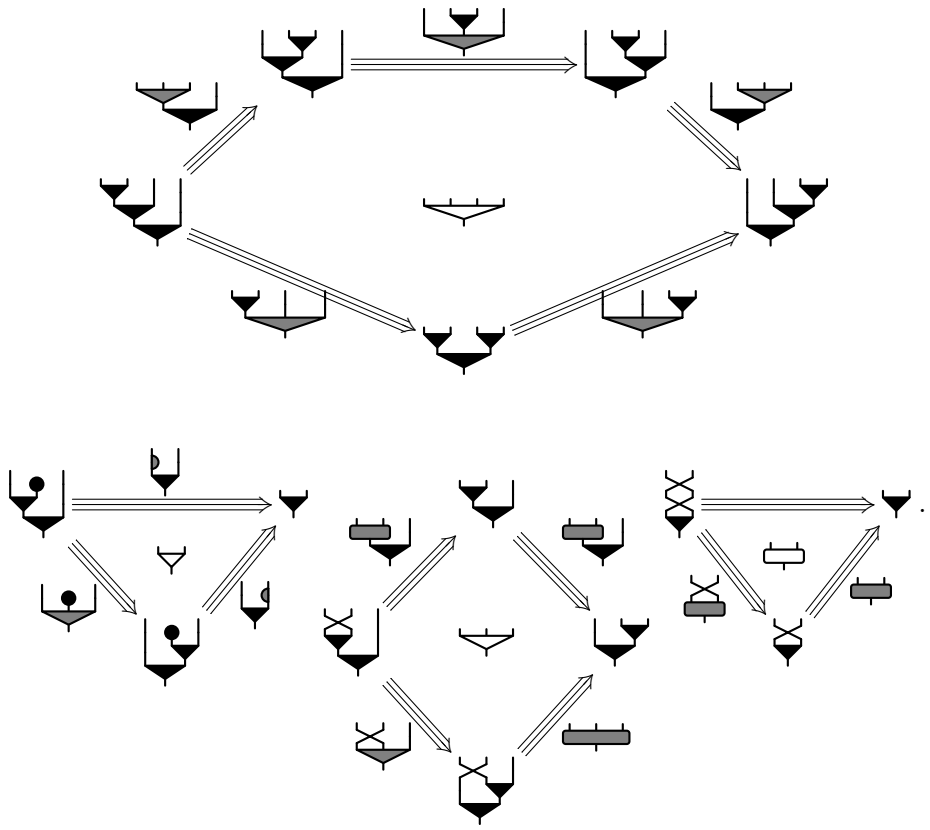
The main idea for correctness in MLL is to give some operation over the graph of a proof net in order to verify that binary rules (Cut and $\otimes$) are well-typed, .e. that is their premises come from two different branching of the associate derivation tree. Following the same idea, the *jumps* assignations in Girard's correctness criterion aim to verify if a $\bot$ cell is attached to a correct branch of the derivation tree. Therefor, boxes in exponentials isolate a sub-net of a proof net in order to verify the correct application of a promotion rule, i.e. if the sequent in the context is of of type ?Γ.

Appendix B

Refinement of Kelly theorem for SMC

In this section we give the complete construction of the 4-cells corresponding to the *Kelly* and *weak-Kelly* critical peaks of the rewriting system $\mathfrak{F}$.

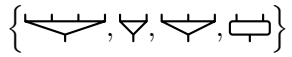
In this construction, we assume the following 4-cells, corresponding to symmetric monoidal categories axioms, to be defined:



Moreover assume to be defined also the 4-cells which borders a solution of a trivial or strongly trivial critical peak. We denote all these 4-cells with the same symbol $\odot$. This choice is justified by the fact that the commutative diagrams corresponding to their border are interpreted by trivial diagrams in the categorical interpretation of the rewriting system.

Every time we build a new 4-cell for a solution of a critical peak ϕ we denote it by $\boxed{\boxed{\phi}}$.

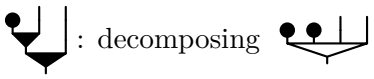
Proposition B.0.1 (Refinement of Kelly's lemma for symmetric monodal category)**.**
For all Kelly-peaks *and* weak-Kelly-peaks*, a* 4*-cell can be defined from set of* 4*-cells*

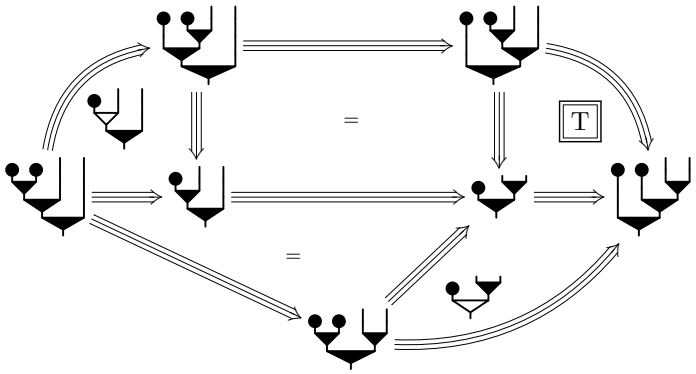


plus the set of 4*-cells with border a solution of a trivial or strongly-trivial critical peak (we note them with the symbol* $\odot$*).*

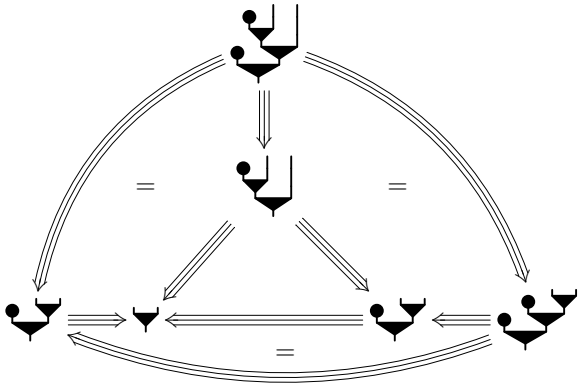
Proof. We now give a list of possible ways to define the 4-cells solution of these critical peaks. These 4-cells are implicitly defined by given a decomposition of another 4-cell.

In order to reduce the size of pictures, we note $\boxed{\boxed{\phi}}$ the 4-cell which border the given solution of a critical peak represented by its source ϕ.

-  : decomposing

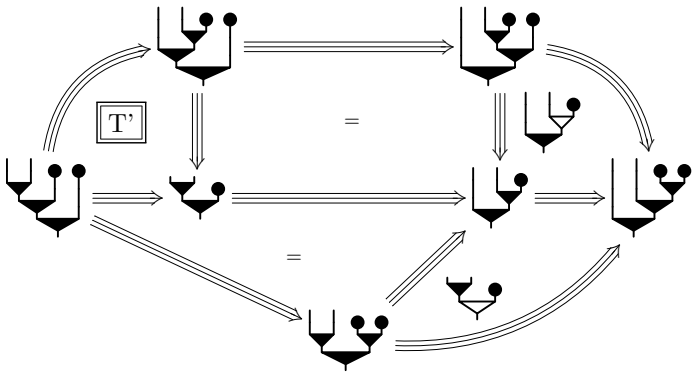


where $\boxed{\boxed{\mathrm{T}}}$ is

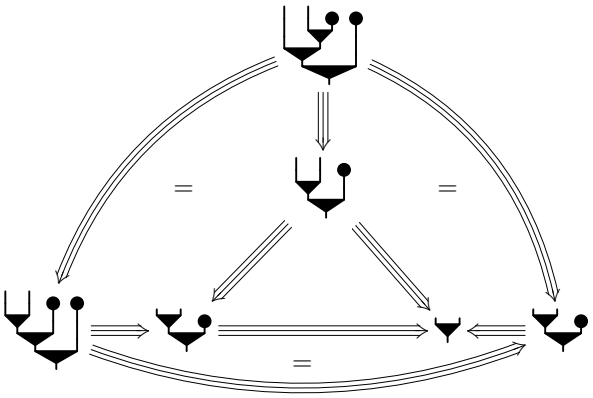


;

-  : decomposing 

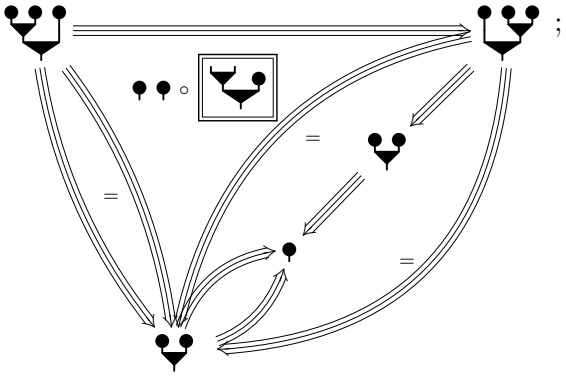


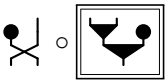
where  is

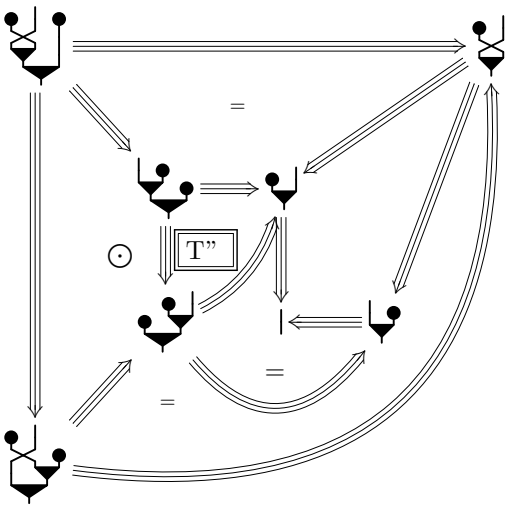


;

•  : decomposing

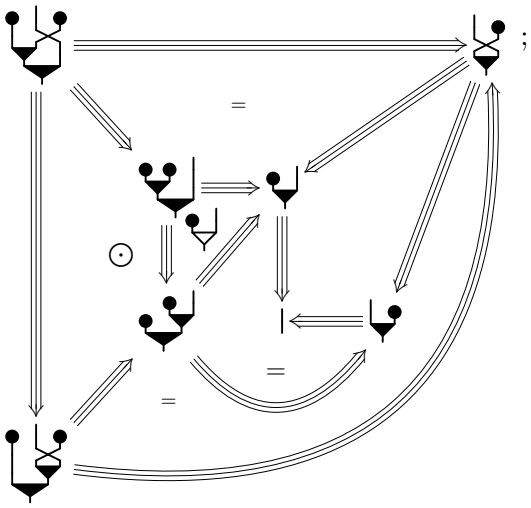


•  : decomposing 

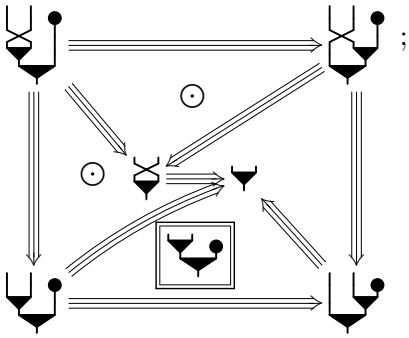


where  = 

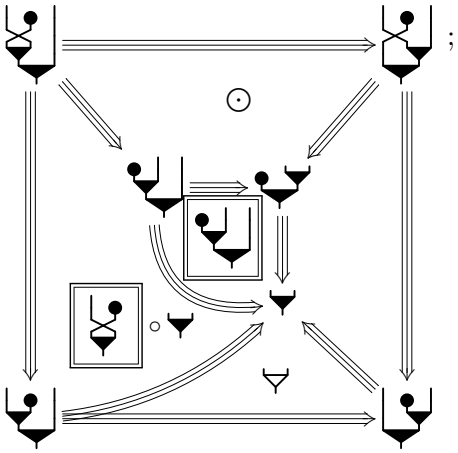
•  : decomposing  ∘ 



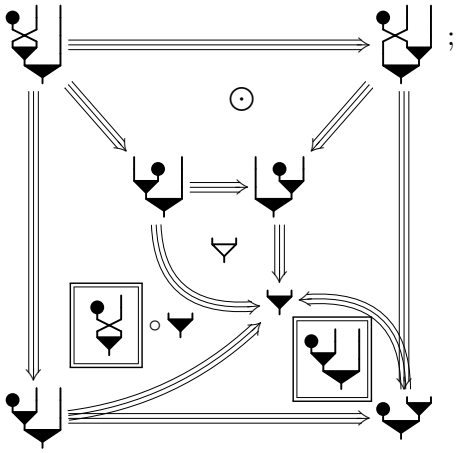
•  : decomposing 



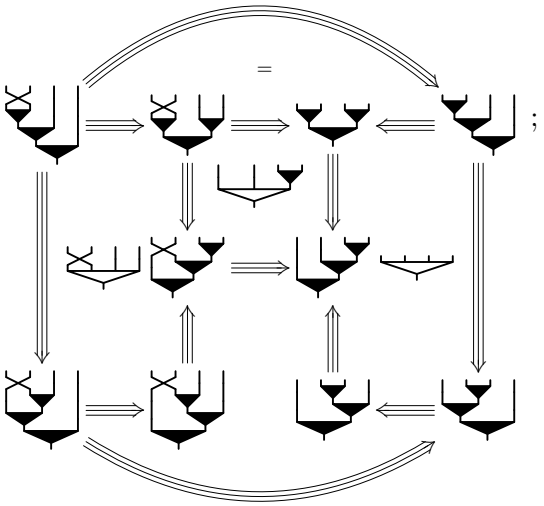
-  : decomposing 



-  : decomposing 

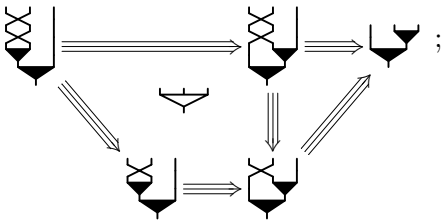


-  : decomposing 

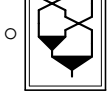


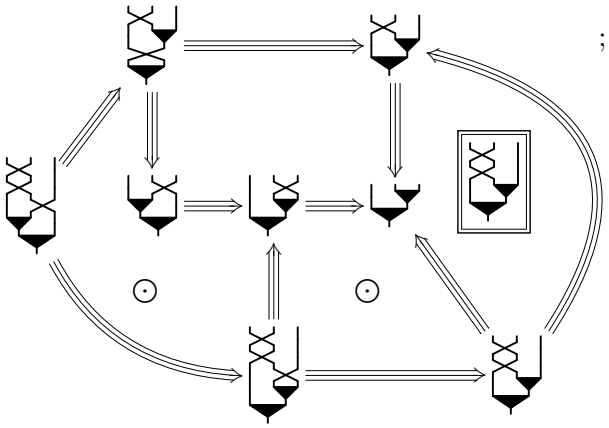
;

-  : decomposing 



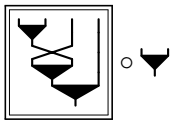
;

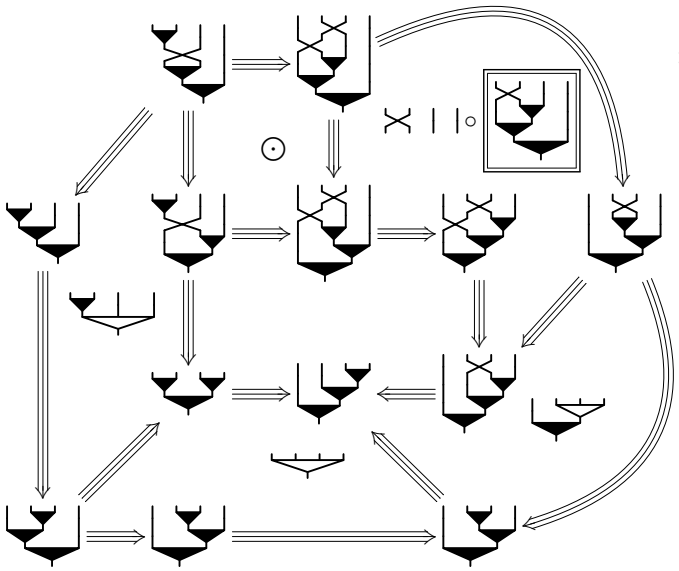
-  : decomposing 

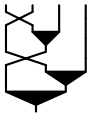
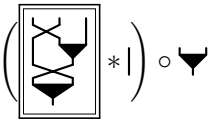


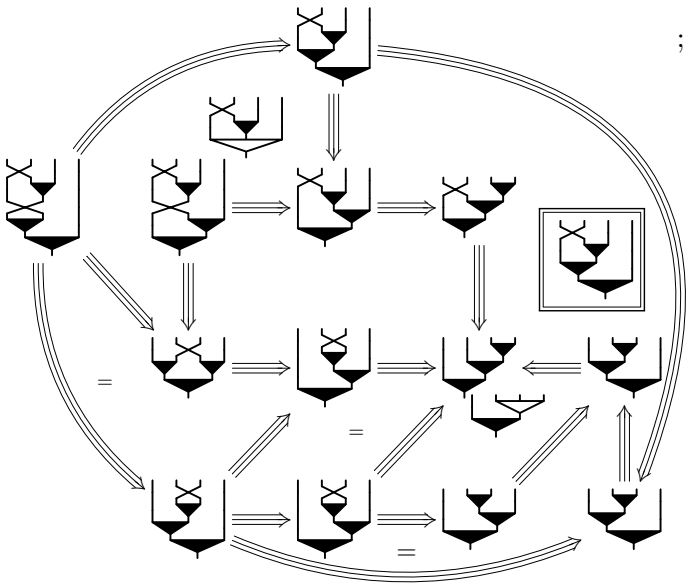
;

•

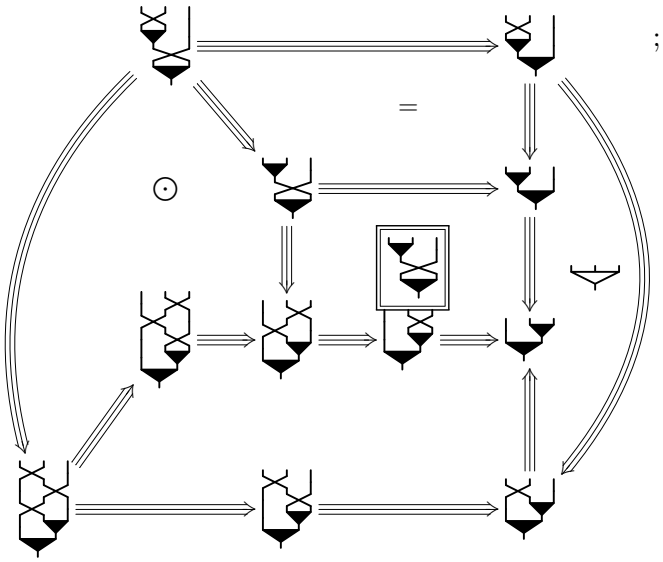
: decomposing



;

•

: decomposing


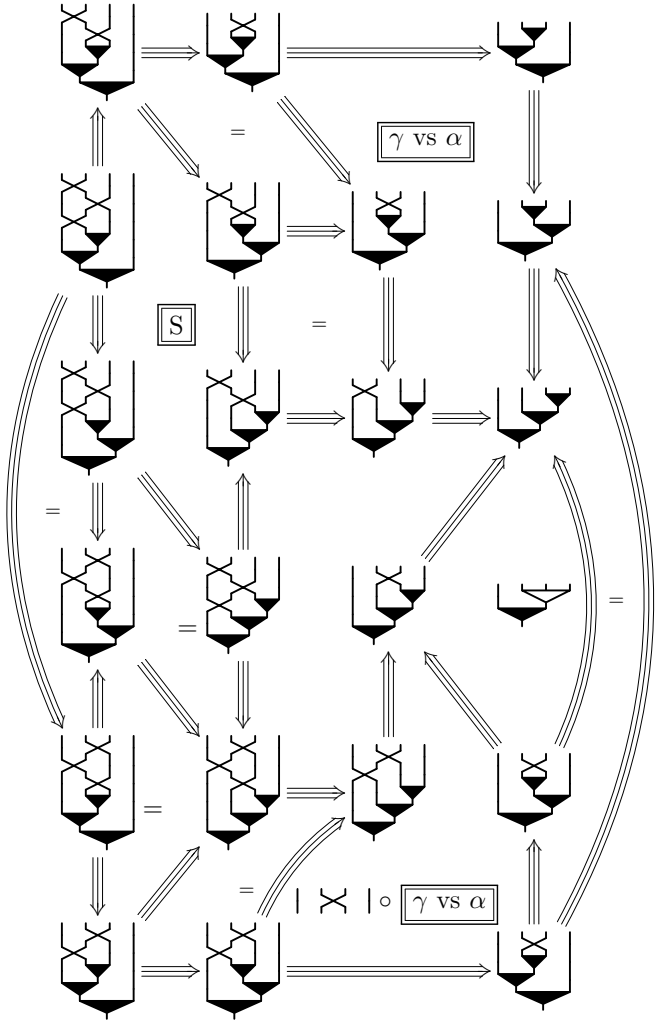

;

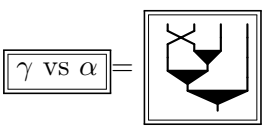
•  : decomposing



•

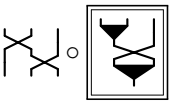
: decomposing

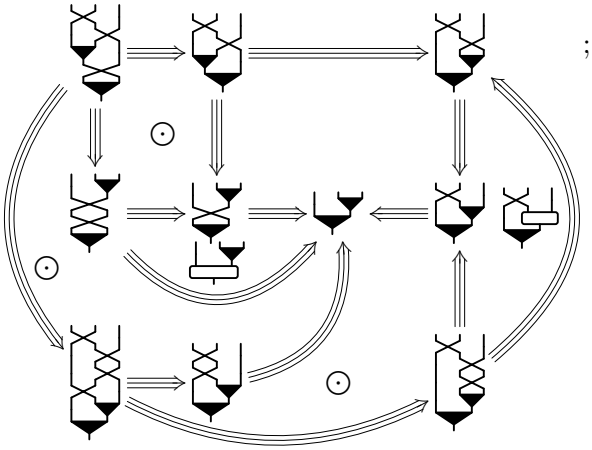


where


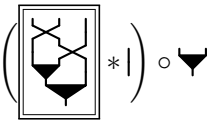
and

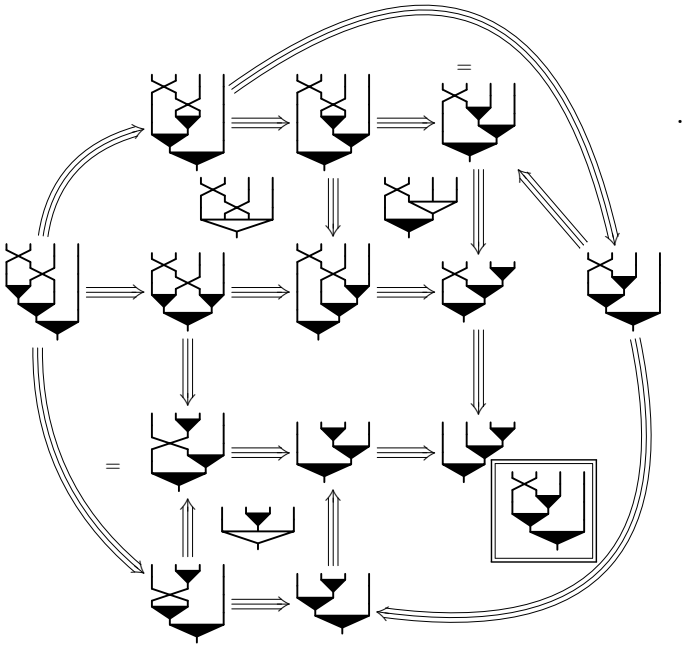
;

-  : decomposing 



;

-  : decomposing 



.

□

Appendix C

On factors and confluences

In this section we present some rewriting systems in order to give some examples of rewriting systems with factors, irreducible factors and their relative homotopical properties.

We recall the definition of the following polygraphs:

- The polygraph of permutations:

$$\mathfrak{S} = \left\{ \begin{array}{c} \mathfrak{S}_0 = \{\square\} \\ \mathfrak{S}_1 = \{|\} \\ \mathfrak{S}_2 = \{\times\} \\ \mathfrak{S}_3 = \left\{ \times\times \Longrightarrow || \,,\ \times| \Longrightarrow |\times \right\} \end{array} \right\}.$$

  $\mathfrak{S}$ has no 0-factor, it has 1-factor but no irreducible 1-factor and it has irreducible 2-factor and 3-factor. The polygraph is confluent with a finite number of critical pairs;

- The polygraph of permutations with restrictions:

$$\mathfrak{S}^\bullet = \left\{ \begin{array}{c} \mathfrak{S}^\bullet_0 = \{\square\} \\ \mathfrak{S}^\bullet_1 = \{|\} \\ \mathfrak{S}^\bullet_2 = \{\times, \bullet\} \\ \mathfrak{S}^\bullet_3 = \left\{ \times\times \Longrightarrow || \,,\ \times| \Longrightarrow |\times \right\} \end{array} \right\}.$$

  $\mathfrak{S}^\bullet S$ has irreducible 0-factor, 1-factor and 2-factor and 3-factor. The polygraph is not confluent and has an infinite number of critical pairs;

- The polygraph of monoidal categories:

$$\mathfrak{M} = \left\{ \begin{array}{c} \mathfrak{M}_0 = \{\square\} \\ \mathfrak{M}_1 = \{|\} \\ \mathfrak{M}_2 = \{\curlyvee, \bullet\} \\ \mathfrak{M}_3 = \left\{ \curlyvee| \Longrightarrow |\curlyvee \,,\ \bullet| \Longrightarrow | \,,\ |\bullet \Longrightarrow | \right\} \end{array} \right\}.$$

$\mathfrak{M}$ has no 0-factor. It has 1-factor but no irreducible 1-factor. It has no 2-factor. The polygraph is confluent with a finite number of critical pairs.

- The polygraph of symmetric monoidal categories

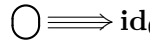
$$\mathfrak{F} = \left\{ \begin{array}{c} \mathfrak{F}_0 = \{\Box\} \\ \mathfrak{F}_1 = \{|\} \\ \mathfrak{F}_2 = \{\curlyvee, \bullet, \times\} \\ \mathfrak{F}_3 = \left\{ \begin{array}{c} \curlyvee \bullet \Longrightarrow \curlyvee\curlyvee, \quad \bullet \bullet \Longrightarrow |, \quad \curlyvee \bullet \Longrightarrow |, \\ \times\times \Longrightarrow || , \quad \times\times\times \Longrightarrow \times\times\times \\ \bullet \times \Longrightarrow | \bullet, \quad \times \bullet \Longrightarrow \bullet | \\ \curlyvee \times \Longrightarrow \times \curlyvee, \quad \times\curlyvee \Longrightarrow \curlyvee\curlyvee, \\ \times\curlyvee \Longrightarrow \curlyvee\times \\ \times\bullet \Longrightarrow \curlyvee, \quad \times\curlyvee \Longrightarrow \curlyvee\curlyvee, \end{array} \right\} \end{array} \right\}.$$

$\mathfrak{F}$ has no 0-factor. It has 1-factor but no irreducible 1-factor. It has irreducible 2-factor and 3-factor. The polygraph is confluent with a finite number of critical pairs.

- The polygraph of paths with loops:

$$\mathfrak{C}^\ell = \left\{ \begin{array}{c} \mathfrak{C}^\ell_0 = \{\Box\} \\ \mathfrak{C}^\ell_1 = \{|\} \\ \mathfrak{C}^\ell_2 = \{\times, \cap, \cup\} \\ \mathfrak{C}^\ell_3 = \left\{ \begin{array}{c} \cap\cup \Longrightarrow |, \quad \cup\cap \Longrightarrow |, \\ \times\times \Longrightarrow || , \quad \times\times\times \Longrightarrow \times\times\times, \\ \times \Longrightarrow \cap, \quad \times \Longrightarrow \cup, \quad \times\cup \Longrightarrow \cup\times, \\ \times\cap \Longrightarrow | \cap, \quad \times\cap \Longrightarrow \cap |, \\ \times\cup \Longrightarrow | \cup, \quad \times\cup \Longrightarrow \cup |, \end{array} \right\}; \end{array} \right\}.$$

$\mathfrak{C}^\ell$ has irreductible 0-factor, 1-factor but not irreducible 1-factor. It has irreducible 2-factor and 3-factor. The polygraph is not confluent and has an infinite number of critical pairs.

- The polygraph of paths without loops: we add to the polygraph $\mathfrak{C}^\ell$ the following 3-cell:  $\mathbf{id}_0$.
  Here, $\mathfrak{C}$ has 0-factor and 1-factor but not irreducible 0-factor nor 1-factor. It has irreducible 2-factor and 3-factor. The polygraph is confluent with a finite number of critical pairs;

- The polygraph used by Yves Guiraud and Philippe Malbos in [35] in order to show a finite polygraph with infinite derivation type:

$$\mathfrak{G} = \left\{ \begin{array}{c} \mathfrak{G}_0 = \{\square\} \\ \mathfrak{G}_1 = \{|\} \\ \mathfrak{G}_2 = \{\bullet, \cap, \cup\} \\ \mathfrak{G}_3 = \left\{ \begin{array}{c} \bullet\cap \Longrightarrow \cap\bullet\,,\ \bullet\cup \Longrightarrow \cup\bullet\,, \\ \cap\cup \Longrightarrow |\,,\ \cup\cap \Longrightarrow | \end{array} \right\} \end{array} \right\}.$$

  $\mathfrak{G}$ has irreducible 0-factor and 1-factor. It has irreducible 2-factor and 3-factor. The polygraph is confluent with an infinite number of critical pairs.

Bibliography

[1] Stål Aanderaa and Daniel E Cohen. Modular machines, the word problem for finitely presented groups and collins' theorem. *Studies in Logic and the Foundations of Mathematics*, 95:1–16, 1980.

[2] Matteo Acclavio. Comparing various proofs of the novikov-boone theorem based on rewriting, 2012.

[3] Matteo Acclavio. A complete proof of coherence for symmetric monoidal categories using rewriting. *arXiv preprint arXiv:1606.01722*, 2016.

[4] Matteo Acclavio. Proof diagrams for multiplicative linear logic. *arXiv preprint arXiv:1606.09016*, 2016.

[5] Jean Andreoli. *Proposition pour une synthese des paradigmes de la programmation logique et de la programmation par objets*. PhD thesis, Paris 6, 1990.

[6] John C Baez and Aaron Lauda. A prehistory of n-categorical physics. *Deep Beauty: Understanding the Quantum World Through Mathematical Innovation*, pages 13–128, 2009.

[7] Michael Barr. *-autonomous categories. In *Lecture Notes in Mathematics 752*. Citeseer, 1979.

[8] Ilya Beylin and Peter Dybjer. Extracting a proof of coherence for monoidal categories from a proof of normalization for monoids. In *International Workshop on Types for Proofs and Programs*, pages 47–61. Springer, 1995.

[9] LA Bokut' et al. *Algorithmic and combinatorial algebra*, volume 255. Springer Science & Business Media, 2012.

[10] William W Boone. The word problem. *Annals of mathematics*, pages 207–265, 1959.

[11] Roberto Bruni, Fabio Gadducci, and Ugo Montanari. Normal forms for algebras of connections. *Theoretical Computer Science*, 286(2):247–292, 2002.

[12] Albert Burroni. Higher-dimensional word problems with applications to equational logic. *Theoretical computer science*, 115(1):43–62, 1993.

[13] Bruce Chandler and Wilhelm Magnus. The history of combinatorial group theory, volume 9 of studies in the history of mathematics and physical sciences, 1982.

[14] Kaustuv Chaudhuri, Dale Miller, and Alexis Saurin. Canonical sequent proofs via multi-focusing. In *Fifth Ifip International Conference On Theoretical Computer Science–Tcs 2008*, pages 383–396. Springer, 2008.

[15] Noam Chomsky. On certain formal properties of grammars. *Information and control*, 2(2):137–167, 1959.

[16] Alonzo Church. A note on the entscheidungsproblem. *The journal of symbolic logic*, 1(01):40–41, 1936.

[17] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical logic*, 28(3):181–203, 1989.

[18] Martin Davis. *Computability & unsolvability*. Courier Corporation, 1958.

[19] Martin Davis. *The undecidable: Basic papers on undecidable propositions, unsolvable problems and computable functions*. Courier Corporation, 2004.

[20] Martin Davis, Ron Sigal, and Elaine J Weyuker. *Computability, complexity, and languages: fundamentals of theoretical computer science*. Newnes, 1994.

[21] Martin Davis, Ron Sigal, and Elaine J Weyuker. *Computability, complexity, and languages: fundamentals of theoretical computer science*. Newnes, 1994.

[22] Roberto Di Cosmo, Delia Kesner, and Emmanuel Polonovski. Proof nets and explicit substitutions. In *International Conference on Foundations of Software Science and Computation Structures*, pages 63–81. Springer, 2000.

[23] Samuel Eilenberg and Saunders MacLane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, 58(2):231–294, 1945.

[24] Samuel Eilenberg and Saunders MacLane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, 58(2):231–294, 1945.

[25] Richard Phillips Feynman. Space-time approach to non-relativistic quantum mechanics. *Reviews of Modern Physics*, 20(2):367, 1948.

[26] Peter Freyd and David N Yetter. Coherence theorems via knot theory. *Journal of Pure and Applied Algebra*, 78(1):49–76, 1992.

[27] Peter J Freyd and David N Yetter. Braided compact closed categories with applications to low dimensional topology. *Advances in Mathematics*, 77(2):156–182, 1989.

[28] Jean-Yves Girard. Linear logic. *Theoretical computer science*, 50(1):1–101, 1987.

[29] Jean-Yves Girard. Proof-nets: the parallel syntax for proof-theory. *Lecture Notes in Pure and Applied Mathematics*, pages 97–124, 1996.

[30] David Griffiths. Introduction to elementary particles. *John Willey & Sons Inc*, 1987.

[31] Stefano Guerrini and Andrea Masini. Parsing mell proof nets. *Theoretical Computer Science*, 254(1):317–335, 2001.

[32] Yves Guiraud. *Présentations d'opérades et systèmes de réécriture*. PhD thesis, Université Montpellier II-Sciences et Techniques du Languedoc, 2004.

[33] Yves Guiraud. Termination orders for three-dimensional rewriting. *Journal of Pure and Applied Algebra*, 207(2):341–371, 2006.

[34] Yves Guiraud. Catex latex patch. https://www.irif.fr/ guiraud/catex/s, 2007.

[35] Yves Guiraud and Philippe Malbos. Higher-dimensional categories with finite derivation type. *Theory and Applications of Categories*, 22(18):420–478, 2009.

[36] Yves Guiraud and Philippe Malbos. Coherence in monoidal track categories. *Mathematical Structures in Computer Science*, 22(06):931–969, 2012.

[37] Yves Guiraud and Philippe Malbos. Higher-dimensional normalisation strategies for acyclicity. *Advances in Mathematics*, 231(3):2294–2351, 2012.

[38] Yves Guiraud and Philippe Malbos. Polygraphs of finite derivation type. *arXiv preprint arXiv:1402.2587*, 2014.

[39] Yves Guiraud, Philippe Malbos, and Samuel Mimram. A homotopical completion procedure with applications to coherence of monoids. In *RTA-24th International Conference on Rewriting Techniques and Applications-2013*, volume 21, pages 223–238. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013.

[40] Willem Heijltjes and Robin Houston. No proof nets for mll with units: Proof equivalence in mll is pspace-complete. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, page 50. ACM, 2014.

[41] Gérard Huet. Initiation à la théorie des catégories, 1985.

[42] Dominic JD Hughes. Simple free star-autonomous categories and full coherence. *Journal of Pure and Applied Algebra*, 216(11):2386–2410, 2012.

[43] Michio Jimbo. Introduction to the yang-baxter equation. *International Journal of Modern Physics A*, 4(15):3759–3777, 1989.

[44] André Joyal and Ross Street. The geometry of tensor calculus, i. *Advances in Mathematics*, 88(1):55–112, 1991.

[45] André Joyal and Ross Street. The geometry of tensor calculus ii. *Draft available at http://www. math. mq. edu. au/˜ street/GTCII. pdf*, 585, 1991.

[46] Deepak Kapur and Paliath Narendran. The knuth-bendix completion procedure and thue systems. *SIAM journal on computing*, 14(4):1052–1072, 1985.

[47] Gregory M Kelly and Miguel L Laplaza. Coherence for compact closed categories. *Journal of Pure and Applied Algebra*, 19:193–213, 1980.

[48] Gregory Maxwell Kelly. On maclane's conditions for coherence of natural associativities, commutativities, etc. *Journal of Algebra*, 1(4):397–402, 1964.

[49] Yves Lafont. Equational reasoning with 2-dimensional diagrams. In *Term Rewriting*, pages 170–195. Springer, 1995.

[50] Yves Lafont. From proof nets to interaction nets. *London Mathematical Society Lecture Note Series*, pages 225–248, 1995.

[51] Yves Lafont. A new finiteness condition for monoids presented by complete rewriting systems (after craig c. squier). *Journal of Pure and Applied Algebra*, 98(3):229–244, 1995.

[52] Yves Lafont. Towards an algebraic theory of boolean circuits. *Journal of Pure and Applied Algebra*, 184(2):257–310, 2003.

[53] Yves Lafont. Algebra and geometry of rewriting. *Applied Categorical Structures*, 15(4):415–437, 2007.

[54] Yves Lafont. Réécriture et problème du mot. *Gazette des mathématiciens*, 120:27–38, 2009.

[55] Yves Lafont and Alain Prouté. Church-rooser property and homology of monoids. *Mathematical structures in computer science*, 1(03):297–326, 1991.

[56] Olivier Laurent, Myriam Quatrini, and Lorenzo Tortora De Falco. Polarized and focalized linear and classical proofs. *Annals of Pure and Applied Logic*, 134(2):217–264, 2005.

[57] F William Lawvere. Functorial semantics of algebraic theories. *Proceedings of the National Academy of Sciences*, 50(5):869–872, 1963.

[58] Tom Leinster. *Higher operads, higher categories*, volume 298. Cambridge University Press, 2004.

[59] Patrick Lincoln and Timothy Winkler. Constant-only multiplicative linear logic is np-complete. *Theoretical computer science*, 135(1):155–169, 1994.

[60] Maxime Lucas. A coherence theorem for pseudonatural transformations. *arXiv preprint arXiv:1508.07807*, 2015.

[61] Saunders Mac Lane. *Categories for the working mathematician*, volume 5. Springer Science & Business Media, 2013.

[62] Saunders MacLane. Natural associativity and commutativity. *Rice Institute Pamphlet-Rice University Studies*, 49(4):28–46, 1963.

[63] Andrei A. Markov. On the impossibility of certain algorithm in the theory of associative systems. pages 587–590, 1947.

[64] Damiano Mazza. Multiport interaction nets and concurrency. In *International Conference on Concurrency Theory*, pages 21–35. Springer, 2005.

[65] Jagdish Mehra and Helge Kragh. The beat of a different drum: The life and science of richard feynman. *American Journal of Physics*, 62(12):1155–1155, 1994.

[66] Paul-André Mellies. Mac lane's coherence theorem expressed as a word problem. 2003.

[67] Paul-André Mellies. Functorial boxes in string diagrams. In *International Workshop on Computer Science Logic*, pages 1–30. Springer, 2006.

[68] Paul-André Mellies. Categorical semantics of linear logic. *Interactive Models of Computation and Program Behaviour, Panoramas et syntheses*, 27:15–215, 2009.

[69] Samuel Mimram. Towards 3-dimensional rewriting theory. *arXiv preprint arXiv:1403.4094*, 2013.

[70] Maxwell Herman Alexander Newman. On theories with a combinatorial definition of" equivalence". *Annals of mathematics*, pages 223–243, 1942.

[71] P.S. Novikov. On algorithmic undecidability of the word problem. *Dokl. Akad. Nauk SSSR*, (4):485–524, 1952.

[72] Michele Pagani and Lorenzo Tortora de Falco. Strong normalization property for second order linear logic. *Theoretical Computer Science*, 411(2):410–444, 2010.

[73] Roger Penrose. Applications of negative dimensional tensors. *Combinatorial mathematics and its applications*, 221244, 1971.

[74] Emil L Post. Formal reductions of the general combinatorial decision problem. *American journal of mathematics*, 65(2):197–215, 1943.

[75] Emil L Post. Recursive unsolvability of a problem of thue. *The Journal of Symbolic Logic*, 12(01):1–11, 1947.

[76] Pierre Rannou. *Réécriture de diagrammes et de Σ-diagrammes*. PhD thesis, PhD thesis, University of Marseille, 2013.

[77] Peter Selinger. A survey of graphical languages for monoidal categories. In *New structures for physics*, pages 289–355. Springer, 2010.

[78] Craig C Squier. Word problems and a homological finiteness condition for monoids. *Journal of Pure and Applied Algebra*, 49(1):201–217, 1987.

[79] Craig C. Squier, Friedrich Otto, and Yuji Kobayashi. A finiteness condition for rewriting systems. *Theoretical Computer Science*, 131(2):271–294, 1994.

[80] Ross Street. Limits indexed by category-valued 2-functors. *Journal of Pure and Applied Algebra*, 8(2):149–181, 1976.

[81] Ross Street. Categorical structures. *Handbook of algebra*, 1:529–577, 1996.

[82] Axel Thue. *Probleme über Veränderungen von Zeichenreihen nach gegebenen Regeln.* Skr. Vid. Kristianaia. I Mat. Naturv. Klasse, 1914.

[83] Heinrich Tietze. Über die topologischen invarianten mehrdimensionaler mannigfaltigkeiten. *Monatshefte für Mathematik und Physik*, 19(1):1–118, 1908.

[84] Lorenzo Tortora De Falco. *Réseaux, cohérence et expériences obsessionnelles*. PhD thesis, 2000.

[85] GS Tseitin. An associative calculus with undecidable problem of equivalence. In *Dokl. Akad. Nauk SSSR*, volume 107, pages 370–371, 1956.

[86] J Vicary, A Kissinger, and K Bar. Globular: an online proof assistant for higher-dimensional rewriting. 2016.