Proof-net as graph, Taylor expansion as pullback

Giulio Guerrieri[1][0000−0002−0469−4279], Luc Pellissier[2], and Lorenzo Tortora de Falco[3]

[1] University of Bath, Department of Computer Science, Bath, UK
g.guerrieri@bath.ac.uk
[2] IRIF, Université Paris Diderot, Paris, France pellissier@irif.fr
[3] Università Roma Tre, Dipartimento di Matematica e Fisica, Rome, Italy
tortora@uniroma3.it

Abstract. We introduce a new graphical representation for multiplicative and exponential linear logic proof-structures, based only on standard labelled oriented graphs and standard notions of graph theory. The inductive structure of boxes is handled by means of a box-tree. Our proof-structures are canonical and allows for an elegant definition of their Taylor expansion by means of pullbacks.

Keywords: linear logic · proof-net · Taylor expansion · graph

1 Introduction

Linear Logic (LL) [14] has been introduced by Girard as a refinement of intuitionnistic and classical logic that isolates the infinitary parts of reasoning under two modalities: the *exponentials* ! and ?. These modalities give a logical status to the operations of memory/hypothesis management such as *copying*/*contraction* or *erasing*/*weakening*: a linear proof corresponds to a program/proof that uses its arguments/hypotheses *linearly*, i.e. only once, while an exponential proof corresponds to a program/proof that can use its arguments/hypotheses at will.

One of the features of LL is that it allows us to represent its proofs as *proof-nets*, a graphical syntax alternative to sequent calculus. Sequent calculus is a standard formalism for several logical systems. However, sequent calculus forces an order among inference rules even when they are evidently independent, a drawback called *bureaucracy*. Proof-nets, instead, are a geometrical, parallel and bureaucracy-free representation of proofs as labeled oriented *graphs*. In proof-nets deductive rules are disposed on the plane, in parallel, and connected only by their causal relation. Clearly, not all graphs that can be written in the language of LL are proof-nets, i.e. represent a proof in LL sequent calculus. Proof-nets are special inhabitants of the wider land of *proof-structures*: they can be characterized, among proof-structures, by abstract (geometric) conditions called correctness criteria [14]. The procedure of cut-elimination can be applied directly to proof-structures, and proof-nets can also be seen as the proof-structures with a good behavior with respect to cut-elimination [3]. Cut-elimination defined on proof-structures is more elegant than in sequent calculus because it drastically reduces the need for

commutative steps, the non-interesting and bureaucratic burden in every sequent calculus proof of cut-elimination. Indeed, in proof-structures there is no last rule, and so most commutative cut-elimination cases simply disappear.

Unfortunately, this is a faithful picture of the advantages of proof-structures only in the multiplicative fragment of LL (MLL) [9], which does not contain exponentials ! and ?, and so it is not sufficiently expressive to encode classical or intuitionistic logic (or the λ-calculus) in. To handle the exponentials Girard was forced to introduce *boxes*. They come with the black-box principle: "boxes are treated in a perfectly modular way: we can use the box B without knowing its content, i.e., another box B' with exactly the same doors would do as well" [14].

According to this principle, boxes forbid interaction between their content and their outer environment. This is evident in the definition of correctness criteria for MELL (the multiplicative-exponential fragment of LL) and in the definition of cut-elimination steps for MELL. Let us consider cut-elimination. Some cut-elimination steps require us to duplicate or erase whole sub-proofs, typically the steps for the !-modality in MELL. Proofs in sequent calculus are tree-shaped and bear a clear notion of last rule, the root of the tree. This property has an obvious but important consequence: given a !-rule r in a sequent calculus proof, there is an evident sub-proof ending with r, the sub-tree rooted in r. Therefore, non-linear cut-elimination steps can easily be defined by duplicating or erasing sub-trees. Switching to proof structures, the situation radically changes, because a proof structure in general has many last rules, one for each formula in the conclusions. Given a rule r it is not clear how to find a sub-proof-structure ending with r. Thus, in order to define cut-elimination steps for the !-modality in MELL proof-structures—which requires to identify some sub-proof-structure—some information has to been added.

The typical solution is to re-introduce part of the bureaucracy in MELL proof-structures, pairing each !-rule with an *explicit box* containing the sub-proof that can be duplicated or erased during cut-elimination. In some fragments of MELL (for instance the intuitionistic one corresponding to the λ-calculus [24] or more in general the polarized one [1]) where proof-structures still have an implicit tree-like structure (since among the conclusions there is always exactly one distinct output, the analogue of sequent calculus last rule), the explicit box is actually not needed. But here we are interested in the full (classical) MELL fragment, where linear negation is involutive and classical duality can be interpreted as the possibility of juggling between different conclusions. Concretely, in the literature mainly two kinds of solution that make use of explicit boxes can be found:

1. A MELL proof-structure is an oriented graph together with some additional information to identify the content and the border of each box. This additional information can be provided either informally, just drawing the border of each box in the graph [14,10,19], but then the definition of MELL proof-structure is not rigorous; or in a more formal way [6,15,7], but then the definition is highly technical and *ad hoc*;
2. A MELL proof-structure is an *inductive* oriented graph [17,22,26,8], i.e. an oriented graph where with any vertex v of type ! is associated another oriented

graph representing the content of the box of v. This inductive solution can be taken to extremes by representing proof-structures with term-like syntax [12].

The drawback of Item 1 is that the definition of MELL proof-structure is not easily manageable because either it is not precise or it is too tricky. Item 2, instead, provides more manageable definitions of MELL proof-structures, but another drawback arises: they intrinsically are *not canonical*, in that there are different inductive presentations of a MELL proof-structure defined up to associativity and commutativity of contractions, neutrality of weakening with respect to contraction, and permutation of weakenings and contractions with box-border.

Our contribution. We present here a purely graphical definition of MELL proof-structures (Section 3), so as to keep Girard's original intuition of a proof-structure as a graph even in MELL. This definition follows the non-inductive approach seen in Item 1: we use n-ary vertices of type ? collapsing weakening, dereliction and contraction (like in [10]). In this way, we get a *canonical* representation of MELL proof-structures. But our definition is completely based on standard notions (recalled in Section 2) coming from the theory of graphs, being formal (with an eye towards complete computer formalization) but avoiding *ad hoc* technicalities to identify the border and the content of a box. The inductive structure of boxes is handled by means of a box-tree: indeed, a MELL proof-structure R is given by an oriented labelled graph $|R|$ plus a tree $\mathcal{A}_R$ (representing the order of the boxes of R) and a graph morphism box_R from $|R|$ to $\mathcal{A}_R$ which allows us to recognize the content and the border of all boxes in R. In this way, our MELL proof-structures are still manageable: sophisticated operations on them, such as the Taylor expansion [11] can be easily defined. As a test of the usability of our MELL proof-structures, we give an elegant definition of their Taylor expansion, by means of *pullbacks* (Section 5).

Moreover, our setting allows us to define in a simple way correctness graphs (used to characterize the proof-structures that are proof-nets, i.e. that correspond to proofs in the LL sequent calculus), as we show in Section 4 for MLL.

Since the main contribution of our work is to provide a new definition of MELL proof-structures, our paper contains several definitions and no new theorems.

2 Preliminaries on graphs

Graphs with half-edges. There are many formalizations of the familiar notion of graph. Here we adopt the one due to [4]:[4] a graph is still a set of edges and a set of vertices, but edges are now split in halves, allowing some of them to be hanging. Splitting every edge in two has at least three features that are of particular interest for representing LL proof-structures:

– two half-edges are connected by an involution, thus defining an edge. The fixpoints of this involution are thus dangling edges, connected to only one

[4] The folklore attributes the definition of graphs with half-edges to Kontsevitch and Manin, but the idea can actually be traced back to Grothendieck's *dessins d'enfant*.

vertex: they are suited to represent the conclusions of a proof-structure. In this way it is also easy to define some intuitive but formally tricky operations such as the graft or the substitution of a graph for another graph (see Example 1);
– given a graph and any of its vertices, it is natural to define the *corolla* of the vertex, that is the vertex itself with all the half-edges connected to it;
– finally, while studying proof-structures, it is always necessary to treat them both as oriented and unoriented graphs. With this definition of graph, an orientation, so as a labeling and a coloring, are structures on top of the structure of the unoriented graph (see Definition 3).

Definition 1 (graph). *A (finite)* graph τ *is a quadruple* $(F_\tau, V_\tau, \partial_\tau, j_\tau)$*, where*

– F_τ *is a finite set, whose elements are called* flags *of* τ*;*
– V_τ *is a finite set, whose elements are called* vertices *of* τ*;*
– $\partial_\tau : F_\tau \to V_\tau$ *is a function associating with each flag its* boundary*;*
– $j_\tau : F_\tau \to F_\tau$ *is an involution.*

The graph τ *is* empty *if* $V_\tau = \emptyset$*.*

A flag that is a fixed point of the involution j_F is a *tail* of τ. A two-element orbit $\{f, f'\}$ of j_F is an *edge* of τ between $\partial_\tau(f)$ and $\partial_\tau(f')$, and f and f' are the *halves* of such an edge. The set of edges of τ is denoted by E_τ.

Given two graphs τ and τ', it is always possible to consider their disjoint union $\tau \sqcup \tau'$ defined as the disjoint union of the underlying sets and functions.

A one-vertex graph with set of flags F and involution the identity function id_F on F is called the *corolla* with set of flags F; it is usually denoted by $*_F$.

Given a graph $\tau = (F_\tau, V_\tau, \partial_\tau, j_\tau)$, a vertex v defines a corolla $\tau_v = (F_v, \{v\}, \partial_\tau|_{F_v}, \mathrm{id}_{F_v})$ where $F_v = \partial_\tau^{-1}(v)$. Every graph can be described as the set of corollas of its vertices, together with the involution glueing the flags in edges.

Definition 2 (graph morphism and isomorphism). *Let* τ, σ *be two graphs. A* graph morphism $h\colon \tau \to \sigma$ *from* τ *to* σ *is a couple of functions* $(h_F\colon F_\tau \to F_\sigma, h_V\colon V_\tau \to V_\sigma)$ *such that* $h_V \circ \partial_\tau = \partial_\sigma \circ h_F$ *and* $h_F \circ j_\tau = j_\sigma \circ h_F$*.*

A graph morphism is injective *if its component functions are. A graph* isomorphism *is a graph morphism whose component functions are bijections.*

The category **Graph** has graphs as objects and morphisms of graphs as morphisms: indeed, graph morphisms compose (by composing the underlying functions) and the couple of identities (on vertices and flags) is neutral. It is a monoidal category, with disjoint union as a monoidal product.

Graphs with structure. Some structure can be put on top of a graph.

Definition 3 (structured graph). *Let* $\tau = (F_\tau, V_\tau, \partial_\tau, j_\tau)$ *be a graph.*

– *A* labeled graph (τ, ℓ_τ) *with labels in* I *is a graph* τ *and a function* $\ell_\tau\colon V_\tau \to I$*.*
– *A* colored graph (τ, c_τ) *is a graph* τ *with a function* $\mathsf{c}_\tau\colon F_\tau \to C$ *such that* $\mathsf{c}_\tau(f) = \mathsf{c}_\tau(f')$ *for the two halves* f, f' *of any edge of* τ*.*

- *An* oriented graph (τ, o_τ) *is a graph* τ *with a function* $\mathsf{o}_\tau \colon F_\tau \to \{\mathbf{in}, \mathbf{out}\}$ *such that* $\mathsf{o}_\tau(f) \neq \mathsf{o}_\tau(f')$ *for the two halves* f, f' *of any edge of* τ*. If* $\mathsf{o}_\tau(f) = \mathbf{out}$ *and* $\mathsf{o}_\tau(f') = \mathbf{in}$*,* $\{f, f'\}$ *is said an* edge *of* τ from $\partial_\tau(f)$ to $\partial_\tau(f')$*;* $\mathbf{in}$*-oriented (resp.* $\mathbf{out}$*-oriented) tails of* τ *are called* inputs *(resp.* outputs*) of* τ*; if* v *is a vertex of* τ*, its* inputs *(resp. its* outputs*) are the elements of the set* $\mathbf{in}_\tau(v) = \partial_\tau^{-1}(v) \cap \mathsf{o}_\tau^{-1}(\mathbf{in})$*; (resp.* $\mathbf{out}_\tau(v) = \partial_\tau^{-1}(v) \cap \mathsf{o}_\tau^{-1}(\mathbf{out})$*);*
- *An* ordered graph $(\tau, <_\tau)$ *is a graph together with an order on the flags.*

Different structures on a graph can combine: for instance, a graph τ can be endowed with both a labeling ℓ_τ and an orientation o_τ.

Graphs can be depicted in diagrammatic form. As a graph is just a disjoint union of corollas glued with the involution, we only need to depict corollas (as in Figure 1, on the left) and place the two halves of an edge next to each other (as in Figure 1, on the right). In oriented graphs, inputs of a corolla are depicted above the corolla, outputs are below; arrows also show the orientation. The color of a flag f (if any) is written next to f. The label of a vertex v (if any) is written inside v. If ordered, flags of a corolla are depicted increasing from left to right.

Example 1. The oriented labeled colored ordered corolla $\tau_{\mathbf{5}} = (*_{\mathbf{5}}, \mathsf{o}_{\mathbf{5}}, \ell_{\mathbf{5}}, \mathsf{c}_{\mathbf{5}}, <_{\mathbf{5}})$ depicted in Figure 1 (on the left) has $*$ as only vertex and $\mathbf{5} = \{0, 1, 2, 3, 4\}$ as set of flags; it is endowed with the order $0 <_{\mathbf{5}} 4$ and $1 <_{\mathbf{5}} 2 <_{\mathbf{5}} 3$, and

- the orientation $\mathsf{o}_{\mathbf{5}} \colon \mathbf{5} \to \{\mathbf{in}, \mathbf{out}\}$ defined by $\mathsf{o}_{\mathbf{5}}(0) = \mathsf{o}_{\mathbf{5}}(4) = \mathbf{out}$ and $\mathsf{o}_{\mathbf{5}}(1) = \mathsf{o}_{\mathbf{5}}(2) = \mathsf{o}_{\mathbf{5}}(3) = \mathbf{in}$,
- the labeling $\ell_{\mathbf{5}} \colon \{*\} \to \{\Pisymbol{pzd}{216}\}$ defined by $\ell_{\mathbf{5}}(*) = \Pisymbol{pzd}{216}$,
- the coloring $\mathsf{c}_{\mathbf{5}} \colon \mathbf{5} \to \{a_0, \dots, a_4\}$ defined by $\mathsf{c}(i) = a_i$ for all $i \in \mathbf{5}$.

Consider also the oriented labeled colored corolla $\sigma_{\mathtt{ax}}$, whose only vertex is labeled by $\mathtt{ax}$, and whose only flags are the outputs 5 (labeled by a_2) and 6 (labeled by a_3). The oriented labeled colored ordered two-vertex graph ρ depicted in Figure 1 (on the right) is obtained from the corollas $\tau_{\mathbf{5}}$ and $\sigma_{\mathtt{ax}}$ by defining the involution $j_\rho \colon \{0, \dots, 6\} \to \{0, \dots, 6\}$ as $j_\rho(i) = j_{\tau_{\mathbf{5}}}(i)$ for $i \in \{0, 1, 4\}$, and $j_\rho(i) = i + 3$ for $i \in \{2, 3\}$, and $j_\rho(i) = i - 3$ for $i \in \{5, 6\}$.

Each enrichment of the structure of graphs introduced in Definition 3 induces a notion of morphism that preserves such a structure, and an associated category. For instance, a morphism $h \colon (\tau, \mathsf{o}_\tau) \to (\sigma, \mathsf{o}_\sigma)$ where (τ, o_τ) and $(\sigma, \mathsf{o}_\sigma)$ are oriented graphs, is a graph morphism $h = (h_F, h_V) \colon \tau \to \sigma$ such that $\mathsf{o}_\sigma \circ h_F = \mathsf{o}_\tau$.

Trees and paths. An *unoriented path* on a graph τ is a finite and even sequence of flags $\varphi = (f_1, \dots, f_{2n})$ for some $n \in \mathbb{N}$ such that, for all $1 \leqslant i \leqslant n$, $j_\tau(f_{2i-1}) = j_\tau(f_{2i})$ and (if $i \neq n$) $\partial_\tau(f_{2i}) = \partial_\tau(f_{2i+1})$. We say that φ is *between* $\partial_\tau(f_1)$ *and* $\partial_\tau(f_{2n})$ if $n > 0$ (and it is a *cycle* if moreover $\partial_\tau(f_1) = \partial_\tau(f_{2n})$), otherwise it is the *empty* (*unoriented*) *path*, which is between any vertex and itself; the *length* of φ is n. Two vertices are *connected* if there is an unoriented path between them.

Let τ be a graph: τ is *connected* if any vertices $v, v' \in V_\tau$ are connected; a *connected component* of τ is a maximal (with respect the inclusion of flags and

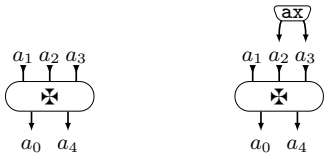


Fig. 1. An oriented labeled colored ordered corolla (on the left), and an oriented labeled colored ordered two-vertex graph (on the right).

vertices) connected sub-graph of τ; τ is *acyclic* (or a *forest*) if it has no cycles; τ is a *tree* if it is a connected forest.

A *rooted tree* τ is an oriented tree such that each vertex has exactly one output. Thus, τ has exactly one output tail: its boundary is called the *root* of τ.

Remark 1. Let τ and τ' be two rooted trees, and $h\colon \tau \to \tau'$ be an oriented graph morphism. As h_F preserves tails and orientation, h_V maps the root of τ to the root of τ'. Rooted trees and oriented graph morphisms form a category **RoTree**.

An *oriented path* on an oriented graph τ is an unoriented path $(f_1, \dots, f_{2n})$ for some $n \in \mathbb{N}$ such that f_{2i-1} is output and f_{2i} is input for all $1 \leqslant i \leqslant n$. Such a path is said to be *from* $\partial_\tau(f_1)$ *to* $\partial_\tau(f_{2n})$ if $n > 0$, otherwise it is the *empty* (oriented) *path*, which is from any vertex to itself.

The set of oriented paths on an oriented tree is finite. As such, given a tree τ, we define its *reflexive-transitive closure*, or *free category*, $\tau^{\circlearrowleft}$ as the oriented graph with same vertices and same tails as τ, and with an edge from v to v' for any oriented path from v to v' in τ. The operator $(\cdot)^{\circlearrowleft}$ lifts to a functor from the category **RoTree** to the category of oriented graphs.

3 DiLL proof-structures

This section is the core of our paper. We define here proof-structures corresponding to some fragments or extension of LL: MELL, DiLL and DiLL_0. Full differential linear logic (DiLL) is an extension of MELL (with the same language as MELL) provided with both promotion rule (i.e. boxes) and co-structural rules (the duals of the structural rules handling the ?-modality) for the !-modality: DiLL_0 and MELL are particular subsystems of DiLL, respectively the promotion-free one (i.e. without boxes) and the one without co-structural rules. As the study of cut-elimination is left to future work, our interest for DiLL is just to have an unitary syntax subsuming both MELL and DiLL_0: this is why, unlike [22,26], our DiLL proof-structures are not allowed to contain a set of DiLL proof-structures inside a box.

Given a countably infinite set of propositional variables $X, Y, Z, \dots$, (MELL) *formulas* (whose set is denoted by $\mathcal{F}_{\mathsf{MELL}}$) are defined by the following grammar:

$$A, B ::= X \mid X^{\perp} \mid \mathbf{1} \mid \bot \mid A \otimes B \mid A \parr B \mid !A \mid ?A$$

Linear negation $(\cdot)^\perp$ is defined via De Morgan laws $\mathbf{1}^\perp = \bot$, $(A \otimes B)^\perp = A^\perp \parr B^\perp$ and $(!A)^\perp = ?A^\perp$, so as to be involutive, i.e. $A^{\perp\perp} = A$ for any formula A. Variables and their negations are *atomic* formulas; $\otimes$ and $\parr$ (resp. ! and ?) are *multiplicative* (resp. *exponential*) *connectives*; $\mathbf{1}$ and $\bot$ are *multiplicative units*.

We equip an oriented graph with labels (specifying the type of the vertices, which is a MELL connective or unit), colors (specifying the type of the flags, which is a MELL formula), and a function that specifies the deepest box each flag or vertex is in; all of them are subject to compatibility conditions.

Definition 4 (module, proof-structure). *A (*DiLL*)* module $M = (|M|, \ell, \mathsf{o}, \mathsf{c}, <)$ *is a labeled (ℓ), oriented (o), colored (c), ordered ($<$) graph $|M|$ such that:*

- $\ell \colon V_{|M|} \to \{\mathtt{ax}, \mathtt{cut}, \mathbf{1}, \bot, \otimes, \parr, ?, !\}$ *associates with each vertex its* type*;*
- $\mathsf{c} \colon F_{|M|} \to \mathcal{F}_{\mathsf{MELL}}$ *associates with each flag its* type*;*
- $<$ *is a strict order on the flags of $|M|$ that is total on the tails of $|M|$ and on the inputs of each vertex labeled by $\parr$ or $\otimes$;*
- *for every vertex $v \in V_{|M|}$,*
  - *if $\ell(v) = \mathtt{cut}$, v has no output and exactly two inputs i_1 and i_2, such that $\mathsf{c}(i_1) = \mathsf{c}(i_2)^\perp$;*
  - *if $\ell(v) = \mathtt{ax}$, v has no inputs and exactly two outputs o_1 and o_2, such that $\mathsf{c}(o_1) = \mathsf{c}(o_2)^\perp$;*
  - *if $\ell(v) \in \{\mathbf{1}, \bot\}$, v has no inputs and only one output o, with $\mathsf{c}(o) = \ell(v)$;*
  - *if $\ell(v) \in \{\otimes, \parr\}$, v has exactly two inputs $i_1 < i_2$ and one output o, such that $\mathsf{c}(o) = \mathsf{c}(i_1)\, \ell(v)\, \mathsf{c}(i_2)$;*
  - *if $\ell(v) \in \{?, !\}$, v has exactly $n \geqslant 0$ inputs $i_1, \dots, i_n$ and one output o, such that $\mathsf{c}(o) = \ell(v)\, \mathsf{c}(i_j)$ for all $1 \leqslant j \leqslant n$;*[5]

In Figure 2 we depicted the corollas associated with all types of vertices.

*A (*DiLL*)* proof-structure *is a tuple $R = (|R|, \mathcal{A}, \mathsf{box})$, where $|R| = (\|R\|, \ell_R, \mathsf{o}_R, \mathsf{c}_R, <_R)$ is a module with no input tails, called the* structured graph *of R (and $\|R\|$ is the* graph *of R). Moreover, the following hold:*

- *$\mathcal{A}$ is a rooted tree with no input tails, called the* box-tree *of R.*
- *$\mathsf{box} \colon |R| \to \mathcal{A}^{\circlearrowleft}$ is a morphism of oriented graphs,*[6] *the* box-function *of R, such that box_F induces a partial bijection from $\bigcup_{v \in V_{\|R\|}, \ell(v) = !} \mathbf{in}_{|R|}(v)$ to the*

[5] This implies that $\mathsf{c}(i_j) = \mathsf{c}(i_k)$ for all $1 \leqslant j, k \leqslant n$.

[6] The structured graph $|R|$ of R is more structured (it is also labeled, colored, ordered) than an oriented graph such as $\mathcal{A}^{\circlearrowleft}$. When we talk of a morphism between two structured graphs where one of the two, say τ, is less structured than the other, say σ, we mean that τ must be only considered with the same structure as σ. Thus, in this case, box is a morphism from $(\|R\|, \mathsf{o}_R)$—discarding ℓ_R, c_R, $<_R$—to $\mathcal{A}^{\circlearrowleft}$.

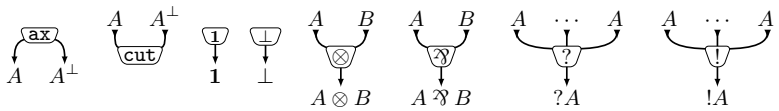


Fig. 2. DiLL cells, with their labels and their typed inputs and outputs.

set of input flags in $\mathcal{A}$.[7] *Moreover, for any vertex* $v \in V_{\|R\|}$ *with* $f \in \mathbf{in}_{|R|}(v)$, *if* $\mathsf{box}_V(\partial_{\|R\|} \circ j_{\|R\|}(f)) \neq \mathsf{box}_V(\partial_{\|R\|}(f))$ *then* $\ell(v) \in \{!, ?\}$.[8]

A proof-structure is empty *(denoted by* ε*) if its graph is empty.*
A MELL proof-structure *is a proof-structure such that:*

- *for all* $v \in V_{\|R\|}$, *if* $\ell(v) = !$ *then* $\mathsf{card}(\mathbf{in}_{|R|}(v)) = 1$;
- box_F *induces a (total) bijection from* $\bigcup_{v \in V_{\|R\|}, \ell(v)=!} \mathbf{in}_{|R|}(v)$ *to the set of input flags in* $\mathcal{A}$.

A DiLL_0 proof-structure *is a proof-structure whose box-tree contains only its root in the set of vertices. A* MLL *(resp.* MLL^-*)* proof-structure *is a* DiLL_0 *proof-structure whose structured graph has no vertices of type* ! *or* ? *(resp.* **1**, $\bot$, ! *or* ?*).*

Given a proof-structure $R = (|R|, \mathcal{A}, \mathsf{box})$, the output tails of $|R|$ are the *conclusions* of R. So, if f is the output of the root of $\mathcal{A}$, the pre-images $f_1, \ldots, f_n$ of f via box_F ordered according to $<_{|R|}$ form a finite sequence of the conclusions of R. The *type* of R is the list $(\mathsf{c}_{|R|}(f_1), \ldots, \mathsf{c}_{|R|}(f_n))$ of the types of the conclusions.

Borrowing the terminology of interaction nets [16,13], if R is a proof-structure, we say that the vertices of $|R|$ are the *cells* of R, the flags of $|R|$ are the *ports* of R.

Remark 2 (box). In our syntax, boxes do not have explicit constructors or cells, hence boxes and depth of a proof structure are recovered in a non-inductive way.

Let $R = (|R|, \mathcal{A}_R, \mathsf{box}_R)$ be a proof-structure. With every flag f of $|R|$ such that $\mathsf{box}_{R_F}(f)$ is an input flag of $\mathcal{A}_R$[9] is associated a *box* B_f, that is the subgraph

[7] This means that for any input flag f' in $\mathcal{A}$ there is exactly one input f of some vertex of type ! in $|R|$ such that $\mathsf{box}_F(f) = f'$; but $\mathsf{box}_F(f)$ need not be an input flag in $\mathcal{A}$ for any input f of some vertex of type ! in $|R|$ (by definition of morphism, $\mathsf{box}_F(f)$ is necessarily an input flag in $\mathcal{A}^{\circlearrowleft}$). Intuitively, a vertex v of type ! represents a generalized co-contraction (in particular, a co-weakening if it has no inputs), and a box is associated with (and only with) each input f of v such that $\mathsf{box}_F(f)$ is an input flag in $\mathcal{A}$ (and not only in $\mathcal{A}^{\circlearrowleft}$): f represents the principal door (in the border) of such a box (note that for $f' \in F_{\|R\|}$, if $f' \neq f$ then $\mathsf{box}_F(f') \neq \mathsf{box}_F(f)$ and that $\mathsf{box}_V(\partial_{\|R\|} \circ j_{\|R\|}(f)) \neq \mathsf{box}_V(\partial_{\|R\|}(f))$ for such a f).

[8] Roughly, it says that the border of a box is made of (inputs of) vertices of type ! or ?.

[9] According to the constraints on box_R, this condition can be fulfilled only by inputs of a cell of type ! (a !-cell, for short) in $|R|$, and an input of a !-cell need not fulfill it; in particular, if R is a MELL proof-structure, then this condition is fulfilled by all and only the inputs of !-cells (and such an input is unique for any !-cell) in $|R|$; but if R is a DiLL_0 proof-structure, then this condition is not fulfilled by any flag in $|R|$ (since $\mathcal{A}_R$ has no inputs) and so box_R is a graph morphism associating the root of $\mathcal{A}_R$ with any vertex of $|R|$. Therefore, in a DiLL_0 proof-structure $\rho = (|\rho|, \mathcal{A}_\rho, \mathsf{box}_\rho)$, $\mathcal{A}_\rho$ and box_ρ do not induce any structure on $|\rho|$: ρ can be identified with $|\rho|$.

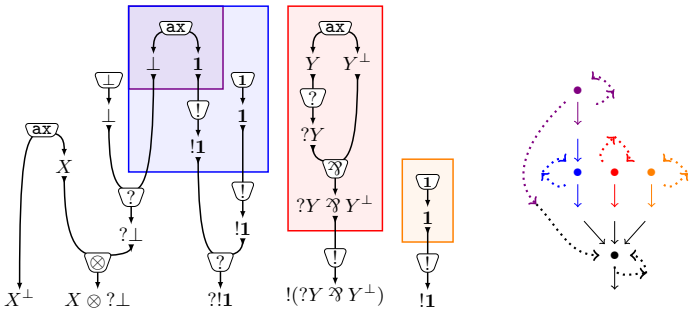


Fig. 3. A MELL proof-structure R with its box-tree $\mathcal{A}_R$. The dotted arrows represent the edges added to $\mathcal{A}_R$ by the reflexive-transitive closure $(\cdot)^{\circlearrowleft}$.

of $|R|$ (which is actually a proof-structure) made up of all the cells v (with their inputs and outputs) such that there is an oriented path on $\mathcal{A}_R$ from $\mathsf{box}_{R_V}(v)$ to $\mathsf{box}_{R_V}(\partial_{\|R\|} \circ j_{\|R\|}(f))$: a *conclusion* of such a box B_f associated with f is every output f' of a vertex v in B_f such that $\partial_{\|R\|} \circ j_{\|R\|}(f')$ is not in B_f. Summing up, every non-root vertex of $\mathcal{A}_R$ represents a box in R, and the root of $\mathcal{A}_R$ represents the parts of R outside all the boxes. The tree-structure of $\mathcal{A}_R$ expresses the nesting condition of boxes.

The *depth* of a cell v of R is the length of the oriented path in $\mathcal{A}_R$ from $\mathsf{box}_R(v)$ to the root of $\mathcal{A}_R$. The *depth* of R is the maximal depth of the cells of R.

Example 2. In Figure 3 a MELL proof-structure R is depicted: the structured graph $|R|$ of R is on the left; the box-tree $\mathcal{A}_R$ of R is on the right. The box-function box_R is kept implicit by means of colors: the colored areas in $|R|$ represent boxes, and the same color is used on $\mathcal{A}_R$ to show where each box is mapped by box_R.

The proof-structures we have just defined are quite rigid: they depend on their carrier-sets of cells and wires. Nonetheless, a precise answer to the question "When two proof-structures can be considered equal?" requires a notion of isomorphism inherited by the notion of graph isomorphism.

Definition 5 (isomorphism of proof-structures). *Let $R = (|R|, \mathcal{A}_R, \mathsf{box}_R)$ and $R' = (|R'|, \mathcal{A}_{R'}, \mathsf{box}_{R'})$ be proof-structures, with $|R| = (\|R\|, \ell_R, \mathsf{o}_R, \mathsf{c}_R, <_R)$ and $|R'| = (\|R'\|, \ell_{R'}, \mathsf{o}_{R'}, \mathsf{c}_{R'}, <_{R'})$. An* isomorphism of proof-structures *$f\colon R \simeq R'$ is a couple $f = (f_{|\cdot|}, f_{\mathsf{box}})$ where:*

- *$f_{|\cdot|}\colon |R| \to |R'|$ is an isomorphism of the structured graphs of R and R',*
- *$f_{\mathsf{box}}\colon \mathcal{A}_R \to \mathcal{A}_{R'}$ is an isomorphism of the box-trees of R and R',*

such that the following diagram commutes

$$\begin{array}{ccc} |R| & \xrightarrow{\mathsf{box}_R} & \mathcal{A}_R^{\circlearrowleft} \\ \downarrow f_{|\cdot|} & & \downarrow f_{\mathsf{box}}^{\circlearrowleft} \\ |R'| & \xrightarrow{\mathsf{box}_{R'}} & \mathcal{A}_{R'}^{\circlearrowleft} \end{array}$$

$$\frac{\vdash \Gamma, A, B, \Delta}{\vdash \Gamma, B, A, \Delta}(\text{exc}) \qquad \frac{}{\vdash A, A^\perp}(\text{ax}) \qquad \frac{\vdash \Gamma, A \qquad \vdash A^\perp, \Delta}{\vdash \Gamma, \Delta}(\text{cut}) \qquad \frac{\vdash \Gamma \qquad \vdash \Delta}{\vdash \Gamma, \Delta}(\text{mix})$$

$$\frac{}{\vdash 1}(\mathbf{1}) \qquad \frac{\vdash \Gamma}{\vdash \Gamma, \bot}(\bot) \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}(\parr) \qquad \frac{\vdash \Gamma, A \qquad \vdash B, \Delta}{\vdash \Gamma, A \otimes B, \Delta}(\otimes) \qquad \frac{}{\vdash}(\text{emp})$$

Fig. 4. Sequent calculi for MLL^- (all rules but (mix), (emp), (**1**), ($\bot$)), MLL (all rules).

Note that if R is isomorphic to a proof-structure R', and R is a MELL or DiLL_0 proof-structure, then R' is respectively a MELL or DiLL_0 proof-structure.

4 Sequent calculi, proof-nets and correctness for MLL

Every proof in the sequent calculus for LL can be translated in a proof-structure with the same conclusions. Figure 4 gives the rules of the sequent calculi for two multiplicative fragments of LL: MLL^- (without units) and MLL (with units and mix). A MLL^- (resp. MLL) *formula* is a MELL formula without exponential connectives and multiplicative units (resp. without exponential connectives). A *sequent* is a finite sequence of (MLL^- or MLL, depending on the context) formulas $A_1, \dots, A_n$. Capital Greek letters $\Gamma, \Delta, \dots$ range over sequents.

Definition 6 (translation, proof-net). *Let* $\mathsf{X} \in \{\mathsf{MLL}^-, \mathsf{MLL}\}$.

With any proof π *in the sequent calculus for* X *and conclusion* $\vdash \Gamma$ *is associated a* X *proof-structure* R_π *with type* Γ*, called the* translation of π*, defined by induction on the size of* π *as follows:*[10]

$$\pi = \frac{}{\vdash A, A^\perp}(\text{ax}) \qquad \rightsquigarrow \qquad |R_\pi| = \quad \overset{ax}{\overbrace{}} \quad A \quad A^\perp$$

$$\pi = \frac{\begin{array}{cc}\vdots\,\pi_1 & \vdots\,\pi_2 \\ \vdash \Gamma, A & \vdash A^\perp, \Delta\end{array}}{\vdash \Gamma, \Delta}(\text{cut}) \qquad \rightsquigarrow \qquad |R_\pi| = \quad \overset{cut}{\underbrace{\boxed{|R_{\pi_1}|} \quad \Gamma \quad A}} \quad \overbrace{\boxed{|R_{\pi_2}|} \quad A^\perp \quad \Delta}$$

$$\pi = \frac{\vdots\,\pi_1}{\vdash \Gamma, A, B, \Delta}(\text{exc}) \qquad \rightsquigarrow \qquad |R_\pi| = \quad \boxed{|R_{\pi_1}|} \quad \Gamma \quad B \quad A \quad \Delta$$

$$\pi = \frac{}{\vdash}(\text{emp}) \qquad \rightsquigarrow \qquad |R_\pi| = |\varepsilon| \quad (\varepsilon \text{ is the empty proof-structure})$$

[10] We write only the graph $|R_\pi|$ of R_π, because its box-tree $\mathcal{A}_{R_\pi}$ and its box-function box_{R_π} are trivial (see Footnote 9).

$$\pi = \dfrac{\vdots\,\pi_1 \quad \vdots\,\pi_2}{\vdash \Gamma \quad \vdash \Delta}(\text{mix}) \qquad \rightsquigarrow \qquad |R_\pi| = \boxed{|R_{\pi_1}|}\ \boxed{|R_{\pi_2}|}$$

$$\pi = \dfrac{}{\vdash \mathbf{1}}(\mathbf{1}) \qquad \rightsquigarrow \qquad |R_\pi| = \mathbf{1}$$

$$\pi = \dfrac{\vdots\,\pi_1}{\vdash \Gamma, \bot}(\bot) \qquad \rightsquigarrow \qquad |R_\pi| = \boxed{|R_{\pi_1}|}\ \bot$$

$$\pi = \dfrac{\vdots\,\pi_1}{\vdash \Gamma, A, B}(\parr) \qquad \rightsquigarrow \qquad |R_\pi| = \boxed{|R_{\pi_1}|}\ \Gamma\ A\ B \to A \parr B$$

$$\pi = \dfrac{\vdots\,\pi_1 \quad \vdots\,\pi_2}{\vdash \Gamma, A \quad \vdash B, \Delta}(\otimes) \qquad \rightsquigarrow \qquad |R_\pi| = \boxed{|R_{\pi_1}|}\ \boxed{|R_{\pi_2}|}\ \Gamma\ A\ B\ \Delta \to A \otimes B$$

A proof-structure R is a X proof-net *(or is* X sequentializable*) if $R = R_\pi$ (i.e. R is the translation of π) for some proof π in the* X *sequent calculus.*

The translation is not surjective (neither injective) over proof-structures, even when we restrict to MLL^- or MLL proof-structures. Purely graph-theoretical conditions, called *correctness criteria*, have been presented in order to characterize the set of sequentializable proof-structures. We give here two among the most celebrated of such correctness criteria, *switching acyclicity* and its variant *switching acyclicity and connectedness*, presented originally in [9]. We define them via the switching operation on a proof-structure R, which roughly consists of "detaching" all inputs but one of every vertex of type $\parr$ in R. This switching can be easily defined in our setting, thanks to modules and involutions.

Definition 7 (switching, correctness graph). *Let R be a* MLL *proof-structure, whose structured graph is $|R|$ and whose (unoriented) graph is $\|R\|$.*

A switching *of R is a function $\mathsf{s}_R \colon \{v \in V_{\|R\|} \mid \ell_{|R|}(v) = \parr\} \to F_{\|R\|}$ such that $\mathsf{s}_R(v)$ is one of the two inputs of v.*

With every switching s_R of R is associated a s_R-correctness graph *$\tau(\mathsf{s}_R)$, which is the (unoriented) graph obtained from $\|R\|$ by replacing the involution $j_{\|R\|} \colon F_{\|R\|} \to F_{\|R\|}$ for $\|R\|$ with $j_{\tau(\mathsf{s}_R)} \colon F_{\|R\|} \to F_{\|R\|}$ defined as follows:*

$$j_{\tau(\mathsf{s}_R)}(f) = \begin{cases} j_{\|R\|}(f) & \text{if } f \text{ is an input of a vertex } v \text{ such that either} \\ & \quad \ell_{|R|}(v) = \parr \text{ and } \mathsf{s}_R(v) = f, \text{ or } \ell_{|R|}(v) \neq \parr; \\ f & \text{otherwise.} \end{cases}$$

A MLL *proof-structure R is* switching acyclic *(resp.* switching acyclic and connected*) if every correctness graph of R is acyclic (resp. acyclic and connected).*

Theorem 1 (Sequentialization, [9]).

1. *A* MLL^- *proof-structure is* MLL^- *sequentializable iff it is switching acyclic and connected.*
2. *A* MLL *proof-structure is* MLL *sequentializable iff it is switching acyclic.*

The definitions and the results of this section can be easily generalized to DiLL_0 and MELL proof-structures.

5 The Taylor expansion

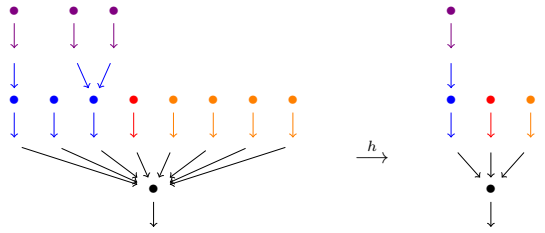
The *Taylor expansion* [11] of a MELL (or more in general a DiLL) proof-structure R is a (usually infinite) set of DiLL_0 proof-structures: roughly speaking, each element of the Taylor expansion of R is obtained from R by replacing each box B in R with n_B copies of its content (for some $n_B \in \mathbb{N}$), recursively on the depth of R. Note that n_B depends not only on B but also on which "copy" of all boxes containing B we are considering. Up to now (with the exception of [15]), the Taylor expansion of MELL proof-structure is defined globally and inductively [19,21]: with every MELL proof-structure R is directly associated its Taylor expansion (the whole set!) by induction on the depth of R.

We adopt an alternative non-inductive approach, which strongly refines [15]: the Taylor expansion is defined pointwise (see Example 3 and Figure 5). Indeed, proof-structures have a tree structure that is made explicit through their box-function. The definition of the Taylor expansion of a proof-structure uses this tree structure: first we define how to "expand" a tree via the notion of thick subtree [5] (Definition 8), then we take all the expansions of the tree structure of a proof-structure and we *pull* them *back* to the underlying graphs (Definition 9), finally we forget the tree structures associated with them (Definition 10).

Definition 8 (thick subtree [5]). *Let τ be a rooted tree. A* thick subtree *of τ is a pair (σ, h) of a rooted tree σ and a graph morphism $h\colon \sigma \to \tau$.*

As in analysis, an addend of the Taylor expansion of an analytical function f is an approximant of f, here if $\mathcal{A}$ is the box-tree of a proof-structure R, a thick subtree of $\mathcal{A}$ is a sort of approximant of $\mathcal{A}$ taking recursively a number of copies (possibly 0) of each input of the vertices of $\mathcal{A}$, i.e. of each box of R.

Example 3. The following is (a graphical presentation of) a thick subtree (τ, h) of the box-tree $\mathcal{A}_R$ of the proof-structure R in Figure 3, where the graph morphism $h\colon \tau \to \mathcal{A}_R$ is depicted chromatically (same color means same image via h).



Intuitively, τ is obtained from $\mathcal{A}_R$ by taking 3 copies of the blue box, 1 copy of the red box, 4 copies of the orange box; in the first (resp. second; third) copy of the blue box, 1 copy (resp. 0 copies; 2 copies) of the purple box has been taken.

The crucial point is to pull back the expansion of trees to proof-structures. In Appendix A we recall the definition of pullback in the category of graphs.

Definition 9 (proto-Taylor expansion). *Let $R = (|R|, \mathcal{A}_R, \mathsf{box}_R)$ be a proof-structure. The* proto-Taylor expansion *of R is set $\mathfrak{T}_R^{\mathsf{proto}}$ of thick subtrees of $\mathcal{A}_R$.*

Let $t = (\tau_t, h_t) \in \mathfrak{T}_R^{\mathsf{proto}}$. The t-expansion *of R is the pullback (R_t, p_t, p_R):*

$$\begin{array}{ccc} R_t & \xrightarrow{p_t} & \tau_t^{\circlearrowleft} \\ \downarrow{\scriptstyle p_R} & & \downarrow{\scriptstyle h_t^{\circlearrowleft}} \\ |R| & \xrightarrow{\mathsf{box}_R} & \mathcal{A}_R^{\circlearrowleft} \end{array}$$

computed in the category of graphs and graph morphisms.[11]

Given a proof-structure R and $t = (\tau_t, h_t) \in \mathfrak{T}_R^{\mathsf{proto}}$, the t-expansion (R_t, p_t, p_R) of R is a naked graph. In order for it to be lifted into a DiLL_0 proof-structure, we need to define more structure on it, using either t or R.

- Oriented, labeled and colored structures on the graph $|R|$ are defined through functions defined on the flags and vertices of $|R|$; hence, by precomposing with the graph morphism $p_R = (p_{R_F}, p_{R_V})\colon R_t \to |R|$, this transports to a structure of oriented labeled and colored graph on R_t;
- the order on the flags of R_t is defined as the order induced by their image in $|R|$: $f < f'$ if and only if $p_{R_F}(f) < p_{R_F}(f')$;
- let $[\tau_t]$ be the tree made up only of the root of τ_t and its output and let $\iota\colon \tau_t \to [\tau_t]$ be the graph morphism sending all the vertices of τ_t to the root of τ_t; $\iota^{\circlearrowleft}$ induces by post-composition a morphism $\overline{h_t} = \iota^{\circlearrowleft} \circ p_t\colon R_t \to [\tau_t]^{\circlearrowleft}$.

With its structure of oriented, labeled, ordered and colored graph, the triple $(R_t, [\tau_t], \overline{h_t})$ is a DiLL_0 proof-structure.

[11] This means that $\tau_t^{\circlearrowleft}$ and $\mathcal{A}_R^{\circlearrowleft}$ are considered as (unoriented) graphs, see also Footnote 6.

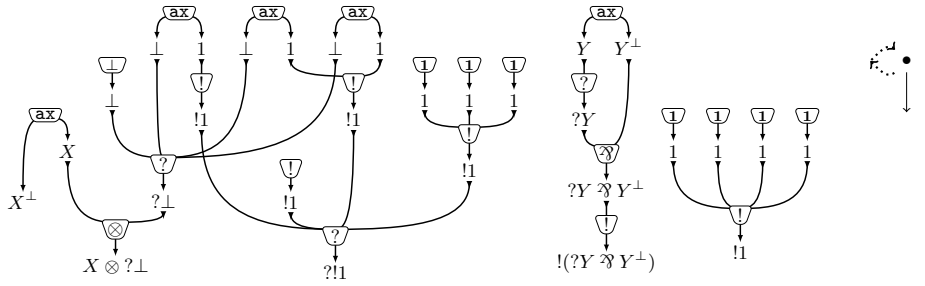


Fig. 5. The element of the Taylor expansion of the MELL proof-structure R in Fig. 3 obtained from the element of $\mathfrak{T}_R^{\mathsf{proto}}$ depicted in Ex. 3.

Definition 10 (Taylor expansion). *Let R be a proof-structure. The* Taylor expansion *of R is the set $\mathfrak{T}_R = \{(R_t, [\tau_t], \overline{h_t}) \mid t = (\tau_t, h_t) \in \mathfrak{T}_R^{\mathsf{proto}}\}$.*

An element of the Taylor expansion of a proof-structure is thus a DiLL_0 proof-structure. It has much less structure than the pullback (R_t, p_t, p_R), which defines a DiLL_0 proof-structure R_t coming with its projections $p_t \colon R_t \to \tau_t^{\circlearrowleft}$ and $p_R \colon R_t \to |R|$. In particular, a cell in R_t is labelled (through the projections) by the cell of $|R|$ and the branch of the box-tree of R it arose from. But $(R_t, [\tau_t], \overline{h_t})$ where R_t is without its projections p_t and p_R loses the correspondence with $R = (|R|, \mathcal{A}_R, \mathsf{box}_R)$ (see Fig. 5). Reconstructing such projections, starting only from an element of the Taylor expansion, can be seen as the core of the works on the injectivity of the Taylor expansion, see [7,15].

Remark 3. From the definition it follows that each element of the Taylor expansion of a proof-structure R has the *same conclusions* and the *same type* as R. More precisely, let R be a proof-structure and ρ be in the Taylor expansion of R: f is a conclusion of ρ and an output of a cell v of ρ if and only if $p_{R_F}(f)$ is a conclusion of R and an output of a cell $p_{R_V}(v)$ of R. And $\mathsf{c}_{|\rho|}(f) = \mathsf{c}_{|R|}(p_{R_F}(f))$ (i.e. the type of f in ρ is the same as the type of $p_{R_F}(f)$ in R) and $\ell_{|\rho|}(v) = \ell_{|R|}(p_{R_V}(v))$ (i.e. the type of v in ρ is the same as the type of $p_{R_V}(v)$ in R).

Remark 4. One could go further and define an *incomplete Taylor expansion* of a proof-structure, where some boxes are expanded, but not all. This extension fits into this framework: the absence of boxes in an element $(R_t, [\tau_t], \overline{h_t})$ of the Taylor expansion owes only to the fact that $[\tau_t]$ is a root. By replacing this root by a rooted tree, we keep track of which boxes are expanded and which are not.

6 Conclusions

Cut-elimination. The fact that we get a *canonical* (as explained in Section 1) representation of MELL proof-structures is not only an aesthetic matter: it has important consequences on the definition of cut elimination, because it avoids the presence of the bureaucratic commutative-steps. As a notable consequence,

as proved in [2], the proof of strong normalization for MELL becomes quite elegant and much easier than with non-canonical MELL proof-structures [23]. The canonicity of our definition of MELL proof-structures paves the way to such a smooth cut elimination; we plan to work out this issue in future work.

Taylor expansion and relational semantics. Relational semantics is the simplest denotational model of LL. It can be seen as a degenerate case of Girard's coherent semantics [14]: formulas are interpreted as sets and proof-structures as relations between them. It is well-known that, given a MELL proof-structure R, there is a correspondence between certain equivalence classes on its relational semantics $[\![R]\!]$ and the elements of its Taylor expansion $\mathfrak{T}_R$: in particular, two cut-free MELL proof-structures with atomic axioms have the same relational semantics if and only if they have the same Taylor expansion. This equivalence, which relates the syntactic notion of Taylor expansion to the semantic notion of relational model, holds only with a *canonical* representation MELL proof-structures, such as ours.

Mix and forests. In an ongoing work, we are naturally led to consider several proof-structures at the same time and to "mix" them in a single proof-structure. Our definition of proof-structure (Definition 4) is perfectly suited for this purpose. Indeed, the definition of a box-tree lends itself to a generalization: considering not trees, but forests of boxes. The graph morphism condition implies, if its image is a forest, that the proof-structure contains several connected components; and each inverse image of a tree in the forest is actually a proof-structure.

So, by slightly generalizing the definition, we can consider a list of proof-structures as a whole proof-structure, while respecting their individuality, contrarily to all of them having the same image through box. This allows us to mimic the situation of the `mix` rule of sequent calculus: taking two proofs and considering it one can be done by merging two roots of a box-forest.

A most general Taylor expansion: Milner's absorption. It is possible to go farther in the definition of the Taylor expansion and to specify a new box-tree that need not be trivial. This allows for instance to expand some boxes and not all; and even to expand partially a box: copying alongside a box its contents and (co-)contracting the box with the copies.

This is reminiscent of the π-calculus and of Mazza's parsimonious λ-calculus [18], where the exponentials verify the isomorphism $!A \simeq A \otimes !A$.

Other boxes. Boxes for other connectives of linear logic have been considered in various works: quantifiers (both first-order and second order [14]), fix-points [20] and additives [14,25]. The boxing tree represents a sequential structure that is added on top of a proof-structure. All these connective share in common to require such a sequentialization.

As the different kind of sequentialization need to merge correctly, we believe this approach to be adapted without problems to other kinds of boxes, paving the way to a unified notion of proof-structures for a richer system than MELL.

References

1. Accattoli, B.: Compressing polarized boxes. In: 28th Annual Symposium on Logic in Computer Science (LICS 2013). pp. 428–437. IEEE Computer Society (2013). https://doi.org/10.1109/LICS.2013.49
2. Accattoli, B.: Linear logic and strong normalization. In: 24th International Conference on Rewriting Techniques and Applications (RTA 2013). LIPIcs, vol. 21, pp. 39–54. Schloss Dagstuhl (2013). https://doi.org/10.4230/LIPIcs.RTA.2013.39
3. Béchet, D.: Minimality of the correctness criterion for multiplicative proof nets. Mathematical Structures in Computer Science **8**(6), 543–558 (1998)
4. Borisov, D.V., Manin, Y.I.: Generalized Operads and Their Inner Cohomomorphisms, pp. 247–308. Birkhäuser Basel, Basel (2008). https://doi.org/10.1007/978-3-7643-8608-5_4
5. Boudes, P.: Thick subtrees, games and experiments. In: Typed Lambda Calculi and Applications, 9th International Conference (TLCA 2009). Lecture Notes in Computer Science, vol. 5608, pp. 65–79 (2009). https://doi.org/10.1007/978-3-642-02273-9
6. de Carvalho, D., Tortora de Falco, L.: The relational model is injective for multiplicative exponential linear logic (without weakenings). Ann. Pure Appl. Logic **163**(9), 1210–1236 (2012)
7. de Carvalho, D.: Taylor expansion in linear logic is invertible. Logical Methods in Computer Science **14**(4), 1–73 (2018). https://doi.org/10.23638/LMCS-14(4:21)2018
8. de Carvalho, D., Pagani, M., Tortora de Falco, L.: A semantic measure of the execution time in linear logic. Theor. Comput. Sci. **412**(20), 1884–1902 (2011). https://doi.org/10.1016/j.tcs.2010.12.017
9. Danos, V., Regnier, L.: The structure of multiplicatives. Arch. Math. Log. **28**(3), 181–203 (1989). https://doi.org/10.1007/BF01622878
10. Danos, V., Regnier, L.: Proof-nets and the Hilbert Space. In: Proceedings of the Workshop on Advances in Linear Logic. pp. 307–328. Cambridge University Press (1995)
11. Ehrhard, T., Regnier, L.: Uniformity and the Taylor expansion of ordinary lambda-terms. Theoretical Computer Science **403**(2-3), 347–372 (2008)
12. Ehrhard, T.: A new correctness criterion for MLL proof nets. In: Joint Meeting of the Twenty-Third Conference on Computer Science Logic and the Twenty-Ninth Symposium on Logic in Computer Science (CSL-LICS '14). pp. 38:1–38:10. ACM (2014). https://doi.org/10.1145/2603088.2603125
13. Ehrhard, T., Regnier, L.: Differential interaction nets. Theor. Comput. Sci. **364**(2), 166–195 (2006). https://doi.org/10.1016/j.tcs.2006.08.003
14. Girard, J.Y.: Linear logic. Theoretical Computer Science **50**(1), 1–101 (1987). https://doi.org/10.1016/0304-3975(87)90045-4
15. Guerrieri, G., Pellissier, L., Tortora de Falco, L.: Computing connected proof(-structure)s from their Taylor expansion. In: 1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016). LIPIcs, vol. 52, pp. 20:1–20:18. Schloss Dagstuhl (2016). https://doi.org/10.4230/LIPIcs.FSCD.2016.20
16. Lafont, Y.: Interaction nets. In: Seventeenth Annual ACM Symposium on Principles of Programming Languages (POPL 1990). pp. 95–108. ACM Press (1990). https://doi.org/10.1145/96709.96718
17. Laurent, O.: Polarized proof-nets and lambda-μ-calculus. Theor. Comput. Sci. **290**(1), 161–188 (2003). https://doi.org/10.1016/S0304-3975(01)00297-3

18. Mazza, D.: Simple parsimonious types and logarithmic space. In: 24th Annual Conference on Computer Science Logic (CSL 2015). LIPIcs, vol. 41, pp. 24–40. Schloss Dagstuhl (2015). https://doi.org/10.4230/LIPIcs.CSL.2015.24
19. Mazza, D., Pagani, M.: The separation theorem for differential interaction nets. In: Logic for Programming, Artificial Intelligence, and Reasoning, 14th International Conference (LPAR 2007). Lecture Notes in Computer Science, vol. 4790, pp. 393–407. Springer (2007). https://doi.org/10.1007/978-3-540-75560-9_29
20. Montelatici, R.: Polarized Proof Nets with Cycles and Fixpoints Semantics. In: Typed Lambda Calculi and Applications. pp. 256–270. Springer, Berlin, Heidelberg, Berlin, Heidelberg (Jun 2003). https://doi.org/10.1007/3-540-44904-3_18
21. Pagani, M., Tasson, C.: The inverse taylor expansion problem in linear logic. In: Proceedings of the 24th Annual Symposium on Logic in Computer Science (LICS 2009). pp. 222–231. IEEE Computer Society (2009). https://doi.org/10.1109/LICS.2009.35
22. Pagani, M.: The cut-elimination theorem for differential nets with promotion. In: Typed Lambda Calculi and Applications, 9th International Conference, (TLCA 2009). Lecture Notes in Computer Science, vol. 5608, pp. 219–233. Springer (2009). https://doi.org/10.1007/978-3-642-02273-9_17
23. Pagani, M., Tortora de Falco, L.: Strong normalization property for second order linear logic. Theor. Comput. Sci. **411**(2), 410–444 (2010). https://doi.org/10.1016/j.tcs.2009.07.053, `https://doi.org/10.1016/j.tcs.2009.07.053`
24. Solieri, M.: Geometry of resource interaction and Taylor-Ehrhard-Regnier expansion: a minimalist approach. Mathematical Structures in Computer Science **28**(5), 667–709 (2018). https://doi.org/10.1017/S0960129516000311
25. Tortora de Falco, L.: The additive multiboxes. Annals of Pure and Applied Logic **120**(1-3), 65–102 (Apr 2003). https://doi.org/10.1016/S0168-0072(02)00042-8
26. Tranquilli, P.: Intuitionistic differential nets and lambda-calculus. Theor. Comput. Sci. **412**(20), 1979–1997 (2011). https://doi.org/10.1016/j.tcs.2010.12.022

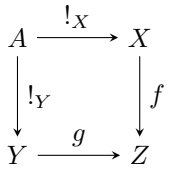
Technical Appendix

A Computing a pullback in the category of graphs

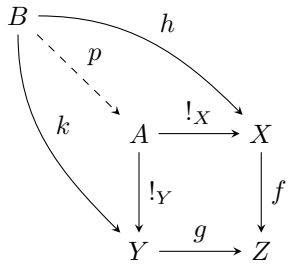
The category of graphs has all pullbacks, a fact that we use extensively. We recall here all the definitions and facts that are packed in that affirmation.

Definition 11 (pullback). *Let $\mathcal{C}$ be a category. Let X, Y, and Z be three objects of $\mathcal{C}$ and $f : X \to Z$ and $g : Y \to Z$ be two arrows of $\mathcal{C}$.*

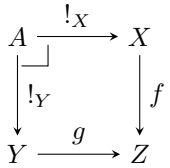
The pullback *of X and Y along f and g is the triple $(A, !_X, !_Y)$ such that the diagram*



commutes and, for every other $(B, h : B \to X, k : B \to Y)$ making the same diagram commute, there exists a unique arrow $p : B \to A$ such that:



It is unique (up to unique isomorphism), and it is customary to write $X \times_Z Y$ a pullback of X and Y over Z (leaving f and g implicit) and a pullback diagram with a corner:



All pullbacks exist in the category of graphs. Explicitely, let $\tau = (F_\tau, V_\tau, \partial_\tau, j_\tau)$, $\sigma = (F_\sigma, V_\sigma, \partial_\sigma, j_\sigma)$ and $\rho = (F_\rho, V_\rho, \partial_\rho, j_\rho)$ be three graphs and $g : \sigma \to \tau$, $h : \rho \to \tau$ be two graph morphisms. Consider the two sets

$$F = \{(f_1, f_2) \in F_\sigma \times F_\rho \mid g_F(f_1) = h_F(f_2)\}$$
$$V = \{(v_1, v_2) \in V_\sigma \times V_\rho \mid g_V(v_1) = h_V(v_2)\}$$

They are both equiped with two projections, which we will write $\pi^F_\sigma, \pi^F_\rho, \pi^V_\sigma, \pi^V_\rho$. Let $f \in F$.

$$\begin{aligned} g_V \circ \partial_\sigma \circ \pi^F_\sigma(f) &= \partial_\tau \circ g_F \circ \pi^F_\sigma(f), \text{ because } g \text{ is a graph morphism} \\ &= \partial_\tau \circ h_F \circ \pi^F_\rho(f), \text{ by definition of } F \\ &= h_V \circ \partial_\rho \circ \pi^F_\rho(f), \text{ because } h \text{ is a graph morphism} \end{aligned}$$

Hence, we can define $\partial : F \to V$ by $\partial(f) = (\partial_\sigma \circ \pi^F_\sigma(f), \partial_\rho \circ \pi^F_\rho(f))$. In the same way, we define $j : F \to F$ by $j(f) = (j_\sigma \circ \pi^F_\sigma(f), j_\rho \circ \pi^F_\rho(f))$, and check that it is an involution.

Hence $\sigma \times_\tau \rho = (F, V, \partial, j)$ is a graph and $\pi_\sigma = (\pi^F_\sigma, \pi^V_\sigma) : \sigma \times_\tau \rho \to \sigma$ and $\pi_\rho = (\pi^F_\rho, \pi^V_\rho) : \sigma \times_\tau \rho \to \rho$ are graph morphisms.

$$\begin{array}{ccc} \sigma \times_\tau \rho & \xrightarrow{\pi_\rho} & \rho \\ \downarrow{\scriptstyle \pi_\sigma} & & \downarrow{\scriptstyle h} \\ \sigma & \xrightarrow{g} & \tau \end{array}$$

Consider now any $\mu = (F_\mu, V_\mu, \partial_\mu, j_\mu)$ such that the diagram

$$\begin{array}{ccc} \mu & \xrightarrow{p} & \rho \\ \downarrow{\scriptstyle q} & & \downarrow{\scriptstyle h} \\ \sigma & \xrightarrow{g} & \tau \end{array}$$

commutes. For $f \in F_\mu$, let $r_F(f) = (p_F(f), q_F(f))$ and for $v \in V_\mu$, let $r_V(v) = (p_V(v), q_V(v))$. We check that it defines a graph morphism $r : \mu \to \sigma \times_\tau \rho$ and it factorises p and q.