GLUING RESOURCE PROOF-STRUCTURES: INHABITATION AND INVERTING THE TAYLOR EXPANSION

GIULIO GUERRIERI, LUC PELLISSIER, AND LORENZO TORTORA DE FALCO

University of Bath, Department of Computer Science, Bath, United Kingdom
e-mail address: g.guerrieri@bath.ac.uk

Université Paris Est Créteil, LACL, F-94010 Créteil, France
e-mail address: luc.pellissier@lacl.fr

Università Roma Tre, Dipartimento di Matematica e Fisica, Rome, Italy
e-mail address: tortora@uniroma3.it

Abstract. A Multiplicative-Exponential Linear Logic (MELL) proof-structure can be expanded into a set of resource proof-structures: its Taylor expansion. We introduce a new criterion characterizing those sets of resource proof-structures that are part of the Taylor expansion of some MELL proof-structure, through a rewriting system acting both on resource and MELL proof-structures. As a consequence, we also prove semi-decidability of the type inhabitation problem for cut-free MELL proof-structures.

1. Introduction

Resource λ-calculus and the Taylor expansion. Girard's linear logic (LL, [Gir87]) is a refinement of intuitionistic and classical logic that isolates the infinitary parts of reasoning in two (dual) modalities: the *exponentials* ! and ?. They give a logical status to the operations of memory management such as *copying* and *erasing*: a linear proof corresponds—via the Curry–Howard isomorphism—to a program that uses its argument *linearly*, *i.e.* exactly once, while an exponential proof corresponds to a program that can use its argument at will.

The intuition that linear programs are analogous to linear functions (as studied in linear algebra) while exponential programs mirror a more general class of analytic functions got a technical incarnation in Ehrhard's work [Ehr02, Ehr05] on LL-based denotational semantics for the λ-calculus. This investigation has been then internalized in the syntax, yielding the *resource λ-calculus* [ER03, ER08], inspired by [Bou93]: there, copying and erasing are forbidden and replaced by the possibility to apply a function to a *bag* of resource λ-terms which specifies how many times (in a finite number) an argument can be linearly passed to the function, so as to represent only bounded computations.

The *Taylor expansion* [ER08] (more precisely, its support; but here and throughout the paper we do not consider the rational coefficients in the Taylor expansion) associates with an ordinary λ-term a—generally infinite—set of resource λ-terms, recursively approximating the

Key words and phrases: linear logic, Taylor expansion, proof-structure, pullback, natural transformation.

Preprint submitted to
Logical Methods in Computer Science

usual application: the Taylor expansion of the λ-term MN is made of resource λ-terms of the form $t[u_1, \dots, u_n]$, where t is a resource λ-term in the Taylor expansion of M, and $[u_1, \dots, u_n]$ is a bag of arbitrarily finitely many (possibly 0) resource λ-terms in the Taylor expansion of N. Roughly, the idea is to decompose a program into a set of purely "resource-sensitive programs", all of them containing only bounded (although possibly non-linear) calls to inputs. The notion of Taylor expansion has had many applications in the theory of the λ-calculus, *e.g.* in the study of linear head reduction [ER06a], normalization [PTV16, Vau17], Böhm trees [BHP13, KMP20, BM20], λ-theories [MR14], intersection types [MPV18]. In general, understanding the relation between a program and its Taylor expansion renews the logical approach to the quantitative analysis of computation started with the inception of LL.

A natural question is the *inverse Taylor expansion problem*: how to characterize which sets of resource λ-terms are contained in the Taylor expansion of a same λ-term? Ehrhard and Regnier [ER08] defined a simple *coherence* binary relation such that a finite set of resource λ-terms is included in the Taylor expansion of a λ-term if and only if the elements of this set are pairwise coherent. Coherence is crucial in many structural properties of the resource λ-calculus, such as in the proof that in the λ-calculus normalization and Taylor expansion commute [ER06a, ER08].

We aim to solve the inverse Taylor expansion problem in the more general context of LL, more precisely in the *multiplicative-exponential fragment* MELL of LL, being aware that for MELL no coherence relation can solve it (see below). A side effect of this investigation is apparently unrelated to the notion of Taylor expansion: we characterize MELL formulas that are inhabited by cut-free MELL proof-structures, so as to prove semi-decidability of the *type inhabitation problem* for cut-free MELL proof-structures (again, see below).

Proof-nets, proof-structures and their Taylor expansion: seeing trees behind graphs. In MELL, linearity and the sharp analysis of computations naturally lead to represent proofs in a more general *graph*-like syntax instead of a term-like or tree-like one.[1] Indeed, linear negation is involutive and classical duality can be interpreted as the possibility of juggling between different conclusions, without a distinguished output [Par92]. Graphs representing proofs in MELL are called *proof-nets*: their syntax is richer and more expressive than the λ-calculus (which corresponds to an intuitionistic implicative fragment of MELL). Contrary to λ-terms, proof-nets are special inhabitants of the wider land of *proof-structures*. A proof-structure is any "graph" that can be build in the language of proof-nets and it need not represent a proof in MELL. Proof-nets can be characterized, among proof-structures, by abstract (geometric) conditions called correctness criteria [Gir87].

Proof-structures are well-behaved for performing computations: indeed, the procedure of cut-elimination can be applied to proof-structures, and proof-nets can also be seen as the proof-structures with a good behavior with respect to cut-elimination [Béc98]. Furthermore, proof-structures can be interpreted in denotational models and proof-nets can be characterized among them by semantic means [Ret97]. It is then natural to attack problems in the general framework of proof-structures. In our work, correctness plays no role at all, hence we will consider proof-structures and not only proof-nets. MELL proof-structures are a particular kind of graphs, whose edges are labeled by MELL formulas and vertices by MELL connectives, and for which special subgraphs are highlighted, the *boxes*, representing the parts of the proof-structure that can be copied and discarded (*i.e.* called an unbounded number of times).

[1] A term-like object is essentially a tree, with one output (its root) and many inputs (its other leaves).

A box is delimited from the rest of a proof-structure by exponential modalities: its border is made of one !-cell, its principal door, and arbitrarily many ?-cells, its auxiliary doors. Boxes are either nested or disjoint (they cannot partially overlap), so as to add a tree-like structure to proof-structures *aside* from their graph-like nature.

As in λ-calculus, one can define [ER06b] box-free *resource* (or DiLL_0) *proof-structures*[2], where !-cells make resources available boundedly, and the *Taylor expansion* of MELL proof-structures into these resource proof-structures, that recursively copies the content of the boxes an arbitrary number of times. In fact, as somehow anticipated by Boudes [Bou09], such a Taylor expansion operation can be carried on any tree-like structure. This primitive, abstract, notion of Taylor expansion can then be pulled back to the structure of interest, as shown in [GPT19] and put forth again here.

The question of coherence for proof-structures. The *inverse Taylor expansion problem* has a natural counterpart for MELL proof-structures: given a set of resource proof-structures, is there a MELL proof-structure the Taylor expansion of which contains the set? Pagani and Tasson [PT09] give the following answer: it is possible to decide whether a finite set of resource proof-structures is a subset of the Taylor expansion of a same MELL proof-structure (and even possible to do it in nondeterministic polynomial time); but unlike the λ-calculus, the structure of the relation "being part of the Taylor expansion of a same proof-structure" is *much more* complicated than a binary (or even n-ary) coherence. Indeed, for any $n > 1$, it is possible to find $n + 1$ resource proof-structures such that any n of them are in the Taylor expansion of some MELL proof-structure, but there is no MELL proof-structure whose Taylor expansion has all the $n+1$ as elements (see our Example 8.6 and [Tas09, pp. 244-246]).

In this work, we introduce a new combinatorial criterion, *glueability*, for deciding whether a set of resource proof-structures is a subset of the Taylor expansion of some MELL proof structure, based on a *rewriting system* on lists of MELL formulas. Our criterion is more general and simpler than the one of [PT09], which is limited to the *cut-free* case with *atomic axioms* and characterizes only *finite* sets: we do not have these limitations. We believe that our criterion is a useful tool for studying proof-structures. We conjecture that it can be used to show that a binary coherence relation exists for resource proof-structures satisfying a suitable geometric restriction. It might also shed light on correctness and sequentialization.

As the proof-structures we consider are typed, an unrelated difficulty arises: a resource proof-structure ρ of type A might not be in the Taylor expansion of any cut-free MELL proof-structure, not because it does not respect the structure imposed by the Taylor expansion, but because there is no cut-free MELL proof-structure of type A, and ρ can "mask" this "untypeability".[3] To solve this issue, we enrich the resource (but not MELL) proof-structure syntax with a "universal" proof-structure: a special ✠-cell (*daimon*) that can have any number of outputs of any types, representing information plainly missing (see Section 10 for more details and the way this matter is handled by Pagani and Tasson [PT09]).

[2]Aka differential proof-structure [dC16], differential net [ER06b, MP07, dC18], simple net [PT09].

[3]Similarly, in the λ-calculus, there is no closed λ-term of type $X \to Y$ with $X \neq Y$ atomic, but the resource λ-term $(\lambda f.f)[]$ can be given that type: the empty bag $[]$ kills any information on the argument.

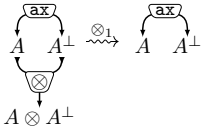
Our contribution. This paper is the long version of [GPT20] and we keep its structure and main results: the *glueability* criterion that solves the inverse Taylor problem in MELL proof-structures (Theorem 8.3), and the way we prove it, which consists in showing that the Taylor expansion defines a *natural transformation* (Theorem 7.4) from the realm of resource proof-structures to the realm of MELL proof-structures (see Section 2 for an informal explanation). With respect to [GPT20], the main novelties are:

(1) Following [GPT19], we introduce rigorous definitions of all the notions of graph theory (Section 3) involved in the definition of proof-structures and Taylor expansion. In this way, we can present here a purely graphical definition of MELL proof-structures (Section 4), so as to keep Girard's original intuition of a proof-structure as a graph even in MELL and to avoid *ad hoc* technicalities to identify the border and the content of a box. This is not only an aesthetic issue but also practical, because our MELL proof-structures are manageable: sophisticated operations on them can be easily defined. For instance, we give an elegant definition of their Taylor expansion by means of *pullbacks* (Section 5).
(2) With respect to [GPT20], here we use daimons in a different and more limited way. In particular, unlike [GPT20], our MELL proof-structures do not contain any ✠-cell. Thus, our results refer to a more standard and interesting definition of MELL proof-structures, and we deal with less syntactic categories than in [GPT20], simplifying the presentation and fixing some technical inaccuracies in a few definitions and lemmas in [GPT20].
(3) As a consequence of the previous point, our results—in particular glueability and natural transformation—are more informative, and allow us also to solve the apparently unrelated *inhabitation problem* for cut-free MELL proof-structures (it could not be solved with the results proven in [GPT20]): our rewrite system *semi-decides* if, for a list Γ of MELL formulas, there is a cut-free MELL proof-structure of type Γ. We only provide a semi-algorithm: we can guess a rewrite sequence and check its correctness; but there is no bound on its length. Inhabitation problems are well-studied in many type systems for the λ-calculus, but no results are in the literature for MELL proof-structures. Our contribution may shed sone light on the open problem of decidability of MELL.
(4) Akin to [GPT20], to simplify the presentation, we first focus on proof-structures restricted to atomic axioms, then Section 9 shows how to lift our results to the non-atomic case. Compared to [GPT20], such a lift is more elegant and requires less *ad hoc* adjustments.

2. Outline and technical issues

The rewritings. The essence of our rewrite system is not located in proof-structures but in lists of MELL formulas (Definition 6.1). In a very down-to-earth way, the rewrite system is generated by elementary steps akin to rules of sequent calculus read from the *bottom up*: they act on a list of conclusions, analogous to a monolaterous right-handed sequent. These steps can be seen as morphisms in a category **Sched** whose objects are lists of MELL formulas, and are actually more sequentialized than sequent calculus rules, as they do not allow for commutation. For instance, the rule corresponding to the introduction of a $\otimes$ on the i-th formula, is defined as $\otimes_i : (\gamma_1, \dots, \gamma_{i-1}, A \otimes B, \gamma_{i+1}, \dots, \gamma_n) \to (\gamma_1, \dots, \gamma_{i-1}, A, B, \gamma_{i+1}, \dots, \gamma_n)$.

These rewrite steps then act on MELL proof-structures, coherently with their type, by modifying (most of the times, erasing) a cell immediately above the conclusion of the proof-structure. Formally, this means that there is a functor qMELL from **Sched** to the category **Rel** of sets



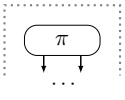
and relations, associating with a list of MELL formulas the set of MELL proof-structures with these conclusions, and with a rewrite step a relation implementing it (Definition 6.4). The rules *deconstruct* the proof-structure, starting from its conclusions. The rule $\otimes_1$ acts by removing a $\otimes$-cell on the first conclusion, replacing it by two conclusions.

These rules can only act on specific proof-structures, and indeed, capture a lot of their structure: $\otimes_i$ can be applied to a MELL proof-structure R if and only if R has a $\otimes$-cell in the conclusion i (as opposed to, say, an axiom). So, in particular, every proof-structure is completely characterized by any sequence rewriting it to the empty proof-structure.

Naturality. The same rules act also on sets of resource proof-structures, defining the functor $\mathfrak{P}\mathsf{qDiLL}_0$ from the category **Sched** of rewrite steps into the category **Rel** (Definition 7.3). When carefully defined, the Taylor expansion induces a *natural transformation* from $\mathfrak{P}\mathsf{qDiLL}_0$ to qMELL (Theorem 7.4). By applying this naturality repeatedly, we get our characterization (Theorem 8.3): a set of resource proof-structures Π is a subset of the Taylor expansion of a MELL proof-structure if and only if Π is *glueable*, that is, there is a sequence rewriting Π to the singleton of the *empty* proof-structure.

The naturality property is not only a mean to get our characterization, but also an interesting result in itself: natural transformations can often be used to express fundamental properties in a mathematical context. In this case, the *Taylor expansion is natural* with respect to the possibility of building a (MELL or resource) proof-structure by adding a cell to its conclusions or boxing it. Said differently, naturality of the Taylor expansion roughly means that the rewrite rules that deconstruct a MELL proof-structure R and a set of resource proof-structures in the Taylor expansion of R mimic each other.

Quasi-proof-structures and mix. Our rewrite rules consume proof-structures from their conclusions. The rule corresponding to boxes in MELL opens a box by deleting its principal door (a !-cell) and its border, while for a resource proof-structure it deletes a !-cell and separates the different copies of the content of the box (possibly) represented by such a !-cell. This operation is problematic in a twofold way. In a resource proof-structure, where the border of boxes is not marked, it is not clear how to identify such copies. On the other side, in a MELL proof-structure the content of a box is not to be treated as if it were at the same level as what is outside of the box: it can be copied many times or erased, while what is outside boxes cannot, and treating the content in the same way as the outside suppresses this distinction, which is crucial in LL. So, we need to remember that the content of a box, even if it is at depth 0 (*i.e.* not contained in any other box) after erasing the box wrapping it by means of our rewrite rules, is not to be mixed with the rest of the structure at depth 0.



In order for our proof-structures to provide this information, we need to generalize them and consider that a proof-structure can have not just a tree of boxes, but a *forest*: this yields the notion of *quasi-proof-structure* (Definition 4.4). In this way, according to our rewrite rules, opening a box by deleting its principal door amounts to taking a box in the tree and disconnecting it from its root, creating a new tree. We draw this in a quasi-proof-structure by surrounding elements having the same root with a dashed line, open from the bottom, remembering the phantom presence of the border of the box, even if it was erased. This allows one to open the box only when it is "alone", surrounded by a dashed line (see Definition 6.3).

This is not merely a technical remark, as this generalization gives a status to the `mix` rule of LL: indeed, mixing two proofs amounts to taking two proofs and considering them as one, without any other modifications. Here, it amounts to taking two proofs, each with its box-tree, and considering them as one by merging the roots of their trees (see the mix step in Definition 6.3). We embed this design decision up to the level of formulas, which are segregated in different zones that have to be mixed before interacting (see the notion of partition of a finite sequence of formulas in Section 4.1).

Geometric invariance and emptiness: the filled Taylor expansion. The use of forests instead of trees for the nesting structure of boxes, where the different roots are thought of as the contents of long-gone boxes, has an interesting consequence in the Taylor expansion: indeed, an element of the Taylor expansion of a proof-structure contains an arbitrary number of copies of the contents of the boxes, in particular *zero*. If we think of the part at depth 0 of a MELL proof-structure as inside an invisible box, its content can be deleted in some elements of the Taylor expansion just as any other box.[4] As erasing completely the conclusions would cause the Taylor expansion not preserve the conclusions (which would lead to technical complications), we introduce the *filled Taylor expansion* (Definition 5.7), which contains not only the elements of the usual Taylor expansion, but also elements of the Taylor expansion where one component has been erased and replaced by a ✠-cell (*daimon*), representing lack of information, apart from the number and types of the conclusions. Roughly, a ✠-cell is a placeholder for any DiLL_0 proof-structure of given conclusions.

Glueability and cut-free glueability. Our glueability criterion is based on local rewritings, which yields a geometric and modular approach to the inverse Taylor expansion problem. From the geometric point of view, there is nothing special in the elementary rewrite step corresponding to the cut rule, and it is natural to prove our glueability criterion in presence of cuts (Theorem 8.3.1). But from the proof-theoretical point of view, the gulf separating cut-free proof-structures from the others shows up: for every MELL formula A, the set of MELL proof-structures with conclusion of type A is never empty, while this might very well be the case if we restrict to cut-free MELL proof-structures (see Example 8.5 and Remark 8.8). The modularity of our proofs allows to straightforwardly adapt the criterion to the cut-free case (Theorem 8.3.2) and thus to state correctly (and easily solve) the *type inhabitation problem* for (cut-free) MELL proof-structures (Theorem 8.9).

Outline. Section 3 recalls some preliminary notions on graph theory. In Section 4 we define (MELL and DiLL_0) proof-structures and quasi-proof-structures with atomic axioms. Section 5 defines the notion of Taylor expansion. Section 6 introduces the rewriting rules on lists of lists of formulas and lifts them to MELL quasi-proof-structures via the functor qMELL. In Section 7 we lift the rewriting rules to DiLL_0 quasi-proof-structures via the functor $\parr\mathsf{qDiLL}_0$ and we show our first main result: the Taylor expansion induces a natural transformation between the two functors. Section 8 proves our other main results: the solution of the inverse Taylor expansion problem (via a glueability criterion) and the solution of the type inhabitation problem in MELL. In Section 9 we show how to adapt our method when axioms are not necessarily atomic. Section 10 concludes with some final remarks.

[4]The dual case, of copying the contents of a box, poses no problem in our approach.

3. Preliminaries on graphs

Graphs with half-edges. There are many formalizations of the familiar notion of graph. Here we adopt the one due to [BM07]:[5] 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 four features of particular interest to represent LL proof-structures:

- two half-edges are connected by an involution, thus defining an edge linking two vertices (possibly the same vertex). The fixed points of this involution are "hanging" edges, linked to a vertex only on one endpoint: they are well suited for representing the conclusions of a proof-structure. In this way it is also easy to define some intuitive but formally tricky operations such as grafting/substituting a graph into/for another graph (see Example 3.4);
- given any vertex v in a graph τ, it is natural to define the *corolla* of v, that is v itself with the half-edges linked to it; τ is the union of its corollas, glued together by the involution;
- while studying proof-structures, it is often necessary to treat them both as directed and undirected graphs. With this definition of graph, an orientation, a labeling and a coloring, are structures on top of the structure of the undirected graph (see Definition 3.3);
- this definition of graph allows a uniform syntax to represent both proof-structures and other structures of interest (*e.g.*, the box-tree of a proof-structure). In this way, we avoid appealing to *ad hoc* conditions in the definitions of proof-structure and Taylor expansion, which are then more compact and rely only on notions from graph theory.

Definition 3.1 (graph)**.** A (finite[6]) *graph* is a quadruple $\tau = (F_\tau, V_\tau, \partial_\tau, j_\tau)$, 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;
- $\partial_\tau \colon F_\tau \to V_\tau$ is a function associating with each flag its *endpoint*;
- $j_\tau \colon F_\tau \to F_\tau$ is an involution, *i.e.* $(j_\tau \circ j_\tau)(f) = f$ for any $f \in F_\tau$.

The graph τ is *empty* if $V_\tau = \emptyset$.[7]

A flag that is a fixed point of the involution j_τ is a *tail* of τ. A two-element orbit $\{f, f'\}$ of j_τ is an *edge* of τ between vertices $\partial_\tau(f)$ and $\partial_\tau(f')$, and f, f' are the *halves* of the edge.

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 a *corolla* (the endpoint of each flag is the only vertex); 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 gluing some flags in edges.

Definition 3.2 (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.

[5] 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*.

[6] The finiteness condition on F_τ and V_τ can be dropped, so as to allow for possibly infinite graphs. We require it because we only deal with finite graphs.

[7] This implies that ∂_τ is the empty function and $F_\tau = \emptyset$ (as F_τ is the domain of ∂_τ).

Intuitively, a graph morphism preserves tails and edges. The category **Graph** has graphs as objects and graph morphisms as arrows: indeed, graph morphisms compose (by composing the underlying functions) and the couple of identities (on vertices and flags) is neutral for such a composition. 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 3.3 (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) *with colors in* C 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.
- A *directed 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* $\partial_\tau(f')$; **in**-oriented (resp. **out**-oriented) tails of τ are called *inputs* (resp. *outputs*) *of* τ; if v is a vertex of τ, the *inputs* (resp. *outputs*) *of* v 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 τ can combine, *e.g.* τ can be endowed with both a labeling ℓ_τ and an orientation o_τ. Roughly, a graph is labeled (resp. colored) when labels are associated with its vertices (resp. edges and "hanging" edges). In a directed graph, an input (resp. output) of a vertex v is a—half or hanging—edge incoming in (resp. outgoing from) v.

Graphs can be depicted diagrammatically. As a graph is just a disjoint union of corollas glued by the involution, we only need to depict corollas (Figure 1, on the left) and place the two halves of an edge next to each other (Figure 1, on the right). In directed graphs, inputs of a corolla are depicted above, 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. Dots $\cdots$ above or below a flag stand for an arbitrary number (possibly 0) of flags (see Figures 8f and 9j).

Example 3.4. The directed labeled colored ordered corolla $\mathbf{5} = (*_{\mathbf{5}}, \mathsf{o}_{\mathbf{5}}, \ell_{\mathbf{5}}, \mathsf{c}_{\mathbf{5}}, <_{\mathbf{5}})$ depicted in Figure 1 (on the left) has $*$ as only vertex and $F_{\mathbf{5}} = \{0, 1, 2, 3, 4\}$ as set of flags; it is endowed with the order $0 <_{\mathbf{5}} 4$ and $1 <_{\mathbf{5}} 2 <_{\mathbf{5}} 3$, the labeling $\ell_{\mathbf{5}}(*) = \maltese$, and the orientation $\mathsf{o}_{\mathbf{5}} \colon F_{\mathbf{5}} \to \{\mathbf{in}, \mathbf{out}\}$ 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 coloring $\mathsf{c}_{\mathbf{5}} \colon F_{\mathbf{5}} \to \{a_0, \dots, a_4\}$ defined by $\mathsf{c}(i) = a_i$ for all $i \in F_{\mathbf{5}}$.

Let $\sigma_{\mathtt{ax}}$ be the directed labeled colored corolla, whose only vertex is labeled by $\mathtt{ax}$, and whose only flags are the outputs 5 (colored by a_2) and 6 (colored by a_3). The directed labeled colored ordered two-vertex graph ρ in Figure 1 (on the right) is obtained by "grafting" $\sigma_{\mathtt{ax}}$ into $\mathbf{5}$, *i.e.* from $\sigma_{\mathtt{ax}}$ and $\mathbf{5}$ by defining the involution $j_\rho \colon \{0, \dots, 6\} \to \{0, \dots, 6\}$ as $j_\rho(i) = j_{\mathbf{5}}(i)$ 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 3.3 induces a notion of morphism that preserves such a structure, and an associated category. For instance, given two directed graphs (τ, o_τ) and $(\sigma, \mathsf{o}_\sigma)$, a *directed graph morphism* $h \colon (\tau, \mathsf{o}_\tau) \to (\sigma, \mathsf{o}_\sigma)$ is a graph morphism $h = (h_F, h_V) \colon \tau \to \sigma$ such that $\mathsf{o}_\sigma \circ h_F = \mathsf{o}_\tau$; this means that h_F maps input (resp. output) flags of τ into input (resp. output) flags of σ.

FIGURE 1. A directed labeled colored ordered corolla **5** (left), and a directed labeled colored ordered two-vertex graph ρ (right), see Example 3.4.

Trees and paths. An *undirected 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 $1 \leqslant i \leqslant n$, $j_\tau(f_{2i-1}) = f_{2i} \neq f_{2i-1}$ and (if $i \neq n$) $\partial_\tau(f_{2i}) = \partial_\tau(f_{2i+1})$ with $f_{2i} \neq f_{2i+1}$. 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* (*undirected*) *path*, which is between any vertex and itself; the *length* of φ is n. Two vertices are *connected* if there is an undirected 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 to the inclusion of flags and 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. Note that a forest can be seen as a list of trees, each tree is a connected component.

A *rooted tree* τ is a directed tree such that each vertex has exactly one output. Thus, by finiteness, τ has exactly one output tail f: the endpoint of f is called the *root* of τ.

Remark 3.5. Let τ and τ' be two rooted trees, and $h \colon \tau \to \tau'$ be a directed graph morphism. As h_F preserves tails and orientation, h_V maps the root of τ to the root of τ'. Rooted trees and directed graph morphisms form a category **RoTree**.

A *directed path* on a directed graph τ is an undirected path $\varphi = (f_1, \dots, f_{2n})$ for some $n \in \mathbb{N}$ where f_{2i-1} is output and f_{2i} is input for all $1 \leqslant i \leqslant n$. We say that φ is *from* $\partial_\tau(f_1)$ *to* $\partial_\tau(f_{2n})$ if $n > 0$; otherwise it is the *empty* (directed) *path*, from any vertex to itself.

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

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 involved in that affirmation.

Definition 3.6 (pullback)**.** Let $\mathcal{C}$ be a category. Let X, Y, Z be objects of $\mathcal{C}$, and $f \colon X \to Z$ and $g \colon Y \to Z$ be arrows of $\mathcal{C}$. A *pullback* of f and g is the triple $(P, !_X, !_Y)$ where P is an object of $\mathcal{C}$ and $!_X \colon P \to X$ and $!_Y \colon P \to Y$ are arrows of $\mathcal{C}$ such that diagram (3.1) commutes and, for any $(Q, h \colon Q \to X, k \colon Q \to Y)$ making the same diagram commute, there is a unique arrow $u \colon Q \to P$ factorizing h and k, *i.e.* such that diagram (3.2) commutes.

$$\begin{array}{ccc} P & \xrightarrow{!_X} & X \\ {\scriptstyle !_Y}\downarrow & & \downarrow{\scriptstyle f} \\ Y & \xrightarrow{g} & Z \end{array} \quad (3.1) \qquad \begin{array}{ccc} Q & \xrightarrow{h} & \\ & \searrow^{u} & \\ & P \xrightarrow{!_X} X & \\ {\scriptstyle k}\searrow & {\scriptstyle !_Y}\downarrow & \downarrow{\scriptstyle f} \\ & Y \xrightarrow{g} Z & \end{array} \quad (3.2) \qquad \begin{array}{ccc} X \times_Z Y & \xrightarrow{!_X} & X \\ {\scriptstyle !_Y}\downarrow & \lrcorner & \downarrow{\scriptstyle f} \\ Y & \xrightarrow{g} & Z \end{array} \quad (3.3)$$

A pullback $(P, !_X, !_Y)$ of $f\colon X \to Z$ and $g\colon Y \to Z$ is unique (up to unique isomorphism); P is usually denoted by $X \times_Z Y$ (leaving f, g implicit) and depicted like in diagram (3.3).

All pullbacks exist in the category **Graph** of graphs. Let us show it explicitly. 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 graphs, and let $g = (g_F, g_V)\colon \sigma \to \tau$ and $h = (h_F, h_V)\colon \rho \to \tau$ be graph morphisms. Consider the sets

$$F = \{(f_\sigma, f_\rho) \in F_\sigma \times F_\rho \mid g_F(f_\sigma) = h_F(f_\rho)\} \quad V = \{(v_\sigma, v_\rho) \in V_\sigma \times V_\rho \mid g_V(v_\sigma) = h_V(v_\rho)\}.$$

They are both equipped with their natural projections $\pi^F_\sigma\colon F \to F_\sigma$, $\pi^F_\rho\colon F \to F_\rho$ and $\pi^V_\sigma\colon V \to V_\sigma$, $\pi^V_\rho\colon V \to V_\rho$. 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 $\partial\colon F \to V$ by $\partial(f) = (\partial_\sigma \circ \pi^F_\sigma(f), \partial_\rho \circ \pi^F_\rho(f))$. In the same way, we define $j\colon F \to F$ by $j(f) = (j_\sigma \circ \pi^F_\sigma(f), j_\rho \circ \pi^F_\rho(f))$, and check that it is an involution.

Thus, $\sigma \times_\tau \rho = (F, V, \partial, j)$ is a graph and $\pi_\sigma = (\pi^F_\sigma, \pi^V_\sigma)\colon \sigma \times_\tau \rho \to \sigma$ and $\pi_\rho = (\pi^F_\rho, \pi^V_\rho)\colon \sigma \times_\tau \rho \to \rho$ are graph morphisms such that diagram (3.4) below commutes.

$$\begin{array}{ccc} \sigma \times_\tau \rho & \xrightarrow{\pi_\rho} & \rho \\ {\scriptstyle \pi_\sigma}\downarrow & & \downarrow{\scriptstyle h} \\ \sigma & \xrightarrow{g} & \tau \end{array} \quad (3.4) \qquad \begin{array}{ccc} \mu & \xrightarrow{p} & \rho \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle h} \\ \sigma & \xrightarrow{g} & \tau \end{array} \quad (3.5)$$

Consider now any graph $\mu = (F_\mu, V_\mu, \partial_\mu, j_\mu)$ and graph morphisms $p = (p_F, p_V)\colon \mu \to \rho$ and $q = (q_F, q_V)\colon \mu \to \sigma$ such that diagram (3.5) above commutes. For $f \in F_\mu$, let $r_F(f) = (p_F(f), q_F(f))$ and for $v \in V_\mu$, let $r_V(v) = (p_V(v), q_V(v))$. It is easy to check that it defines a graph morphism $r = (r_F, r_V)\colon \mu \to \sigma \times_\tau \rho$ that is the unique one to factorize p and q. Therefore, $(\sigma \times_\tau \rho, \pi_\rho, \pi_\sigma)$ is the pullback of h and g.

Roughly, the pullback $\sigma \times_\tau \rho$ is obtained as a sort of "lax intersection" of the "similar" vertices of σ and ρ (they are "similar" when they are sent to the same vertex in τ, by g_V and h_V respectively), and keeping whatever flags that are in the "intersection". So, $\sigma \times_\tau \rho$ is the maximal graph that is "compatible" with both σ and ρ.

The pullback construction lifts to directed, labeled, colored and ordered graphs.

4. MELL PROOF-STRUCTURES AND QUASI-PROOF-STRUCTURES

We present here a purely graphical definition of MELL proof-structures, following the non-inductive approach of [GPT19]: a MELL proof-structure R is essentially a labeled directed graph $|R|$ together with some additional information to identify its boxes. Our definition is completely based on standard notions (recalled in Section 3) from graph theory: it is formal

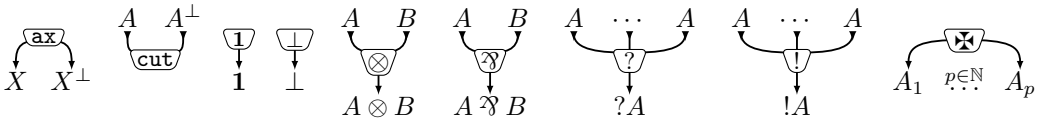


FIGURE 2. Cells, with their labels and their typed inputs and outputs.

(with an eye towards complete computer formalization) but avoids *ad hoc* technicalities to identify boxes. Indeed, the inductive and ordered structure of the boxes of R is recovered by means of a tree $\mathcal{A}_R$ 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. We use n-ary vertices of type ? collapsing weakening, dereliction and contraction (like in [DR95]). In this way, we get a *canonical* representation of MELL proof-structures with respect to operations like associativity and commutativity of contractions, or neutrality of weakening with respect to contraction.

4.1. **MELL formulas and lists.** Given a countably infinite set of propositional variables $X, Y, Z, \dots$, MELL *formulas* are defined by the following inductive grammar:

$$A, B ::= X \mid X^\perp \mid \mathbf{1} \mid \bot \mid A \otimes B \mid A \parr B \mid !A \mid ?A \qquad (\text{set: } \mathcal{F}orm_{\mathsf{MELL}})$$

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

For a list $\Gamma = (A_1, \dots, A_m)$ of MELL formulas, a *partition* of Γ is a list $\Gamma' = (\Gamma_1; \cdots; \Gamma_n)$ of lists of MELL formulas such that there are $0 = i_0 < \cdots < i_n = m$ with $\Gamma_j = (A_{i_{j-1}+1}, \dots, A_{i_j})$ for all $1 \leqslant j \leqslant n$; the partition Γ' of Γ is also denoted by $(A_1, \dots, A_{i_1}; \cdots; A_{i_{n-1}+1}, \dots, A_m)$, with lists separated by semicolons. The *flattening* of Γ' is Γ. The *empty list* is denoted by ε.

4.2. **Proof-structures.** We define here proof-structures corresponding to some subsystems and extensions 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 [Pag09, Tra11], our DiLL proof-structures are not allowed to contain a set of DiLL proof-structures inside a box. We reuse the syntax of proof-structures given in [GPT19], based on the graph notions introduced in Section 3. Note that, unlike [PT09], we allow the presence of cuts (vertices of type `cut`).

Definition 4.1 (module, proof-structure)**.** A (DiLL) *module* $M = (|M|, \ell, \mathsf{o}, \mathsf{c}, <)$ is a labeled (ℓ), directed (o), colored (c), ordered ($<$) graph $|M|$ such that:

- the map $\ell\colon V_{|M|} \to \{\mathtt{ax}, \mathtt{cut}, \mathbf{1}, \bot, \otimes, \parr, ?, !, \maltese\}$ associates with each vertex v its *type* $\ell(v)$;
- the map $\mathsf{c}\colon F_{|M|} \to \mathcal{F}orm_{\mathsf{MELL}}$ associates with each flag f its *type* $\mathsf{c}(f)$;
- $<$ is a strict order on the flags of $|M|$ that is total on the tails of $|M|$ and on the inputs of each vertex of type $\parr$ or $\otimes$;
- for every vertex $v \in V_{|M|}$,

– if $\ell(v) = \mathtt{cut}$, v has no output and 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 two outputs o_1 and o_2, with $\mathsf{c}(o_1) = \mathsf{c}(o_2)^\perp$ atomic;[8]
– 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 two inputs $i_1 < i_2$ and one output o, with $\mathsf{c}(o) = \mathsf{c}(i_1)\,\ell(v)\,\mathsf{c}(i_2)$;
– if $\ell(v) \in \{?, !\}$, v has $n \geqslant 0$ inputs $i_1, \dots, i_n$ and one output o, such that $\mathsf{c}(o) = \ell(v)\,\mathsf{c}(i_j)$ for all $1 \leqslant j \leqslant n$;[9]
– if $\ell(v) = \maltese$, v has no inputs and $p \geqslant 0$ outputs $o_1, \dots, o_p$.[10]

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

A (DiLL) *proof-structure* is a triple $R = (|R|, \mathcal{A}, \mathsf{box})$ 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 $\|R\| = (F_{\|R\|}, V_{\|R\|}, \partial_{\|R\|}, j_{\|R\|})$ is the *graph* of R);
- $\mathcal{A}$ is a rooted tree with no input tails, called the *box-tree* of R;[11]
- $\mathsf{box} = (\mathsf{box}_F, \mathsf{box}_V) \colon |R| \to \mathcal{A}^{\circlearrowleft}$ is a directed graph morphism,[12] called the *box-function* of R, such that box_F induces a partial bijection from the set $\bigcup_{v \in V_{\|R\|}, \ell(v) = !} \mathbf{in}_{|R|}(v)$ of inputs of the vertices of type ! in $|R|$ and the set of inputs flags in $\mathcal{A}$.[13] Also, for any vertex $v \in V_{\|R\|}$ and any input f of v, if $\mathsf{box}_V(\partial_{\|R\|} \circ j_{\|R\|}(f)) \neq \mathsf{box}_V(\partial_{\|R\|}(f))$ then $\ell(v) \in \{!, ?\}$.[14]

A (DiLL) proof-structure $R = (|R|, \mathcal{A}, \mathsf{box})$ is said:

(1) MELL if the structured graph $|R|$ has no vertices of type $\maltese$ and
  - all vertices in $|R|$ of type ! have exactly one input;
  - the partial bijection induced by box_F is total.[15]
(2) DiLL_0 (or *resource*) if $\mathcal{A}$ contains only the root with its output, and either $|R|$ has no vertices of type $\maltese$ or $|R|$ is a *daimon*, *i.e.* $|R|$ has only one vertex and it is of type $\maltese$.
(3) *Empty* (which is both DiLL_0 and MELL) if the structured graph $|R|$ and the box-tree $\mathcal{A}$ are empty graphs. It is denoted by ε.
(4) *Cut-free* if the structured graph $|R|$ has no vertices of type $\mathtt{cut}$, otherwise it is *with cuts*.

Our MELL proof-structures correspond to usual MELL proof-structures (as in [dCT12]). Our DiLL_0 proof-structures correspond to usual DiLL_0 proof-structures (as in [ER06b]) except

[8] We deal with *atomic axioms* only to simplify the presentation in the next sections. The general case with non-atomic axioms (*i.e.* $\mathsf{c}(o_1)$ and $\mathsf{c}(o_2)$ are still dual but not necessarily atomic) is discussed in Section 9.

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

[10] Note that there are not conditions on the number of outputs $o_1, \dots, o_p$ or on their types.

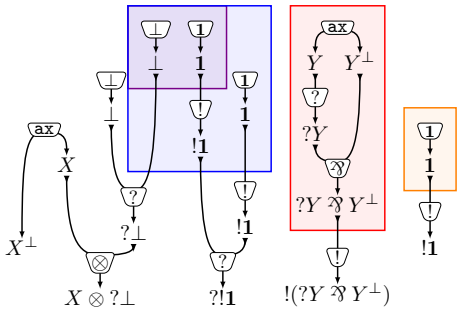
[11] Intuitively, $\mathcal{A}$ represents the tree-structure of the nested boxes of R, see Remark 4.2 below.

[12] 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 σ, is less structured than the other, say τ, 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 $\mathcal{A}^{\circlearrowleft}$.

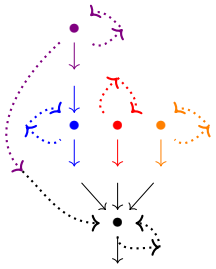
[13] It means that for any input flag f' in $\mathcal{A}$ there is exactly one input f of some vertex of type ! in $|R|$ such that $\mathsf{box}_F(f) = f'$; but, for any input f of some vertex of type ! in $|R|$, $\mathsf{box}_F(f)$ need not be an input flag in $\mathcal{A}$ (by definition of directed graph 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 (by definition of directed graph morphism, $\mathsf{box}_V(\partial_{\|R\|} \circ j_{\|R\|}(f)) \neq \mathsf{box}_V(\partial_{\|R\|}(f))$ for such a f, and if $f \neq f' \in F_{\|R\|}$ then $\mathsf{box}_F(f') \neq \mathsf{box}_F(f)$).

[14] Roughly, it says that the border of a box is made of inputs of vertices of type ! or ? in $|R|$.

[15] It means that for any input flag f' in $\mathcal{A}$ there is exactly one vertex of type ! in $|R|$ whose unique input f is such that $\mathsf{box}_F(f) = f'$; and, if f is the (only) input some vertex of type ! in $|R|$, $\mathsf{box}_F(f)$ is an input flag in $\mathcal{A}$. Intuitively, a box is associated with (and only with) the unique input of each vertex of type ! in $|R|$.



(A) The structured graph $|S|$.



(B) The box-tree $\mathcal{A}_S$ (without dotted lines) and the reflexive-transitive closure $\mathcal{A}_S^{\circlearrowleft}$ of $\mathcal{A}_S$ (with also dotted lines).

FIGURE 3. A MELL proof-structure $S = (|S|, \mathcal{A}_S, \mathsf{box}_S)$.

that we allow also (proof-structures that are) *daimons*. Daimons will be used in the Taylor expansion to deal with the content of a box taken 0 times (see Section 5). Our interest for DiLL proof-structures is just to have a unitary syntax subsuming both MELL and DiLL_0 without considering cut-elimination: for this reason, unlike [Pag09, Tra09, Tra11], our DiLL proof-structures are not allowed to contain a sum of DiLL proof-structure inside a box.[16]

Given a proof-structure $R = (|R|, \mathcal{A}, \mathsf{box})$, the output tails of $|R|$ are the *conclusions* of R. The *type* of R is the list of the types of these conclusions, ordered according to $<_{|R|}$. We often identify the conclusions of R with a finite initial segment of $\mathbb{N}$.

Borrowing the terminology from [ER06b, Laf90], in proof-structures, we speak of *cells* instead of vertices, and of *ℓ-cell* for a cell of type ℓ. An *hypothesis* is a cell without inputs.

Remark 4.2 (box)**.** In our syntax, boxes do not have explicit constructors or cells, hence boxes and depth of a proof structure $R = (|R|, \mathcal{A}_R, \mathsf{box}_R)$ are recovered in a non-inductive way.

Roughly, any non-root vertex v in $\mathcal{A}_R$ induces a subgraph of $\mathcal{A}_R^{\circlearrowleft}$ made up of all vertices "above v" with their inputs and outputs: the preimage of this subgraph through box_R is the *box* of v in $|R|$. The preimage of the root of $\mathcal{A}_R$ through box_R is the part of $|R|$ outside any box.

More precisely, with every flag f of $|R|$ such that $\mathsf{box}_{R_F}(f)$ is an input flag of $\mathcal{A}_R$[17] is associated a *box* B_f, that is the subgraph of $|R|$ (which is actually a proof-structure) made up of all the cells v (with their inputs and outputs) such that there is a directed path on $\mathcal{A}_R$ from $\mathsf{box}_{R_V}(v)$ to $\mathsf{box}_{R_V}(\partial_{\|R\|} \circ j_{\|R\|}(f))$ (note that f and the !-cell of which f is an input are not in B_f). A *conclusion* of such a box B_f is any output flag f' in B_f such that $\partial_{\|R\|} \circ j_{\|R\|}(f')$ is not in B_f. The tree-structure of $\mathcal{A}_R$ expresses the usual *nesting condition* of boxes: two boxes in $|R|$ are either disjoint or contained one in the other.

The *depth* of a cell v of R is the length of the directed 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.

[16] This restriction is without loss of generality, because a box with a sum inside is equivalent (see [Tra09]) to a co-contraction (a vertex of type ! with several inputs) of the boxes of each term of the sum. Also, the vertex of type ✠ corresponds to the empty sum inside a box: roughly, it is the content of an "empty box".

[17] By the constraints on box_R, this condition can be fulfilled only by inputs of !-cells 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, this condition is not fulfilled by any flag in $|R|$ (since $\mathcal{A}_R$ has no inputs) and so box_R is a directed graph morphism associating the root of $\mathcal{A}_R$ with any cell of R. Thus, in a DiLL_0 proof-structure $\rho = (|\rho|, \mathcal{A}_\rho, \mathsf{box}_\rho)$, there are no boxes, $\mathcal{A}_\rho$ and box_ρ do not induce any structure on $|\rho|$: ρ can be identified with $|\rho|$.

Example 4.3. In Figure 3 a MELL proof-structure $S = (|S|, \mathcal{A}_S, \mathsf{box}_S)$ is depicted. The box-function box_S is kept implicit by means of colors: the colored areas in $|S|$ represent boxes (the preimages of non-root vertices of $\mathcal{A}_S$ through box_S), and the same color is used on $\mathcal{A}_S$ to show where each box is mapped by box_S.

4.3. **Quasi-proof-structures.** We need to consider more general structures in order to accommodate our rewrite rules, as discussed in Section 2, p. 5 (the rewrite rules are defined in Sections 6 and 7). This is why we extend all the definitions to *tuples* of proof-structures.

Definition 4.4 (quasi-proof-structure)**.** A (DiLL) *quasi-proof-structure* is a tuple $R = (R_1, \dots, R_n)$ of proof-structures; for all $1 \leqslant i \leqslant n$, R_i is a *component* of R.

For convenience, given a proof-structure $R_i = (|R_i|, \mathcal{A}_{R_i}, \mathsf{box}_{R_i})$ for all $1 \leqslant i \leqslant n$, we denote the quasi-proof-structure $R = (R_1, \dots, R_n)$ as $R = (|R|, \mathcal{F}_R, \mathsf{box}_R)$, where $|R| = (|R_1|, \dots, |R_n|)$ is the *structured graph* of R, and $\mathcal{F}_R = (\mathcal{A}_{R_1}, \dots, \mathcal{A}_{R_n})$ is the *box-forest* of R, and $\mathsf{box}_R = (\mathsf{box}_{R_1}, \dots, \mathsf{box}_{R_n})$ is the *box-function* of R.

A *conclusion* of R is any conclusion of any component of R.

A quasi-proof-structure R is MELL (resp. DiLL_0) if all components of R are MELL (resp. DiLL_0) proof-structures.

The short notation $R = (|R|, \mathcal{F}_R, \mathsf{box}_R)$ for quasi-proof-structures makes sense because the structured graph $|R| = (|R_1|, \dots, |R_n|)$ can be seen as the disjoint union of its components $|R_1|, \dots, |R_n|$, and similarly for $\mathcal{F}$ and box_R. In particular, the box-forest $\mathcal{F}_R$ is the disjoint union of the box-trees $\mathcal{A}_{R_1}, \dots, \mathcal{A}_{R_n}$ of $R_1, \dots, R_n$. The box-function box_R locates not only the boxes on $|R|$, but also the different components of R on $|R|$ (see also Remark 4.6).

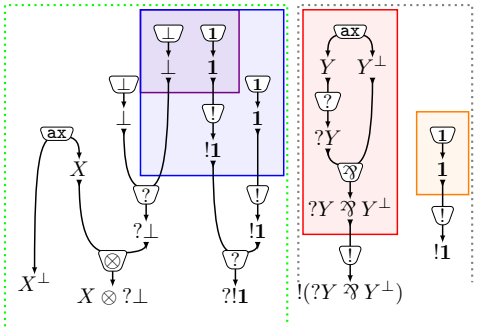
The only delicate point is the definition of the order $<_{|R|}$ for the conclusions of the quasi-proof-structure $R = (R_1, \dots, R_n)$, given the order $<_{|R_i|}$ for each component $|R_i|$. Given the conclusions f, f' of R, we set $f <_{|R|} f'$ if either f is a conclusion of R_i and f' is a conclusion of $R_{i'}$ with $i < i'$, or f and f' are conclusions of the same component R_j and $f <_{|R_j|} f'$.

We often identify the conclusions of R with a finite initial segment of $\mathbb{N}$, and its order.

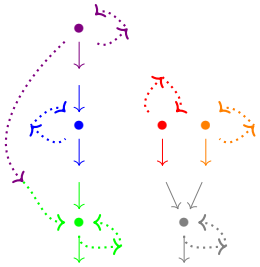
The *type* of a quasi-proof-structure $R = (R_1, \dots, R_n)$ is a list $\Gamma = (\Gamma_1, \dots, \Gamma_n)$ of lists of MELL formulas such that Γ_i is the type of R_i for all $1 \leqslant i \leqslant n$. When $n = 1$, we often identify R and R_1 (a quasi-proof-structure with only one component and a proof-structure), or Γ and Γ_1 (the type of a quasi-proof-structure with only one component and the type of a proof-structure). So, the type of a quasi-proof-structure R determines if R is a proof-structure.

Example 4.5. In Figure 4 a MELL quasi-proof-structure R is depicted. The colored areas represent the preimages of boxes, and each dashed box represents a component of R (in other examples, the absence of a dashed line means that there is only one component).

Remark 4.6 (components)**.** A subtle difference arises between DiLL_0 proof-structure and DiLL_0 quasi-proof-structure. Both have no boxes. In the former, its box-tree and box-function do not induce any structure on its structured graph, so we can identify a DiLL_0 proof-structure with its structured graph. In the latter, its box-forest and box-function do induce a structure on its structured graph: indeed, its box-forest is made up only of roots with their output and its box-function separates the components of its structured graph, by mapping the vertices to the roots. So, we cannot identify a DiLL_0 quasi-proof-structure with its structured graph.



(A) The structured graph $|R|$.



(B) The box-forest $\mathcal{F}_R$ (without dotted lines) and the reflexive-transitive closure $\mathcal{F}_R^{\circlearrowleft}$ of $\mathcal{F}_R$ (with also dotted lines).

FIGURE 4. A MELL quasi-proof-structure $R = (|R|, \mathcal{F}_R, \mathsf{box}_R)$.

5. The Taylor expansion

The *Taylor expansion* $\mathcal{T}(R)$ [ER08] of a MELL (or more in general DiLL) proof-structure R is a possibly infinite set of DiLL_0 proof-structures: roughly, each element of $\mathcal{T}(R)$ 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. Usually, the Taylor expansion of MELL proof-structure is defined globally and inductively [MP07, PT09]: with every MELL proof-structure R is directly associated its Taylor expansion (the whole set!) by induction on the depth of R.

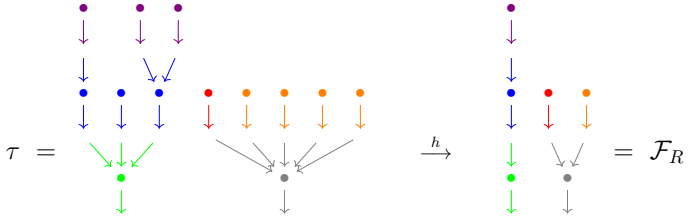
Following [GPT19], we adopt an alternative non-inductive approach, which strongly refines [GPT16]: the Taylor expansion is defined pointwise (see Example 5.2 and Figure 5). Indeed, proof-structures have a tree structure made explicit by their box-function. The definition of the Taylor expansion uses this tree structure: first, we define how to "*expand*" a tree via the notion of thick subtree [Bou09] (Definition 5.1; roughly, it states the number of copies of each box to be taken, recursively), we then take all the expansions of the tree structure of a proof-structure and we *pull* the approximations *back* to the underlying graphs (Definition 5.4), finally we *forget* the tree structures associated with them (Definition 5.5).

An advantage of our approach is that it smoothly generalizes to quasi-proof-structures.

Definition 5.1 (thick subtree and subforest)**.** Let σ be a rooted tree. A *thick subtree* of σ is a pair (τ, h) of a rooted tree τ and a directed graph morphism $h = (h_F, h_V) \colon \tau \to \sigma$.

Let $\sigma = (\sigma_1, \dots, \sigma_n)$ be a forest of rooted trees. A *thick subforest* of σ is a tuple $((\tau_1, h_1), \dots, (\tau_n, h_n))$ where (τ_i, h_i) is a thick subtree of σ_i for all $1 \leqslant i \leqslant n$. It is denoted by (τ, h), where $\tau = (\tau_1, \dots, \tau_n)$ and $h = (h_1, \dots, h_n)$.

Example 5.2. The following is a graphical presentation of a thick subforest (τ, h) of the box-forest $\mathcal{F}_R$ of the quasi-proof-structure in Figure 4, where the directed graph morphism $h = (h_F, h_V) \colon \tau \to \mathcal{F}_R$ is depicted chromatically (same color means same image via h).



Intuitively, it means that τ is obtained from $\mathcal{F}$ by taking 3 copies of the blue box, 1 copy of the red box and 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.

Remark 5.3 (roots)**.** If (τ, h) is a thick subforest of a forest σ of rooted trees, h establishes a bijection between the roots of τ and the roots of σ, by definition of directed graph morphism.

The crucial point is to pull back the expansion of a forest to quasi-proof-structures.

Definition 5.4 (proto-Taylor expansion)**.** Let $R = (|R|, \mathcal{F}_R, \mathsf{box}_R)$ be a quasi-proof-structure. The *proto-Taylor expansion* of R is the set $\mathcal{T}^{\mathsf{proto}}(R)$ of thick subforests of $\mathcal{F}_R$.

Let $t = (\tau_t, h_t) \in \mathcal{T}^{\mathsf{proto}}(R)$. The *t-expansion* of R is the pullback (R_t, p_t, p_R) below, computed in the category of directed graphs and directed graph morphisms.[18]

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

Given a quasi-proof-structure R and $t = (\tau_t, h_t) \in \mathcal{T}^{\mathsf{proto}}(R)$, the directed graph R_t inherits the types of its vertices and flags by pre-composition of $\ell_{|R|}$ and $\mathsf{c}_{|R|}$ with the graph morphism $p_R \colon R_t \to |R|$. The order on the flags of R_t is induced by the one on $|R|$ via p_R.

Let $[\tau_t]$ be the forest made up of the roots of τ_t and $\iota \colon \tau_t \to [\tau_t]$ be the graph morphism sending each vertex of τ_t to the root below it; $\iota^{\circlearrowleft}$ induces by post-composition a morphism $\overline{h_t} = \iota^{\circlearrowleft} \circ p_t \colon R_t \to [\tau_t]^{\circlearrowleft}$. The triple $(R_t, [\tau_t], \overline{h_t})$ is a DiLL_0 quasi-proof-structure, and it is a DiLL_0 proof-structure if R is a proof-structure. We can now define the *Taylor expansion* $\mathcal{T}(R)$ of a quasi-proof-structure R (an example of an element of a Taylor expansion is in Figure 5).

Definition 5.5 (Taylor expansion)**.** Let R be a quasi-proof-structure. The *Taylor expansion* of R is the set of DiLL_0 quasi-proof-structures $\mathcal{T}(R) = \{(R_t, [\tau_t], \overline{h_t}) \mid t = (\tau_t, h_t) \in \mathcal{T}^{\mathsf{proto}}(R)\}$.

An element $(R_t, [\tau_t], \overline{h_t})$ of the Taylor expansion of a quasi-proof-structure R has much less structure than the pullback (R_t, p_t, p_R): the latter indeed is a DiLL_0 quasi-proof-structure R_t coming with its projections $|R| \xleftarrow{p_R} R_t \xrightarrow{p_t} \tau_t^{\circlearrowleft}$, which establish a precise correspondence between cells and flags of R_t and cells and flags of R: a cell in R_t is labeled (via the projections) by both the cell of $|R|$ and the branch of the box-forest 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.

Remark 5.6 (conclusions)**.** From the definition and Remark 5.3, it follows the Taylor expansion preserves conclusions and type: each element of the Taylor expansion of a quasi-proof-structure R has the *same conclusions* and the *same type* as R. More precisely, there is a bijection φ from the conclusions of a quasi-proof-structure R to the ones in each element ρ of $\mathcal{T}(R)$ such that i and $\varphi(i)$ have the same type and the same root (*i.e.* $\mathsf{box}_R(i) = \mathsf{box}_\rho(\varphi(i))$ up to the bijection of Remark 5.3). So, the types of R and ρ are the same (as a list of lists).

[18] So, $|R|$ is considered as directed graph (see Footnote 12), forgetting that it is colored, labeled, ordered.

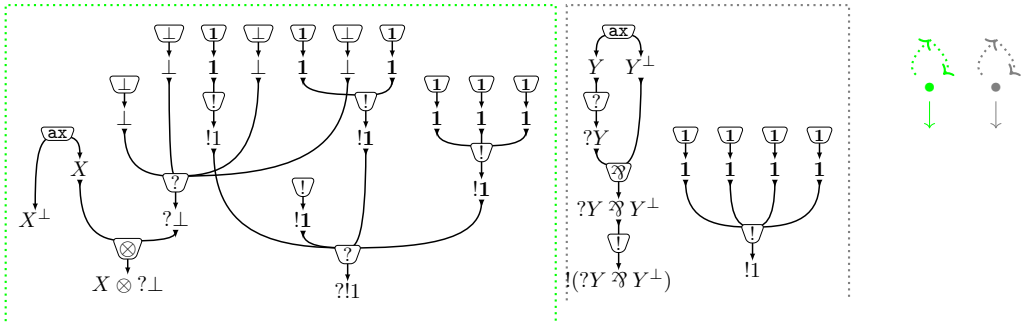


FIGURE 5. The element ρ of the Taylor expansion of the MELL quasi-proof-structure R in Figure 4 obtained from the element of $\mathcal{T}^{\mathsf{proto}}(R)$ in Example 5.2.

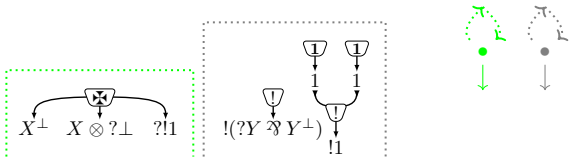


FIGURE 6. An element of the filled Taylor expansion of the MELL quasi-proof-structure R in Figure 4, obtained as an emptying of ρ in Figure 5.

5.1. **The filled Taylor expansion.** As discussed in Section 2 (p. 6), the rewriting rules we will introduce in Section 6 need to "represent" the emptiness introduced by the Taylor expansion (taking 0 copies of a box) so as to preserve the conclusions. Thus, an element of the *filled Taylor expansion* $\mathcal{T}^{\Pi}(R)$ of a quasi-proof-structure R (an example is in Figure 6) is obtained from an element of $\mathcal{T}(R)$ where some components can be erased and replaced by daimons with the same conclusions (hence $\mathcal{T}(R) \subseteq \mathcal{T}^{\Pi}(R)$). The filled Taylor expansion (and not the plain one) will play a crucial role in Section 7 to define a natural transformation.

Definition 5.7 (emptying, filled Taylor expansion)**.** The *emptying* of a DiLL_0 proof-structure ρ is a daimon whose conclusions and type are the same as ρ.

An *emptying* of a DiLL_0 *quasi*-proof-structure ρ is a DiLL_0 quasi-proof-structure obtained from ρ by replacing some components of ρ (possibly none) with their emptying.

The *filled Taylor expansion* $\mathcal{T}^{\Pi}(R)$ of a quasi-proof-structure R is the set of all the emptyings of every element of its Taylor expansion $\mathcal{T}(R)$.

If $\rho = (R_t, [\tau_t], \overline{h_t}) \in \mathcal{T}(R)$, and $r_1, \dots, r_n$ are some roots of $[\tau_t]$, the *emptying of ρ on $r_1, \dots, r_n$*, denoted by $\rho_{r_1,\dots,r_n}$, is the element of $\mathcal{T}^{\Pi}(R)$ obtained by substituting a daimon for each component of ρ corresponding to the root r_i, for all $1 \leqslant i \leqslant n$.

Remark 5.8 (conclusions of emptying)**.** By construction, the filled Taylor expansion preserves conclusions and type, as the Taylor expansion (Remark 5.6): the conclusions and type of a quasi-proof-structure R and of *any* emptying of *any* element of $\mathcal{T}(R)$ are the same.

Remark 5.9 (Taylor of the empty)**.** For the empty proof-structure ε, $\mathcal{T}(\varepsilon) = \mathcal{T}^{\Pi}(\varepsilon) = \{\varepsilon\}$.

Remark 5.10 (connection)**.** The Taylor expansion does not "create connection", the filled Taylor expansion can "create connection" only in the component that has been filled by a daimon. More precisely, let i, j be the conclusions of a quasi-proof-structure $R = (|R|, \mathcal{F}_R, \mathsf{box}_R)$

$$
\begin{aligned}
(\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i),\mathsf{c}(i{+}1),\Gamma'_k;\cdots;\Gamma_n) &\xrightarrow{\mathtt{exc}_i} (\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i{+}1),\mathsf{c}(i),\Gamma'_k;\cdots;\Gamma_n)\\
(\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i),\mathsf{c}(i{+}1),\Gamma'_k;\cdots;\Gamma_n) &\xrightarrow{\mathtt{mix}_i} (\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i);\mathsf{c}(i{+}1),\Gamma'_k;\cdots;\Gamma_n)\\
(\Gamma_1;\cdots;\Gamma_k;\mathsf{c}(i),\mathsf{c}(i{+}1);\Gamma_{k+2};\cdots;\Gamma_n) &\xrightarrow{\mathtt{ax}_i} (\Gamma_1;\cdots;\Gamma_k;\Gamma_{k+2};\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=\mathsf{c}(i{+}1)^\perp\\
(\Gamma_1;\cdots;\Gamma_k;\cdots;\Gamma_n) &\xrightarrow{\mathtt{cut}^i} (\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i),\mathsf{c}(i{+}1);\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=\mathsf{c}(i{+}1)^\perp\\
(\Gamma_1;\cdots;\Gamma_k;\mathsf{c}(i);\Gamma_{k+2};\cdots;\Gamma_n) &\xrightarrow{\mathbf{1}_i} (\Gamma_1;\cdots;\Gamma_k;\Gamma_{k+2};\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=\mathbf{1}\\
(\Gamma_1;\cdots;\Gamma_k;\mathsf{c}(i);\Gamma_{k+2};\cdots;\Gamma_n) &\xrightarrow{\perp_i} (\Gamma_1;\cdots;\Gamma_k;\Gamma_{k+2};\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=\perp\\
(\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i);\cdots;\Gamma_n) &\xrightarrow{\otimes_i} (\Gamma_1;\cdots;\Gamma_k,A,B;\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=A\otimes B\\
(\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i);\cdots;\Gamma_n) &\xrightarrow{\parr_i} (\Gamma_1;\cdots;\Gamma_k,A,B;\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=A\parr B\\
(\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i);\cdots;\Gamma_n) &\xrightarrow{?_{c_i}} (\Gamma_1;\cdots;\Gamma_k,?A,?A;\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=?A\\
(\Gamma_1;\cdots;\Gamma_k,\mathsf{c}(i);\cdots;\Gamma_n) &\xrightarrow{?_{d_i}} (\Gamma_1;\cdots;\Gamma_k,A;\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=?A\\
(\Gamma_1;\cdots;\Gamma_k;\mathsf{c}(i);\Gamma_{k+2};\cdots;\Gamma_n) &\xrightarrow{?_{w_i}} (\Gamma_1;\cdots;\Gamma_k;\Gamma_{k+2};\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=?A\\
(\Gamma_1;\cdots;?\Gamma_k,\mathsf{c}(i);\cdots;\Gamma_n) &\xrightarrow{\mathsf{Box}_i} (\Gamma_1;\cdots;?\Gamma_k,A;\cdots;\Gamma_n) \text{ with } \mathsf{c}(i)=!A
\end{aligned}
$$

FIGURE 7. The generators of **Sched**. In the source Γ of each arrow, $\mathsf{c}(i)$ is the i^{th} formula in the flattening of Γ (the arrow keeps track of i).

and let $\rho = (|\rho|, \mathcal{F}_\rho, \mathsf{box}_\rho) \in \mathcal{T}(R)$ and $\rho_{r_1,\dots,r_n}$ be the emptying of ρ on the roots $r_1,\dots,r_n$. If i and j are not connected in the (undirected) graph $\|R\|$ of R, then i and j are not connected in the (undirected) graph $\|\rho\|$ of ρ. And if i and j are connected in the (undirected) graph of $\|\rho_{r_1,\dots,r_n}\|$ of $\rho_{r_1,\dots,r_n}$, then i and j are output tails of the same ✠-cell in $\rho_{r_1,\dots,r_n}$.

6. Means of destruction: unwinding MELL quasi-proof-structures

Our aim is to deconstruct (MELL or DiLL$_0$) proof-structures from their conclusions. To do that, we introduce the category of schedulings. The arrows of this category are sequences of deconstructing rules, acting on lists of lists of MELL formulas. These arrows act through functors on quasi-proof-structures, exhibiting their sequential structure.

Definition 6.1 (the category **Sched**)**.** Let **Sched** be the category of schedulings whose

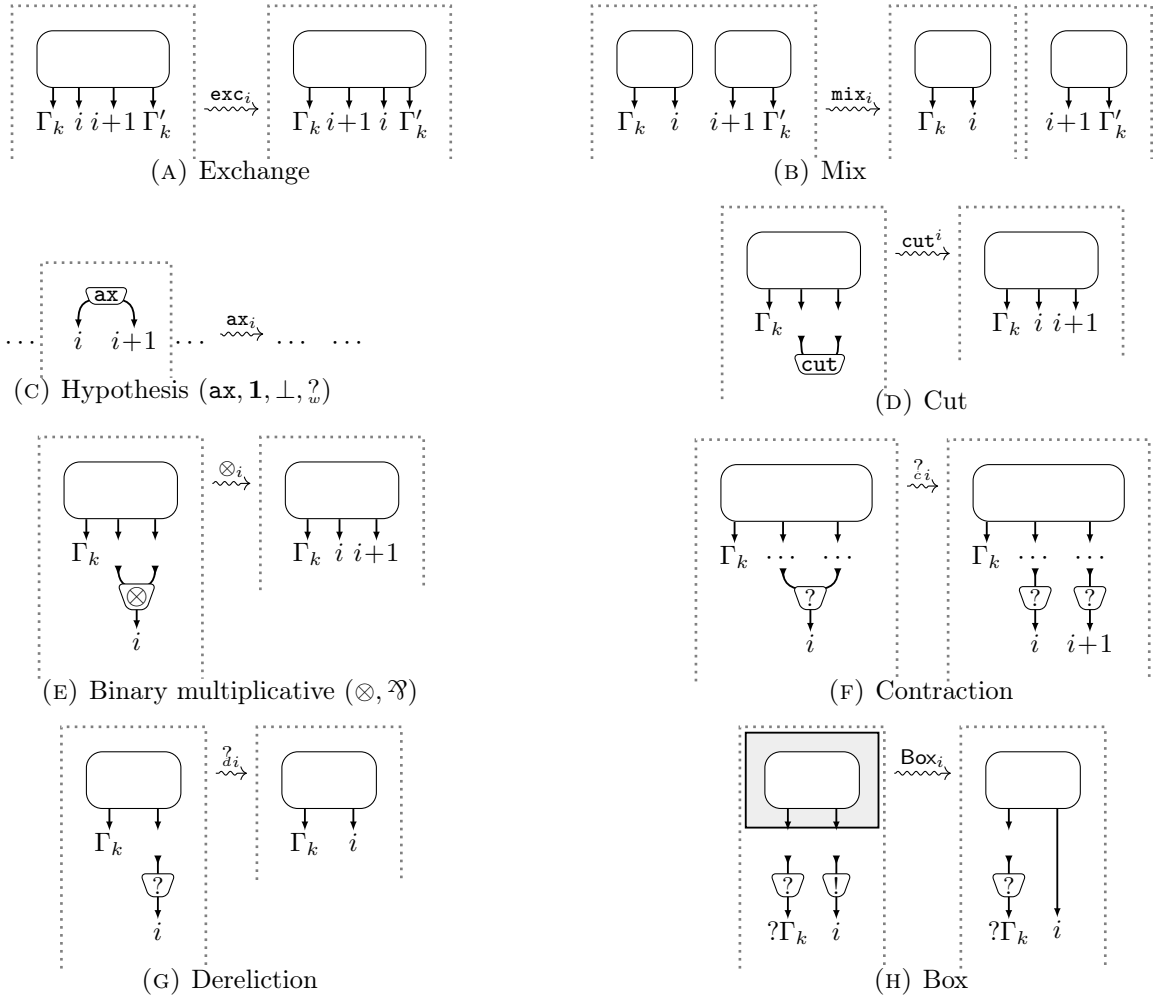
- objects are lists $\Gamma = (\Gamma_1;\cdots;\Gamma_n)$ of lists of MELL formulas;
- arrows are freely generated by composition of the *elementary schedulings* in Figure 7; identities are the empty sequence of elementary schedulings.

We call a *scheduling* any arrow $\xi\colon \Gamma \to \Gamma'$. We write the composition of schedulings by juxtaposition in the diagrammatic order; so, if $\xi\colon \Gamma \to \Gamma'$ and $\xi'\colon \Gamma' \to \Gamma''$, then $\xi\xi'\colon \Gamma \to \Gamma''$.

Example 6.2. $\parr_1\ \parr_2\ \parr_3\ \otimes_1\ \otimes_3\ \mathtt{exc}_1\ \mathtt{exc}_2\ \mathtt{mix}_2\ \mathtt{ax}_1\ \mathtt{exc}_2\ \mathtt{mix}_2\ \mathtt{ax}_1\ \mathtt{ax}_1$ is a scheduling from $\big((X\otimes Y^\perp)\parr((Y\otimes Z^\perp)\parr(X^\perp\parr Z))\big)$ to the empty list ε of lists of MELL formulas.

The category **Sched** acts on MELL quasi-proof-structures, exhibiting a sequential structure in their construction. For Γ a list of lists of MELL formulas, qMELL(Γ) is the set of MELL quasi-proof-structures of type Γ. To ease the reading of the rewrite rules acting on a MELL quasi-proof-structure R, we only draw the relevant component of R and omit the components left unchanged; *e.g.*, if we consider an ax-cell whose outputs are the conclusions i and $i{+}1$, and it is the only cell in a component, we write 

ignoring the rest of R.



(A) Exchange (B) Mix (C) Hypothesis ($\mathtt{ax}, \mathbf{1}, \bot, ?_w$) (D) Cut (E) Binary multiplicative ($\otimes, \parr$) (F) Contraction (G) Dereliction (H) Box

FIGURE 8. The action of elementary schedulings on MELL quasi-proof-structures.

Definition 6.3 (action of schedulings on MELL quasi-proof-structures)**.** An elementary scheduling $a\colon \Gamma \to \Gamma'$ defines a relation $\overset{a}{\rightsquigarrow} \subseteq \mathsf{qMELL}(\Gamma) \times \mathsf{qMELL}(\Gamma')$, called the *action* of a, as the smallest relation containing all the cases in Figure 8, with the following remarks:

exchange: $\overset{\mathtt{exc}_i}{\rightsquigarrow}$ swaps the order of two consecutive conclusions in the same component.

mix: read in reverse, $\overset{\mathtt{mix}_i}{\rightsquigarrow}$ relates a quasi-proof-structure with two components with a quasi-proof-structure with the same module but the two roots of such components merged.

hypothesis: if $a \in \{\mathtt{ax}_i, \mathbf{1}_i, \bot_i, ?_{w\,i}\}$, $\overset{a}{\rightsquigarrow}$ deletes a cell without inputs that is the only cell in its component. We have drawn the axiom case in Figure 8c, the others vary only by their number of conclusions.

cut: read in reverse, $\overset{\mathtt{cut}^i}{\rightsquigarrow}$ relates a quasi-proof-structure with two conclusions i and $i+1$ with the quasi-proof-structure where these two conclusions are cut by a $\mathtt{cut}$-cell of depth 0. This rule, from left to right, is nondeterministic (as there are many possible cuts).

binary multiplicatives: if $a \in \{\otimes_i, \parr_i\}$, $\overset{a}{\rightsquigarrow}$ deletes a cell labeled by a binary multiplicative connective, $\otimes$ or $\parr$. We have only drawn the $\otimes$ case in Figure 8e, the $\parr$ case is similar.

contraction: $\overset{?}{\underset{c\,i}{\rightsquigarrow}}$ splits a ?-cell with $h+k$ inputs into two ?-cells (so, it duplicates a ?-cell) with h and k inputs, respectively ($h, k \geqslant 0$). The rule, from left to right, is nondeterministic.

dereliction: $\overset{?}{\underset{d\,i}{\rightsquigarrow}}$ only applies if the ?-cell (with 1 input) does not shift a level in the box-forest, *i.e.* it is not just outside a box (otherwise $\overset{?}{\underset{d\,i}{\rightsquigarrow}}$ would not yield a MELL quasi-proof-structure).

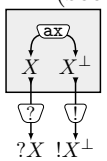
box: $\overset{\mathsf{Box}_i}{\rightsquigarrow}$ only applies if a box (and its border) is alone in its component.

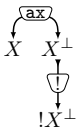
The rewrite relation is extended by composition of relations to define, for any scheduling $\xi\colon \Gamma \to \Gamma'$, a relation $\overset{\xi}{\rightsquigarrow} \subseteq \mathsf{qMELL}(\Gamma) \times \mathsf{qMELL}(\Gamma')$, called the *action* of ξ.[19]

Except $\mathtt{mix}_i$, $\mathtt{exc}_i$ and $\underset{c\,i}{?}$, the action of an elementary scheduling is a rewrite rule on quasi-proof-structures that destroys either a $\mathtt{cut}$-cell of depth 0 or a cell whose output is a conclusion. The action of a scheduling is just a composition of these rewrite rules.

Among the actions of elementary schedulings in Figure 8, the cases mix, box and hypothesis are the only ones that change the box-forest of a quasi-proof-structure: mix splits a tree in two distinct trees by splitting its root, box merges a root of a tree with a non-root vertex just above it, while hypothesis discards a tree made up only of a root with its output. For instance, the action $\mathtt{mix}_i$ rewrites the MELL proof-structure S in Figure 3 into the MELL quasi-proof-structure R in Figure 4, when i is the conclusion of S of type $?!\mathbf{1}$.

The way the action of an elementary scheduling is defined (Definition 6.3) automatically rules out the possibility that a MELL quasi-proof-structure rewrites to a module that is not a MELL quasi-proof-structure. For instance, the elementary scheduling $\underset{d\,1}{?}$ cannot act on the MELL proof-structure Q below, because it would yield a module Q' that is not a MELL quasi-proof-structure, as in Q' it would be impossible to define a box-function that fulfills the constraints of Definition 4.1 for MELL (see Footnote 14); Q' is just a DiLL_0 proof-structure.

$Q =$ 

$\overset{?}{\underset{d\,1}{\rightsquigarrow}}$ 

$= Q'$

Actions of schedulings can been seen as arrows in the category **Rel** of sets and relations.

Definition 6.4 (functor qMELL)**.** We define a functor $\mathsf{qMELL}\colon \mathbf{Sched} \to \mathbf{Rel}$ by:

- *on objects:* $\mathsf{qMELL}(\Gamma)$ is the set of MELL quasi-proof-structures of type Γ;
- *on arrows:* for any $\xi\colon \Gamma \to \Gamma'$, $\mathsf{qMELL}(\xi)$ is $\overset{\xi}{\rightsquigarrow}\colon \mathsf{qMELL}(\Gamma) \to \mathsf{qMELL}(\Gamma')$ (Definition 6.3).

Lemmas 6.5 and 6.6 and Proposition 6.7 express some properties of our rewrite rules.

Lemma 6.5 (co-functionality)**.** *Let $\xi\colon \Gamma \to \Gamma'$ be a scheduling. The action $\overset{\xi}{\rightsquigarrow}$ of ξ is a co-function from $\mathsf{qMELL}(\Gamma)$ to $\mathsf{qMELL}(\Gamma')$, that is, a function $\overset{\xi}{\rightsquigarrow}^{\mathrm{op}}\colon \mathsf{qMELL}(\Gamma') \to \mathsf{qMELL}(\Gamma)$.*

Given the MELL quasi-proof-structures R and R', we say that an elementary scheduling a *applies* to R if there are elementary schedulings $\mathtt{exc}_{i_1}, \dots, \mathtt{exc}_{i_n}$ such that $R \overset{\mathtt{exc}_{i_1} \cdots \mathtt{exc}_{i_n} a}{\rightsquigarrow} R'$.

[19] More explicitly, composition of actions means that, given the schedulings $\xi\colon \Gamma \to \Gamma'$ and $\xi'\colon \Gamma' \to \Gamma''$ and the quasi-proof-structure $R \in \mathsf{qMELL}(\Gamma)$ and $R'' \in \mathsf{qMELL}(\Gamma'')$, $R \overset{\xi\xi'}{\rightsquigarrow} R''$ if and only if there exists $R' \in \mathsf{qMELL}(\Gamma')$ such that $R \overset{\xi}{\rightsquigarrow} R'$ and $R' \overset{\xi'}{\rightsquigarrow} R''$ (often denoted by $R \overset{\xi}{\rightsquigarrow} \overset{\xi'}{\rightsquigarrow} R''$).

Lemma 6.6 (applicability of actions)**.** *Let R be a non-empty* MELL *quasi-proof-structure. Then an elementary scheduling $a \in \{\mathtt{mix}_i, \mathtt{ax}_i, \mathbf{1}_i, \bot_i, \otimes_i, \parr_i, ?_{ci}, ?_{di}, ?_{wi}, \mathtt{cut}^i, \mathsf{Box}_i\}$ applies to R, for some conclusion i of R (or some input i of a* cut*-cell of depth 0 in R). And if $a = ?_{ci}$ then:*

- *either i is the output of a ?-cell without inputs, in which case also $\mathtt{mix}_i$ or $?_{wi}$ applies to R;*
- *or i is the output of a ?-cell with exactly one input, in which case also $?_{di}$ applies to R;*
- *or $?_{ci}$ applies to R by splitting a ?-cell into two ?-cells with at least one input each.*

No elementary scheduling applies to ε.

Lemmas 6.5 and 6.6 are proven by simple inspection of the rewrite rules of Figure 8. In the proof of Lemma 6.5, the case $\xi = ?_{ci}$ is slightly delicate: given $R' \in \mathsf{qMELL}(\Gamma')$, there is (a unique) $R \in \mathsf{qMELL}(\Gamma)$ such that $R \overset{?_{ci}}{\rightsquigarrow} R'$ because any conclusion of R' of type $?A$ is the output of a ?-cell (thanks to atomic axioms, Footnote 8; for non-atomic axioms, see Section 9).

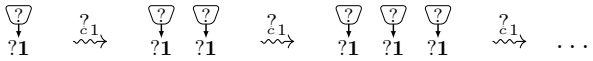
Proposition 6.7 (confluent termination)**.** *Let R be a* MELL *quasi-proof-structure of type Γ. Then, there exists a scheduling $\xi \colon \Gamma \to \varepsilon$ such that $R \overset{\xi}{\rightsquigarrow} \varepsilon$.*

Proof. By Lemma 6.6, it is enough show that the size of MELL quasi-proof-structures is left unchanged by $\overset{\mathtt{exc}_i}{\rightsquigarrow}$ and strictly decreases for each other action in Figure 8 (and for $\overset{?_{ci}}{\rightsquigarrow}$, we can assume that it splits a ?-cell into two with at least one input each), according to the following definition of size. The *size* of a quasi-proof-structure R is the triple (p, q, r) where:

- p is the (finite) multiset of the number of inputs of each ?-cell in R;
- q is the number of cells in R;
- r is the (finite) multiset of the number of conclusions of each component of R.

Multisets are well-ordered as usual, triples are well-ordered lexicographically. □

Actions of schedulings define a confluent normalizing rewrite system on MELL quasi-proof-structures (Proposition 6.7) that is not strongly normalizing, because of the actions of $\mathtt{exc}_i$ and, more significantly, $?_{ci}$. An example of a rewriting that can be extended indefinitely is below.

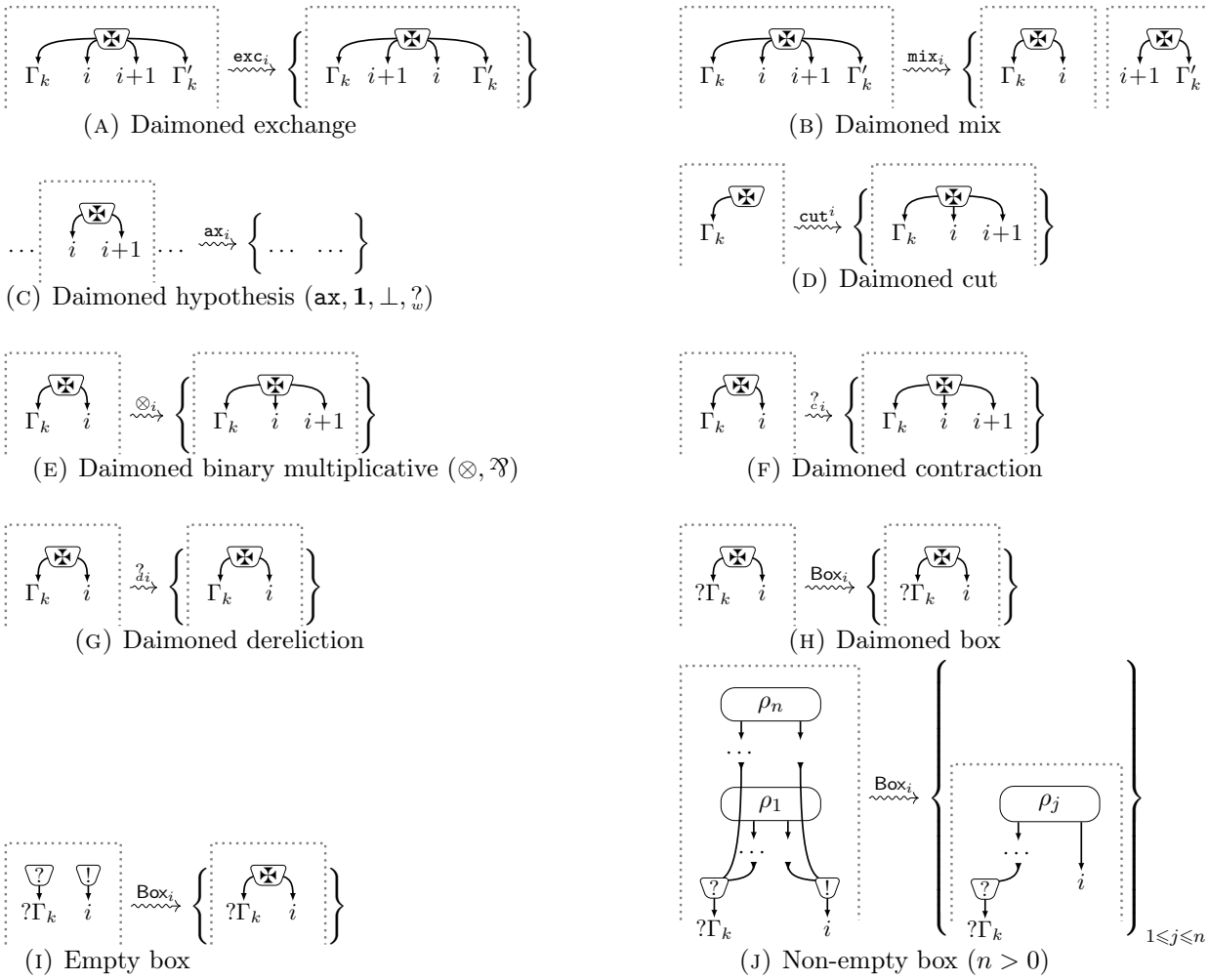


The action of a scheduling ξ such that $R \overset{\xi}{\rightsquigarrow} \varepsilon$, read backward, can be seen as a sequence of elementary steps to build the MELL quasi-proof-structure R from the empty one ε. A detailed example of a rewriting from some R to ε is in Figure 11 (Section 8). Intuitively, a scheduling $\xi \colon (\Gamma) \to \varepsilon$ "encodes" a way to build a MELL proof-structure of type Γ.

7. Naturality of unwinding DiLL_0 quasi-proof-structures

We show here how the actions of schedulings act on DiLL_0 quasi-proof-structures, mimicking the behavior of the actions of schedulings on MELL quasi-proof-structures seen in Section 6. Schedulings are the same, the novelty is that DiLL_0 quasi-proof-structures have no boxes and might have daimons. The bridge between the MELL and DiLL_0 frameworks is given by the filled Taylor expansion, which actually defines a *natural transformation* (Theorem 7.4).

For Γ a list of lists of MELL formulas, $\mathsf{qDiLL}_0(\Gamma)$ is the set of DiLL_0 quasi-proof-structures of type Γ. For any set X, its powerset is denoted by $\mathfrak{P}(X)$.



(A) Daimoned exchange (B) Daimoned mix

(C) Daimoned hypothesis ($\mathbf{ax}, \mathbf{1}, \bot, ?_w$) (D) Daimoned cut

(E) Daimoned binary multiplicative ($\otimes, \parr$) (F) Daimoned contraction

(G) Daimoned dereliction (H) Daimoned box

(I) Empty box (J) Non-empty box ($n > 0$)

FIGURE 9. The action of elementary schedulings on DiLL_0 quasi-proof-structures.

Definition 7.1 (action of schedulings on DiLL_0 quasi-proof-structures)**.** An elementary scheduling $a\colon \Gamma \to \Gamma'$ defines a relation $\overset{a}{\rightsquigarrow} \subseteq \mathsf{qDiLL}_0(\Gamma) \times \mathfrak{P}(\mathsf{qDiLL}_0(\Gamma'))$, called the *action* of a, as the smallest relation containing all the rules in Figure 8—except Figure 8h, and with the same remarks—and in Figure 9. We extend it to a relation on $\mathfrak{P}(\mathsf{qDiLL}_0(\Gamma)) \times \mathfrak{P}(\mathsf{qDiLL}_0(\Gamma'))$ by the multiplication of the monad powerset $X \mapsto \mathfrak{P}(X)$.[20] The rewrite relation on $\mathfrak{P}(\mathsf{qDiLL}_0(\Gamma)) \times \mathfrak{P}(\mathsf{qDiLL}_0(\Gamma'))$ is extended by composition of relations to define, for any scheduling $\xi\colon \Gamma \to \Gamma'$, a relation $\overset{\xi}{\rightsquigarrow}\colon \mathfrak{P}(\mathsf{qDiLL}_0(\Gamma)) \to \mathfrak{P}(\mathsf{qDiLL}_0(\Gamma'))$, called the *action* of ξ.

Roughly, all the rewrite rules in Figure 9—except Figures 9i and 9j—mimic the behavior of the corresponding rule in Figure 8 using a ✠-cell. The action in Figure 9i creates a ✠-cell, whose outputs are all the conclusions of the component: if we just erased a co-weakening (a !-cell without inputs) with output of type $!A$, we would lose the information about the type

[20] More explicitly, given $\Pi \subseteq \mathsf{qDiLL}_0(\Gamma)$ and $\Pi' \subseteq \mathsf{qDiLL}_0(\Gamma')$, $\Pi \overset{a}{\rightsquigarrow} \Pi'$ means that $\Pi' = \bigcup\{\rho' \subseteq \mathsf{qDiLL}_0(\Gamma') \mid \exists\, \rho \in \Pi : \rho \overset{a}{\rightsquigarrow} \rho'\}$. Thus, $\rho \overset{a}{\rightsquigarrow} \rho' \subseteq \Pi'$ for every $\rho \in \Pi$. And, in particular, $\emptyset \overset{a}{\rightsquigarrow} \emptyset$. Roughly, the same rewrite rule applies to the same conclusion of each element of a set of DiLL_0 quasi-proof-structures.

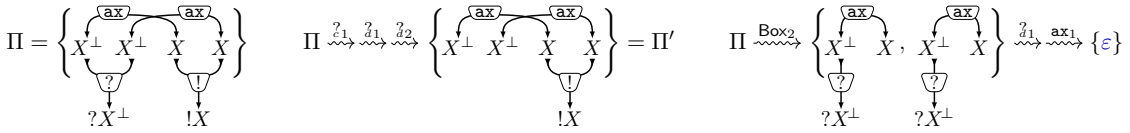


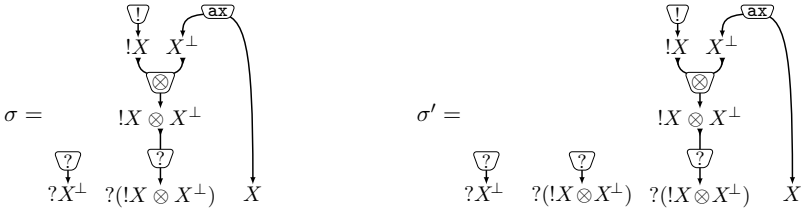
FIGURE 10. Examples of action of schedulings for DiLL_0.

A; this is why a ✠-cell is needed. Intuitively, a co-weakening represents a box taken 0 times, so there is no information about the content of the box, apart from its type. The daimon is a "universal" DiLL_0 proof-structure representing information plainly missing.

The action in Figure 9j requires that, on the left of $\overset{\mathsf{Box}_i}{\rightsquigarrow}$, ρ_j is not connected to $\rho_{j'}$ for $j \neq j'$, except for the !-cell and the ?-cells in the conclusions. Read in reverse, the rule associates with a non-empty finite set of DiLL_0 quasi-proof-structures $\{\rho_1, \dots, \rho_n\}$ the merging of $\rho_1, \dots, \rho_n$, that is the DiLL_0 quasi-proof-structure depicted on the left of $\overset{\mathsf{Box}_i}{\rightsquigarrow}$. Intuitively, $\rho_1, \dots, \rho_n$ represent $n > 0$ possible copies of a box, which then have to be analyzed "in parallel": this is why the action rewrites to a *set* of DiLL_0 quasi-proof-structures.

Note that a ✠-cell can be erased only by the action of elementary scheduling called hypothesis in Figure 9: $\mathtt{ax}_i, \mathbf{1}_i, \bot_i, ?_{wi}$ (Figure 9c).

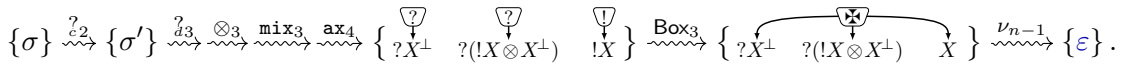
Example 7.2. Consider the DiLL_0 proof-structures σ and σ' below.



For every $n > 0$, the singleton $\{\sigma\}$ rewrites to $\{\varepsilon\}$ by the action of the scheduling $\nu_n \colon (?X^\perp, ?(!X \otimes X^\perp), X) \to \varepsilon$, defined inductively by:

$$\nu_1 = ?_{d\,2} \otimes_2 \mathtt{mix}_2\ \mathtt{ax}_3\ \mathsf{Box}_2\, ?_{d\,1}\, \mathtt{ax}_1 \qquad \nu_{n+1} = ?_{c\,2}\, ?_{d\,3} \otimes_3 \mathtt{mix}_3\ \mathtt{ax}_4\ \mathsf{Box}_3\, \nu_n\,.$$

In particular, for every $n > 1$, the scheduling ν_n acts in the following way:



The rewriting for DiLL_0 quasi-proof-structures behaves differently than in MELL quasi-proof-structures. In DiLL_0, Lemma 6.6 and Proposition 6.7 are false. For instance, in Figure 10 no elementary scheduling applies to the singleton Π' of a DiLL_0 proof-structure. Note that $\Pi \overset{?_{c\,1}}{\rightsquigarrow} \overset{?_{d\,1}}{\rightsquigarrow} \overset{?_{d\,2}}{\rightsquigarrow} \Pi'$, but there is a scheduling whose action rewrites Π to $\{\varepsilon\}$ (see Figure 10 on the right), so the rewriting in DiLL_0 is not confluent and need not terminate in $\{\varepsilon\}$.

Definition 7.3 (functor $\mathfrak{P}\mathsf{qDiLL}_0$)**.** We define a functor $\mathfrak{P}\mathsf{qDiLL}_0 \colon \mathbf{Sched} \to \mathbf{Rel}$ by:

- *on objects:* for Γ a list of lists of MELL formulas, $\mathfrak{P}\mathsf{qDiLL}_0(\Gamma) = \mathfrak{P}(\mathsf{qDiLL}_0(\Gamma))$, the set of sets of DiLL_0 quasi-proof-structures of type Γ;
- *on arrows:* for any $\xi \colon \Gamma \to \Gamma'$, $\mathfrak{P}\mathsf{qDiLL}_0(\xi)$ is $\overset{\xi}{\rightsquigarrow} \colon \mathfrak{P}\mathsf{qDiLL}_0(\Gamma) \to \mathfrak{P}\mathsf{qDiLL}_0(\Gamma')$ (Def. 7.1).

We can now compare the functors qMELL and $\mathfrak{P}\mathsf{qDiLL}_0$ from **Sched** to **Rel**.

Theorem 7.4 (naturality)**.** *The filled Taylor expansion defines a natural transformation*

$$\mathfrak{T}^{\Pi}\colon \mathfrak{P}\mathsf{qDiLL}_0 \Rightarrow \mathsf{qMELL}\colon \mathbf{Sched} \to \mathbf{Rel}$$

by: for Γ *a list of lists of formulas,* $(\Pi, R) \in \mathfrak{T}^{\Pi}_{\Gamma}$ *iff* $\Pi \subseteq \mathcal{T}^{\Pi}(R)$ *and the type of* R *is* Γ*.*

In other words, diagram (7.1) below commutes for every scheduling $\xi\colon \Gamma \to \Gamma'$.

$$\begin{array}{ccc} \mathfrak{P}\mathsf{qDiLL}_0(\Gamma) & \xrightarrow{\mathfrak{P}\mathsf{qDiLL}_0(\xi)} & \mathfrak{P}\mathsf{qDiLL}_0(\Gamma') \\ {\scriptstyle \mathfrak{T}^{\Pi}_{\Gamma}}\downarrow & & \downarrow{\scriptstyle \mathfrak{T}^{\Pi}_{\Gamma'}} \\ \mathsf{qMELL}(\Gamma) & \xrightarrow{\mathsf{qMELL}(\xi)} & \mathsf{qMELL}(\Gamma') \end{array} \quad (7.1) \qquad \begin{array}{ccc} \Pi & \overset{\xi}{\rightsquigarrow} & \Pi' \\ {\scriptstyle \mathfrak{T}^{\Pi}_{\Gamma}}\downarrow & & \downarrow{\scriptstyle \mathfrak{T}^{\Pi}_{\Gamma'}} \\ R & \overset{\xi}{\rightsquigarrow} & R' \end{array} \quad (7.2) \qquad \begin{array}{ccc} \Pi & \overset{\xi}{\rightsquigarrow} & \Pi' \\ {\scriptstyle \mathfrak{T}^{\Pi}_{\Gamma}}\downarrow & & \downarrow{\scriptstyle \mathfrak{T}^{\Pi}_{\Gamma'}} \\ R & \overset{\xi}{\rightsquigarrow} & R' \end{array} \quad (7.3)$$

That is, given $\Pi \overset{\xi}{\rightsquigarrow} \Pi'$ with $\Pi' \subseteq \mathcal{T}^{\Pi}(R')$, we can simulate backwards the rewriting to R (here the co-functionality of actions on qMELL expressed by Lemma 6.5 comes in handy) so that $R \overset{\xi}{\rightsquigarrow} R'$ and $\Pi \subseteq \mathcal{T}^{\Pi}(R)$ (square (7.2)); and conversely, given $R \overset{\xi}{\rightsquigarrow} R'$, we can simulate the rewriting for any $\Pi \subseteq \mathcal{T}^{\Pi}(R)$, so that $\Pi \overset{\xi}{\rightsquigarrow} \Pi'$ for some $\Pi' \subseteq \mathcal{T}^{\Pi}(R')$ (square (7.3)). Note that, in both squares, the same kind of action is performed on DiLL_0 and on MELL.

Proof. $\mathfrak{T}^{\Pi}$ is a family of arrows of **Rel** indexed by the objects in **Sched**. We have to show that squares (7.2)–(7.3) commute for any scheduling $\xi\colon \Gamma \to \Gamma'$, and actually, it is enough to show commutation for elementary schedulings. Let $a\colon \Gamma \to \Gamma'$ be an elementary scheduling.

(1) *Square* (7.3)*:* Let us prove that $\mathsf{qMELL}(a) \circ \mathfrak{T}^{\Pi}_{\Gamma} \subseteq \mathfrak{T}^{\Pi}_{\Gamma'} \circ \mathfrak{P}\mathsf{qDiLL}_0(a)$.

Let $(\Pi, R') \in \mathsf{qMELL}(a) \circ \mathfrak{T}^{\Pi}_{\Gamma}$. Let $R = (|R|, \mathcal{F}, \mathsf{box}) \in \mathsf{qMELL}(\Gamma)$ be a witness of composition, that is an element such that $(\Pi, R) \in \mathfrak{T}^{\Pi}_{\Gamma}$ and $R \overset{a}{\rightsquigarrow} R'$. If $\Pi = \emptyset$ then we are done because $\Pi \overset{a}{\rightsquigarrow} \emptyset$ (see Footnote 20) and $\emptyset \subseteq \mathcal{T}^{\Pi}(R')$. So, we suppose that $\Pi \neq \emptyset$.

Let $p\colon t \to \mathcal{F}$ be a thick subforest of $\mathcal{F}$, let $r_1, \dots, r_n$ be some roots in $\mathcal{F}$ (and in t, by Remark 5.3), and let $\rho_{r_1 \dots r_n} \in \Pi$ be the element of the filled Taylor expansion $\mathcal{T}^{\Pi}(R)$ of R associated with p and $r_1, \dots, r_n$ (*i.e.* $\rho_{r_1 \dots r_n}$ is the emptying on $r_1, \dots, r_n$ of the element of $\mathcal{T}(R)$ obtained via the thick subforest (t, p) of $\mathcal{F}$, see Definition 5.7). By Remark 5.8, we can identify the conclusions of R and of $\rho_{r_1 \dots r_n}$. The case $a = \mathtt{exc}_i$ is trivial since it just swaps the order of two consecutive conclusions. Other cases for a:

- If $a = \mathtt{mix}_i$, then, in R, the conclusions $1, \dots, i, i+1, \dots k$ are exactly the conclusions of a root in the box-forest of R, and i and $i+1$ are not connected in R (seen as an undirected graph, *i.e.* in $\|R\|$) By Definitions 5.5 and 5.7, since $\rho_{r_1 \dots r_n} \in \Pi \subseteq \mathcal{T}^{\Pi}(R)$, we have that the conclusions $1, \dots, i, i+1, \dots k$ are exactly the conclusions of a root r in the box-forest of $\rho_{r_1 \dots r_n}$, and we have two possibilities according to Remark 5.10:
  - the connected components of i and $i+1$ are disjoint in $\rho_{r_1 \dots r_n}$;
  - i and $i+1$ belong to the same connected component of $\rho_{r_1, \dots, r_n}$, and this component is a ✠-cell with conclusions $1, \dots, i, i+1, \dots k$.

  In both cases the rule $\mathtt{mix}_i$ is also applicable in $\rho_{r_1 \dots r_n}$, yielding a DiLL_0 quasi-proof-structure ρ'. The box-forest $\mathcal{F}'$ of R' is obtained from the box-forest $\mathcal{F}$ of R by replacing a root b by two roots b_1, b_2. Let $p' : t' \to \mathcal{F}'$ be such that all the boxes $d \neq b_1, b_2$ have the same inverse image as by p: $p'^{-1}(d) = p^{-1}(d)$, and, $p'^{-1}(b_1) = p^{-1}(b) \times \{1\}$, $p'^{-1}(b_2) = p^{-1}(b) \times \{2\}$. We verify that ρ' is the element of the filled Taylor expansion $\mathcal{T}^{\Pi}(R')$ of R' associated with p'.
- If $a \in \{\mathtt{ax}_i, \mathbf{1}_i, \bot_i, ?_{wi}\}$, let k be such that the rule a acts on the conclusions $i-k, \dots, i$ in R ($k \in \{0, 1\}$), and let ℓ be the type of the cell in R whose outputs are $i-k, \dots, i$.

In $\rho_{r_1\dots r_n}$ there is a cell of type ℓ or $\maltese$ whose outputs are the same conclusions. Clearly a is applicable to $\rho_{r_1\dots r_n}$, which yields a DiLL_0 quasi-proof-structure ρ'.
The box-forest $\mathcal{F}'$ of R' is obtained from the box-forest $\mathcal{F}$ of R by erasing a root b. Let $p' : t' \to \mathcal{A}'$ be such that all the boxes $d \neq b$ have the same inverse image than by p: $p'^{-1}(d) = p^{-1}(d)$. We verify that ρ' is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R')$ of R' associated with p'.

- If $a \in \{\otimes_i, \parr_i, ?_{d\,i}, ?_{c\,i}\}$, let k be such that the rule a acts on the conclusions $i-k,\dots,i$ of a component of R, and let ℓ be the type of the cell in R whose output is the conclusion i. In $\rho_{r_1\dots r_n}$ either there is a ℓ-cell whose output tail is i, or the component containing the conclusion i is a daimon with conclusions $i-k,\dots,i$. Clearly a is applicable to $\rho_{r_1\dots r_n}$, which yields a DiLL_0 quasi-proof-structure ρ'.
  R' has the same box-forest $\mathcal{F}$ as R. We verify that ρ' is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R')$ of R' associated with p.
- If $a = \mathtt{cut}^i$, let c be the $\mathtt{cut}$-cell of depth 0 in R to which the rule is applied. Like in the previous case, the $\mathtt{cut}$-cell c has either one image in $\rho_{r_1\dots r_n}$ or is represented by a $\maltese$-cell. In both cases, $\mathtt{cut}^i$ is applicable to $\rho_{r_1\dots r_n}$, yielding ρ'.
  R' has the same box-forest $\mathcal{F}$ as R. We verify that ρ' is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R')$ of R' associated with p.
- If $a = \mathsf{Box}_i$, let k be such that the rule a acts on the conclusions $i-k,\dots,i$ of a component of R. In $\rho_{r_1\dots r_n}$ we have one of the following possibilities:
  (a) $\rho_{r_1\dots r_n}$ consists of a unique $\maltese$-cell with the same conclusions $i-k,\dots,i$;
  (b) $\rho_{r_1\dots r_n}$ consists of a !-cell with output tail i and no inputs; and of k ?-cells, each of which has no inputs and one output tail;
  (c) there are a !-cell above the conclusion i and a ?-cell above each of the other k conclusions; and the other cells of this component of $\rho_{r_1,\dots,r_n}$ can be identified by their image $1,\dots,\ell$ in t: we have ℓ pairwise disconnected sub-quasi-proof-structures $\pi_1,\dots,\pi_\ell$.

  In any case, the rule Box_i can be applied, yielding either a family $\rho'_1,\dots,\rho'_\ell$ of DiLL_0 proof-structures or a DiLL_0 quasi-proof-structure ρ'_1. More precisely, in the first (resp. second, third) case, we apply the Daimoned (resp. Empty, Non-empty) box rule, see Figure 9h (resp. Figure 9i, Figure 9j).
  The box-forest $\mathcal{F}'$ of R' is obtained from the box-forest $\mathcal{F}$ of R by erasing the root of the conclusions $i-k,\dots,i$: the new root b' of this tree of $\mathcal{F}'$ is the unique vertex connected to the root of $\mathcal{F}$ (its unique son). We have $p^{-1}(b') = \{b'_1,\dots,b'_\ell\}$, and ℓ trees $t'_1,\dots,t'_\ell$, where b'_i is the root of t'_i. The morphisms $p'_i : t'_i \to \mathcal{F}'$ are defined accordingly, and ρ'_i is the element of filled Taylor expansion $\mathcal{T}^{\maltese}(R')$ of R' associated with p'_i.

(2) *Square* (7.2)*:* Let us prove that $\mathfrak{T}^{\maltese}_{\Gamma'} \circ \mathfrak{P}\mathsf{qDiLL}_0(a) \subseteq \mathsf{qMELL}(a) \circ \mathfrak{T}^{\maltese}_{\Gamma}$.

Let $(\Pi, R') \in \mathfrak{T}^{\maltese}_{\Gamma'} \circ \mathfrak{P}\mathsf{qDiLL}_0(a)$ with $R' = (|R'|, \mathcal{F}', \mathsf{box}')$. Let Π' be a witness of composition, *i.e.* a set in $\mathfrak{P}\mathsf{qDiLL}_0(\Gamma')$ such that $\Pi \overset{a}{\rightsquigarrow} \Pi'$ and $(\Pi', R') \in \mathfrak{T}^{\maltese}_{\Gamma'}$.

We want to exhibit a MELL quasi-proof-structure R such that $R \overset{a}{\rightsquigarrow} R'$ and Π is a part of the filled Taylor expansion $\mathcal{T}^{\maltese}(R)$ of R. By co-functionality of $\mathsf{qMELL}(a)$ (Lemma 6.5), we have a candidate for such an R: the preimage of R' by this co-functional relation $\overset{a}{\rightsquigarrow}$. In other terms, if $a\colon \Gamma \to \Gamma'$ and $R' \in \mathsf{qMELL}(\Gamma')$, then there exists a unique $R = (|R|, \mathcal{F}, \mathsf{box}) \in \mathsf{qMELL}(\Gamma)$ such that $R \overset{a}{\rightsquigarrow} R'$ (Lemma 6.5). We only have to check that Π is a part of the filled Taylor expansion $\mathcal{T}^{\maltese}(R)$ of such a R, *i.e.* $\Pi \subseteq \mathcal{T}^{\maltese}(R)$.

Let $\rho \in \Pi$ and $\{\rho'_1, \dots, \rho'_n\} \subseteq \Pi'$ such that $\rho \overset{a}{\rightsquigarrow} \{\rho'_1, \dots, \rho'_n\}$. In all cases except $a = \mathsf{Box}_i$, we have $n = 1$ *i.e.* $\{\rho'_1, \dots, \rho'_n\} = \{\rho'_1\}$. For $a \neq \mathsf{Box}_i$, let $p' \colon t' \to \mathcal{F}'$ be a thick subforest of $\mathcal{F}'$, let $r'_1, \dots, r'_k$ be some roots in $\mathcal{F}'$ (and in t', by Remark 5.3), and let $\rho'_1 = \rho'_{r'_1 \dots r'_k} \in \Pi$ be the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R')$ of R' associated with p' and $r'_1, \dots, r'_k$ (*i.e.* $\rho'_{r'_1 \dots r'_k}$ is the emptying on $r'_1, \dots, r'_k$ of the element of $\mathcal{T}(R')$ obtained via the thick subforest (t', p') of $\mathcal{F}'$, see Definition 5.7). By Remark 5.8, we can identify the conclusions of R' and of $\rho'_{r'_1 \dots r'_k}$. The case $a = \mathtt{exc}_i$ is trivial since it just swaps the order of two consecutive conclusions. Other cases for a:

- If $a \in \{\mathtt{ax}_i, \mathbf{1}_i, \bot_i, ?_{wi}\}$, let r be the root in $\mathcal{F}$ of the conclusion i in R and let s be the root in t of the conclusion i in ρ. $\mathcal{F}$ is the juxtaposition of $\mathcal{F}'$ and the root r. Let t be the juxtaposition of t' and the root s, and $p \colon t \to \mathcal{F}$ be defined as p' over t' and $p(s) = r$. We check that ρ is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R)$ of R associated with p and $r'_1, \dots, r'_k$.
- If $a \in \{\mathtt{cut}^i, \otimes_i, \parr_i, ?_{di}, ?_{ci}\}$, then $\mathcal{F} = \mathcal{F}'$, $p = p'$ and we check that ρ is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R)$ of R associated with p and $r'_1, \dots, r'_k$.
- If $a = \mathtt{mix}_i$, let s'_1 and s'_2 be the respective roots in $\mathcal{F}'$ of the conclusions i and $i+1$ in R', let r be the root of the conclusion i in R. Let t be equal to t' except that the two roots s'_1 and s'_2 are merged into the root s in t, and let p be defined as $p(s) = r$ and $p = p'$ over the rest of t. We check that ρ is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R)$ of R associated with p and $r'_1, \dots, r'_k$.
- If $a = \mathsf{Box}_i$, we describe the rule for Non-empty box (Figure 9j); the easier rules for Daimoned and Empty boxes (Figures 9h and 9i) are left to the reader. Let $p'_1 \colon t'_1 \to \mathcal{F}'$, $\dots$, $p'_n \colon t'_n \to \mathcal{F}'$ be the thick subforests of $\mathcal{F}'$, let $r'_1, \dots, r'_k$ be some roots in $\mathcal{F}'$ (and in $t'_1, \dots, t'_n$, by Remark 5.3). For all $1 \leqslant i \leqslant n$, let ρ'_i be the element of the filled Taylor expansions $\mathcal{T}^{\maltese}(R')$ of R' associated with p'_i and $r'_1, \dots, r'_k$ (*i.e.* ρ'_i is the emptying on $r'_1, \dots, r'_k$ of the element of $\mathcal{T}(R')$ obtained via the thick subforest (t'_i, p'_i) of $\mathcal{F}'$, see Definition 5.7). Each forest t'_i differs from $\mathcal{F}'$ only in one tree. Let t be the forest made of all the trees on which the forests $t'_1, \dots, t'_n$ do not differ, and of the union of the trees on which the forests differ, joined by a root; define p from $p'_1, \dots, p'_n$ accordingly. We check that ρ is the element of the filled Taylor expansion $\mathcal{T}^{\maltese}(R)$ of R associated with p and $r'_1, \dots, r'_k$. □

8. Glueability of DiLL_0 quasi-proof-structures and inhabitation

Naturality (Theorem 7.4) allows us to prove our main original results:

(1) a characterization of the sets of DiLL_0 proof-structures that are in the Taylor expansion of some MELL proof-structure (*inverse Taylor expansion problem*, Theorem 8.3);
(2) a characterization of the lists of MELL formulas inhabitated by some cut-free MELL proof-structure (*type inhabitation problem* for cut-free MELL proof-structures, Theorem 8.9).

Both characterizations are *constructive*, in the sense that, when we claim the existence of an object (a MELL proof-structure satisfying a certain property), we are actually able to construct a witness of it, thanks to our rewriting rules. However, our procedure is *nondeterministic* because the rewriting is not unique.

8.1. **Glueability.** We introduce the *glueability criterion* to solve the inverse Taylor expansion problem. We actually solve it in two different domains. We look for a solution:
(1) in the set of MELL proof-structures (Theorem 8.3.1);
(2) in the set of cut-free MELL proof-structures (Theorem 8.3.2).

We say that a scheduling is *cut-free* if does not contain the elementary scheduling $\mathtt{cut}^i$.

Lemma 8.1 (cut-free scheduling)**.** *Let R be a* MELL *quasi-proof-structure and ξ be a scheduling such that $R \overset{\xi}{\rightsquigarrow} \varepsilon$. We have that R is cut-free if and only if ξ is cut-free.*

Proof. Let R be MELL quasi-proof-structure. If R is cut-free and $R \overset{a}{\rightsquigarrow} R'$, then $a \neq \mathtt{cut}^i$ and R' is cut-free, because the action $R \overset{\mathtt{cut}^i}{\rightsquigarrow} R'$ requires the presence of a $\mathtt{cut}$-cell in R.

Conversely, if R is with cuts and $R \overset{a}{\rightsquigarrow} R'$ where R' is cut-free, then $a = \mathtt{cut}^i$, because among the rewrite rules in Figure 8 only $\overset{\mathtt{cut}_i}{\rightsquigarrow}$ erases a $\mathtt{cut}$-cell.

A straightforward induction on the length of the scheduling ξ allows us to conclude. □

Definition 8.2 (glueability, cut-free glueability)**.** A set Π of DiLL_0 quasi-proof-structures of type Γ is *glueable* (resp. *cut-free glueable*) if there exists a scheduling (resp. cut-free scheduling) $\xi \colon \Gamma \to \varepsilon$ such that $\Pi \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$.

Theorem 8.3 (glueability criterion)**.** *Let Π be a non-empty set of* DiLL_0 *proof-structures without $\maltese$-cells.*
(1) *Π is glueable if and only if $\Pi \subseteq \mathcal{T}(R)$ for some* MELL *proof-structure R.*
(2) *Π is cut-free glueable if and only if $\Pi \subseteq \mathcal{T}(R)$ for some cut-free* MELL *proof-structure R.*

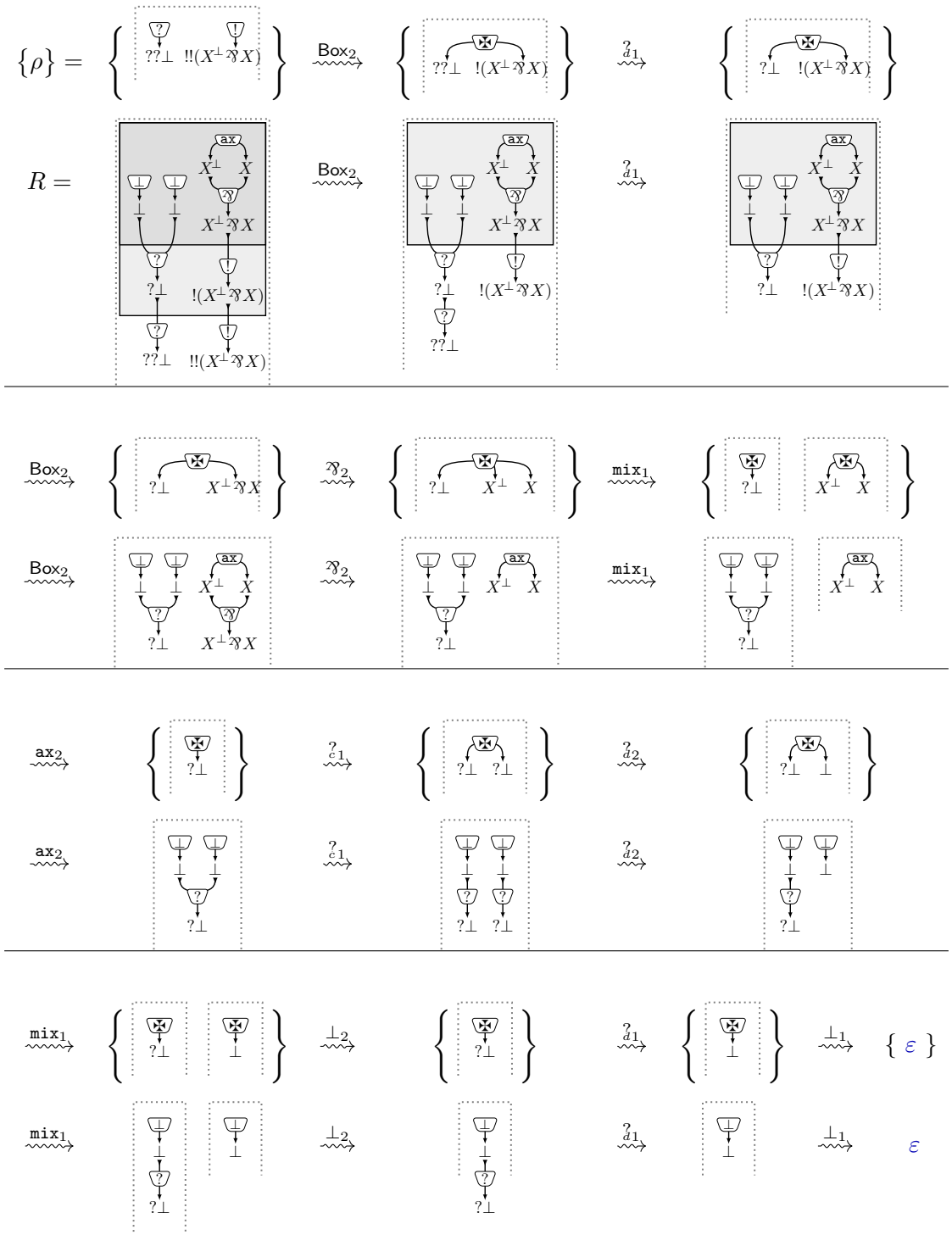
Proof. (1) If $\emptyset \neq \Pi \subseteq \mathcal{T}(R)$ for some MELL proof-structure R, then by termination (Proposition 6.7) $R \overset{\xi}{\rightsquigarrow} \varepsilon$ for some scheduling $\xi \colon (\Gamma) \to \varepsilon$, where Γ is the type of R (and of each element of Π, by Remark 5.6). So, $\Pi \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$ by naturality (Theorem 7.4, as $\mathcal{T}^{\maltese}(\varepsilon) = \{\varepsilon\}$).

Conversely, if $\Pi \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$ for some scheduling $\xi \colon (\Gamma) \to \varepsilon$ (where Γ is the type of each element of Π), then by naturality (Theorem 7.4, as $\mathcal{T}^{\maltese}(\varepsilon) = \{\varepsilon\}$) $\Pi \subseteq \mathcal{T}^{\maltese}(R)$ for some MELL quasi-proof-structure $R \overset{\xi}{\rightsquigarrow} \varepsilon$, and so $\Pi \subseteq \mathcal{T}(R)$ since each element of Π is without $\maltese$-cells. The (MELL) quasi-proof-structure R is indeed a proof-structure as each element of Π is a (DiLL_0) proof-structure of type Γ, and by Remark 5.6 the type of R is Γ.

(2) Just repeat the same proof as above, paying attention that:
- in the "if" part, if R is cut-free, then the scheduling ξ is cut-free by Lemma 8.1;
- in the "only if" part, if $\Pi \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$ for some cut-free scheduling $\xi \colon (\Gamma) \to \varepsilon$, then by Lemma 8.1 R is cut-free, since $R \overset{\xi}{\rightsquigarrow} \varepsilon$ according to naturality (Theorem 7.4). □

Theorem 8.3 means that the action of a scheduling rewriting a set Π of DiLL_0 proof-structures to the empty proof-structure can be seen, read in reverse, as a sequence of elementary steps to build a MELL proof structure that contains Π in its Taylor expansion.

Example 8.4. An example of the action of a scheduling starting from the singleton of a DiLL_0 proof-structure ρ and ending in $\{\varepsilon\}$ is in Figure 11 (it is by no means the shortest possible rewriting). When replayed backward, by naturality (Theorem 7.4) it induces a MELL proof-structure R such that $\rho \in \mathcal{T}(R)$. In Figure 11 we represent simultaneously the rewriting from Π to $\{\varepsilon\}$ and the rewriting from R to ε.



Scheduling $\mathsf{Box}_2\, ?_{d1}\, \mathsf{Box}_2\, ⅋_2\, \mathtt{mix}_1\, \mathtt{ax}_2\, ?_{c1}\, ?_{d2}\, \mathtt{mix}_1\, \bot_2\, ?_{d1}\, \bot_1 : (??\bot, !!(X^\bot ⅋ X)) \to \varepsilon$

FIGURE 11. Action of a scheduling witnessing that $\rho \in \mathcal{T}(R)$.

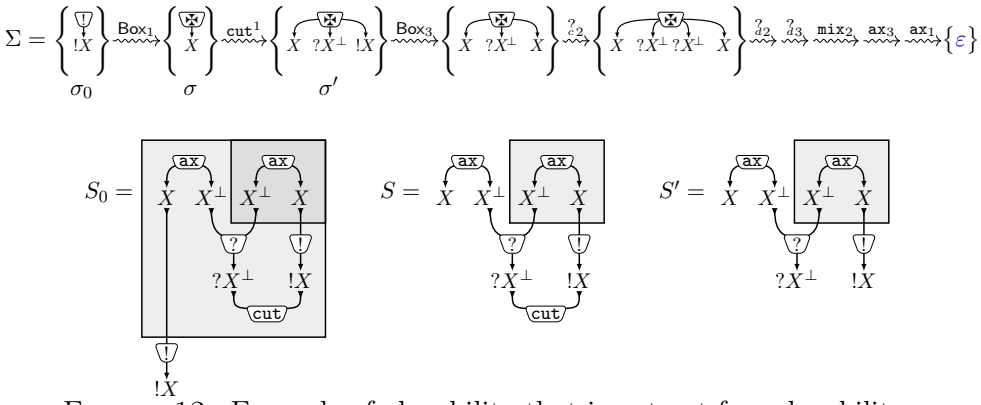
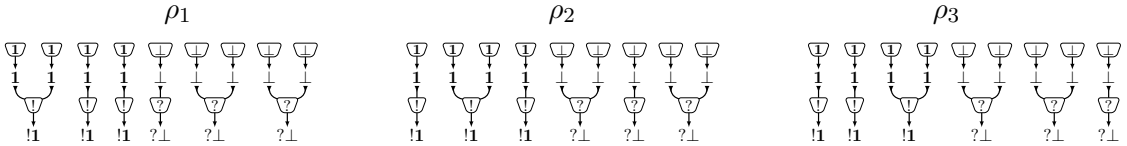


FIGURE 12. Example of glueability that is not cut-free glueability.

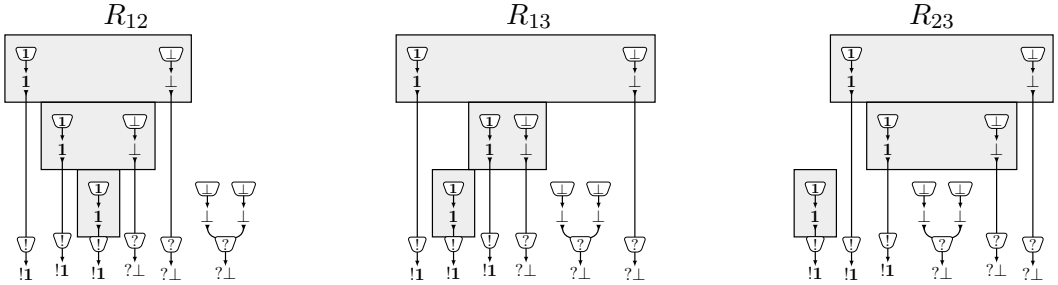
We introduced two different glueability criteria (Theorems 8.3.1 and 8.3.2) because, given a non-empty set Π of *cut-free* DiLL_0 proof-structures without ✠-cells, the existence of a solution to the inverse Taylor expansion problem depends on whether we look for it among MELL proof-structures or among cut-free MELL proof-structures (Example 8.5).

Example 8.5. Let $\Sigma = \{\sigma_0\}$ as in Figure 12. This singleton of a cut-free DiLL_0 proof-structure without ✠-cells is glueable (as shown by the action of the scheduling in Figure 12) but not cut-free glueable (as $\{\sigma_0\} \overset{\mathsf{Box}_1}{\rightsquigarrow} \{\sigma\}$ and, apart from $\mathtt{cut}^i$, no elementary schedulings apply to $\{\sigma\}$ because of its type). Indeed, σ_0 "masks" any information about the content B of the box represented by its !-cell, apart from its type $!X$ which does not allow B to be a cut-free MELL proof-structure (see also Section 8.2). Glueability (with cuts) of $\{\sigma\}$ is related to the fact that there is a MELL proof-structure of type X with cuts, for every atomic formula X (see Remark 8.8). Note that $\sigma_0 \in \mathcal{T}(S_0)$, and $\sigma \in \mathcal{T}^{✠}(S)$ and $\sigma' \in \mathcal{T}^{✠}(S')$, where S_0, S and S' are the MELL proof-structures with cuts represented in Figure 12.

Example 8.6. The three DiLL_0 proof-structures ρ_1, ρ_2, ρ_3 below are not glueable as a whole, but are glueable two by two (this is a slight variant of the example in [Tas09, pp. 244-246]). In fact, there is no MELL proof-structure whose Taylor expansion contains ρ_1, ρ_2, ρ_3, but any pair of them is in the Taylor expansion of some MELL proof-structure.



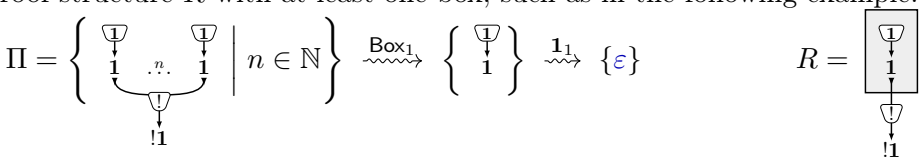
Indeed, for any $i, j \in \{1, 2, 3\}$ with $i \neq j$, ρ_i and ρ_j are in the Taylor expansion of R_{ij} below.



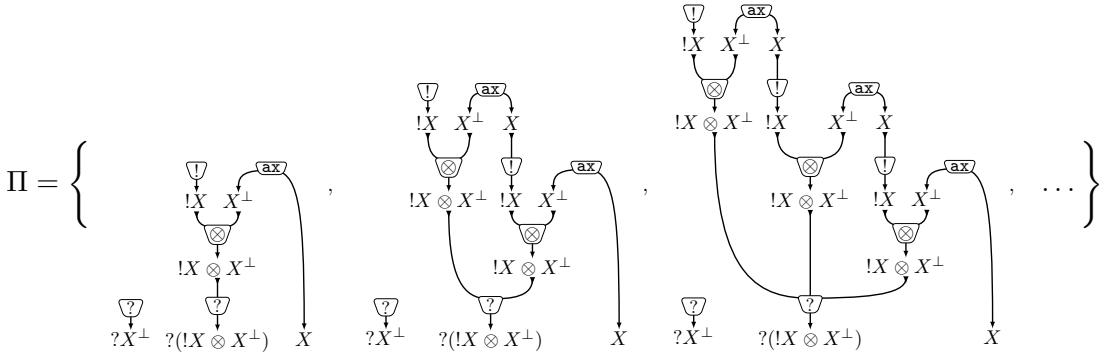
Example 8.6 can be generalized to finite sets of any cardinality: for any $n > 1$, there is a finite set Π of $n+1$ DiLL_0 proof-structures that is not glueable as a whole, but all subsets of Π of at most n elements are glueable. This means that, for any $n > 1$, glueability of a set Π of DiLL_0 proof-structures is not reducible to test that all subsets of Π with at most n elements satisfy any n-ary coherence relation, unless (obviously) Π is a finite set with at most n elements. As explained in Section 1, this difficulty in the inverse Taylor expansion problem is typical of MELL and does not arise in the λ-calculus.

Remark 8.7 (infinite sets)**.** The glueability criterion (Theorem 8.3) is not limited to finite sets of DiLL_0 proof-structures, it holds for infinite sets too. This raises the question of the relationship between glueability and finite glueability, namely: when every finite subset of a given infinite set of DiLL_0 proof-structures is glueable, is the infinite set itself glueable too? The answer is negative. Let Π be an *infinite* set of DiLL_0 proof-structures such that *every finite* subset of Π is glueable. There are two cases for Π.

- Π itself is glueable. It is typically the case when Π is exactly the Taylor expansion of a MELL proof-structure R with at least one box, such as in the following example.



  We might say that Π is *infinite in width*: infinity is related to the number of copies chosen for the boxes of R, a number that has to be finite but can be arbitrarily large.
- Π is not glueable. This is the case in the following set (we draw three elements only).



  Every finite subset of Π is glueable, but the whole infinite set Π is not. We might say that Π is *infinite in depth*: in Π, for any $n > 0$ there is a DiLL_0 proof-structure with n !-cells. Any finite subset Π_n of Π (n is the maximal number of !-cells in the elements of Π_n) is included in the Taylor expansion of a MELL proof-structure with depth n, and indeed the action of the scheduling ν_n defined in Example 7.2 rewrites Π_n to $\{\varepsilon\}$.[21] However, the whole Π can only been seen as the Taylor expansion of an "infinite" proof-structure with infinitely deep nested boxes. Such infinite proof-structures are not in the syntax of MELL. An ongoing research relates them to non-well-founded proofs for fixed-point logics [DS19, DPS21].

In the inverse Taylor expansion problem, both kinds of infinity—in width and in depth—arise already in the λ-calculus, they are not specific to MELL. An instance of a set of resource λ-terms that is infinite in width is the Taylor expansion of the λ-term xy. The set

[21] In fact, if ρ_h is the element of Π with exactly h !-cells, $\{\rho_h\} \overset{\nu_k}{\rightsquigarrow} \{\varepsilon\}$ for all $k \geq h$.

$\Pi = \{\lambda f.\lambda x.f[\,],\ \lambda f.\lambda x.f[f[\,]],\ \lambda f.\lambda x.f[f[f[\,]]],\ \dots\}$ of resource λ-terms is an example of infinity in depth: every finite subset of Π is included in the Taylor expansion of some λ-term (a Church numeral), but Π is not contained in the Taylor expansion of any λ-term.

8.2. **Inhabitation.** We can characterize type inhabitation in MELL proof-structures. We first put the question of type inhabitation for MELL proof-structures in a proper way.

Remark 8.8 (non-cut-free inhabitation is trivial)**.** For every atomic formula X there is a MELL proof-structure of type X with cuts, the one called S shown in Figure 12.

Also, for every MELL formula A (resp. list Γ of MELL formulas), there is a MELL proof-structure of type A (resp. Γ) with cuts. Indeed, for A it is enough to "η-expand" S, by replacing any $\mathtt{ax}$-cell in S with the "canonical" MELL proof-structure of type $A, A^{\perp}$. For $\Gamma = (A_1, \dots, A_n)$, just repeat the same procedure for each A_i separately, and then juxtapose the MELL proof-structures to yield a unique MELL proof-structure of type Γ.

Therefore, type inhabitation for MELL proof-structures is trivial, if $\mathtt{cut}$-cells are allowed.

Remark 8.8 says that type inhabitation for MELL proof-structures is non-trivial only if we restrict to *cut-free* MELL proof-structures. And given two propositional variables $X \neq Y$, there is no *cut-free* MELL proof-structure of type X or Y, X. So, the question is: given a list Γ of MELL formulas, is there a cut-free MELL proof-structure of type Γ? Following the intuition that a scheduling $\xi\colon (\Gamma) \to \varepsilon$ "encodes" a MELL proof-structure of type Γ, we investigate the type inhabitation problem for cut-free MELL proof-structures via schedulings.

Let us see the action of cut-free schedulings on daimons. For some daimon σ there is no cut-free scheduling ξ such that $\{\sigma\} \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$. For instance, if $X \neq Y$ are propositional variables and ρ is one of the two daimons in Figure 13a, then—because of the type—apart from $\mathtt{cut}^i$, no elementary schedulings can be applied to σ_0, while the only other elementary schedulings that can be applied to σ_1 are $\mathtt{mix}_1$ and $\mathtt{exc}_1$, but their actions do not rewrite $\{\sigma_2\}$ to $\{\varepsilon\}$. Indeed, a ✠-cell can be erased only by the action of elementary scheduling called hypothesis: $\mathtt{ax}_i, \mathbf{1}_i, \perp_i, ?_{wi}$ (Figure 9c), but none of them applies because of the type of σ_0 and σ_1.

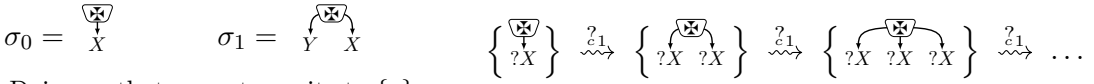
Actually, cut-free schedulings acting on daimons are a way to characterize the (lists of) MELL formulas that are the type of some cut-free MELL proof-structure.

Theorem 8.9 (type inhabitation for cut-free MELL proof-structures)**.** *Let Γ be a list of* MELL *formulas and ρ be the daimon of type Γ. There is a cut-free* MELL *proof-structure of type Γ if and only if $\{\rho\} \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$ for some cut-free scheduling $\xi\colon (\Gamma) \to \varepsilon$.*

Proof. If R is a cut-free MELL proof-structure of type Γ, then $R \overset{\xi}{\rightsquigarrow} \varepsilon$ for some cut-free scheduling $\xi\colon (\Gamma) \to \varepsilon$, by Proposition 6.7 and Lemma 8.1. By naturality (Theorem 7.4), $\{\rho\} \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$ since $\rho \in \mathcal{T}^{✠}(R)$ (indeed ρ is the emptying of any element of $\mathcal{T}(R)$) and $\mathcal{T}^{✠}(\varepsilon) = \{\varepsilon\}$.

Conversely, if $\{\rho\} \overset{\xi}{\rightsquigarrow} \{\varepsilon\}$ for some cut-free scheduling $\xi\colon (\Gamma) \to \varepsilon$, then by naturality (Theorem 7.4) there is a MELL quasi-proof-structure $R \overset{\xi}{\rightsquigarrow} \varepsilon$ such that $\rho \in \mathcal{T}^{✠}(R)$ (since $\mathcal{T}^{✠}(\varepsilon) = \{\varepsilon\}$ according to Remark 5.9). As Γ is a list of MELL formulas and the type of R (by Remark 5.6), R is a proof-structure. By Lemma 8.1, it R is cut-free. □

The proof of Theorem 8.9 says that the action of a cut-free scheduling rewriting a daimon to the empty proof-structure can be seen, when read backward, as a sequence of elementary steps to build a cut-free MELL proof-structure from scratch. Consider now the following *procedure*: given a list Γ of MELL formulas as input, let ρ_Γ be the daimon of type Γ and



(A) Daimons that cannot rewrite to $\{\varepsilon\}$ via any cut-free schedulings.

(B) Infinite rewriting from a daimon.

FIGURE 13. Examples of rewritings for daimons.

(1) for all $n > 0$, consider all the rewritings from $\{\rho_\Gamma\}$ with at most n steps, *i.e.*, the actions of all cut-free schedulings (finite in number) made of at most n elementary schedulings;
(2) as soon as the singleton $\{\varepsilon\}$ is reached by the action of a scheduling ξ, accept.

Theorem 8.9 guarantees that such a procedure accepts Γ if and only if there is a cut-free MELL proof-structure R of type Γ, and by naturality (Theorem 7.4), the cut-free scheduling ξ that accepts Γ allows us to construct such a R by reading ξ backward. This procedure:

- *semi-decides* the type inhabitation problem for cut-free MELL proof-structures;
- *decides* the type inhabitation problem for cut-free MELL proof-structures *without contractions* (*i.e.* without ?-cells with at least two inputs).

Indeed, in general, there is no bound on the length of the rewritings on DiLL_0, and the action of the elementary scheduling $?_{c\,i}$ on a daimon can create an infinite rewrite sequence (see Figure 13b). So, the procedure is not guaranteed to halt. Moreover, it halts in the case without contractions: the action of each elementary scheduling other than $\mathtt{cut}_i$ on DiLL_0 decreases the size of the type of a daimon (for a suitable notion of size), except for the action of $?_{c\,i}$—and of $\mathtt{exc}_i$, but it is not relevant here—which corresponds to add a contraction (a ?-cell with at least two inputs) to the MELL proof-structure under construction.

Remark 8.10 (on decidability of MELL)**.** Our characterization of type inhabitation for cut-free MELL proof-structures (Theorem 8.9) can be seen in relation to the question of deciding if a MELL formula is provable: indeed, a MELL formula A is provable if and only if there is a *correct* cut-free MELL proof-structure of type A (for some notion of correctness).

In the propositional setting, MLL provability is decidable and **NP**-complete [Kan94], while LL provability is undecidable [LMSS92]; a longstanding open problem is whether provability in MELL is decidable or not. Our result offers a fresh perspective on it.

Take again the procedure above with input Γ, and replace its second step by:

(2) if the singleton $\{\varepsilon\}$ is reached by the action of a cut-free scheduling ξ, construct the cut-free MELL proof-structure R of type Γ by reading ξ backward;
(3) check if R is correct for a suitable correctness criterion that characterizes all and only the MELL proof-structures corresponding to a derivation in MELL sequent calculus [TdF03];
(4) as soon as a correct (cut-free) MELL proof-structure is found, accept.

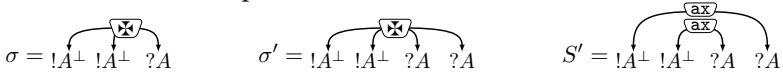
As explained above, there may be infinitely many cut-free schedulings $\xi\colon (\Gamma) \to \varepsilon$, and so infinitely many cut-free MELL proof-structures of type Γ. But they are finite in number in the case without contractions. Since correctness of a MELL proof-structure is decidable, this procedure *semi-decides* the problem of provability in MELL (is a given sequent provable in MELL sequent calculus?) and *decides* the problem of provability in MELL *without contractions* (is a given sequent provable in MELL sequent calculus without the contraction rule?). The fact that our procedure restricts its search to cut-free objects is without loss of generality, by cut-elimination. Semi-decidability of MELL and decibability of MELL without contractions are not new results; another (and simpler) proof relies on the fact that in MELL sequent calculus every inference rule other than contraction and cut, read bottom-up, decreases the

size of a sequent, for a suitable notion of size. What is new is the analogy between the type inhabitation problem for cut-free MELL proof-structures and the provability problem in MELL.

Summing up, contraction is the only reason that makes hard to prove whether inhabitation for cut-free MELL proof-structures, as well as MELL provability, are decidable or not. This seems to suggest that *correctness* (the fact that a MELL proof-structure is a proof of a MELL formula) does not play an essential role in the question if MELL provability is decidable or not. However, for the time being, we are not able to reduce decidability of provability in MELL to decidability of type inhabitation for cut-free MELL proof-structures.

9. Non-atomic axioms

Our (quasi-)proof-structures (Definitions 4.1 and 4.4, Figure 2) deal with *atomic axioms* only: the two outputs of any ax-cell are dual atomic formulas. We can relax the definitions and allow also the presence of *non-atomic axioms*: the outputs of any ax-cell are typed by dual MELL formulas, not necessarily atomic. We can extend our results to this more general setting, with some technical complications. The only trouble is with *exponential axioms*, where the outputs of an ax-cell are typed by MELL formulas $!A^\perp, ?A$. Indeed, consider the daimons σ and σ' and the MELL proof-structure S' below.



We have that $\{\sigma\} \overset{?}{\underset{c}{\rightsquigarrow}}_3 \{\sigma'\} \subseteq \mathcal{T}^{\Phi}(S')$. But no elementary scheduling $?_3$ can be applied backwards to S' (because of the lack of a ?-cell), which *breaks naturality* (Theorem 7.4). This is actually due to the fact that, with exponential axioms, co-functionality (Lemma 6.5) fails for the rewrite rule $\overset{?}{\underset{c}{\rightsquigarrow}}_i$: read from right to left, $\overset{?}{\underset{c}{\rightsquigarrow}}_i$ is a functional but not total relation.

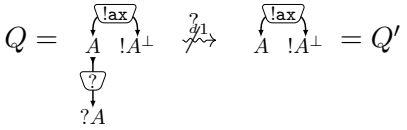
The solution we propose here to deal with non-atomic axioms asks for a technical refinement of only few notions, the rest is unchanged and goes smoothly. The definitions of module, proof-structure and quasi-proof-structure (Definitions 4.1 and 4.4) change as follows.

(1) In modules, non-atomic axioms are allowed, except the exponential ones. We add a new type of cells, the !ax-cells, with no inputs and two outputs of type $!A^\perp$ and A.
(2) In proof-structures (and hence in quasi-proof-structures), we require that, for every !ax-cell whose outputs have type $!A^\perp$ and A, the output o of type A is an input of a ?-cell (more precisely, o is an half of an edge whose other half is the input of a ?-cell).

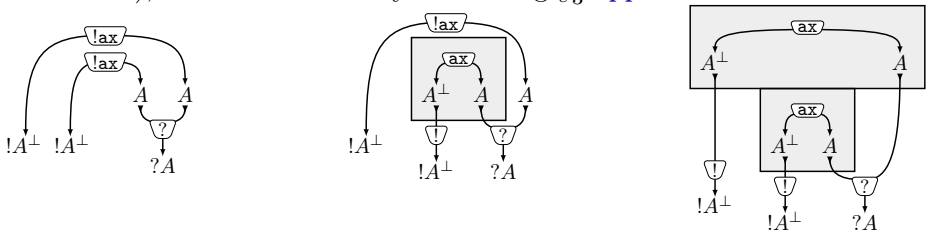
As a consequence, we add the new elementary scheduling $!\mathtt{ax}_i$ below to the list in Figure 7, $(\Gamma_1;\cdots;\Gamma_k;\mathsf{c}(i),\mathsf{c}(i{+}1);\Gamma_{k+2};\cdots;\Gamma_n) \xrightarrow{!\mathtt{ax}_i} (\Gamma_1;\cdots;\Gamma_k;\Gamma_{k+2};\cdots;\Gamma_n)$ with $\mathsf{c}(i) = ?A = \mathsf{c}(i{+}1)^\perp$ whose action—also in its daimoned version—is like the hypothesis in Figures 8c and 9c:

The way the action of an elementary scheduling is defined (Definitions 6.3 and 7.1) automatically rules out the possibility that a quasi-proof-structure rewrites to a module that is not a quasi-proof-structure. For instance, the elementary scheduling $?_1$ does not apply to

the (MELL and DiLL_0) proof-structure Q below, because it would yield a module Q' that is not a quasi-proof-structure (see Item 2 above).



Apart from the changes we mentioned, all definitions and propositions in Sections 4 to 8 remain unaffected and valid (just add $!\mathtt{ax}_i$ to the list of elementary schedulings that can apply to R in Lemma 6.6). In particular, the counterexample to naturality shown at the beginning of this section does not apply here, as S' is not a MELL (quasi-)proof-structure (because of its exponential axioms). In our framework with non-atomic axioms, the only MELL proof-structures containing σ' in their filled Taylor expansion are the ones below (up to the order of their conclusions), and the elementary scheduling $?_{c\,3}$ applies backward to each of them.



The way we propose here to deal with non-atomic axioms is more elegant and less *ad hoc* than the one we presented in [GPT20, Section 7]. In the latter, we allowed the outputs of a ✠-cell to be inputs of a ?-cell, and we changed the action of the elementary scheduling $?_{c\,i}$ for DiLL_0 quasi-proof-structures in the daimoned case (Figure 9f), by explicitly requiring the presence of a ?-cells above the conclusion i. Consequently, in [GPT20] we had to redefine the filled Taylor expansion in the non-atomic case, and thus to recheck that naturality holds for *all* the rewrite rules. Here instead we do not have to redefine the filled Taylor expansion and we have to check naturality (Theorem 7.4) in the non-atomic case only for the action of $!\mathtt{ax}_i$.

As another possible solution to deal with non-atomic axioms, instead of using generalized ?-cells in Definition 4.1 (with an arbitrary number of inputs of type A and one output of type $?A$, as in [DR95]), we could have used different kinds of ?-cells for dereliction (one input of type A, one output of type $?A$), contraction (two inputs of type $?A$, one output of type $?A$), and weakening (no inputs, one output of type $?A$), as in [Gir87]. Such a choice should be supported by a rework of the elegant definition of the Taylor expansion of a MELL proof-structure via the notion of pullback (Definitions 5.4 and 5.5), since it collapses dereliction, contraction and weakening in a generalized ?-cell.

10. Conclusions and perspectives

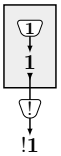
Daimons for empty boxes. Our glueability criterion (Theorem 8.3) solves the inverse Taylor expansion problem in an "asymmetric" way: we characterize the sets of DiLL_0 proof-structures without ✠-cells that are included in the Taylor expansion of some MELL proof-structure, but in general DiLL_0 proof-structures might contain ✠-cells (while MELL proof-structures cannot, see Definition 4.1). Daimons and emptyings are needed to get a natural transformation (via the *filled* Taylor expansion, Theorem 7.4), which is the main ingredient to prove our glueability criterion. But we are interested in frameworks without ✠-cells. This asymmetry

is technically inevitable, due to the fact that a glueable set of DiLL_0 proof-structures might not contain any information on the content of some box, when they take 0 copies of it.

Comparison with Pagani and Tasson [PT09]. Pagani and Tasson's solution of the inverse Taylor expansion problem is less general than ours, because it characterizes *finite* sets of DiLL_0 proof-structures that are included in the Taylor expansion of some *cut-free* MELL proof-structure with *atomic axioms*. We do not have these limitations (finite, cut-free, atomic axioms), and we can smoothly adapt our criterion to the cut-free case (Theorem 8.3.2, which is not trivial as explained in Example 8.5). In [PT09], the restrictions to cut-free and to atomic axioms aim to simplify their presentation and might be overcome. But the restriction to *finite* sets of DiLL_0 proof-structures is somehow essential in their approach. Roughly, their machinery takes a finite set Π of DiLL_0 proof-structures as input, juxtaposes its elements in a unique graph π and then runs a rewrite relation defined by means of some tokens that go through the whole π and try to merge its components in a MELL proof-structure whose Taylor expansion contains Π, if it exists. If Π were an infinite set, π would be an infinite graph and, apart from the technical intricacies of dealing with infinite objects, it would be impossible to provide a characterization in that case. In particular, their approach could not distinguish whether Π is infinite in width (which can be in the Taylor expansion of some MELL proof-structure) or infinite in depth (which cannot, see Remark 8.7). Our approach, instead, defines a rewrite relation on a single—finite—element of Π and extends it to the whole (possibly infinite) Π by requiring that the same rewrite rule applies to each element of Π (Definition 7.1). Thus, we can accommodate the case where Π is infinite.

We believe that our rewriting rules are also simpler than the ones in [PT09] and rely on a more abstract and less *ad hoc* property (naturality), that allows us to prove also semi-decidability of another problem: *type inhabitation* for cut-free MELL proof-structures.

Finally, Pagani and Tasson's solution of the inverse Taylor expansion problem is affected by another limitation, even though not particularly emphasized in [PT09]: their Theorem 2 (analogous to the "only if" part of our Theorem 8.3) assumes not only that their rewriting starting from a set of DiLL_0 proof-structures terminates but also that it ends on a MELL proof-structure, according to their definition of MELL proof-structure. This is limiting when in the set of DiLL_0 proof-structures there is no information about the content of a box. For instance, consider the singletons Π and Π' of DiLL_0 proof-structures below.

$\Pi = \left\{ \right.$ $\left. \right\}$ $\qquad$ $\Pi' = \left\{ \right.$ $\left. \right\}$ $\qquad$ $R =$ 

Pagani and Tasson's rewrite rules do not distinguish the two singletons, each one is included in the Taylor expansion of cut-free MELL proof-structures with an "empty box", due to the lack of information. So their notion of MELL proof-structure is wider and non-standard (because it allows the presence of empty boxes). On the contrary, our rewrite rules distinguish Π and Π': via the action of cut-free schedulings, the former can rewrite to $\{\varepsilon\}$ and is included in the Taylor expansion of the MELL proof-structure R above, the latter cannot rewrite to $\{\varepsilon\}$ and is not part of the Taylor expansion of any cut-free MELL proof-structure. Summing up, our characterization follows the standard notion of MELL proof-structures (unlike [PT09]) and is more fine-grained and informative than the one in [PT09].

The λ-calculus, connectedness and coherence. Our rewriting system and glueability criterion might help to prove that a binary coherence relation can solve the inverse Taylor expansion problem for MELL proof-structures fulfilling some geometric property related to connectedness, despite the impossibility for the full MELL fragment. Such a coherence would extend the coherence criterion for resource λ-terms. Note that our glueability criterion is actually an extension of the criterion for resource λ-terms. Indeed, in the case of the λ-calculus, there are three rewrite steps, corresponding to abstraction, application and variable (which can be encoded in our rewrite steps), and coherence is defined inductively: if a set of resource λ-terms is coherent, then any set of resource λ-term that rewrites to it is also coherent.

Presented in this way, the main difference between the λ-calculus and "connected" MELL (concerning the inverse Taylor expansion problem) would not be because of the rewriting system but because the structure of any resource λ-term univocally determines the rewriting path, while, for DiLL_0 proof-structures, we have to quantify existentially over all possible paths. This is an unavoidable consequence of the fact that proof-structures do not have a tree-structure, contrary to λ-terms.

Moreover, it is possible to match and mix different sequences of rewriting. Indeed, consider three DiLL_0 proof-structures pairwise glueable. Proving that they are glueable as a whole amounts to computing a rewriting path from the rewriting paths witnessing the three glueabilities. Our paths were designed with that mixing-and-matching operation in mind, in the particular case where the boxes are connected. This is reminiscent of [GPT16], where we also showed that a certain property enjoyed by the λ-calculus can be extended to proof-structures, provided they are connected inside boxes. We leave it as future work.

Functoriality and naturality. Our functorial point of view on proof-structures might unify many results. Let us cite two of them.

- A sequent calculus proof of $\vdash \Gamma$ can be translated into a path from the empty sequence into Γ. This could be the starting point for the formulation of a new correctness criterion.
- The category **Sched** can be extended with a higher structure—transforming it from a category into a 2-category—which allows cut-elimination to be represented as a 2-arrow. The functors qMELL and $\mathfrak{P}\mathsf{qDiLL}_0$ can also be extended to 2-functors, so as to prove via naturality that cut-elimination and the Taylor expansion commute.

References

[Béc98] Denis Béchet. Minimality of the correctness criterion for multiplicative proof nets. *Mathematical Structures in Computer Science*, 8(6):543–558, 1998.

[BHP13] Pierre Boudes, Fanny He, and Michele Pagani. A characterization of the Taylor expansion of lambda-terms. In *Computer Science Logic 2013 (CSL 2013)*, volume 23 of *LIPIcs*, pages 101–115. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[BM07] Dennis V. Borisov and Yuri I. Manin. *Generalized Operads and Their Inner Cohomomorphisms*, volume 265 of *Progress in Mathematics*, pages 247–308. Birkhäuser Basel, Basel, 2007.

[BM20] Davide Barbarossa and Giulio Manzonetto. Taylor subsumes Scott, Berry, Kahn and Plotkin. *Proc. ACM Program. Lang.*, 4(POPL):1:1–1:23, 2020.

[Bou93] Gérard Boudol. The lambda-calculus with multiplicities (abstract). In *4th International Conference on Concurrency Theory (CONCUR '93)*, volume 715 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 1993.

[Bou09] Pierre Boudes. Thick subtrees, games and experiments. In *Typed Lambda Calculi and Applications, 9th International Conference (TLCA 2009)*, volume 5608 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2009.

[dC16] Daniel de Carvalho. The relational model is injective for multiplicative exponential linear logic. In *25th Annual Conference on Computer Science Logic (CSL 2016)*, volume 62 of *LIPIcs*, pages 41:1–41:19. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[dC18] Daniel de Carvalho. Taylor expansion in linear logic is invertible. *Logical Methods in Computer Science*, 14(4), 2018.

[dCT12] Daniel de Carvalho and Lorenzo Tortora de Falco. The relational model is injective for multiplicative exponential linear logic (without weakenings). *Annals of Pure and Applied Logic*, 163(9):1210–1236, 2012.

[DPS21] Abhishek De, Luc Pellissier, and Alexis Saurin. Eliminating infinitely many cuts in non-wellfounded μMLL proof-nets. Submitted, available on the authors' webpage, 2021.

[DR95] Vincent Danos and Laurent Regnier. Proof-nets and the Hilbert Space. In *Proceedings of the Workshop on Advances in Linear Logic*, pages 307–328. Cambridge University Press, 1995.

[DS19] Abhishek De and Alexis Saurin. Infinets: The parallel syntax for non-wellfounded proof-theory. In *TABLEAUX 2019 - 28th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, volume 11714 of *Lecture Notes in Computer Science*. Springer, 2019.

[Ehr02] Thomas Ehrhard. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12(5):579–623, 2002.

[Ehr05] Thomas Ehrhard. Finiteness spaces. *Mathematical Structures in Computer Science*, 15(4):615–646, 2005.

[ER03] Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theoretical Computer Science*, 309(1-3):1–41, 2003.

[ER06a] Thomas Ehrhard and Laurent Regnier. Böhm Trees, Krivine's Machine and the Taylor Expansion of Lambda-Terms. In *Second Conference on Computability in Europe (CiE 2006)*, volume 3988 of *Lecture Notes in Computer Science*, pages 186–197. Springer, 2006.

[ER06b] Thomas Ehrhard and Laurent Regnier. Differential interaction nets. *Theoretical Computer Science*, 364(2):166–195, 2006.

[ER08] Thomas Ehrhard and Laurent Regnier. Uniformity and the Taylor expansion of ordinary lambda-terms. *Theoretical Computer Science*, 403(2-3):347–372, 2008.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–101, 1987.

[GPT16] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Computing connected proof(-structure)s from their Taylor expansion. In *1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016)*, volume 52 of *LIPIcs*, pages 20:1–20:18. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[GPT19] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Proof-net as graph, Taylor expansion as pullback. In *Logic, Language, Information, and Computation - 26th International Workshop (WoLLIC 2019)*, volume 11541 of *Lecture Notes in Computer Science*, pages 282–300. Springer, 2019.

[GPT20] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Glueability of resource proof-structures: inverting the Taylor expansion. In *28th EACSL Annual Conference on Computer Science Logic, CSL 2020*, volume 152 of *LIPIcs*, pages 24:1–24:18. Schloss Dagstuhl, 2020.

[Kan94] Max I. Kanovich. The complexity of Horn fragments of linear logic. *Ann. Pure Appl. Log.*, 69(2-3):195–241, 1994.

[KMP20] Emma Kerinec, Giulio Manzonetto, and Michele Pagani. Revisiting call-by-value Böhm trees in light of their Taylor expansion. *Log. Methods Comput. Sci.*, 16(3), 2020.

[Laf90] Yves Lafont. Interaction nets. In *Seventeenth Annual ACM Symposium on Principles of Programming Languages (POPL 1990)*, pages 95–108. ACM Press, 1990.

[LMSS92] Patrick Lincoln, John Mitchell, Andre Scedrov, and Natarajan Shankar. Decision problems for propositional linear logic. *Annals of Pure and Applied Logic*, 56(1):239 – 311, 1992.

[MP07] Damiano Mazza and Michele Pagani. The separation theorem for differential interaction nets. In *Logic for Programming, Artificial Intelligence, and Reasoning, 14th International Conference (LPAR 2007)*, volume 4790 of *Lecture Notes in Computer Science*, pages 393–407. Springer, 2007.

[MPV18] Damiano Mazza, Luc Pellissier, and Pierre Vial. Polyadic approximations, fibrations and intersection types. *PACMPL*, 2(POPL):6:1–6:28, 2018.

[MR14] Giulio Manzonetto and Domenico Ruoppolo. Relational graph models, Taylor expansion and extensionality. *Electronic Notes in Theoretical Compututer Science*, 308:245–272, 2014.

[Pag09] Michele Pagani. The cut-elimination theorem for differential nets with promotion. In *Typed Lambda Calculi and Applications, 9th International Conference, (TLCA 2009)*, volume 5608 of *Lecture Notes in Computer Science*, pages 219–233. Springer, 2009.

[Par92] Michel Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In *Logic Programming and Automated Reasoning,International Conference (LPAR '92)*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[PT09] Michele Pagani and Christine Tasson. The inverse Taylor expansion problem in linear logic. In *24th Annual Symposium on Logic in Computer Science (LICS 2009)*, pages 222–231. IEEE Computer Society, 2009.

[PTV16] Michele Pagani, Christine Tasson, and Lionel Vaux. Strong normalizability as a finiteness structure via the Taylor expansion of λ-terms. In *Foundations of Software Science and Computation Structures - 19th International Conference (FOSSACS 2016)*, volume 9634 of *Lecture Notes in Computer Science*, pages 408–423. Springer, 2016.

[Ret97] Christian Retoré. A semantic characterisation of the correctness of a proof net. *Mathematical Structures in Computer Science*, 7(5):445–452, 1997.

[Tas09] Christine Tasson. *Sémantiques et Syntaxes Vectorielles de la Logique Linéaire*. PhD thesis, Université Paris Diderot, France, December 2009.

[TdF03] Lorenzo Tortora de Falco. Additives of linear logic and normalization - part I: a (restricted) church-rosser property. *Theor. Comput. Sci.*, 294(3):489–524, 2003.

[Tra09] Paolo Tranquilli. Confluence of pure differential nets with promotion. In *Computer Science Logic, 23rd international Workshop, CSL 2009, 18th Annual Conference of the EACSL*, volume 5771 of *Lecture Notes in Computer Science*, pages 500–514. Springer, 2009.

[Tra11] Paolo Tranquilli. Intuitionistic differential nets and lambda-calculus. *Theor. Comput. Sci.*, 412(20):1979–1997, 2011.

[Vau17] Lionel Vaux. Taylor expansion, lambda-reduction and normalization. In *26th EACSL Annual Conference on Computer Science Logic (CSL 2017)*, volume 82 of *LIPIcs*, pages 39:1–39:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.