EPiC Series in Computing

Volume 100, 2024, Pages 464–483

Proceedings of 25th Conference on Logic for Programming, Artificial Intelligence and Reasoning

Confluence for Proof Nets via Parallel Cut Elimination

Giulio Guerrieri[1], Giulia Manara[2], Lorenzo Tortora de Falco[3], and Lionel Vaux Auclair[4]

[1] University of Sussex, Department of Informatics, Brighton, UK. g.guerrieri@sussex.ac.uk
[2] Université Paris Cité, IRIF, Paris, France. manara@irif.fr
[3] Università Roma Tre, Dipartimento di Matematica e Fisica, Rome, Italy; GNSAGA, Istituto Nazionale di Alta Matematica. tortora@uniroma3.it
[4] Aix-Marseille Université, I2M, Marseille, France. lionel.vaux@univ-amu.fr

Abstract

We provide the first proof of confluence for cut elimination in multiplicative-exponential linear logic proof-nets that is not based on Newman's lemma or strong normalization, not even indirectly. To achieve our goal, we take inspiration from Tait and Martin-Lof's method based on parallel reduction for the lambda-calculus, even though the wilder realm of untyped proof-nets makes the proof subtler and more challenging.

1 Introduction

A rewriting system is *confluent* if, for every term t, any two reduction sequences from t can be extended with some additional (possibly none) reduction steps to reach a common result. Confluence simplifies the study of the convergence of programs (represented by terms in the rewriting system) because it ensures that the ordering in which reductions are chosen does not make a difference to the result: of any two possible redexes in a term, the reduction of one instead of the other will never preclude the possibility of eventually reaching the same final result (a normal form), if any. In a way, confluence ensures the modularity of computation.

The Curry-Howard correspondence [6, 19] relates proofs to programs, and cut elimination (a proof rewriting technique to show the coherence of a proof system) to program execution. Thus, confluence of cut elimination has a computational relevance. It is well-known that the β-reduction of the λ-calculus is confluent, both in the typed and the untyped (hence non normalizing) settings: by the Curry-Howard correspondence, this gives confluence for cut elimination in natural deduction for intuitionistic logic. Cut elimination in intuitionistic sequent calculus LJ is also essentially confluent, but only up to structural commutations of rules due to the more *bureaucratic* nature of sequent calculus; by contrast, cut elimination in classical sequent calculus LK is fundamentally non confluent.

Girard [14] introduced linear logic (LL), a refinement of LJ and LK allowing a finer analysis of the use of hypotheses/resources in proofs/programs via the introduction of two dual modalities, the exponentials ! and ?. In LL (more precisely, in its one-sided sequent presentation) a proof of a formula of the form $!A$ (representing a resource available at will in cut elimination) can be deduced from a proof of A, provided the other formulas in the conclusion sequent are all of

the form $?B$. Girard also gave a graphical syntax to represent LL proofs via special directed graphs whose edges are labeled by LL formulas: *proof nets*. These are the analogue of natural deduction for LL, in the sense that they quotient out the bureaucracy of sequent calculus: the confluence of cut elimination is thus naturally expressed on proof nets. In LL proof nets, a formula $!A$ is introduced by means of a "box" marking the context of ?-formulas. Boxes are the only sub-graphs of a proof net that can be erased or duplicated at will during cut elimination.

Proof nets belong to a wider set of directed graphs labeled by LL formulas: *proof structures*. Not all proof structures are proof nets (that is, represent a proof in LL), but a geometrical criterion characterizes all and only the proof nets among the proof structures. Such a criterion, called AC boils down to an acyclicity property of proof structures. Cut elimination steps for LL can be defined on proof structures. And being AC (which is preserved by cut elimination steps) guarantees that cut elimination has good rewriting properties, such as *strong normalization* (every reduction sequence is finite), confluence and the fact that normal forms (the proof structures where no cut elimination step can be applied) are cut-free [14, 7, 11, 21].

Cut elimination actually generalizes to *untyped* proof structures, where edges are not labeled by LL formulas: for AC untyped proof structures, cut elimination is still confluent, and hence it has a clear computational meaning (despite their lack of a logical content), but weak—and hence strong—normalization fails (see [21, Fig. 9–10]) and normal forms are not necessarily cut-free [21, 10]. However, dropping AC, cut elimination for (typed or untyped) proof structures is not confluent (a counterexample is in [21, Figure 12]), not even in the *multiplicative-exponential fragment* of LL (MELL), in which the untyped λ-calculus can be encoded.

Goal. In the literature, for instance [7, 21], all the proofs of confluence of cut elimination for (typed or untyped) AC proof structures rely on Newman's lemma, which states that a reduction is confluent if it is strongly normalizing and *locally confluent* (that is, whenever two distinct reduction steps are applied to the same term, there is a term that is reachable from both, by applying some additional—possibly none—reduction steps). Newman's lemma can be applied directly to cut elimination for typed AC proof structures, but not to cut elimination for the untyped ones, since it is not strongly normalizing, as we have mentioned. In the untyped case, Newman's lemma is applied to some variant of cut elimination (finite development [7], or some subsets of the cut elimination steps that are separately strongly normalizing [21, 2]), and confluence of cut elimination is then deduced from confluence of the variant.

Our goal is to give a proof of confluence for MELL AC untyped proof structures that is not based on Newman's lemma and strong normalization, not even indirectly. We have chosen:

- MELL because it is expressive enough and its proof structures are easier to define than in LL;
- AC proof structures because otherwise confluence fails, as we have mentioned;
- untyped proof structures because cut elimination is strongly normalizing in the typed setting, and we do not want to use this property, not even surreptitiously; of course, the fact that our proof works for untyped proof structures means that it also works in the typed case.

This way, we disentangle confluence from strong normalization for MELL AC proof structures, which is a conceptual achievement and a technical simplification of the proof of confluence.

Method. To achieve our goal we use a basic result in rewriting theory. Given a reduction $\to$ and its reflexive-transitive closure $\to^*$ (the iteration of $\to$ an arbitrary—possibly zero—number of times), to prove that $\to$ is confluent it is enough to have a reduction $\dashrightarrow$ such that

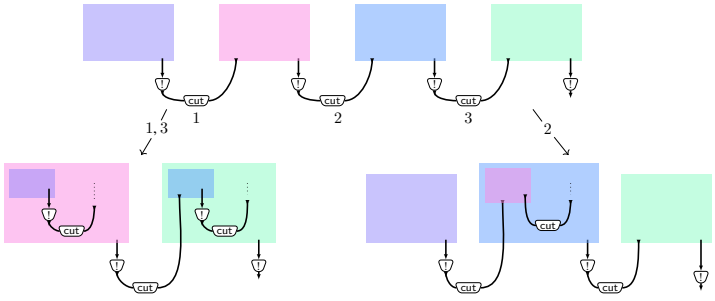


Figure 1: A chain of exponential cuts (box/box).

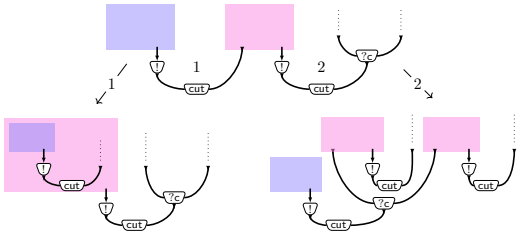


Figure 2: A chain of exponential cuts (box/box and box/contraction).

$\to \subseteq \dashrightarrow \subseteq \to^*$ and $\dashrightarrow$ is *diamond*: whenever two distinct reduction steps can be applied to the same term, there is a term that is reachable from both, by applying a single reduction step.

Let us see how this method had been successfully applied to the untyped λ-calculus: there, β-reduction $\to_\beta$ is neither strongly normalizing (indeed, $\delta\delta \to_\beta \delta\delta \to_\beta \dots$ for $\delta = \lambda x.xx$) nor diamond. For a counterexample to the diamond, consider the span below, where t has two redexes:

$$t_1 = (\lambda x.xx)z \;\; {}_\beta\!\leftarrow \;\; t = (\lambda x.xx)((\lambda y.y)z) \;\; \to_\beta \;\; ((\lambda y.y)z)((\lambda y.y)z) = t_2$$

The terms t_1 and t_2 have a common reduct zz, but it cannot be reached in one single $\to_\beta$ step. Indeed, the step $t \to_\beta ((\lambda y.y)z)((\lambda y.y)z)$ duplicates the rightmost redex in t.

To prove that $\to_\beta$ is confluent, Tait and Martin-Löf solved the problem introducing a technique, later simplified by Takahashi [23], based on *parallel β-reduction* $\Rightarrow_\beta$: in a single step, parallel β-reduction can reduce an arbitrary number (possibly zero) of existing redexes in a term. This leads to the desired properties: $\Rightarrow_\beta$ is diamond and $\to_\beta \subseteq \Rightarrow_\beta \subseteq \to_\beta^*$. Note how $\Rightarrow_\beta$ solves the problem presented above in just one (parallel) step: $(\lambda x.xx)z \Rightarrow_\beta zz \;{}_\beta\!\Leftarrow ((\lambda y.y)z)((\lambda y.y)z)$. This proof of confluence of $\to_\beta$ is one of the simplest and most elegant: it does not need to resort to Newman's lemma, and only relies on the two key properties of $\Rightarrow_\beta$.

Our approach to prove confluence of cut elimination for MELL AC untyped proof structures is then to adapt the notion of *parallel reduction* to this framework, in the spirit of what was done in [5] for the multiplicative fragment of LL (MLL).

Challenges. Choosing the right syntax for proof structures to obtain the diamond property for parallel reduction turned out to be a nontrivial problem. Examples in Figures 1 and 2, both involving the exponential cut elimination steps, show how the choice of standard syntax (with different ?-nodes for dereliction, weakening and contraction) prevents parallel reduction from being diamond. In Figure 1 the topmost proof structure can be reduced to the one on the

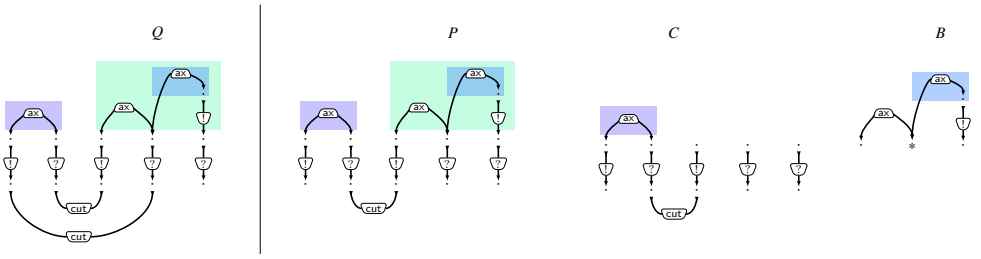


Figure 3: Q and P are MELL proof structures: P is AC but Q is not. P can be decomposed into the prestructure B (the content of the green box of P) and the prestructure C (the context).

right, eliminating the box/box cut 2, or in the one on the left, eliminating the box/box cuts 1 and 3: the only way to close this pair is to reduce to the net where the four boxes are fully nested, but to do so requires boxes to cross *more than one* border in a single step. In Figure 2 the topmost proof structure has a box/box cut 1 and a box/contraction cut 2 : to be diamond, parallel reduction from the right proof structure needs to *duplicate* the violet box *and* to put each copy *inside* the pink boxes.

These problems vanish by choosing a different syntax, namely the one introduced by Regnier [22, 9] with *generalized* ?*-nodes* (an arbitrary number—possibly zero—of premises of type A, one conclusion of $?A$). Exponential cut elimination simultaneously duplicates and opens boxes making them cross arbitrarily many box borders: exactly what we need.

Another difficulty is the fact that, unlike λ-terms, (typed or untyped) proof structures are *not* defined *inductively*. A λ-term t can be seen as a tree whose root allows one to identify its subterms, providing a *decomposition* of t so that parallel cut elimination can be easily defined inductively. On the other hand, an MELL proof structure is a graph without a clear notion of root, thus it is not evident how to decompose it, so as to define parallel cut elimination similarly to the λ-calculus. As boxes represent sub-proof structures, we exploit them: the idea is to *decompose* an MELL proof structure G with a box of content B, as the union of B and a context H in which B is pluggable in a specific way. However, in the syntax with generalized ?-nodes the content of a box is not necessarily a proof structure. In P in Figure 3, the content of the green box is the graph B in Figure 3: the blue box in B has a conclusion (denoted by $*$ in the figure) that is not plugged to a ?-node: this is not an MELL proof structure. Thus we consider a *more general* notion, called MELL *prestructures*: in these, exactly one conclusion of each box is linked to a !-node but the other conclusions are either linked to ? (as is required in proof structures), or left dangling as conclusions of the prestructure (as in the above example).

A third challenge comes with the fact that not every choice of a box is suitable for the purpose of defining parallel cut elimination. It is necessary to identify a box that is computationally independent of its context, meaning that cut elimination outside of the box does not change its content. In the λ-calculus the inductive term syntax eases the identification of an external redex, but this is trickier in MELL (pre)structures because of their inherent parallelism as graphs. Anyway, it is still possible to identify dependencies between boxes. A box B_1 *(computationally) depends* on a box B_2 if one of the non-! conclusions of B_1 is cut with the !-node of B_2. To define parallel cut elimination for MELL prestructures, we need to identify a box that has no dependencies with other boxes, that is, *none of its non-*! *conclusions is itself a premise of a cut*. We say that such a box is *external*. For example, in the proof structure P in Figure 3, the green box is external while the violet is not; in the proof structure Q in Figure 3, both the green and violet boxes are not external. Again we can find an analogy with the λ-calculus. In

particular, consider the inductive step below in the definition of parallel β-reduction:

$$\frac{s \Rightarrow_\beta s' \qquad u \Rightarrow_\beta u'}{(\lambda x.s)u \Rightarrow_\beta s'[u'/x]}\ \star$$

Let $\bar{t}$ be the MELL untyped proof structure encoding the λ-term t. The external redex $(\lambda x.s)u$ fired in $\star$ corresponds to a cut between a conclusion of $\overline{\lambda x.s}$ and the !-node in the border of a box containing the translation of $\overline{u}$. This box is always *external*. In every proof structure that is the image of some λ-term, there always exists an external box, which is not necessarily unique. More generally, what about MELL (untyped) proof structures, not necessarily in the image of the translation from the λ-calculus? A condition that guarantees the existence of an external box is *being* AC. In Figure 3, the proof structure P is AC and contains one external box, the green one, while Q is not AC and is with no external box.

To properly define parallel cut elimination, we also need a formal description of the exponential cut elimination step. We achieved this by defining a general notion of substitution of prestructures $H[B/x]$: this duplicates a prestructure B and cuts its copies against a set x of conclusions of H. Considering an exponential cut in some structure G between a box and a ?-node, and writing B for the content of the box and H for the context of the cut (that is, G with the cut, the box and the ?-node removed), $H[B/x]$ is the result of eliminating the cut.

Summing up. Finally, we can define the analogue of $\star$ for parallel cut elimination on MELL AC untyped prestructures as follows (this is the key ingredient in the definition):

$$\frac{H \Rightarrow H' \qquad B \Rightarrow B'}{H[B/x] \Rightarrow H'[B'/x]}$$

Thanks to this, we can prove confluence of cut elimination for MELL AC untyped prestructures, in a way similar to Tait–Martin-Löf's technique for the λ-calculus [23].

2 Notations and Preliminaries on Hypergraphs

Notations. Let X be a set. The cardinality of X is $|X|$, and the powerset of X is $\mathsf{P}(X)$. A function is *empty* if its domain is the empty set $\emptyset$. Given $f\colon X \to Y$, $X' \subseteq X$ and $Y' \subseteq Y$, we set:

- $f\restriction_{X'}\colon X' \to Y$ is the *restriction* of f to X', i.e. $f\restriction_{X'}(x) = f(x)$ for all $x \in X'$;
- $f(X') = \{f(x) \mid x \in X'\}$ is the *image* of X' via f;
- $f^{-1}(Y') = \{x \in X \mid f(x) \in Y'\}$ is the *preimage* of Y' via f; and $f^{-1}(y) = f^{-1}(\{y\})$ for $y \in Y$;
- if $g\colon W \to Z$ with $X \cap W = \emptyset$, then $f \cup g\colon (X \cup W) \to (Y \cup Z)$ is the *union* of f and g, i.e. $(f \cup g)(a) = f(a)$ if $a \in X$ and $(f \cup g)(a) = g(a)$ if $a \in W$.

For a partial function $f\colon X \rightharpoonup Y$, its *domain* and *image* are noted $\mathsf{dom}(f)$ and $\mathsf{im}(f)$ respectively. Given $R \subseteq A \times B$ and $\hat{A} \subseteq A$, the *restriction* of R to $\hat{A}$ is $R\restriction_{\hat{A}} = \{(a,b) \in R \mid a \in \hat{A}\}$.

We identify nested tuples with their flattened version, e.g. $(a,b,c) = (a,(b,c)) = ((a,b),c)$.

Hypergraphs. For technical convenience, we represent LL proof structures by (directed, labeled, ordered) *hypergraphs*, instead of graphs usually used in the LL literature [9, 11, 20, 10, 21, 15, 17]. We recall some basic notions about hypergraphs, as they may differ from the literature [12, 13]. We fix a countably infinite set $\mathcal{V}$ of *names* for the nodes and edges of our hypergraphs.

Definition 2.1 (Hypergraph)**.** *A (*undirected*)* hypergraph *is a triple* $(\mathcal{N}, \mathcal{E}, \mathsf{hyper})$*, where* $\mathcal{N} \subseteq \mathcal{V}$ *and* $\mathcal{E} \subseteq \mathcal{V}$ *are disjoint finite sets of* nodes *and (*hyper*)*edges *respectively,* $\mathsf{hyper}\colon \mathcal{E} \to \mathsf{P}(\mathcal{N}) \setminus \{\emptyset\}$ *and* $\mathcal{N} = \bigcup_{e\in\mathcal{E}} \mathsf{hyper}(e)$*. If* $n, m \in \mathsf{hyper}(e)$ *for some* $e \in \mathcal{E}$*, then* e links n *to* m*.*

This definition allows *multiple edges* (hyper may not be injective) and *dangling edges* ($e \in \mathcal{E}$ with $|\mathsf{hyper}(e)| = 1$, which we do not consider as loops in the definition of path, see Footnote 2 on p. 471). If $|\mathsf{hyper}(e)| \leq 2$ for all $e \in \mathcal{E}$, then the hypergraph is a *(multi-)graph.* The condition $\bigcup_{e\in\mathcal{E}} \mathsf{hyper}(e) = \mathcal{N}$ means that there are no isolated nodes (i.e. without any edge linked to them). Two hypergraphs are *disjoint* if the sets of their nodes and edges are.

Some structures can be put on top of a hypergraph: direction, labeling, order.

Definition 2.2 (Directed, labeled, ordered hypergraph)**.** *Let* $H = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$ *be a hypergraph.*

- *A* directed hypergraph *is* $D = (H, \mathsf{t}, \mathsf{s})$ *where* $\mathsf{t}\colon \mathcal{E} \to \mathsf{P}(\mathcal{N})$ *and* $\mathsf{s}\colon \mathcal{E} \to \mathsf{P}(\mathcal{N})$ *are such that* $\mathsf{t}(e) \cup \mathsf{s}(e) = \mathsf{hyper}(e)$ *for all* $e \in \mathcal{E}$*;*[1] *the elements of* $\mathsf{t}(e)$ *and* $\mathsf{s}(e)$ *are respectively the* targets *and* sources *of* e*. The* inputs *(or* hypotheses*) and* outputs *(or* conclusions*) of* D *are respectively the elements of* $\mathsf{in}(D) = \mathcal{N} \setminus \bigcup \mathsf{im}(\mathsf{t})$ *and* $\mathsf{out}(D) = \mathcal{N} \setminus \bigcup \mathsf{im}(\mathsf{s})$*.*
- *A* labeled hypergraph *is* (H, l) *where* $\mathsf{l}\colon \mathcal{E} \to A$ *associates with every edge its* label *in* A*.*
- *An* ordered hypergraph *is* $(H, <)$ *where* $<$ *is a strict partial order on* $\mathcal{N}$*.*

Let $D = (\mathcal{N}, \mathcal{E}, \mathsf{hyper}, \mathsf{t}, \mathsf{s})$ be a directed hypergraph: the *underlying* (undirected) hypergraph of D is $H_D = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$. As hyper in D can be deduced from t and s, we often omit it when defining a directed hypergraph. Given $e \in \mathcal{E}$ in D, the set of *predecessors* and *successors* of e are $\mathsf{pre}_D(e) = \{f \in \mathcal{E} : \mathsf{t}(f) \cap \mathsf{s}(e) \neq \emptyset\}$ and $\mathsf{suc}_D(e) = \{f \in \mathcal{E} : \mathsf{s}(f) \cap \mathsf{t}(e) \neq \emptyset\}$ respectively. By abuse of notation, we identify an hypergraph consisting of only one edge with the edge itself.

Different structures on a hypergraph H can combine modularly, for instance H can be endowed with a labeling l and an order $<$, so as to form a labeled ordered hypergraph $(H, \mathsf{l}, <)$.

We need a notion of structure-preserving map between hypergraphs.

Definition 2.3 (Morphism)**.** *Let* $H = (\mathcal{N}_H, \mathcal{E}_H, \mathsf{hyper}_H)$*,* $J = (\mathcal{N}_J, \mathcal{E}_J, \mathsf{hyper}_J)$ *be hypergraphs.*

*A (*plain*)* hypergraph morphism $\tau\colon H \to J$ *from* H *to* J *is a pair of functions* $(\tau_{\mathcal{N}}\colon \mathcal{N}_H \to \mathcal{N}_J$*,* $\tau_{\mathcal{E}}\colon \mathcal{E}_H \to \mathcal{E}_J)$ *such that* $\tau_{\mathcal{N}}(\mathsf{hyper}_H(e)) \subseteq \mathsf{hyper}_J(\tau_{\mathcal{E}}(e))$ *for all* $e \in \mathcal{E}_H$*.*

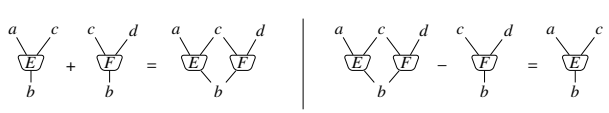
Given the directed hypergraphs $D = (H, \mathsf{t}_D, \mathsf{s}_D)$*,* $F = (J, \mathsf{t}_F, \mathsf{s}_F)$ *a* directed hypergraph morphism $\delta = (\delta_{\mathcal{N}}, \delta_{\mathcal{E}})\colon D \to F$ *is a hypergraph morphism such that* $\delta_{\mathcal{N}}(\mathsf{t}_D(e)) \subseteq \mathsf{t}_F(\delta_{\mathcal{E}}(e))$ *and* $\delta_{\mathcal{N}}(\mathsf{s}_D(e)) \subseteq \mathsf{s}_F(\delta_{\mathcal{E}}(e))$ *for all* $e \in \mathcal{E}_H$*.*

Given the labeled hypergraphs $G = (H, \mathsf{l}_G\colon \mathcal{E}_H \to A_G)$*,* $I = (J, \mathsf{l}_I\colon \mathcal{E}_J \to A_I)$ *with* $A_G \subseteq A_I$*, a* labeled hypergraph morphism $\delta = (\delta_{\mathcal{N}}, \delta_{\mathcal{E}})\colon G \to I$ *is a hypergraph morphism with* $\mathsf{l}_I \circ \delta_{\mathcal{E}} = \mathsf{l}_G$*.*

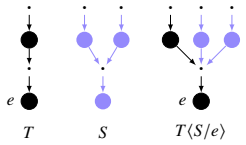
Given the ordered hypergraphs $O = (H, <_O)$*,* $P = (J, <_P)$*, an* ordered hypergraph morphism $\delta = (\delta_{\mathcal{N}}, \delta_{\mathcal{E}})\colon O \to P$ *is a hypergraph morphism where* $m <_O n$ *implies* $\delta_{\mathcal{N}}(m) <_P \delta_{\mathcal{N}}(n)$*.*

A plain or directed and/or labeled and/or ordered hypergraph morphism $\tau = (\tau_{\mathcal{N}}, \tau_{\mathcal{E}})\colon H \to J$ *is an* isomorphism *if there is a plain or directed and/or labeled and/or ordered hypergraph morphism* $\sigma = (\sigma_{\mathcal{N}}, \sigma_{\mathcal{E}})\colon J \to H$ *such that* $\sigma_{\mathcal{N}}$ *and* $\sigma_{\mathcal{E}}$ *are the inverses of* $\tau_{\mathcal{N}}$ *and* $\tau_{\mathcal{E}}$*, respectively.*

[1] So, there are three kinds of dangling edges ($e \in \mathcal{E}$ with $|\mathsf{t}(e) \cup \mathsf{s}(e)| = 1$): those coming in their target without a source, those going out their source without a target, and the loops going out and coming in the same node.



(a) Sum (on the left) and difference (on the right) of two hypergraphs (a, b, c, d are nodes, E, F are edges).



(b) Grafting the rooted tree S into T on e.

Figure 4: Some operations on hypergraphs (edges are depicted by lines or arrows with a bullet).

Sum of hypergraphs. Two hypergraphs $H = (\mathcal{N}_H, \mathcal{E}_H, \mathsf{hyper}_H)$ and $J = (\mathcal{N}_J, \mathcal{E}_J, \mathsf{hyper}_J)$ are *compatible* if $\mathcal{E}_H \cap \mathcal{E}_J = \emptyset$. The *sum* of compatible hypergraphs is the union of their edges and nodes: $H+J = (\mathcal{N}_H \cup \mathcal{N}_J, \mathcal{E}_H \cup \mathcal{E}_J, \mathsf{hyper}_H \cup \mathsf{hyper}_J)$. Note that the nodes in $\mathcal{N}_H \cap \mathcal{N}_J$ are merged, see the example in Figure 4a. If $\mathcal{N}_H \cap \mathcal{N}_J = \emptyset$, $H+J$ is the *disjoint sum* of H and J. Sums can be extended to directed hypergraphs: the sum of compatible $S_1 = (H_1, \mathsf{t}_1, \mathsf{s}_1)$ and $S_2 = (H_2, \mathsf{t}_2, \mathsf{s}_2)$ is $S_1 + S_2 = (S_1 + S_2, \mathsf{t}_1 \cup \mathsf{t}_2, \mathsf{s}_1 \cup \mathsf{s}_2)$. Similarly for labeled and/or ordered hypergraphs.

Sub-hypergraph, difference and intersection of hypergraphs. Let $J = (\mathcal{N}_J, \mathcal{E}_J, \mathsf{hyper}_J)$ and $H = (\mathcal{N}_H, \mathcal{E}_H, \mathsf{hyper}_H)$ be hypergraphs.

J is a *sub-hypergraph* of H if $\mathcal{N}_J \subseteq \mathcal{N}_H$, $\mathcal{E}_J \subseteq \mathcal{E}_H$, $\mathsf{hyper}_J(e) = \mathsf{hyper}_H(e) \cap \mathcal{N}_J$ for all $e \in \mathcal{E}_J$.

The *difference* $H-J = (\mathcal{N}_{H-J}, \mathcal{E}_{H-J}, \mathsf{hyper}_{H-J})$ removes J from H and is defined by: $\mathcal{E}_{H-J} = \mathcal{E}_H \setminus \mathcal{E}_J$ and $\mathcal{N}_{H-J} = \mathsf{hyper}_H(\mathcal{E}_{H-J})$ and $\mathsf{hyper}_{H-J} = \mathsf{hyper}_H \restriction_{\mathcal{E}_{H-J}}$. If $\mathsf{hyper}_H(e) = \mathsf{hyper}_J(e)$ for all $e \in \mathcal{E}_H \cap \mathcal{E}_J$, the *intersection* of H and J is $H \cap J = (\mathcal{N}_H \cap \mathcal{N}_J, \mathcal{E}_H \cap \mathcal{E}_J, \mathsf{hyper}_H \restriction_{\mathcal{E}_H \cap \mathcal{E}_J})$.

These notions can be extended to directed, labeled and ordered hypergraphs taking the restriction of the respective additional functions (and for the intersection, provided that the respective functions coincides on the shared edges).

Node substitution. Given a hypergraph $H = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$, a name n, a set $\mathbf{n}$ of names such that if $\mathbf{n} = \emptyset$ then $n \notin \mathcal{N}$, and $\varphi \colon \mathcal{E} \rightharpoonup \mathbf{n}$ such that $e \in \mathsf{dom}(\varphi)$ iff $n \in \mathsf{hyper}(e)$, we define the operation of *node substitution*, noted $H\,[\mathbf{n}/n]_\varphi$, that replaces the node n in H (if any) with the elements of $\mathbf{n}$ according to φ. Formally, the hypergraph $H\,[\mathbf{n}/n]_\varphi = (\mathcal{N}', \mathcal{E}, \mathsf{hyper}')$ is defined by:

$$\mathcal{N}' = \begin{cases} (\mathcal{N} \setminus \{n\}) \cup \mathbf{n} & \text{if } n \in \mathcal{N} \\ \mathcal{N} & \text{otherwise} \end{cases} \qquad \mathsf{hyper}'(e) = \begin{cases} (\mathsf{hyper}(e) \setminus \{n\}) \cup \{\varphi(e)\} & \text{if } n \in \mathsf{hyper}(e) \\ \mathsf{hyper}(e) & \text{otherwise.} \end{cases}$$

We set $\mathsf{sub}\,(H, n, \mathbf{n}) = \{H\,[\mathbf{n}/n]_\varphi \mid \varphi \colon \mathcal{E} \rightharpoonup \mathbf{n}$, and $e \in \mathsf{dom}(\varphi)$ iff $n \in \mathsf{hyper}(e)\}$, and $H\,[\mathbf{n}/n]$ denotes a generic element of $\mathsf{sub}\,(H, n, \mathbf{n})$. Note that when $\mathbf{n} \cap \mathcal{N} = \emptyset$, if the node n is linked to several edges and φ is injective, then $H\,[\mathbf{n}/n]_\varphi$ *linearizes* n, that is, disentangles n into distinct nodes, one for each edge that was linked to n in H. If $|\mathbf{n}| = 1$, $H\,[\mathbf{n}/n]_\varphi$ just *renames* the node n. If $n \notin \mathcal{N}$, then $H\,[\mathbf{n}/n]_\varphi = H$. Node substitution can be naturally extended to labeled and/or directed hypergraphs. Given an ordered hypergraph $O = (H, <)$, we set $O[\mathbf{n}/n]_\varphi = (H[\mathbf{n}/n]_\varphi, <')$ where $<' = < \cap (\mathcal{N}' \times \mathcal{N}')$ and $\mathcal{N}'$ is defined as above.

Refreshing. Given a hypergraph $H = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$ and a set $\mathcal{I} \subseteq \mathcal{N}$, a *$\mathcal{I}$-refreshing* of H is a pair (H', δ) where $H' = (\mathcal{N}', \mathcal{E}', \mathsf{hyper}')$ is a hypergraph compatible with H and $\delta = (\delta_\mathcal{N}, \delta_\mathcal{E}) \colon H \to H'$ is a hypergraph isomorphism such that $\delta_\mathcal{N}(n) = n$ if and only if $n \in \mathcal{I}$ (note that this implies that $\mathcal{N} \cap \mathcal{N}' = \mathcal{I}$). We denote by $\mathsf{refresh}\,(H, \mathcal{I})$ the set of $\mathcal{I}$-refreshings of H. The notion of $\mathcal{I}$-refreshing is extended to directed and/or labeled and/or ordered hypergraphs but just requiring that δ is a directed and/or labeled and/or ordered hypergraph isomorphism.

Trees and paths. A (*undirected*) *path* ρ in a hypergraph $H = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$ is a finite sequence $\rho = (n_0, e_1, n_1, \dots, e_k, n_k)$ alternating nodes and edges in H where $e_i \in \mathcal{E}$ and $n_{i-1}, n_i \in \mathsf{hyper}(e_i)$ for all $1 \leq i \leq k$, and $n_i \neq n_{i+1}$ for all $0 \leq i < k$, and $e_i \neq e_{i+1}$ for all $1 \leq i < k$.[2] The *length* of ρ is $\mathsf{len}(\rho) = k$; if $k = 0$, ρ is an *empty* path. If $k > 0$ and $n_0 = n_k$ (resp. $n_0 = n_k$ and $n_i \neq n_j$ for all $0 \leq i \neq j \leq k$), ρ is a *cycle* (resp. *minimal cycle*). The set of paths on H is noted $\mathsf{paths}(H)$.

A path $\rho = (n_0, e_1, n_1, \dots, e_k, n_k)$ in a directed hypergraph $D = (H, \mathsf{t}, \mathsf{s})$ is *directed* if $n_{i-1} \in \mathsf{s}(e_i)$ and $n_i \in \mathsf{t}(e_i)$ for all $1 \leq i \leq k$; we then say that ρ is *from* n_0 *to* n_k, noted $n_0 \,\rho\, n_k$. The set of directed paths in D is noted $\mathsf{dpaths}(D)$.

Two nodes in a hypergraph $H = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$ are *connected* if there is a path between them. A hypergraph is *connected* if any two distinct nodes are connected. A *connected component* of H is a maximal (with respect to the inclusions in $\mathcal{N}$ and $\mathcal{E}$) connected sub-hypergraph of H.

A *forest* is a hypergraph with no cycles. A *tree* is a connected forest. Note that a forest can be seen as the disjoint sum of trees where each tree is a connected component.

A *rooted tree* is a directed non-empty tree $T = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s})$ where $|\mathsf{s}^{-1}(\{n\})| = 1$ for all $n \in \mathcal{N}$ and $|\mathsf{s}(e)| = 1$ for all $e \in \mathcal{E}$, and $|\mathsf{t}(e)| = 1$ for all $e \in \mathcal{E}$ except one, the *root edge* $\mathsf{root}(T)$ of T, for which $\mathsf{t}(\mathsf{root}(T)) = \emptyset$. A *rooted subtree* S of a tree T is a sub-hypergraph of T that is a rooted tree and $\mathsf{pre}_S = \mathsf{pre}_T \restriction_{\mathcal{E}_S}$. The set of rooted subtrees of a tree T is noted $\mathsf{rTrees}(T)$, and the function $\mathsf{rsubtree}_T : \mathcal{E} \to \mathsf{rTrees}(T)$ associates with every $e \in \mathcal{E}$ the rooted tree $S \in \mathsf{rTrees}(T)$ such that $\mathsf{root}(S) = e$. A *rooted forest* is the disjoint sum of rooted trees.

The number of directed paths in a rooted tree $T = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s})$ is finite. Hence, we can define the *height* of a rooted tree T as $\mathsf{heig}(T) = \max\{\mathsf{len}(\rho) \in \mathbb{N} \mid \rho \in \mathsf{dpaths}(T)\}$, and the *reflexive-transitive closure* $T^{\circlearrowleft}$ of T as the directed hypergraph obtained by extending the links of each edge as follows: its sources are the same, its targets are all nodes reachable from its sources by a (possibly empty) direct path. Formally, $T^{\circlearrowleft} = (\mathcal{N}, \mathcal{E}, \mathsf{t}^{\circlearrowleft}, \mathsf{s})$ where, for all $e \in \mathcal{E}$, $\mathsf{t}^{\circlearrowleft}(e) = \{n \in \mathcal{N} \mid \exists\, m \in \mathsf{s}(e)\, \exists\, \rho \in \mathsf{dpaths}(T) : m \,\rho\, n\}$. Note that if T, U are rooted tees and $\tau : T \to U$ is a directed hypergraph morphism, then τ is also a directed hypergraph morphism from $T^{\circlearrowleft}$ to $U^{\circlearrowleft}$.

Operations on trees. Given two rooted trees, we can merge them so that the result of this union is still a rooted tree. Hence, we describe how to graft a rooted tree S into an edge e of another rooted tree T, yielding a rooted tree $T\langle S/e\rangle$. Formally, given two disjoint rooted trees $T = (\mathcal{N}_T, \mathcal{E}_T, \mathsf{t}_T, \mathsf{s}_T)$ and $S = (\mathcal{N}_S, \mathcal{E}_S, \mathsf{t}_S, \mathsf{s}_S)$ with $\mathsf{root}(S) = s$, the operation of *grafting* S into T on $e \in \mathcal{E}_T$, is $T\langle S/e\rangle = T + ((S - s)\,[\{m\}/n])$, where $\mathsf{s}_S(s) = \{n\}$ and $\mathsf{s}_T(e) = \{m\}$ (see Figure 4b).

Given the pairwise disjoint rooted trees $T, S_1, \dots, S_k$ and $e_1, \dots, e_k \in \mathcal{E}_T$ (not necessarily distinct), $T\langle (S_1, \dots, S_k)/(e_1, \dots, e_k)\rangle$ is the iteration $k \geq 0$ times of grafting S_i into T on e_i.

3 Prestructures

We present here a purely graphical definition of MELL proof structures, following the *non-inductive* approach of [16, 17, 18]: a MELL (*proof*) *structure* R is basically a labeled directed graph $|R|$ (here called MELL *shape*) together with additional information to identify its 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 all boxes in R. We use generalized n-ary ?-links collapsing weakening, dereliction and contraction, like in [22, 9]. 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 for contraction.

[2] Note that a path cannot loop on a node. This restriction simplifies the definition of cycle.

Figure 5: Edges for MELL shapes, with their directions and labels.

Our definition of MELL structure is completely based on standard mathematical tools (recalled in Section 2) from graph theory: it is rigorous and formal (with an eye to computer formalization) but avoids *ad hoc* technicalities to identify boxes. Indeed, one of the benefits of our purely graphical approach is that notions involving MELL structures such as isomorphism (see below), Taylor expansion (see [18]) and correctness graph (see [16]) are elegant and manageable.

Unlike [16, 17, 18], we actually use (directed, labeled, ordered) *hypergraphs* instead of graphs, because this allows a more concise definition of the crucial operation of substitution (Section 4). Thus, hyperedges are labeled by MELL connectives and linked to each other by means of nodes. Akin to [21, 10], to get more generality, we consider *untyped* MELL structures, that is, their nodes are not labeled by MELL formulas. Adapting our definition to the typed case is trivial.

We start with the definition of a MELL *shape*, the directed, labeled, ordered hypergraph underlying a MELL structure, without information about boxes. We actually generalize MELL shapes to *pseudoshapes*, since they are used later to define some technical operations.

Definition 3.1 (Pseudoshape, shape)**.** *A (*MELL*) pseudoshape $P = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s}, \mathsf{l}, <)$ is a directed (t, s), labeled (l), ordered $(<)$ hypergraph $H_P = (\mathcal{N}, \mathcal{E}, \mathsf{hyper})$ such that:*

- *the map $\mathsf{l} : \mathcal{E} \to \{\mathsf{cut}, \mathsf{ax}, ⅋, \otimes, !, ?, 1, \bot, ✠\}$ associates with each $e \in \mathcal{E}$ its* type *$\mathsf{l}(e)$, and e is then called a $\mathsf{l}(e)$-(*hyper*)*edge*; a* cut *(resp.* axiom*) is a* cut*-edge (resp.* ax*-edge);*
- *for every $e \in \mathcal{E}$ we have that $\mathsf{t}(e) \cap \mathsf{s}(e) = \emptyset$*[3] *and (see Figure 5):*
  - *if $\mathsf{l}(e) = \mathsf{cut}$ then $|\mathsf{s}(e)| = 2$ and $\mathsf{t}(e) = \emptyset$,*
  - *if $\mathsf{l}(e) = \mathsf{ax}$ then $\mathsf{s}(e) = \emptyset$ and $|\mathsf{t}(e)| = 2$,*
  - *if $\mathsf{l}(e) = ✠$ then $\mathsf{s}(e) = \emptyset$ and $\mathsf{t}(e) \neq \emptyset$,*
  - *if $\mathsf{l}(e) \in \{\otimes, ⅋\}$ then $|\mathsf{s}(e)| = 2$ and $|\mathsf{t}(e)| = 1$,*
  - *if $\mathsf{l}(e) \in \{!, ?\}$ then $|\mathsf{s}(e)| = 1$ and $|\mathsf{t}(e)| = 1$,*
  - *if $\mathsf{l}(e) \in \{1, \bot\}$ then $\mathsf{s}(e) = \emptyset$ and $|\mathsf{t}(e)| = 1$;*
- *$<$ is a total strict order defined only on $\mathsf{s}(e)$, for each ⅋- and $\otimes$-edge $e \in \mathcal{E}$, separately;*[4]
- *for all $n \in \mathcal{N}$, if $|\{f \in \mathcal{E} \mid n \in \mathsf{t}(f)\}| > 1$ then $n \in \mathsf{out}(P)$ or $n \in \mathsf{s}(e)$ for some ?-edge $e \in \mathcal{E}$;*[5]
- *for every $n \in \mathcal{N} \setminus \mathsf{out}(P)$, there is exactly one $e \in \mathcal{E}$, noted $\bar{\mathsf{s}}(n)$, such that $n \in \mathsf{s}(e)$.*[6]

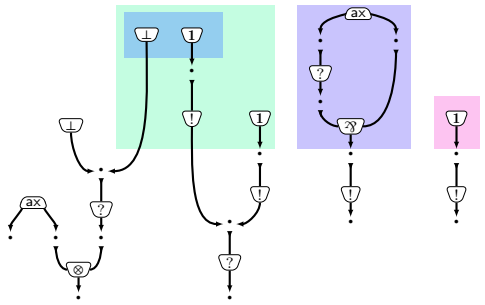
Given $\dagger \in \{1, \bot, \otimes, ⅋, !, ?\}$, we set $\dagger P = \{e \in \mathcal{E} : \mathsf{l}(e) = \dagger\}$. Given $e \in ?P$, e is a contraction *(resp.* dereliction*;* weakening*) if $|\mathsf{pre}_P(e)| > 1$ (resp. $|\mathsf{pre}_P(e)| = 1$; $\mathsf{pre}_P(e) = \emptyset$).*

A MELL shape *is a pseudoshape $P = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s}, \mathsf{l}, <)$ with no ✠-edge and such that if $n \in \mathsf{in}(P)$ then $n \in \mathsf{s}(e)$ for some $e \in ?P \cup !P$.*

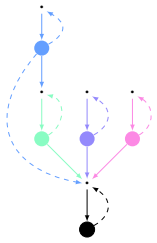
Note that a ?-edge e always has exactly one source: in this setting, the usual premises of a ?-link are the predecessor edges of e (if any). We also allow !-edges to be without a predecessor: when it is so, there is no box associated with the !-edge (see below).

Definition 3.2 (Kind of cut)**.** *Given a pseudoshape $P = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s}, \mathsf{l}, <)$ and a cut $e \in \mathcal{E}$, we say:*

[3] This condition means that a pseudoshape is a directed hypergraph with no loops (i.e. cycles of length 1).
[4] $<$ identifies the left and right source of each ⅋-/$\otimes$-edge, to define the multiplicative cut elimination step.
[5] So, the only nodes that can be targets of more than one edge are outputs or sources of ?-edges.
[6] It means that all nodes that are not outputs of P are the source of exactly one edge.



(a) The MELL shape $|P|$ (without colors).



(b) The box-tree $\mathcal{T}$ (without dotted lines) and its reflexive-transitive closure $\mathcal{T}^{\circlearrowleft}$ (with dotted lines).

Figure 6: A MELL structure $P = (|P|, \mathcal{T}, \mathsf{box})$.

- *e is* exponential *if* $\mathsf{l}(\mathsf{pre}_P(e)) = \{!, ?\}$*, and* active *if also* $\mathsf{pre}_P(i) \neq \emptyset$ *for the* !*-edge* $i \in \mathsf{pre}_P(e)$*;*
- *e is* multiplicative *if* $\mathsf{l}(\mathsf{pre}_P(e)) = \{⅋, \otimes\}$*; and e is* unit *if* $\mathsf{l}(\mathsf{pre}_P(e)) = \{\bot, 1\}$*;*
- *e is* axiom *if* $\mathsf{l}(f) = \mathsf{ax}$ *for some* $f \in \mathsf{pre}_P(e)$*.*

We define a MELL *(proof) structure* P as a MELL shape $|P|$ together with a tree representing the nesting of boxes in P (its root stands for the part of P outside any box) and a directed hypergraph morphism associating with each edge in P the smallest box containing it, if any.

Definition 3.3 (Prestructure, structure)**.** *A (*MELL*)* prestructure *is a tuple* $P = (|P|, \mathcal{T}, \mathsf{box})$ *where* $|P| = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s}, \mathsf{l}, <)$ *is a* MELL *shape,* $\mathcal{T} = (\mathcal{N}_\mathcal{T}, \mathcal{E}_\mathcal{T}, \mathsf{t}_\mathcal{T}, \mathsf{s}_\mathcal{T})$ *is a rooted tree called the* box-tree *of* P*, and* $\mathsf{box} = (\mathsf{box}_\mathcal{N}, \mathsf{box}_\mathcal{E}) \colon |P| \to \mathcal{T}^{\circlearrowleft}$ *is a directed hypergraph morphism such that:*

- $\mathsf{box}_\mathcal{N}(\mathsf{out}(|P|)) \subseteq \mathsf{s}_\mathcal{T}(\mathsf{root}(\mathcal{T}))$*;*[7]
- $\mathsf{box}_\mathcal{E} \colon \mathcal{E} \to \mathcal{E}_\mathcal{T}$ *induces a bijection from* $\bigcup_{e \in !|P|} \mathsf{pre}_{|P|}(e)$ *to* $\mathcal{E}_\mathcal{T} \setminus \{\mathsf{root}(\mathcal{T})\}$*, and for all* $e \in !|P|$*, if* $\mathsf{pre}_{|P|}(e) = \{f\}$ *then* $\mathsf{box}_\mathcal{E}(e) \neq \mathsf{box}_\mathcal{E}(f)$*;*[8]
- *for all* $e \in \mathcal{E}$*, if there exists* $f \in \mathsf{pre}_{|P|}(e)$ *such that* $\mathsf{box}_\mathcal{E}(e) \neq \mathsf{box}_\mathcal{E}(f)$ *then* $\mathsf{l}(e) \in \{!, ?\}$*.*[9]

We set $\mathcal{N}_P = \mathcal{N}$*,* $\mathcal{E}_P = \mathcal{E}$*,* $\mathsf{t}_P = \mathsf{t}$*,* $\mathsf{s}_P = \mathsf{s}$*,* $\mathsf{l}_P = \mathsf{l}$*,* $<_P = <$*,* $\mathsf{pre}_P = \mathsf{pre}_{|P|}$*,* $\mathsf{suc}_P = \mathsf{suc}_{|P|}$*,* $\mathsf{in}(P) = \mathsf{in}(|P|)$*,* $\mathsf{out}(P) = \mathsf{out}(|P|)$*,* $\dagger P = \dagger |P|$ *for all* $\dagger \in \{\otimes, ⅋, 1, \bot, ?, !\}$*. The* depth *of* P *is* $\mathsf{depth}(P) = \mathsf{heig}(\mathcal{T})$*, the* depth *of an edge* e *in* P *is* $\mathsf{depth}_P(e) = \mathsf{heig}(\mathcal{T}) - \mathsf{heig}(\mathsf{rsubtree}_\mathcal{T}(\mathsf{box}_\mathcal{E}(e)))$*.*

The prestructure P *is a* MELL *(*proof*)* structure *if* $\mathsf{in}(P) \subseteq \bigcup \mathsf{s}(?P)$[10] *and*

1. $\mathsf{t}(e) \cap \mathsf{out}(P) = \emptyset$*, for all* $e \in \mathcal{E}$ *such that* $\mathsf{box}_\mathcal{E}(e) \neq \mathsf{root}(\mathcal{T})$*;*[11]
2. *for all* $n \in \mathcal{N}$*, if* $|\{e \in \mathcal{E} \mid n \in \mathsf{t}(e)\}| > 1$ *then* $n \in \bigcup \mathsf{s}(?P)$*.*[12]

From the definition of directed hypergraph morphism, once $\mathsf{box}_\mathcal{E}$ is defined, $\mathsf{box}_\mathcal{N}$ is uniquely determined. Therefore we will often use the notation box to refer to $\mathsf{box}_\mathcal{E}$, and define $\mathsf{box}_\mathcal{E}$ only.

[7] It means that every output n of $|P|$ is outside any boxes, independently of the edges of which n is target.
[8] Hence, all and only the !-edges with a predecessor are associated with a box, to which they do not belong.
[9] It means that only the outputs of $|P|$ or the source of some ?- and !-edge can be in the border of a box.
[10] So, the inputs of a MELL structure can only be sources of ?-edges, thus there is no !-edge without its box.
[11] So, the targets of an edge with nonzero depth cannot be outputs of P, but must be the source of other edges.
[12] This means that the only nodes that can be targets of more than one edge are sources of ?-edges.

In our syntax, boxes do not have explicit constructors, hence the boxes of a MELL *structure* $P = (|P|, \mathcal{T}, \mathsf{box})$ are recovered in a non-inductive way. Roughly, given a subtree of $\mathcal{T}$ rooted at a non-root edge e, its preimage via box in $|P|$ is the *box* represented by e in $\mathcal{T}$. The preimage of the root of $\mathcal{T}$ via box is the part of $|P|$ at depth 0, that is, outside any box. The tree-structure of $\mathcal{T}$ expresses the usual *nesting* condition of boxes: two boxes in $|P|$ are either disjoint or contained one in the other. Since there is a one-to-one correspondence between boxes in $|P|$ and !-edges of P (see Footnotes 8 and 10), we can talk of the box of a !-edge.

A *prestructure* slightly generalizes a MELL structure: !-edges without predecessors (and so without a box) are allowed, as well as outputs at nonzero depth. This is needed to define parallel cut elimination that reduces independently both a cut inside a box (without the !- and ?-edges in the border) and a cut outside the box (with a !-edge without predecessors), see Section 7.

Example 3.4. Figure 6 depicts a MELL structure $P = (|P|, \mathcal{T}, \mathsf{box})$, where box is kept implicit by means of colors: the colored areas in $|P|$ represent boxes (the preimages of non-root edges of $\mathcal{T}$ via box), and the same color is used on $\mathcal{T}$ to show where each box is mapped by box. The MELL shape $|P|$ plus the colors for boxes (Figure 6a) uniquely determines the prestructure P: this is why in other figures we depict a prestructure as a MELL shape plus colors for its boxes. In particular, a prestructure with depth 0 (i.e. no boxes) can be identified with its MELL shape.

The MELL prestructures we have defined depend on the carrier sets for nodes and edges. It is natural to abstract from them, and identify two MELL prestructures that differ only in the name of their nodes and edges, through a handy notion of isomorphism (i.e. structure-preserving bijection) inherited from the one of hypergraph isomorphism defined in Section 2.

Definition 3.5 (Prestructure isomorphism)**.** *Let $P = (|P|, \mathcal{T}, \mathsf{box})$ and $P' = (|P'|, \mathcal{T}', \mathsf{box}')$ be prestructures. A* prestructure isomorphism $\tau \colon P \to P'$ *from P to P' is a pair $(\tau_{|\cdot|}, \tau_{\mathsf{tree}})$ where $\tau_{|\cdot|} \colon |P| \to |P'|$ is a directed, labeled, ordered hypergraph isomorphism, and $\tau_{\mathsf{tree}} \colon \mathcal{T} \to \mathcal{T}'$ is a directed hypergraph isomorphism, such that the diagram below commutes.*

$$\begin{array}{ccc} |P| & \xrightarrow{\mathsf{box}} & \mathcal{T}^{\circlearrowleft} \\ \downarrow{\scriptstyle \tau_{|\cdot|}} & & \downarrow{\scriptstyle \tau_{\mathsf{tree}}} \\ |P'| & \xrightarrow{\mathsf{box}'} & \mathcal{T}'^{\circlearrowleft} \end{array}$$

(The diagram actually stands for two diagrams: one for the nodes of $|P|$, $\mathcal{T}^{\circlearrowleft}$, $|P'|$, $\mathcal{T}'^{\circlearrowleft}$, and one for their edges)

We write $P \cong P'$ if there is a prestructure isomorphism $\tau \colon P \to P'$.

The idea is that MELL prestructures are identified up to the prestructure isomorphism $\cong$. Note that in a prestructure isomorphism $\tau = (\tau_{|\cdot|}, \tau_{\mathsf{tree}})$, the hypergraph isomorphisms $\tau_{|\cdot|}$ and τ_{tree} uniquely determine their mappings on nodes, once their mappings on edges are defined. Therefore, we will often define them on edges only.

Adding and deleting an edge in a prestructure. Let $P = (|P|, \mathcal{T}, \mathsf{box})$ be a prestructure with $|P| = (\mathcal{N}, \mathcal{E}, \mathsf{t}, \mathsf{s}, \mathsf{l}, <)$. We define the prestructures obtained by *adding* and *deleting* an edge in P from underneath, that is, in the outputs of P (which can be at any depth in a prestructure).

Let $(\mathcal{N}_e, \{e\}, \mathsf{t}_e, \mathsf{s}_e, \mathsf{l}_e, <_e)$ be a MELL shape (by abuse of notation, noted e) made only of an edge $e \notin \mathcal{E}$ with $\mathsf{t}_e(e) \cap \mathcal{N} = \emptyset$, $\mathsf{s}_e(e) \subseteq \mathsf{out}(P)$ and $\mathsf{l}_e(e) \notin \{!, ?\}$. Suppose that $\mathsf{box}(f) = \mathsf{box}(f')$ for all $f, f' \in \partial(P, e) = \{f \in \mathcal{E} \mid \mathsf{t}(f) \cap \mathsf{s}_e(e) \neq \emptyset\}$. We set $P + e = (|P| + e, \mathcal{T}, \mathsf{box}_{P+e})$ where $\mathsf{box}_{P+e}(e) = \mathsf{box}(f)$ if $f \in \partial(P, e) \neq \emptyset$, $\mathsf{box}_{P+e}(e) = \mathsf{root}(\mathcal{T})$ if $\partial(P, e) = \emptyset$, and $\mathsf{box}_{P+e} \restriction_{\mathcal{E}} = \mathsf{box}$.

Let $e \in \mathcal{E}$ be such that $\mathsf{t}(e) \subseteq \mathsf{out}(P)$ and $\mathsf{l}(e) \neq !$: we set $P - e = (|P| - e, \mathcal{T}, \mathsf{box} \restriction_{\mathcal{E} \setminus \{e\}})$.

4 Prestructure Substitution

We introduce a general notion of prestructure *substitution*—a key ingredient to define (parallel) cut elimination—inspired by term substitution for the λ-calculus. Given two prestructures P and B, an output o of B and a name n, *substituting* B for n in P via o, noted $P[B_o/n]$, is the analog of substituting the λ-term s for the free occurrences of the variable x in t: think of the tree-like representations of λ-terms to visualize the analogy. A difference with the λ-calculus is that, since prestructures do not have a tree-like structure or a distinguished output such as the root of a λ-term, the output o of B "leading" the substitution must be declared explicitly.

In prestructures, the role of the variable x is played by the name n, which is an output of P if it is a node of P, and the edges of P whose target is n represent its $k \geq 0$ (free) "occurrences" ($k = 0$ means that n is not a node of P): so, before performing the substitution, n is "linearized" in the nodes $n_1, \dots, n_k$, each one is the target of exactly one of the edges n is the target of (see the definition of node substitution $|P|\,[\{n_1, \dots, n_k\}/n]$, p. 470). For each of these k occurrences, a copy B_i of the MELL shape $|B|$ of the prestructure B should be provided and the substitution is performed by linking the copy o_i of the output b of $|B|$ to n_i via a cut-edge; for all $1 \leq i \leq k$, the copy B_i is obtained by renaming the edges and nodes of $|B|$, in particular its output b, but the other outputs of $|B|$ are not renamed because they are merged when performing the substitution (so, $B_i \in \mathsf{refresh}\,(|B|, \mathsf{out}(B) \setminus \{o\})$ according to the definition on p. 470). This way, B is *duplicated* k times, in particular if $k = 0$ then B is *erased*. In $P[B_o/n]$ the edges of B_i may have changed the boxes containing them, with respect to B: indeed, the new cut-edge linked to o_i is in the same boxes as the edge e_i whose target is n_i; this is obtained by grafting a copy of the box-tree of B into the box-tree of P on the image via box_P of e_i (see the definition of grafting, p. 471).

Some technical conditions must be fulfilled to perform a prestructure substitution $P[B_o/n]$.

Definition 4.1 (Substitutability)**.** *Let $P = (|P|, \mathcal{P}, \mathsf{box}_P)$ and $B = (|B|, \mathcal{B}, \mathsf{box}_B)$ be prestructures and $o \in \mathsf{out}(B) \setminus \mathcal{N}_P$ and n be a name: B is* substitutable *for n in P via o if the following hold:*

- *$|P|$ and $|B|$ are compatible, $n \notin \mathcal{N}_B$, and if $n \in \mathcal{N}_P$ then $n \in \mathsf{out}(P)$;*
- *for every $m \in \mathcal{N}_P \cap \mathcal{N}_B$, $m \in \mathsf{out}(B)$ and either $m \in \mathsf{out}(P)$, or $\bar{\mathsf{s}}_P(m) \in ?P$ has depth 0 in P;*
- *o is the target of only one edge e in B, and $\mathsf{depth}_B(e) = 0$.*

Assuming that $B = (|B|, \mathcal{B}, \mathsf{box}_B)$ is substitutable for n in $P = (|P|, \mathcal{P}, \mathsf{box}_P)$ via o, let us formally define the *substitution* $P[B_o/n]$ of B for n in P via o. Let $e_1, \dots, e_k$ be an enumeration of the edges in $\{e \in \mathcal{E}_P \mid n \in \mathsf{t}_P(e)\}$ ($k = 0$ if $n \notin \mathcal{N}_P$) and $(f_1, \dots, f_k) = (\mathsf{box}_P(e_1), \dots, \mathsf{box}_P(e_k))$. Let $(B_1, \gamma_1), \dots, (B_k, \gamma_k) \in \mathsf{refresh}\,(|B|, \mathsf{out}(B) \setminus \{o\})$ be pairwise compatible with $o_1 = \gamma_1(o), \dots, o_k = \gamma_k(o)$ pairwise distinct, and let $(\mathcal{B}_1, \delta_1), \dots, (\mathcal{B}_k, \delta_k) \in \mathsf{refresh}\,(\mathcal{B}, \emptyset)$ be pairwise compatible. For all $i \in \{1, \dots, k\}$, let $\mathsf{cut}_i \notin \mathcal{N}_P \cup \mathcal{N}_B$ be a cut-edge whose sources are n_i and o_i. We define $P[B_o/n] = (|P[B_o/n]|, \mathcal{T}, \mathsf{box})$ as follows:

$$|P[B_o/n]| = |P|\,[\{n_1, \dots, n_k\}/n] + B_1 + \dots + B_k + \mathsf{cut}_1 + \dots + \mathsf{cut}_k$$

$$\mathcal{T} = \mathcal{P}\langle(\mathcal{B}_1, \dots, \mathcal{B}_k)/(f_1, \dots, f_k)\rangle$$

$$\mathsf{box}(e) = \begin{cases} \mathsf{box}_P(e_i) & \text{if } e = \mathsf{cut}_i \text{ or } \mathsf{depth}_B(\gamma_i^{-1}(e)) = 0 \\ \delta_i \circ \mathsf{box}_B \circ \gamma_i^{-1}(e) & \text{if } \mathsf{depth}_B(\gamma_i^{-1}(e)) > 0 \\ \mathsf{box}_P(e) & \text{otherwise.} \end{cases}$$

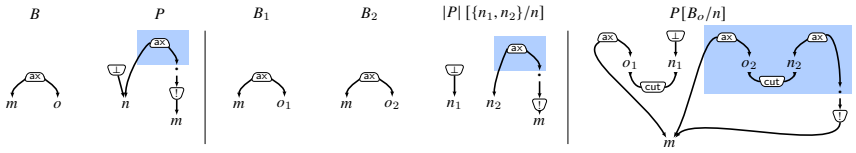


Figure 7: A prestructure substitution $P[B_o/n]$ (where $o, m, n, o_1, o_2, n_1, n_2$ are nodes).

An example of prestructure substitution is in Figure 7. From now on, given two prestructures P, B, and $o \in \mathsf{out}(B) \setminus \mathcal{N}_P$ and a name n, when we write $P[B_o/n]$ we assume B to be substitutable for n in P via o. Substitutability guarantees that prestructures are closed under substitution.

Lemma 4.2. *If P, B are prestructures and $b \in \mathsf{out}(B) \setminus \mathcal{N}_P$, then $P[B_o/n]$ is a prestructure.*

Remark 4.3. If $n \notin \mathsf{out}(P)$ then $P[B_o/n] = P$, as $\{f \in \mathcal{N}_P \mid n \in \mathsf{t}_P(f)\} = \emptyset$ and $|P|\,[\emptyset/n] = |P|$.

Prestructure substitution is commutative under a reasonable independence hypothesis.

Lemma 4.4 (Independence)**.** $P[A_p/n][B_o/m] \cong P[B_o/m][A_p/n]$ *whenever* $n \notin \mathcal{N}_B$ *and* $m \notin \mathcal{N}_A$.

5 Decomposing Prestructures

To define a notion of parallel cut elimination for prestructures analogously to parallel β-reduction for the λ-calculus, we need to decompose a prestructure into two prestructures and perform parallel cut elimination separately on them. As explained in Section 1, this operation is delicate and requires to use prestructures, instead of the more restrictive MELL structures.

When a prestructure $P = (|P|, \mathcal{T}, \mathsf{box})$ contains an !-edge b at depth 0 and with a predecessor, then P can be seen as "union" of two "sub-prestructures" of P: one is the *content of the box* of b, the other one is the *context* into which the box of b is plugged (see Figure 3). We provide two possible ways of decomposing a prestructure based on this idea.

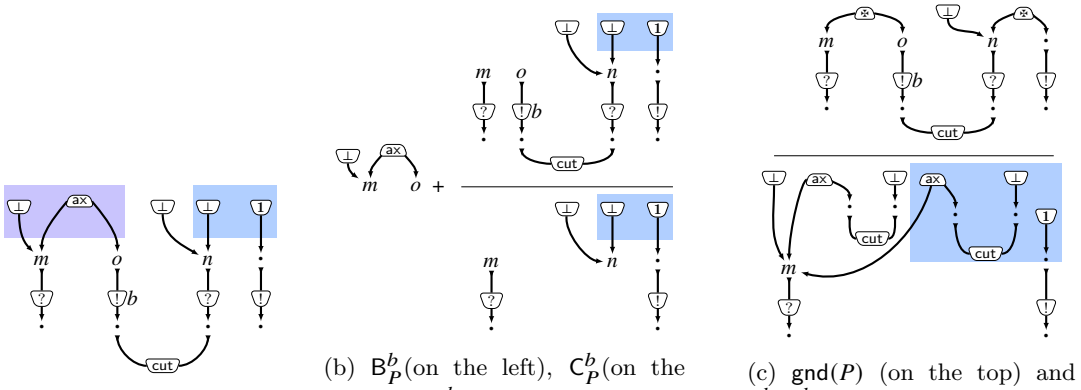
The content of a box. Every !-edge b of a prestructure $P = (|P|, \mathcal{T}, \mathsf{box})$ with $\mathsf{pre}_P(b) = \{b_0\}$ is associated with a prestructure called the *content of the box of b*, denoted by B^b_P, or simply B^b when no ambiguity arises. All information about this "sub-prestructure" of P is retrievable from box. Formally, $\mathsf{B}^b = (|\mathsf{B}^b|, \mathcal{B}, \mathsf{box}_{\mathsf{B}^b})$ where $|\mathsf{B}^b| = (\mathcal{N}_{|\mathsf{B}^b|}, \mathcal{E}_{|\mathsf{B}^b|}, \mathsf{t}_{|\mathsf{B}^b|}, \mathsf{s}_{|\mathsf{B}^b|}, \mathsf{l}_{|\mathsf{B}^b|}, <_{|\mathsf{B}^b|})$ and:

$$\mathcal{B} = \mathsf{rsubtree}_{\mathcal{T}}(\mathsf{box}(b_0)) \qquad \mathcal{E}_{|\mathsf{B}^b|} = \bigcup \mathsf{box}^{-1}(\mathcal{B}) \qquad \mathcal{N}_{|\mathsf{B}^b|} = \bigcup_{e \in \mathcal{E}_{|\mathsf{B}^b|}} (\mathsf{t}_P(e) \cup \mathsf{s}_P(e))$$

$$\mathsf{t}_{|\mathsf{B}^b|} = \mathsf{t}_P \restriction_{\mathcal{E}_{|\mathsf{B}^b|}} \quad \mathsf{s}_{|\mathsf{B}^b|} = \mathsf{s}_P \restriction_{\mathcal{E}_{|\mathsf{B}^b|}} \quad \mathsf{l}_{|\mathsf{B}^b|} = \mathsf{l}_P \restriction_{\mathcal{E}_{|\mathsf{B}^b|}} \quad <_{|\mathsf{B}^b|} = <_P \cap (\mathcal{N}_{|\mathsf{B}^b|} \times \mathcal{N}_{|\mathsf{B}^b|}) \quad \mathsf{box}_{\mathsf{B}^b} = \mathsf{box} \restriction_{\mathcal{E}_{|\mathsf{B}^b|}}$$

The !-edge b is the *principal port* of B^b. The set $\mathsf{bord}(b) = \{e \in \mathcal{E}_{|\mathsf{B}^b|} \mid \mathsf{suc}_P(e) \cap \mathcal{E}_{|\mathsf{B}^b|} = \emptyset\}$ is the *border* of B^b and $\mathsf{aux}(b) = \mathsf{suc}_P(\mathsf{bord}(b)) \setminus \{b\}$ is the set of *auxiliary ports* of B^b. Note that the principal and auxiliary ports of B^b are edges of P but not of B^b, and that $\mathsf{l}(\mathsf{aux}(b)) = \{?\}$.

First decomposition. Let $P = (|P|, \mathcal{T}, \mathsf{box})$ be a prestructure and $\mathsf{B}^b = (|\mathsf{B}^b|, \mathcal{B}, \mathsf{box}_{\mathsf{B}^b})$ be the content of the box of a !-edge b of P at depth 0 with a predecessor: the part of P that is not in B^b is still a prestructure, called the *context* of B^b (in P), denoted by C^b_P, or simply C^b when no ambiguity arises. Formally, $\mathsf{C}^b = (|\mathsf{C}^b|, \mathcal{T} - \mathcal{B}, \mathsf{box} \restriction_{\mathcal{E}_{\mathsf{C}^b}})$ where $|\mathsf{C}^b| = |P| - |\mathsf{B}^b|$ (the directed, labeled and ordered hypergraph obtained in this way is still a shape). Since P can be entirely



(a) A MELL structure P.

(b) B^b_P(on the left), C^b_P(on the top), and E^b_P(at the bottom).

(c) $\mathsf{gnd}(P)$ (on the top) and $\mathsf{E}^b[\mathsf{B}^b_o/n]$ (at the bottom).

Figure 8: A MELL structure $P = (|P|, \mathcal{T}, \mathsf{box})$, its two decompositions with respect to the !-edge b, the MELL structure $\mathsf{E}^b[\mathsf{B}^b_o/n]$ obtained by exponential cut elimination from P, and $\mathsf{gnd}(P)$.

rebuilt from B^b and C^b, we say that B^b and C^b *decompose* P and we write $P \equiv (\mathsf{C}^b, \mathsf{B}^b)$. See Figure 8b for an example, from the MELL structure P in Figure 8a: note that B^b_P and C^b_P are prestructures and not MELL structures there, even though P is a MELL structure.

Second decomposition. Let $P = (|P|, \mathcal{T}, \mathsf{box})$ be a prestructure and $\mathsf{B}^b = (|\mathsf{B}^b|, \mathcal{B}, \mathsf{box}_{\mathsf{B}^b})$ be the content of the box of a !-edge b of P at depth 0 with a predecessor. Suppose also that b is in turn a predecessor of an exponential cut c with $w \in ?P \cap \mathsf{pre}_P(c)$. We define another decomposition for P: drawing apart B^b and the exponential cut c with its predecessors b and w, the remaining part of P is still a prestructure, called the *exponential context of* B^b (in P), denoted by E^b_P or simply E^b when no ambiguity arises. Formally, $\mathsf{E}^b = (|\mathsf{E}^b|, \mathcal{T} - \mathcal{B}, \mathsf{box} \restriction_{\mathcal{E}_{\mathsf{E}^b}})$ with $|\mathsf{E}^b| = ((|\mathsf{C}^b| - c) - w) - b$. Since to rebuild P (up to $\cong$) it is enough to know B^b, E^b and the sources o and n of b and w, respectively (otherwise it is not determined where b and w must be linked to in B^b and/or E^b), we say that E^b and B^b *exponentially decompose* P via n and o, and we write $P \equiv (\mathsf{E}^b, \mathsf{B}^b, o, n)$. Note that the exponential cut c and its predecessors b and w are not edges of B^b or E^b, they are left implicit because for any exponential decomposition of P, it is intended that an exponential cut with its predecessors must be added so as to rebuild P up to $\cong$. See Figure 8b for an example, from the MELL structure P in Figure 8a: again, B^b_P and E^b_P are prestructures and not MELL structures there, even though P is a MELL structure.

Remark 5.1. Let P be a prestructure and b be a !-edge of P with at depth 0 with a predecessor. If $P \equiv (\mathsf{E}^b, \mathsf{B}^b, o, n)$ then B^b is substitutable for n in E^b via o (see Definition 4.1).

6 Acyclicity and External Bang

To give an inductive definition of parallel exponential cut elimination, we need to identify computationally independent "sub-prestructures" of a prestructure P: that is, P can be decomposed into two prestructures S_1 and S_2 such that: (1) all information about P can be retrieved from S_1 and S_2, and (2) parallel exponential cut elimination in S_1 does not affect S_2 (and vice-versa). In Section 5 we have seen how, given a !-edge b of P at depth 0 with $\mathsf{pre}_P(b) \neq \emptyset$, P can be decomposed into the prestructures B^b and C^b (or E^b if b is a predecessor of an exponential cut)

so that (1) is satisfied. For B^b and C^b to satisfy (2) it is necessary that no active exponential cut in C^b has an auxiliary port of B^b as a predecessor, see Figure 3 and the end of Section 1. This last condition concerns exponential cuts only because we parallelize exponential cut elimination steps only, as we will see in Section 7. Keeping this in mind we give the following definition:

Definition 6.1 (External bang)**.** *Let P be a prestructure, and $b \in \mathcal{E}_P$ be an !-edge at depth 0 in P with a predecessor: b is* external *(or an* external bang*) if the successor of every auxiliary port of the content B^b of the box of b is not an active exponential cut; b is* heavily external *(or a* heavily external bang*) if it is external and the successor of every auxiliary port of the content B^b of the box of b is not the predecessor of an active exponential cut.*

Although externality meets condition (2) discussed above, it is heavy externality—and not externality—that is preserved after a (parallel) exponential cut elimination step (see Section 7).

To ensure the presence of an external bang, we introduce an acyclicity condition AC on prestructures: the absence of specific cycles is checked outside the boxes and the acyclicity condition is inductively checked inside every the content of every box at depth 0. To do so, we use the first decomposition introduced in Section 5 and define an intermediate pseudoshape, called the ground pseudoshape, where each box at depth 0 is replaced by a ✠-edge with the same conclusions. For instance, the ground pseudoshape of the MELL structure P in Figure 8a is depicted in Figure 8c on the top.

Definition 6.2 (Ground pseudoshape)**.** *The* ground pseudoshape *of a prestructure P is the pseudoshape $\mathsf{gnd}(P) = \sum_{b\in A} e_b + \bigcap_{b\in A} |\mathsf{C}^b|$ where $A = \{e \in !P \mid \mathsf{depth}_P(e) = 0,\ \mathsf{pre}(e) \neq \emptyset\}$ and, for every $b \in A$, e_b is a ✠-edge whose targets are exactly the elements of $\mathcal{N}_{\mathsf{C}^b} \cap \mathcal{N}_{\mathsf{B}^b}$.*

A switching is defined in the standard way [8] as a boolean function that "chooses" exactly one predecessor for each ?- and ⅋-edge. Every switching on P induces an hypergraph where the ?-/⅋-edges have only one predecessor. It is on these hypergraphs that acyclicity is then tested.

Definition 6.3 (Switching)**.** *A* switching *of a prestructure P is a function* $\mathsf{swch}\colon (?\mathsf{gnd}(P) \cup \parr\mathsf{gnd}(P)) \times \mathcal{E}_{\mathsf{gnd}(P)} \to \{0,1\}$ *such that* $\mathsf{swch}(e_1,e_2) = 1$ *for all* $e_2 \notin \mathsf{pre}_{\mathsf{gnd}(P)}(e_1)$*, and for all* $e \in ?\mathsf{gnd}(P) \cup \parr\mathsf{gnd}(P)$ *there is exactly one* $(e_1,e_2) \in \{(e,f) \mid f \in \mathsf{pre}_{\mathsf{gnd}(P)}(e)\}$ *with* $\mathsf{swch}(e_1,e_2) = 1$.

Any switching swch *of P induces an undirected hypergraph* $\mathsf{swch}(P) = (\mathcal{N}, \mathcal{E}_{\mathsf{gnd}(P)}, \mathsf{hyper})$*, called the* switching hypergraph *of P via* swch*, where* $\mathsf{hyper}(e) = \mathsf{hyper}_{\mathsf{gnd}(P)}(e) \setminus \{n \mid \exists f \in \mathcal{E}_{\mathsf{gnd}(P)} : n \in \mathsf{t}(e) \cap \mathsf{s}(f), \mathsf{swch}(f,e) = 0\}$ *and* $\mathcal{N} = \bigcup_{e\in\mathcal{E}_{\mathsf{gnd}(P)}} \mathsf{hyper}(e)$.

Roughly, $\mathsf{swch}(P)$ is obtained from $\mathsf{gnd}(P)$ by considering each pair (e_1,e_2) of its edges where $\mathsf{l}(e_1) \in \{?, \parr\}$ and looking at their image via swch: if it is 1 then nothing changes, otherwise they are linked in $\mathsf{gnd}(P)$ and they become disconnected in $\mathsf{swch}(P)$.

Definition 6.4 (AC prestructure)**.** *A prestructure $P = (|P|, \mathcal{T}, \mathsf{box})$ is* AC *if:*

- $\mathsf{swch}(P)$ *is acyclic for every switching* swch *of P;*
- *for every $b \in !P$ with $\mathsf{depth}_P(b) = 0$ and $\mathsf{pre}_P(b) \neq \emptyset$, the content B^b of the box of b is* AC.

The following lemma shows that AC is sufficient to ensure the presence of an external bang.

Lemma 6.5. *If $P = (|P|, \mathcal{T}, \mathsf{box})$ is an* AC *prestructure such that $|\mathcal{E}_{\mathcal{T}}| > 1$, then there exists a heavily external bang $b \in \mathcal{E}_P$.*

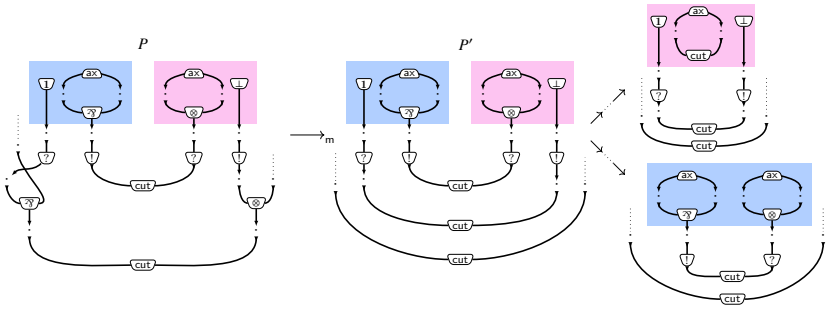


Figure 9: P is confluent with respect to exponential cut elimination, P' is not.

Remark 6.6. Lemma 6.5 is crucial to prove that parallel exponential cut elimination is diamond (see Section 7). The AC hypothesis in Lemma 6.5 can be relaxed because ⅋-, ⊗-, ⊥- and 1-edges do not play any role in its proof. The necessary and sufficient requirement for Lemma 6.5 and for the diamond of parallel exponential cut elimination is a form of "exponential" acyclicity where cycles only made of ax-, !-, ?-, cut- and ✠-edges have to be avoided (for example P in Figure 9 can be considered "exponential" acyclic). Anyway, AC is necessary to prove confluence for exponential *and* multiplicative cut elimination (see Section 8). The reason is that the set of "exponential" acyclic prestructures is not stable with respect to multiplicative cut elimination (the prestructure P in Figure 9 reduces via multiplicative cut elimination to P' which contains an "exponential" cycle). To use Hindley-Rosen lemma [4, Prop. 3.3.5] we need to work on a set that is closed with respect to a generic step of cut elimination: the set of AC prestructures.

Lemma 6.7. *Let P be a prestructure and b be a !-edge of P with at depth 0 with a predecessor. If $P \equiv (\mathsf{E}^b, \mathsf{B}^b, o, n)$ is* AC *then $\mathsf{E}^b[\mathsf{B}^b_o/n]$ is* AC.

7 Exponential Cut Elimination and its Parallel Version

In this section we define *exponential* cut elimination and *parallel exponential* cut elimination in prestructures. Cut elimination steps for other kinds of cut (unit, axiom, multiplicative) are defined in Section 8. To define (parallel) exponential cut elimination, the two kinds of prestructure decompositions (Section 5) and the notion of prestructure substitution (Section 4) play a crucial role. Moreover, we assume that the prestructure is AC to avoid dealing with pathological cases (such as a box whose principal port is cut with one of its auxiliary ports).

Definition 7.1 (Exponential cut elimination)**.** *Let $P = (|P|, \mathcal{T}, \mathsf{box})$ be an* AC *prestructure and $e \in \mathcal{E}_P$ be an active exponential cut. A prestructure P' is obtained from P by an* exponential cut elimination step *on e, noted $P \to_e P'$, when (the definition is by induction on $\mathsf{depth}_P(e)$):*

- $P' = \mathsf{E}^b\left[\mathsf{B}^b_o/n\right]$, *if $\mathsf{depth}_P(e) = 0$ and $P \equiv (\mathsf{E}^b, \mathsf{B}^b, o, n)$, where $b \in \mathsf{pre}_P(e) \cap\, !P$ and $w \in \mathsf{pre}_P(e) \cap\, ?P$ with $\mathsf{s}_P(b) = \{o\}$ and $\mathsf{s}_P(w) = \{n\}$;*
- $P' \equiv (\mathsf{C}^b, \mathsf{B}^b_1)$, *if $\mathsf{depth}_P(e) > 0$ and $P \equiv (\mathsf{C}^b, \mathsf{B}^b)$ with $b \in\, !P$ such that $\mathsf{depth}_P(b) = 0$ and $e \in \mathcal{E}_{\mathsf{B}^b}$ and $\mathsf{B}^b \to_e \mathsf{B}^b_1$.*[13]

[13]Note that if $\mathsf{depth}_P(e) > 0$ then there is $b \in\, !P$ with $\mathsf{depth}_P(b) = 0$ and $e \in \mathcal{E}_{\mathsf{B}^b}$, hence we can decompose $P \equiv (\mathsf{C}^b, \mathsf{B}^b)$ with $\mathsf{depth}_{\mathsf{B}^b}(e) < \mathsf{depth}_P(e)$ and so $\mathsf{B}^b \to_e \mathsf{B}^b_1$ is well defined by induction hypothesis.

We write $P \to_{\mathsf{e}} P'$ if $P \to_e P'$ for some active exponential cut $e \in \mathcal{E}_P$.

For instance, $P \to_e P'$ where P is the MELL structure in Figure 8a, P' is the MELL structure in Figure 8c (at the bottom) and e is the only cut-edge in P.

Parallel exponential cut elimination fires a number of active exponential cuts (possibly none) simultaneously. To define it in a AC prestructure P, we fix an external bang $b \in \mathcal{E}_P$ (if any) as a "point of view" on P, and we consider the content B^b_P of the box of b and the two decompositions of P based on the context C^b_P and the exponential context E^b_P (Section 5). The former decomposition inductively propagates parallel exponential cut elimination inside boxes, the latter one fires an active exponential cut *and* propagate parallel exponential cut elimination.

Definition 7.2 (Parallel exponential cut elimination)**.** *Given an* AC *prestructure P, we write $P \Rightarrow P'$ if either $P \cong P'$ or there is an external bang $b \in \mathcal{E}_P$ and*

1. *either $P \equiv (\mathsf{E}^b, \mathsf{B}^b, o, n)$ and $P' = E'[B'_o/n]$ with $\mathsf{E}^b \Rightarrow E'$ and $\mathsf{B}^b \Rightarrow B'$;*
2. *or $P \equiv (\mathsf{C}^b, \mathsf{B}^b)$ and $P' \equiv (E', B')$ with $\mathsf{C}^b \Rightarrow E'$ and $\mathsf{B}^b \Rightarrow B'$.*

If $P \Rightarrow P'$ and P' is obtained as in Items 1. *or* 2.*, we write $P \Rightarrow_b P'$.*

Note that $\Rightarrow$ is a reflexive (up to $\cong$) binary relation on AC prestructures, and that if there is no external bang in P then P has no active exponential cuts.

The next two lemmas are used to prove that parallel exponential cut elimination is diamond (Theorem 7.5), and rely on the properties of substitution and decomposition. The first one states that the result of a $\Rightarrow_b$ step is independent of the choice of b. The second one is the key lemma about substitution (analogously to parallel β-reduction for the λ-calculus).

Lemma 7.3. *Let $P \not\cong P'$ be prestructures. If $P \Rightarrow P'$ and $b \in \mathcal{E}_P$ is an external bang then $P \Rightarrow_b P'$.*

Lemma 7.4 (Parallel substitution)**.** *Let P, Q be* AC *prestructures such that Q is substitutable for n in P via o. Suppose that for all $e \in P$ such that there exists $m \in \mathsf{out}(Q)$ with $e = \bar{\mathsf{s}}_P(m)$, e is not a predecessor of an active exponential cut. If $P \Rightarrow P'$ and $Q \Rightarrow Q'$, then $P[Q_o/n] \Rightarrow P'[Q'_o/n]$.*

We recall the definitions of confluence and diamond, abstractly. Let $\to$ be a binary relation on a set $\mathcal{A}$, and $\to^*$ be the reflexive-transitive closure of $\to$: $\to$ is *diamond* if for all $a, b, c \in \mathcal{A}$ such that $a \to b$ and $a \to c$, there is $a' \in \mathcal{A}$ such that $b \to a'$ and $c \to a'$; $\to$ is *confluent* if $\to^*$ is diamond. Clearly, diamond implies confluence. Confluence has some important consequences, which we mention to show the relevance of this property. If $\to$ is confluent, then:

- (*uniqueness of normal forms)* for all $a \in \mathcal{A}$, if $a \to^* b$ and $a \to^* c$ for some *normal* $b, c \in \mathcal{A}$ (that is, there are no $b', c' \in \mathcal{A}$ such that $b \to b'$ and $c \to c'$), then $b = c$;
- (*Church-Rosser property*) for all $a, b \in \mathcal{A}$, if $a \leftrightarrow^* b$ (where $\leftrightarrow^*$ is the symmetric, reflexive, transitive closure of $\to$) then $a \to^* c$ and $b \to^* c$ for some $c \in \mathcal{A}$.

Theorem 7.5 (Diamond)**.** *$\Rightarrow$ is diamond on* AC *prestructures (up to $\cong$).*

Corollary 7.6 (Exponential confluence)**.** *$\to_{\mathsf{e}}$ is confluent on* AC *prestructures (up to $\cong$).*

Proof. As $\to_{\mathsf{e}} \subseteq \Rightarrow \subseteq \to^*_{\mathsf{e}}$ (trivial), this is an immediate consequence of Theorem 7.5. □

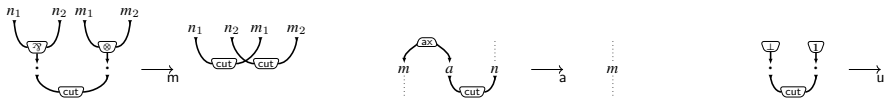


Figure 10: Multiplicative ($\to_{\mathsf{m}}$), axiom ($\to_{\mathsf{a}}$) and unit ($\to_{\mathsf{u}}$) cut elimination.

8 Cut Elimination and its Confluence

In this section, we define the cut elimination steps other than exponential: multiplicative, unit, axiom (see Figure 10). We do not define a parallel version of them. Indeed, we only parallelize *exponential* cut elimination steps because they are the only ones that can duplicate parts of a prestructure, and so they are the only ones requiring parallel reduction to prove their confluence. Confluence of whole (exponential, axiom, unit and multiplicative) cut elimination is then proved via Hindley-Rosen lemma [4, Proposition 3.3.5], after showing that exponential, axiom, unit and multiplicative cut eliminations are separately confluent and commute each other.

Let $P = (|P|, \mathcal{T}, \mathsf{box})$ be a prestructure and c be a cut of P. A prestructure P' is obtained from P by a *non-exponential cut elimination step on* c, noted $P \to_c P'$, if one of the following holds:

- c is a multiplicative cut with predecessors e and f, where $\mathsf{s}_P(e) = \{n_1, n_2\}$ with $n_1 <_P n_2$ and $\mathsf{s}_P(f) = \{m_1, m_2\}$ with $m_1 <_P m_2$, and $P' = (P - c - e - f) + c_1 + c_2$ where $c_i \notin \mathcal{E}_P$ is a cut-edge whose sources are n_i and m_i, for all $i \in \{1, 2\}$;
- if c is an axiom cut having the ax-edge a as a predecessor with $\mathsf{s}_P(c) = \{n, o\}$ and $\mathsf{t}_P(a) = \{m, o\}$ and $m \neq n$, and $P' = (|P'|, \mathcal{T}, \mathsf{box} \restriction_{\mathcal{E}_{P'}})$ with $|P'| = (P - c - a)\,[\{m\}/n]$;
- if c is a unit cut having predecessors e and f, and $P' = ((P - c) - e) - f$.

We write $P \to_{\mathsf{m}} P'$ (resp. $P \to_{\mathsf{a}} P'$; $P \to_{\mathsf{u}} P'$) if $P \to_c P'$ for some multiplicative (resp. axiom; unit) cut $c \in \mathcal{E}_P$. *Cut elimination* is the binary relation $\to = \to_{\mathsf{m}} \cup \to_{\mathsf{a}} \cup \to_{\mathsf{u}} \cup \to_{\mathsf{e}}$.

The property below (stability of AC under cut elimination) is crucial. For MELL untyped proof structures, it has been proved in [10]. Its generalization to prestructures is trivial.

Proposition 8.1. *If P is an AC MELL prestructure (resp. structure) and $P \to_{\mathsf{x}} Q$ with $\mathsf{x} \in \{\mathsf{u}, \mathsf{m}, \mathsf{a}, \mathsf{e}\}$, then Q is an AC MELL prestructure (resp. structure).*

Lemma 8.2 (Commutations)**.** *Let P be a prestructure.*

Let $e, c \in \mathcal{E}_P$ where e is an active exponential cut and c is a multiplicative (resp. axiom; unit) cut. If P is AC and $P \to_e P_1$ and $P \to_c P_2$, then $P_1 \to_{\mathsf{x}}^* P'$ and $P_2 \to_e P''$ for some prestructures $P' \cong P''$, with $\mathsf{x} = \mathsf{m}$ (resp. $\mathsf{x} = \mathsf{a}$; $\mathsf{x} = \mathsf{u}$).

Let $m, u \in \mathcal{E}_P$ where m is a multiplicative cut and u is a unit cut. If $P \to_m P_1$ and $P \to_u P_2$, then $P_1 \to_u P'$ and $P_2 \to_m P'$ for some prestructure P'.

Let $a, c \in \mathcal{E}_P$ where a is an axiom cut and c is an multiplicative (resp. unit) cut. If $P \to_a P_1$ and $P \to_c P_2$, then $P_1 \to_c P'$ and $P_2 \to_a P''$ for some prestructures $P' \cong P''$.

Theorem 8.3. *Cut elimination $\to$ on AC prestructures is confluent (up to $\cong$).*

Proof. It follows from Corollary 7.6, Lemma 8.2, the fact that $\to_{\mathsf{m}}$, $\to_{\mathsf{a}}$ and $\to_{\mathsf{u}}$ are separately confluent up to $\cong$ (immediate, since they do not duplicate or erase anything, they just modify a prestructure locally), and the Hindley-Rosen lemma [4, Proposition 3.3.5]. □

Corollary 8.4. *Cut elimination $\to$ on AC MELL structures is confluent (up to $\cong$).*

Proof. Immediate consequence of Theorem 8.3 and Proposition 8.1. □

Thus, once the confluence of cut elimination on AC prestructures is established (Theorem 8.3), we easily deduced the confluence of cut elimination on AC MELL structures, the result we are interested in. Therefore, extending our study of confluence to prestructures is just a technical device to ease our definitions and proofs, and we are not aware of alternative approaches to use parallel cut elimination, taking into account the issues discussed in Section 1.

Conclusions. We provided the first proof of confluence for cut elimination in (untyped) AC MELL proof structures that is not based on Newman's lemma or strong normalization, not even indirectly. To achieve our goal, we introduced two main tools:

- parallel cut elimination, inspired by Tait and Martin-Lof's on parallel reduction for the λ-calculus, and defined only for exponential cut elimination steps;
- prestructures, generalizing proof structures to ease the definition of parallel cut elimination.

Since proving confluence requires to show that two objects are equal, we developed a formalism to rigorously define the identity on prestructures (the notion of prestructure isomorphism, based on graph-theoretical tools) and the other notions required to prove confluence. Despite some heaviness, we believe that our rigorous approach is needed to avoid hand-waving proofs, and it can be seen as a first step towards a formalization of MELL proof nets in theorem provers.

We defined parallel cut elimination for exponential steps only, because they are the only ones that can duplicate some parts of a prestructure and hence need to be parallelized to prove confluence via Tait and Martin-Lof's method. In a wider perspective, our work paves the way to the general definition of parallel cut elimination for (untyped) AC MELL, not only for exponential steps: as parallel reduction of the λ-calculus has been used to prove several operational properties [23, 1, 3], we believe that parallel cut elimination of untyped MELL can serve as a powerful tool in the general theory of proof nets (and its applications).

Acknowledgments. This project is funded by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 945332.

References

[1] Beniamino Accattoli. An abstract factorization theorem for explicit substitutions. In *23rd International Conference on Rewriting Techniques and Applications (RTA'12)*, volume 15 of *LIPIcs*, pages 6–21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.

[2] Beniamino Accattoli. Exponentials as substitutions and the cost of cut elimination in linear logic. *Log. Methods Comput. Sci.*, 19(4), 2023.

[3] Beniamino Accattoli, Claudia Faggian, and Giulio Guerrieri. Factorization and normalization, essentially. In Anthony Widjaja Lin, editor, *Programming Languages and Systems - 17th Asian Symposium, APLAS 2019, Nusa Dua, Bali, Indonesia, December 1-4, 2019, Proceedings*, volume 11893 of *Lecture Notes in Computer Science*, pages 159–180. Springer, 2019.

[4] Henk Barendregt. *The Lambda Calculus: Its Syntax and Semantics*, volume 103 of *Studies in logic and the foundation of mathematics*. North-Holland, Amsterdam, revised edition, 1984.

[5] Jules Chouquet and Lionel Vaux Auclair. An application of parallel cut elimination in multiplicative linear logic to the taylor expansion of proof nets. *Log. Methods Comput. Sci.*, 17(4), 2021.

[6] Haskell B. Curry and Robert. Feys. *Combinatory Logic*. Number vol. 2 in Combinatory Logic. North-Holland Publishing Company, 1958.

[7] Vincent Danos. *La Logique Linéaire appliquée à l'étude de divers processus de normalisation (principalement du Lambda-calcul)*. PhD thesis, Université Paris VII, 1990.

[8] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Arch. Math. Log.*, 28(3):181–203, 1989.

[9] Vincent Danos and Laurent Regnier. Proof-nets and the hilbert space. In Jean-Yves Girard, Yves Lafont, and Laurent Editors Regnier, editors, *Advances in Linear Logic*, London Mathematical Society Lecture Note Series, page 307–328. Cambridge University Press, 1995.

[10] Daniel de Carvalho, Michele Pagani, and Lorenzo Tortora de Falco. A semantic measure of the execution time in linear logic. *Theoretical Computer Science*, 412(20):1884–1902, 2011. Girard's Festschrift.

[11] Roberto Di Cosmo and Stefano Guerrini. Strong normalization of proof nets modulo structural congruences. In Paliath Narendran and Michaël Rusinowitch, editors, *Rewriting Techniques and Applications, 10th International Conference, RTA-99, Trento, Italy, July 2-4, 1999, Proceedings*, volume 1631 of *Lecture Notes in Computer Science*, pages 75–89. Springer, 1999.

[12] Willibald Dörfler and Derek A. Waller. A category-theoretical approach to hypergraphs. *Archiv der Mathematik*, 34:185–192, 1980.

[13] Giorgio Gallo, Giustino Longo, and Stefano Pallottino. Directed hypergraphs and applications. *Discret. Appl. Math.*, 42(2):177–201, 1993.

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

[15] 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 für Informatik, 2016.

[16] 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.

[17] 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.

[18] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Gluing resource proof-structures: inhabitation and inverting the taylor expansion. *Log. Methods Comput. Sci.*, 18(2), 2022.

[19] William Alvin Howard. The formulae-as-types notion of construction. In Haskell Curry, Hindley B., Seldin J. Roger, and P. Jonathan, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*. Academic Press, 1980.

[20] Olivier Laurent. Polarized proof-nets and lambda-μ-calculus. *Theor. Comput. Sci.*, 290(1):161–188, 2003.

[21] Michele Pagani and Lorenzo Tortora de Falco. Strong normalization property for second order linear logic. *Theoretical Computer Science*, 411(2):410–444, 2010.

[22] Laurent Régnier. *Lambda-calcul et reseaux*. PhD thesis, Université Paris VII, 1992.

[23] Masako Takahashi. Parallel reductions in lambda-calculus. *Inf. Comput.*, 118(1):120–127, 1995.