Compressing Polarized Boxes

Beniamino Accattoli
Carnegie Mellon University
Pittsburgh, PA, USA

***Abstract*—The sequential nature of sequent calculus provides a simple definition of cut-elimination rules that duplicate or erase sub-proofs. The parallel nature of proof nets, instead, requires the introduction of explicit boxes, which are global and synchronous constraints on the structure of graphs. We show that logical polarity can be exploited to obtain an implicit, compact, and natural representation of boxes: in an expressive polarized dialect of linear logic, boxes may be represented by simply recording some of the polarity changes occurring in the box at level 0. The content of the box can then be recovered locally and unambiguously. Moreover, implicit boxes are more parallel than explicit boxes, as they realize a larger quotient. We provide a correctness criterion and study the novel and subtle cut-elimination dynamics induced by implicit boxes, proving confluence and strong normalization.**

Introduction

Gentzen's sequent calculus is a standard formalism for a wide variety of logics. However, sequent calculus forces an order among deduction steps even when they are evidently independent, a drawback called *bureaucracy*. Proof nets, a graphical syntax introduced by Girard in the context of linear logic [1], [2], are a parallel and bureaucracy-free representation of proofs. They also provide new and elegant cut-elimination methods, drastically reducing the need of commutative steps, the non-interesting burden in every sequent calculus proof of cut-admissibility.

Sequents, proof nets, and boxes. Some cut-elimination rules require us to duplicate or erase whole sub-proofs, typically the rules for the ! modality in linear logic. Proofs in sequent calculus are tree-shaped and bear a clear notion of *last* rule, the root of the tree. This property has an obvious but important consequence: given a !-rule r in a proof, there is an evident sub-proof ending on r, the sub-tree rooted in r. Therefore, non-linear cut-elimination rules can easily be defined by duplicating or erasing sub-trees, as in Figure 1.a.

Switching to proof nets, the situation radically changes. Proofs are now represented as graphs, retaining only the causality between rules and without any bureaucracy. A proof net in general has many last rules, one for each formula in the last sequent. Unfortunately, given a rule r it is not clear how to find a sub-proof net ending on r, because the causality retained by proof nets does not provide enough information. Thus, in order to eliminate cuts such as the one in Figure 1.b—which require sub-proof nets—some information has to be added. The typical solution is to re-introduce part of the bureaucracy, pairing each !-rule with an *explicit box* containing the sub-proof to duplicate or erase. The rule is then implemented as in Figure 1.c.

There has been extensive research on boxes, regarding how to represent them [3]–[6], how to decompose their evaluation [4], [5], [7]–[9], how they control cut-elimination [10], [11], how to see them categorically [12], [13], how to use them for the additive connectives [14], and concerning their relationship with explicit substitutions [6], [15]. In particular, the study of boxes and their dynamics has contributed greatly to the understanding of optimal reductions [8], [9], [16], and has also resulted in a new approach to implicit computational complexity [10], [11]. More recently, the discovery of simple implicit boxes for λ-calculus [6] led to a renewal of the theory of explicit substitutions [17]–[19], which in turn led to new results on λ-calculus [17], [20]–[22].

Compressing polarized boxes. In this paper we show that in an expressive polarized dialect of linear logic it is possible to take advantage of the polarized structure of proofs to *compress* explicit !-boxes. The idea is to associate with every explicit box a subset of its content from which it may be reconstructed, exploiting polarity to keep this subset compact.

The logic is Laurent's Polarized Multiplicative and Exponential Linear Logic (MELLP) [23], which may be understood as the fragment of linear logic interpreting classical logic or Parigot's $\lambda\mu$-calculus [24]. This logic derives from Girard's classical system LC [25], [26] and it is built around the concept of polarity. It has strong connections to important topics in proof theory, as game semantics [27]–[29], focalization [30], [31], ludics [32], [33], realizability [34], and the compact representations of proof certificates [35]. But MELLP is also important for programming languages: beyond the $\lambda\mu$-calculus, it is related to CPS translations [36], delimited continuations [37], and the π-calculus [38].

In MELLP, formulas can be either positive or negative, and the exponential modalities ! and ? have a special role in that they change the polarity of a formula. Sequents have an intuitionistic shape: they can have many negative formulas but never more than one positive formula. This structural property forbids permutations involving two positive rules, *i.e.* it forbids positive bureaucracy.

Something interesting happens when one looks at MELLP via proof nets: the rigid positive structure is preserved by the removal of bureaucracy. The important consequence is that now there is a notion of last *positive* rule. Parallelism does not disappear, though: it now concerns only the negatives. The rigid structure of positives is already at work in Laurent's proof nets presentation of MELLP. However, even if polarity provides more structure to his proof nets there is still something missing: the refined coding of causality has yet to allow

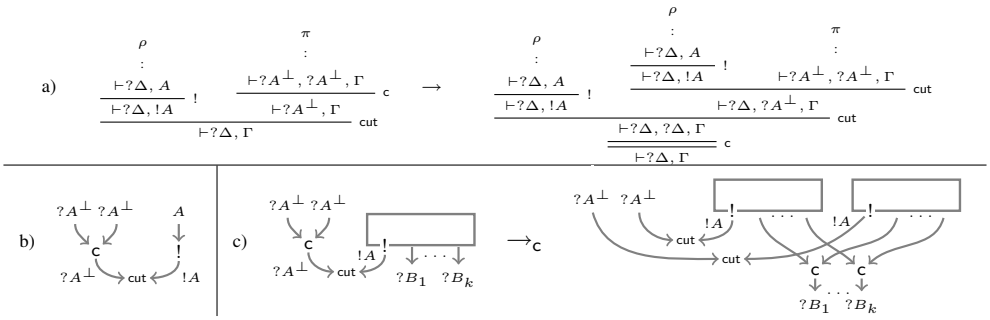

Figure 1. Duplication in linear logic: a) Sequent calculus; b) Proof nets (without boxes); c) Proof nets.

for sub-proof retrieval. Indeed, Laurent uses explicit boxes to implement cut-elimination.

In this paper we show that explicit !-boxes can be replaced by a more compact representation, what we call an **implicit box**. An implicit box is a set of causal information, a set of rules which have to be included in the sub-proof associated with a ! and which are sufficient to reconstruct it. The key point is that polarity provides the structure to keep this set remarkably compact: it is enough to retain the polarity changes (from positive to negative only, given by derelictions and cuts), all other rules can be easily retrieved by the polarity constraints. Implicit boxes add enough information to *locally* reconstruct a sub-proof—what we call the **induced box**—for every positive rule l, which is simply given by the causal past of l. For negative rules there is no induced box, but this is not a problem: !-rules are positive, thus having sub-proofs for positives is enough to implement cut-elimination.

We show that our approach is not an *ad hoc* technique, as we provide a *correctness criterion*, *i.e.* we characterize proof nets with implicit boxes without any reference to explicit boxes nor to sequent calculus. Moreover, our criterion is a natural extension of Laurent's criterion for explicit proof nets. The key technical point here is a representation of implicit boxes as additional edges, usually called *jumps* (introduced by Girard in [39]).

It can be claimed that our solution is canonical: in the case of cut-free proof nets, the implicit box of a ! contains only one dereliction and this dereliction is uniquely determined by the polarized structure, independently of the chosen syntactic formalism. Thus, we are simply taking advantage of an already existing structure, that has already been used in semantical studies [28], [40], but never exploited syntactically before. Our results reveal that polarity has also a fundamental geometric role: *it internalizes boxes*.

Let us point out that via the Curry-Howard correspondence between MELLP and $\lambda\mu$-calculus this work provides implicit boxes for classical logic. In addition, this apparently abstract work has the potential of concrete application: it provides a compact way for representing terms in graphical implementations.

Cut-elimination. We define cut-elimination on the implicit representation, obtaining a new interesting operational semantics. Implicit proof nets are a quotient of Laurent's explicit proof nets, given by the fact that box borders are not stored into implicit boxes. This fact induces new cut-elimination rules, with some good properties and some unexpected behavior:

- *No commutative rule*: since the border of explicit boxes is not explicitly represented, the commutative box-box rule disappears and cut-elimination for implicit proof nets contains only so-called *key* or *principal* cases.
- *Automatic push of negative rules*: weakenings and contractions created by cut-elimination are dynamically pushed out of explicit boxes, without any additional rule (necessary instead in [15], [41], see also [42], [43]), simply as a consequence of the implicit representation.
- *Complex critical pairs*: the absence of explicit borders and their new dynamic shrinking also generates some difficulties with respect to confluence. In particular, there is a new problematic critical pair given by the cut-elimination rules for axioms.

We prove strong normalization and confluence modulo associativity and commutativity of contractions in the propositional case. We first study cut-elimination in the case with atomic axioms only, by relating our formalism to Laurent's original syntax [23], [24], and obtain strong normalization of implicit proof nets by simulating them with explicit ones. Strong normalization is in turn used to prove confluence. Then we reduce the case with arbitrary axioms to the case with atomic axioms by introducing η-expansion rules.

Related work. As previously discussed, alternative representations of boxes have been investigated at length in a variety of studies. At first sight our technique may look similar to the interaction nets encoding of linear logic by Mackie [5], which connects a ! with its border, but in fact it is completely different: the border is not stored in the implicit box. Most research on boxes is simply concerned with evaluation, while ours belongs rather to the those studies concerning the problem of a correctness criterion for the alternative representation (which is a difficult problem). Lamarche's essential nets [3], [44]–[46] were an initial source of inspiration. However, essential nets put jumps on weakenings and box borders, while we place them on polarity changes (*i.e.* derelictions and cuts), allowing weakening to freely float in our implicit nets. This paper extends our previous work with Guerrini on λ- calculus [6] to a classical logical setting (MELLP encodes $\lambda\mu$-calculus [24]), but rather than a slight generalization it is a complete re-

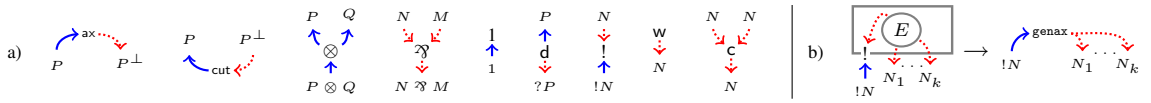

Figure 2. a) MELLP links; b) Collapsing.

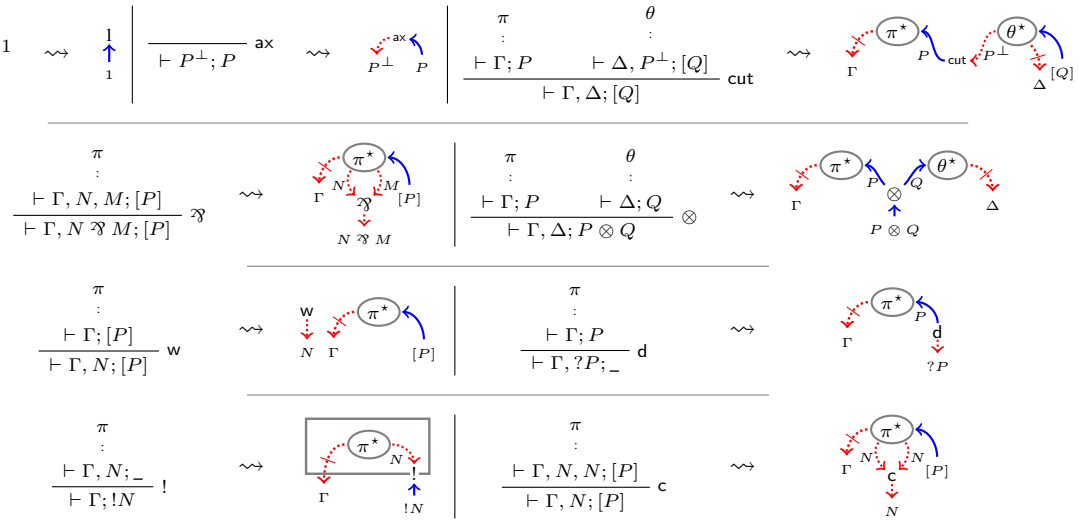

Figure 3. The translation of MELLP sequent proofs to nets.

elaboration with radical differences (see the end of Section II for more details). Our jumps are placed in precise places while those of Di Giamberardino and Faggian [47], [48] (related to L-nets and game semantics [49]) are used in a more liberal way. They can tame floating rules only by attaching them with jumps or by introducing the MIX rules, while our approach circumvent their use.

Plan of the paper. The next section summarizes Laurent's MELLP proof nets with explicit boxes. In Section II we introduce nets with implicit boxes, give a correctness criterion, prove properties of the induced boxes, and relate them statically to explicit boxes. In Section III we study cut-elimination, proving strong normalization and Church-Rosser modulo associativity and commutativity of contractions.

I. MELLP Nets with Explicit Boxes

MELLP. Multiplicative and Exponential Polarized Linear Logic (MELLP) formulas are given by two mutually defined sets, *positive* and *negative* formulas, denoted respectively with P, Q and N, M:

$$\begin{array}{rcccccccc} P, Q & ::= & X & | & 1 & | & P \otimes Q & | & !N \\ N, M & ::= & X^{\perp} & | & \bot & | & N \parr M & | & ?P \end{array}$$

X and $X^{\perp}$ are atomic formulas. The ! connective turns a negative formula into a positive one, and ? does the opposite action. The sequents of MELLP can have two shapes: $\vdash \Gamma \,;\, P$ or $\vdash \Gamma \,;\, \_$, where Γ is a multiset of negative formulas and P, if present, is a positive formula; the place of the eventual positive formula, separated by a semicolon, is called the **stoup**. The rules of MELLP are in Figure 3 (left side of every $\rightsquigarrow$), where $[P]$ means that the stoup may or may not contain a positive formula P. Note that the $\{!, \mathsf{c}, \mathsf{w}\}$-rules do not require negative formulas to be ?-formulas: this is a point on which MELLP departs from linear logic. We deal with MELLP sequent calculus only indirectly: we will relate our syntax to Laurent's proof nets, which are themselves related to sequent calculus.

Nets. Nets for proof systems are usually presented as directed graphs, where the orientation goes from the premise(s) to the conclusion(s) of the logical rules, as in Fig. 1.c. For polarized proof nets Laurent uses a correctness criterion based on directed graphs, but with respect to a different orientation. We prefer to avoid having two orientations, therefore we simplify and use only the *polarized orientation*, *i.e.* the one used by Laurent for correctness. Moreover, we hardcode polarities into the representation: negative edges are in dotted (and red) lines, positive ones are in solid (and blue) lines (see the example in Fig. 4.a, or Fig. 3).

Nets are labelled directed graphs with *pending edges*, *i.e.* some edges may not have a source or a target, but not both. Nodes, called **links**, represent deductive rules and are labeled with an element of $\{\mathsf{ax}, \mathsf{cut}, \otimes, \parr, 1, \mathsf{d}, !, \mathsf{w}, \mathsf{c}\}$. Edges are labeled with a MELLP formula, and an edge is positive/negative if its label is a positive/negative formula. The label of a link forces the number and the labels of its incoming/outcoming edges as shown in Figure 2.a (there is no $\bot$-link because a weakening can introduce no matter which negative formula, so in particular $\bot$). The **conclusions (resp. premises)** of a link, or its **principal (resp. auxiliary) edges**, are those represented below (resp. above) the link symbol. For instance the $\otimes$-link is the source of two edges (labeled P and Q) which are also its premises, and it is the target of one edge (labeled $P \otimes Q$) which is also its conclusion. A link is positive (resp. negative) if it is the source or the target of a positive (resp. negative)

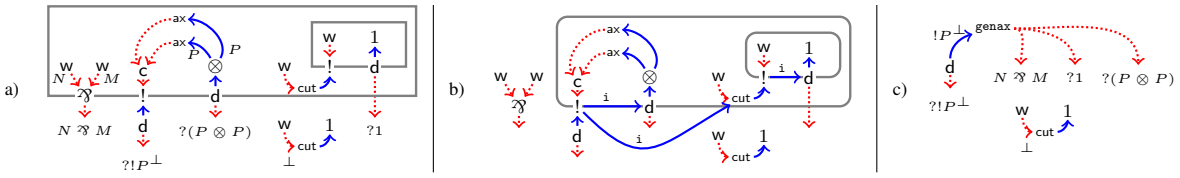

Figure 4. a) The leading examples with explicit boxes b) with implicit (and induced) boxes; c) The 0-depth net of (a).

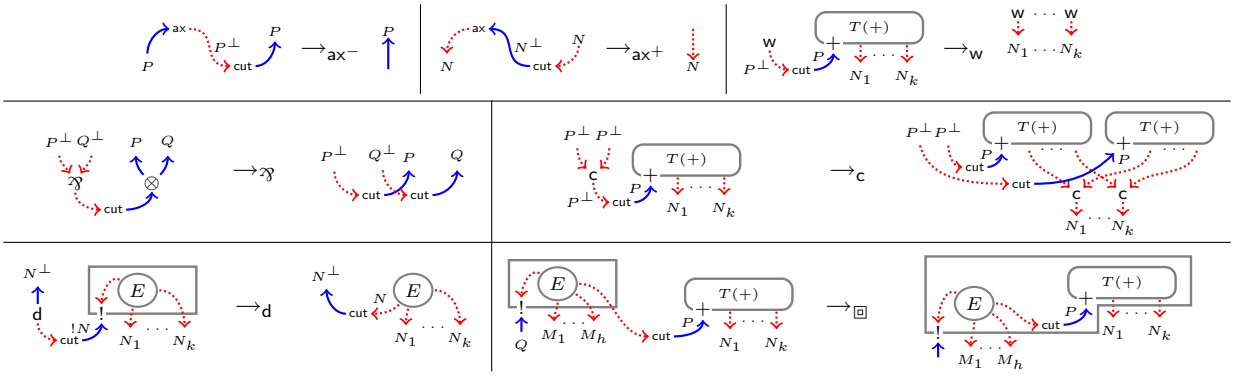

Figure 5. Rewriting rules for MELLP explicit proof nets.

edge. Then the links in $\{\mathsf{d}, !, \mathsf{ax}, \mathsf{cut}\}$ have both polarities, those in $\{\mathsf{w}, ⅋, \mathsf{c}\}$ are only negative, and those in $\{1, \otimes\}$ are only positive.

Definition I.1 (MELLP net with boxes)**.** *A **(MELLP) net** G is a finite set of links from those in Fig. 2.a s.t. every edge is the conclusion of some link. The conclusion edges of G are its pending edges and the **conclusion links** of G are its links with pending edges.*

*A net with explicit boxes E, or simply an **explicit net**, is a net plus for any !-link l a subset $ebox(l)$ of the links of G, called the **explicit box** of l, s.t. $l \in ebox(l)$ and:*

- ***Subnet**: its interior $int(l) := ebox(l) \setminus \{l\}$ is an explicit net, whose boxes are inherited from E.*
- ***Border**: the conclusion edges of $int(l)$ are negative and one of them is the negative premise of l;*
- ***Nesting**: For any two !-links l and l' if $ebox(l) \cap ebox(l') \neq \emptyset$ then $ebox(l) \subseteq ebox(l')$ or $ebox(l') \subseteq ebox(l)$.*

The translation from MELLP proofs to explicit nets is in Fig. 3, where the bar on some negative edges denotes a (multi)set of conclusions.

Leading example (Fig. 4.a, some labels have been omitted): the standard way of representing boxes is to wrap them into a graphical box as in the example. The conclusion (links) of this explicit net are the three d-links plus the ⅋-link.

The **level** of a link is the number of boxes in which it is contained, and the level of a net E is the maximum level of a link of E (the net in Fig. 4.a has level 2).

Paths. A path τ of length $n \in \mathbb{N}$, noted $\tau : l \to^n l'$ (or $\tau : l \to l'$ or $l \to l'$), is an alternated sequence $l = l_1, e_1, l_2, e_2, \dots e_n, l_{n+1} = l'$ of links and edges s.t. e_i has source l_i and target l_{i+1} for $i \in \{1, \dots, n\}$. A path $\tau : l \to^n l'$ is positive (resp. negative), noted $l \to^n_\circ l'$ (resp. $l \to^n_\bullet l'$), if all its nodes and edges are positive (resp. negative). A cycle is a path $l \to^n l$ with $n > 0$. A positive link l is **initial** if it is not a cut and there is no positive path $\tau : l' \to^n_\circ l$ with $n > 0$ (the initial links in Fig. 4.a are the three derelictions).

Correctness. Laurent found a simple criterion for MELLP proof nets.

Definition I.2 (correct net)**.** *Let E be an explicit net. The **0-depth net** E^0 of E is the net containing all the non-! links at level 0 plus a generalized axiom $\mathtt{genax}_1$ for every !-link l at level 0 in E s.t. $\mathtt{genax}_1$ and $ebox(l)$ have the same conclusions,* i.e., *E^0 is obtained from E by applying the transformation in Figure 2.b, called **collapsing**, for every !-box at level 0. Then E is correct if:*

1) ***Rooted DAG**: E^0 is acyclic and has exactly one initial link.*
2) ***Recursive correctness**: $int(l)$ is correct, for every !-link l of E at level 0.*

Consider the net in Fig. 4.a. Its 0-depth net is in Fig. 4.c.

In [23] Laurent proves that an explicit net is the translation of a proof, *i.e.* it is an **explicit proof-net**, if and only if it is correct. Polarity induces the following **matching property** on explicit proof nets (see [23], [24], [50]):

In every !-box there is exactly one dereliction at level 0
(and at level 0 there is at most one dereliction)

This fact—easily proved by induction on the translation—is a structural invariant that will be crucial in the next section (we invite the reader to verify it on the net in Fig. 4.a). Let us provide an intuition. Derelictions may be seen as variable occurrences and !-links as applications in λ-terms[1]. The matching property is the fact that the argument of every applicative

[1] In the usual (call-by-name) encoding of λ-calculus in linear logic (mapping $A \Rightarrow B$ to $!A \multimap B$) the argument of every application is in a !-box.

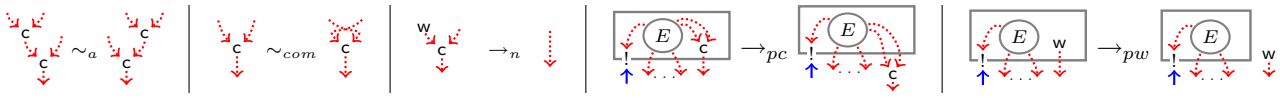

Figure 6. Additional rules and congruences for explicit proof nets.

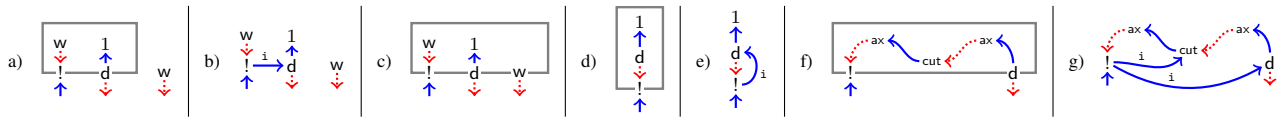

Figure 7. Some examples.

subterm has a different and unique variable occurrence as head variable.

Positive tree. In order to define cut-elimination, Laurent introduces the notion of positive tree. The positive tree is a form of induced box for the positive links $\{1, \mathsf{ax}, \otimes, !\}$, based on explicit !-boxes. Given a $\{1, \mathsf{ax}, \otimes, !\}$-link l the **positive tree** of l, noted $T(l)$, is defined as:

- *Base cases*: if l is a $\{\mathsf{ax}, 1\}$-link then $T(l)$ is l. If l is a !-link then $T(l)$ is $ebox(l)$.
- *Inductive case*: if l is a $\otimes$-link then $T(l)$ is $\{l\} \cup T(l_1) \cup T(l_2)$, where l_1 and l_2 are the target links of the auxiliary edges of l.

By induction on this definition it can be shown that $T(l)$ is a correct explicit net (in a correct net E).

Cut-elimination. The cut-elimination rules for explicit proof nets are in Figure 5, where the box of the positive nodes (generically labeled $+$) is given by their positive tree; we use $\to_{\mathsf{cut}}$ for the union of these rules. In [23] Laurent proves that cut-elimination is confluent and strongly normalizing by means of a translation to linear logic proof nets inducing a simulation. In order to later simulate our nets with Laurent's we are going to extend explicit nets with the rules and congruences in Fig. 6, *i.e.* associativity and commutativity of contractions, neutrality of weakenings for contractions, and permutations of contractions and weakenings with box borders. We define $\equiv_{\mathsf{ac}} := (\sim_{com} \cup \sim_a)^*$, and $\to_{\_} := \to_{pc} \cup \to_{pw} \cup \to_n$. The following theorem about MELLP explicit proof nets has never been proved:

Theorem I.3. *$\to_{\mathsf{cut}} \cup \to_{\_}$ is strongly normalizing modulo $\equiv_{\mathsf{ac}}$.*

However, in [51] Pagani and Tranquilli show that MELL proof nets extended with the same rules and congruences are strongly normalizing (in [51] the authors consider $\to_{pc}$ as a congruence, but the change is harmless since $\to_{pc}$ is trivially strongly normalizing), and it is easily seen that Laurent's translation can still be applied, obtaining a simulation proving the theorem.

II. Implicit Boxes: Statics

Introduction to the technique via simple examples. Consider the proof-net in Fig. 7.a. The content of the explicit box is disconnected. Then, an alternative representation of boxes aiming to provide a local reconstruction procedure has to add some connections to the underlying graph. The idea is that it is enough to exploit the matching property of polarized proof nets and simply connect every bang to the dereliction at level 0 in its box. The example then becomes as in Fig. 7.b. Note that we introduced a positive edge (labeled `i` for *implicit box*), and that in this way the positive subgraph is connected.

With this implicit representation the explicit box of a !-link l is given by the set of links $P(l)$ to which l has a positive path (in the example 1, d and ! itself) plus the negative links which have a negative path to $P(l)$ (the w-link). Applying this procedure to Fig. 7.b (and removing the `i`-edge) we recover Fig. 7.a.

For cut-free proof nets—surprisingly—this is enough to obtain an implicit representation of boxes. Note that we are just taking advantage of an already existing structure, given by the matching property, and there is no arbitrary choice to make. By materializing this implicit structure, the skeleton of explicit boxes is simply given by the set of positive paths plus the negative trees above !-links. The edges given by the matching property do nothing but connect the positive subgraph, because in a cut-free net the only points where polarity changes are derelictions and bangs. Some remarks:

- **Weakenings float**. We do *not* attach weakenings, as Fig. 7.a-b shows (see also Fig. 4.a-b).
- **The implicit box is not the border of the explicit box**. The implicit box of a !-link does not contain the auxiliary conclusions of its explicit box. Consider Fig. 7.d-e, where the dereliction at level 0 is not a conclusion of the explicit box, or Fig. 7.c, where the box has two conclusions but only one of them is attached in Fig. 7.b (see also Fig. 4.a-b).
- **Implicit boxes provide a quotient**. Our technique does not always recover the box one started with. Consider Fig. 7.c: the substitution of the explicit box with the edge given by the matching gives again Fig. 7.b, and the reconstruction gives back Fig. 7.a. The point is that the boxes induced by implicit boxes are obtained from the original ones by pushing weakening, contraction, and par conclusions out of boxes as much as possible (see also Fig. 4.a-b). In particular our representation identifies more proofs than explicit boxes, for instance the nets in Fig. 7.a and c. In other words nets with implicit boxes are a quotient of explicit nets.

Cuts connect links of opposite polarities and break the

continuity of the positive sub-graph, as Fig. 7.f shows. This problem is solved iterating the same idea: we add a positive edge from every !-link l to all the cuts at level 0 in $ebox(l)$, as in Fig. 7.g, so that the positive sub-graph of every box becomes a tree (see also Fig. 4.b).

Definition II.1 (implicit net)**.** *An **implicit net** I is a net plus for every !-link l an implicit box $ibox(l)$,* i.e. *a pair $(\mathsf{d}_l, \mathsf{cuts}_l)$ where d_l is a* d*-link and cuts_l is a a set of* cut*-links, and such that $\mathsf{d}_l \neq \mathsf{d}_{l'}$ and $\mathsf{cuts}_l \cap \mathsf{cuts}_{l'} = \emptyset$ for any two !-links $l \neq l'$.*

As already discussed informally, we represent implicit boxes by introducing new edges, so-called **jumps**: if l is a !-link then there is a positive edge (labeled with i) from l to the dereliction d_l and also to every cut-link $l' \in \mathsf{cuts}_l$.

Correctness. We do more than just represent boxes in this compact way, as we provide a correctness criterion for implicit proof nets. Polarized paths are defined as before, but now they can also use the edges induced by implicit boxes, *i.e.* the jumps, which are considered as positive edges.

Jumps may introduce cycles, as in Fig. 7.g (or Fig. 4.a-b). Rather than introducing a notion of correction graph we modify the acyclicity requirement of Laurent's criterion, and ask that every cycle uses the negative premise of a !.

We also need a condition guaranteeing the well-formedness of boxes. Given a positive link l its **positive trunk** $P(l)$ is the set of links s.t. l has a positive path to them, and its **negative arborescence** $P(l)^N$ is the set of links having a non-empty negative path to a link in $P(l)$ (in Fig. 4.b the negative arborescence of the external ! is given by the axioms, the contraction and two weakenings). Remark that $P(l)^N$ may contain positive links (d and ax links, see also Fig. 7.e), because some links are both positive and negative.

Definition II.2 (correct net)**.** *Let I be an implicit net. I is **correct** when:*

- ***Rooted* !*-graph**: there is only one initial link and every cycle passes through the negative edge of a !-link.*
- ***Box**: for any positive link l the positive links in $P(l)^N$ are also in $P(l)$[2].*

As an example, the net in Fig. 4.b (considered with the jumps but without the box) is correct, in particular it has only one initial link, which is the dereliction that is not the target of a jump.

The rooted !-graph condition allows a correct net to have cycles, but since !-links have only positive outgoing edges we get that the positive (resp. negative) subgraph of a correct implicit net is acyclic. A simple inspection of the shape of the links shows that moreover the positive (resp. negative) subgraph of a correct net is a forest, which is a fundamental property, to be referred as to **the forest property**.

Induced boxes. Implicit boxes induce ordinary explicit boxes, as we are going to show.

[2]It is easy to see that it is enough to consider only the case where l is a !-link and to ask that all the positive links with a negative path to l or cuts_l (and not to $P(l)$) are in $P(l)$.

Definition II.3 (induced box)**.** *Let I be a correct implicit net, l a positive link which is not a cut. The **induced box** $\square(l)$ of l is the set of links $P(l) \cup P(l)^N$.*

In Fig. 4.b the boxes induced by implicit boxes are represented with round corners.

The premises/conclusions orientation can be recovered from the polarized one by simply reversing positive edges (including jumps). With respect to that orientation the induced box of l is nothing else but the set of links with a path (possibly using jumps) to l, *i.e. its causal past* (test this fact on Fig. 4.b). Remark also that given a link l the reconstruction of $\square(l)$ is straightforward, it is enough to follow polarized paths. In particular, the reconstruction is a *local* process.

It is easy to see that $\square(l)$ is a net. We consider it also an implicit net, with respect to the implicit boxes inherited by I, because by definition any !-link l has a positive path to all the positive links in $ibox(l)$, and so $ibox(l) \subseteq P(l) \subseteq \square(l)$.

Definition II.4 (subnet)**.** *Let I be an implicit net. A subnet of I is a subset J of the links of I which is a correct implicit net and s.t. the implicit box of l in J and in I coincide, for any !-link $l \in J$.*

The following theorem collects the properties of induced boxes. We need the following terminology: a cut which does not belong to any implicit box is a **ground cut**.

Theorem II.5 (properties of induced boxes)**.** *Let I be a correct net, l, l' positive links of I which are not cuts.*

1) ***Positive connected skeleton**: $l' \in \square(l)$ iff $l' \in P(l)$.*
2) ***Containment**: $\square(l) \supseteq \square(l')$ iff $l \to_\circ l'$.*
3) ***Correctness**: $\square(l)$ is a subnet of I of initial link l;*
4) ***Kingdom**: Moreover, it is the smallest such one,*
5) ***Border**: the conclusions of $\square(l)$ are l and a possibly empty set of $\{\mathsf{d}, \mathsf{ax}\}$-links, to which l has a positive path. Moreover, $\square(l)$ has no ground cut.*
6) ***Nesting**: if $\square(l) \cap \square(l') \neq \emptyset$ then $\square(l) \subseteq \square(l')$ or $\square(l') \subseteq \square(l)$.*
7) ***Collapsibility**: the collapse of $\square(l)$ preserves correctness.*

Relating implicit and explicit boxes. It is easy to **read-back** an explicit net I^e from an implicit correct net I: just define $ebox(l)$ as $\square(l)$ for every !-link l (and discard implicit boxes). Conversely, given a correct explicit net E its translation is the implicit net E^i obtained by defining $ibox(l)$ as the implicit box given by the dereliction and the cuts at level 0 in $ebox(l)$. For instance, the translation of the net in Fig. 4.a is the net in Fig. 4.b. The read-back of Fig. 4.b is the same net just without the jumps but with the grey induced boxes considered as explicit ones.

By Theorem II.5.5 the explicit boxes of I^e—coming from induced boxes—close on axioms and derelictions only; we say that **they have minimal borders**. By pushing as much as possible weakening, contractions, and $\parr$-links out of explicit boxes, every explicit net E can be unambiguously turned into an explicit net having minimal borders, noted E^m.

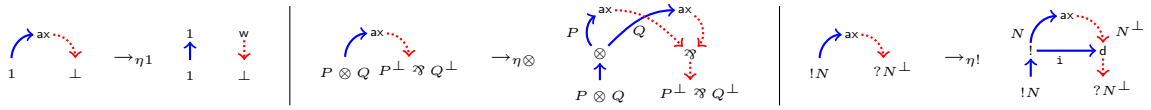

Figure 8. η-expansion rules

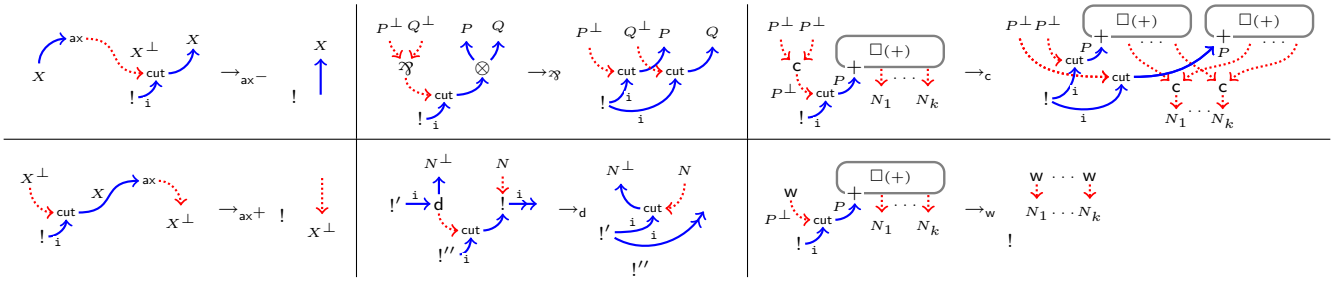

Figure 9. Rewriting rules for implicit proof nets.

Collapsibility of induced boxes and the nesting properties of explicit/implicit box are used to prove the relation between implicit and explicit correct nets:

Theorem II.6 (implicit/explicit correct nets)**.** *Let I and E be a correct implicit and explicit net, respectively, l a $\{1, \mathsf{ax}, !, \otimes\}$-link in I and l^e the corresponding link in I^e. Then:*

1) ***Correctness**: I^e and E^i are correct;*
2) ***Explicitation**: $(I^e)^i = I$, $(E^i)^e = E^m$ and $(\cdot)^e$ is injective.*
3) ***Positive trees and induced boxes**: $T(l^e) = (\square(l))^e$.*

Remark that point 2 is in fact a *sequentialization* theorem: explicit proof nets can be sequentialized, *i.e.* be read-back as sequent calculus proofs, and so—composing read-backs—implicit proof nets can be sequentialized. Then, from now on a correct implicit net is an **implicit proof net**.

Implicit nets are less bureaucratic and more parallel than explicit ones, *i.e.* $(\cdot)^i$ is not injective. Indeed, if $E^m = F^m$ then $E^i = F^i$. In particular, the translation identifies nets differing for permutations of $\parr$-links with box borders (Fig. 4.a-b), which is a novelty.

We conclude this section by sketching a comparison with our previous work [6], which developed implicit boxes for the λ-calculus with explicit substitutions[3]:

1) *Orientation*: the two works use two incompatible orientations of the proof net. In [6] the initial link correspond to the output (*i.e.* the outermost constructor) of the λ-term, while here it corresponds to the head variable.
2) *Jumps*: in [6] jumps are associated with all and only the exponential cuts (corresponding to explicit substitutions). While here jumps are associated with derelictions (corresponding to variable occurrences) and not necessarily to exponential cuts, nor necessarily to all cuts.
3) *Quotient*: the use of jumps in [6] induces a less parallel syntax. Explicit boxes already quotient more than the formalism in [6] (see [52]), and here we quotient more than explicit boxes.
4) *Language*: MELLP encodes the $\lambda\mu$-calculus (with explicit substitutions), while the approach in [6] is limited to the λ-calculus (with explicit substitutions).

Summing up, our study of implicit boxes for MELLP is not a slight extension of [6], but a complete re-elaboration, which catches a more expressive language, in a more parallel way, providing new insights on the structural role of polarity.

III. Implicit Boxes: Dynamics

Simplifying hypothesis. To simplify the study of the dynamics of implicit proof nets we use two assumptions:

1) The net is contained in a box. More precisely, we assume that every cut is in some implicit box. In this way we avoid to double the cut-elimination rules for the cases when the cut is out of every implicit box. This assumption is not essential, it just simplifies the presentation.
2) We allow axioms only with atomic formulas, *i.e.* we restrict to η-expanded nets (the η-expansion rules are in Figure 8). The general case is discussed after the results on η-expanded nets.

Cut-elimination rules. The set of rewriting rules and equivalences for implicit proof nets is in Figure 9. The non-linear steps ($\to_{\mathsf{c}}$ and $\to_{\mathsf{w}}$) are implemented by duplicating or erasing the induced box of the positive premise of the cut (note the round corners).

Cut-elimination directly acts on implicit boxes by erasing, duplicating, or moving jumps. *Erasing*: the rules $\to_{\mathsf{ax}-}$, $\to_{\mathsf{ax}+}$, and $\to_{\mathsf{w}}$ erase the cut link and also remove it from the implicit box $ibox(!)$, *i.e.* they erase the jump. *Duplicating*: the rules $\to_{\mathsf{c}}$ and $\to_{\parr}$ extend $ibox(!)$, duplicating the jump. *Composing*: the rule $\to_{\mathsf{d}}$ is where implicit boxes are composed (an example is in Fig. 10.a). First, the propagated cut enters into $\square(!')$, which is why it has a jump from $!'$ and not from $!''$. Second, also the content of $\square(!)$—where $!$ is the eliminated !-link—enters $\square(!')$, which is why the jumps on $!$ (represented with a double arrow) pass to $!'$ in the reduct.

[3] In [6] the graphical syntax in use is not proof nets, but a proof nets-inspired formalism having the constructors of the λ-calculus as link. Here we compare them as if [6] used proof nets, via the call-by-name encoding of λ-calculus into linear logic.

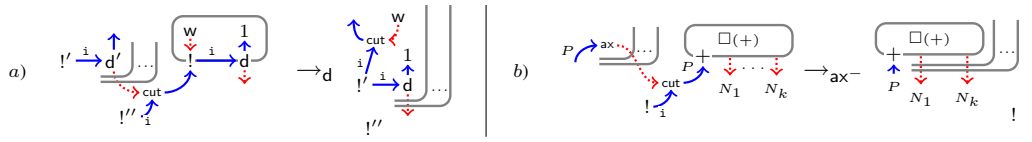

Figure 10. The rules $\to_{\mathsf{d}}$ and $\to_{\mathsf{ax}-}$ merge induced boxes (a is only an example).

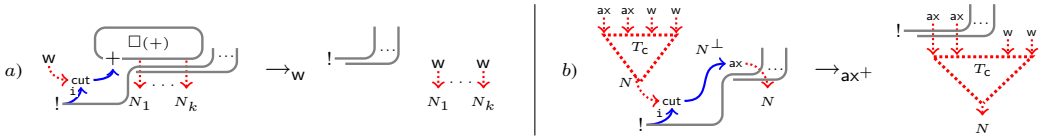

Figure 11. a) Dynamic pushing of created weakenings; b) $\to_{\mathsf{ax}+}$ and box borders.

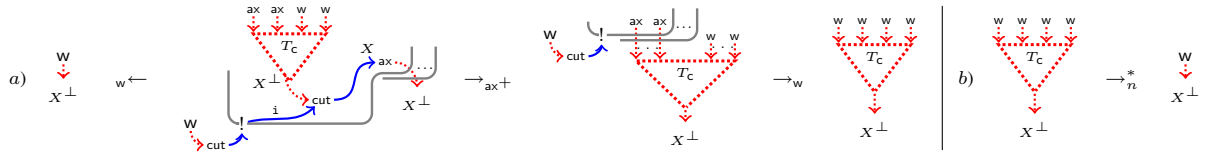

Figure 12. a) Critical pair between $\to_{\mathsf{w}}$ and $\to_{\mathsf{ax}+}$; b) Neutrality closes the pair.

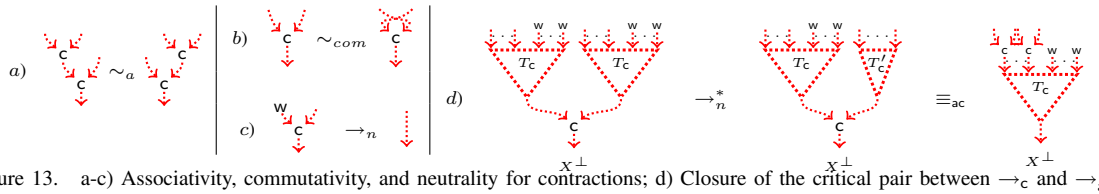

Figure 13. a-c) Associativity, commutativity, and neutrality for contractions; d) Closure of the critical pair between $\to_{\mathsf{c}}$ and $\to_{\mathsf{ax}+}$.

Cut-elimination also acts on induced boxes. In particular, there is an implicit action on the border of induced boxes. Here is where the novelties about cut-elimination appear. It is not difficult to explicit this indirect action, because we know that borders are given by axioms and derelictions (Theorem II.5.5). There are three actions:

1) **Merging of boxes without any commutative rule**: both the rules $\to_{\mathsf{d}}$ and $\to_{\mathsf{ax}-}$ act on box borders, incorporating a big-step box-box rule, as it becomes evident if one represents induced boxes as in Fig. 10. Remark that there is no rule corresponding to the commutative box-box rule for explicit nets, whose role is subsumed by rules $\to_{\mathsf{d}}$ and $\to_{\mathsf{ax}-}$. Therefore, we get only so-called *key* or *principal* cases of cut-elimination.
2) **Dynamic pushing of created contractions and weakenings**: the non-linear rules $\to_{\mathsf{c}}$ and $\to_{\mathsf{w}}$ introduce contractions and weakenings where previously there were dereliction and axioms. But induced boxes cannot close on contractions and weakenings, which are in fact automatically pushed out of boxes as much as possible. Figure 11.a shows the case for weakenings, the case for contractions is analogous.
3) **Automatic permutations on the border**: the rule $\to_{\mathsf{ax}+}$, acting on the (axiom) border of some induced boxes, induce permutations of negative links with box borders, see Fig. 11.b, where T_{c} is the (possibly empty) tree of contractions rooted in N. Remark that our assumption on η-expanded nets forbids $\parr$s to be in T_{c} and d-links to be leaves of T_{c}.

Let $\to$ be the union of all the rewriting rules in Fig. 9. As expected we have:

Proposition III.1 (Cut-elimination preserves correctness)**.** *Let I be an implicit proof net. If $I \to J$ then J is an implicit proof net.*

Strong normalization by simulation. We now show that implicit proof nets can be simulated by explicit proof nets. As already explained, the simulation of $\to_{\mathsf{d}}$ and $\to_{\mathsf{ax}-}$ steps requires a preliminary sequence of commutative steps, while $\to_{\mathsf{w}}$, $\to_{\mathsf{c}}$, and $\to_{\mathsf{ax}+}$ steps require to push the created weakenings and contractions out of explicit boxes.

Proposition III.2 (Simulation via read-back)**.** *Let I be an η-expanded implicit proof net. The read-back $(\cdot)^e$ induces a simulation:*

1) ***Rules*** ax^- ***and*** d ***use explicit commutation****: $I \to_x J$ implies $I^e \to_{\boxdot}^* \to_x J^e$, for $x \in \{\mathsf{ax}^-, \mathsf{d}\}$.*
2) ***The non-linear rules push created weakenings and contraction****: $I \to_{\mathsf{w}} I$ implies $I^e \to_{\mathsf{w}} \to_{pw}^* J^e$, and $I \to_{\mathsf{c}} J$ implies $I^e \to_{\mathsf{c}} \to_{pc}^* J^e$.*
3) ***Rule*** $\to_{\mathsf{ax}+}$ ***and border permutations****: $I \to_{\mathsf{ax}+} J$ implies $I^e \to_{\mathsf{ax}+} \to_{pc}^* \to_{pw}^* J^e$.*
4) ***Multiplicative case****: $I \to_{\parr} J$ implies $I^e \to_{\parr} J^e$.*

From strong normalization of explicit nets (Theorem I.3) and Prop. III.2 we get:

Corollary III.3. *MELLP η-expanded implicit proof nets are strongly normalizing.*

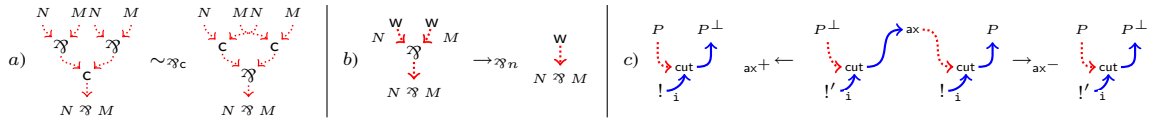

Figure 14. a-b) Additional rules for $\parr$; c) New problematic critical pair.

Confluence. The new features of cut-elimination have a consequence: they require to work modulo associativity of contractions and to add neutrality of weakenings with respect to contractions. Indeed, the critical pairs between the non-linear rules and $\to_{\mathsf{ax}+}$ can be closed only using associativity and neutrality. Figure 12.a shows the critical pair justifying the neutrality rule (omitting irrelevant details), and Fig. 12.b shows how neutrality closes the pair. The similar pair where the weakening cut is replaced by a contraction cut requires associativity and neutrality in order to be closed (Fig.13.d shows how to close this pair).

Implicit proof nets are then considered modulo $\equiv_{\mathsf{ac}}$ which is the equivalence defined as $(\sim_{com} \cup \sim_a)^*$, see Fig. 13.a-b (commutativity is not essential), and now $\to$ denotes the cut-elimination rules plus $\to_n$ (Fig. 13.c). The next proposition shows that the read-back commutes with $\equiv_{\mathsf{ac}}$ and $\to_n$.

Proposition III.4. *Let I be an η-expanded implicit proof net. If $I \equiv_{\mathsf{ac}} J$ (resp. $I \to_n J$) then J is correct, η-expanded and $I^e \equiv_{\mathsf{ac}} J^e$ (resp. $I^e \to_n J^e$).*

Now, from Proposition III.2 and Theorem I.3 it follows:

Corollary III.5. *η-expanded implicit proof nets are strongly normalizing modulo $\equiv_{\mathsf{ac}}$.*

In the case of reduction modulo an equivalence relation there are various notions of confluence. The strongest is Church-Rosser modulo, which by a lemma due to Huet [53] follows from strong normalization whenever the system is locally confluent modulo and locally coherent modulo [54] (Lemma 14.3.11, pp 773-774). We need a definition: given I and J they are **joinable modulo** $\equiv_{\mathsf{ac}}$, noted $I \heartsuit J$, iff exists I', J' s.t. $I \to^* I'$, $J \to^* J'$ and $I' \equiv_{\mathsf{ac}} J'$.

Proposition III.6. *For η-expanded implicit proof nets the relation $\to$ is:*

1) ***Locally confluent modulo*** $\equiv_{\mathsf{ac}}$*: if $I \to J_1$ and $I \to J_2$ then $J_1 \heartsuit J_2$.*
2) ***Locally coherent with*** $\sim_a$ ***and*** $\sim_{com}$*: if $I \to J$ and $I \sim_a I'$ or $I \sim_{com} I'$ then $I' \heartsuit J$.*

The use of $\sim_a$ and $\sim_{com}$ in the statement of local coherence is not an error: Huet's lemma requires the property only for the generators $\sim_a$ and $\sim_{com}$ of $\equiv_{\mathsf{ac}}$. We then conclude with:

Corollary III.7 (Church-Rosser modulo $\equiv_{\mathsf{ac}}$)**.** *For η-expanded implicit proof nets $\to$ is Church-Rosser modulo $\equiv_{\mathsf{ac}}$: if $I(\to \cup \leftarrow \cup \equiv_{\mathsf{ac}})^* J$ then $I \heartsuit J$.*

The general case. We now reduce the non-η-expanded case to the use of atomic axioms via η-expansion. Consider the η-rules in Figure 8, and let $\to_\eta := \to_{\eta 1} \cup \to_{\eta \otimes} \cup \to_{\eta !}$. They are clearly strongly normalizing (the formulas on the axioms strictly decrease in size) and confluent (all redexes are disjoint) modulo $\equiv_{\mathsf{ac}}$ (they do not interact with contractions). Then, given an implicit proof net I, it makes sense to consider its η-normal form $\eta(I)$. Note that the axiom rule applies to atomic axioms only, non-atomic axioms have to be η-expanded first.

Proposition III.8. *Let I be an implicit proof net, $x \in \{\mathsf{ax}^+, \mathsf{ax}^-, \parr, \mathsf{d}, \mathsf{w}, \mathsf{c}, n\}$, and $y \in \{a, com\}$.*

1) ***Local commutation****: $J_1 \,{}_\eta\!\leftarrow I \to_x J_2$ implies exists H s.t. $J_1 \to_x H \,{}^*_\eta\!\leftarrow J_2$.*
2) ***Local coherence****: $J_1 \equiv_y I_1 \to_\eta I_2$ implies exists J_2 s.t. $J_1 \to_\eta J_2 \equiv_y I_2$.*
3) ***Projection of*** $\to_x$*: If $I \to_x J$ then $\eta(I) \to_x \eta(J)$.*
4) ***Projection of*** $\equiv_{\mathsf{ac}}$*: If $I \equiv_{\mathsf{ac}} J$ then $\eta(I) \equiv_{\mathsf{ac}} \eta(J)$*

It is not difficult to derive strong normalization by projection on the η-expanded case, and then infer Church-Rosser as before.

Corollary III.9. *On implicit proof nets the reduction $\to \cup \to_\eta$ is strongly normalizing and Church-Rosser modulo $\equiv_{\mathsf{ac}}$.*

Despite these are remarkably strong results, there still is space for improvement. A detail that the purist would not judge completely satisfactory is that η-reduction is necessary in order to compute normal forms. Unfortunately, the removal of η-reduction has to face serious difficulties. Axiom rules for non-atomic axioms induce two problems that break the relationship with explicit boxes and forbid to apply standard techniques.

The first problem is that the automatic permutation of negative links with box borders, which pushes a tree T_{c} of contractions out of boxes (Fig. 12.a), may now also involve $\parr$-links (*i.e.* $\parr$-links may be part of T_{c}). These permutations, which forced us to work modulo associativity and to add the neutrality rule (Fig. 12.b and 13.d), now force to work with a similar equation and a similar rule for $\parr$s, shown in Fig. 14.a-b. It is easy to extend Laurent's nets with these rules, the problem is rather that Laurent's translation of MELLP to MELL does not simulate them, and so strong normalization cannot be obtained via a simulation.

The second problem is that a new problematic critical pair appears, the one in Fig. 14.c (note the difference between ! and !′ in the two reducts). The difficulty is that the critical pair is *non-local*, in the sense that the pair does not contain enough information in order to be closed, one has to look to the context of the rule, as in [55]–[57]. But in contrast to those cases here the context cannot be rewritten independently of the pair, because reductions depend on implicit boxes, *i.e.* on where jumps are attached. So even local confluence is non-trivial and requires a new technique. Unfortunately, the state

of the art in terms of proof of strong normalization for proof nets relies on local confluence [51], [58].

On η-expanded nets this second problem also vanishes, because the positive premise of the right cut is always an axiom, and so the critical pair can be closed by simply removing both cuts.

Conclusions

We showed how polarity can be exploited in order to get a purely local and implicit implementation of explicit boxes. Our syntax is supported by a correctness criterion, a detailed analysis of its relationship with explicit boxes, and results about cut-elimination.

Acknowledgments. This work was partially supported by the Qatar National Research Fund under grant NPRP 09-1107-1-168.

References

[1] J.-Y. Girard, "Linear logic," *Theoretical Computer Science*, vol. 50, pp. 1–102, 1987.

[2] ——, "Proof-nets: The parallel syntax for proof-theory," in *Logic and Algebra*. Marcel Dekker, 1996, pp. 97–124.

[3] F. Lamarche, "Proof Nets for Intuitionistic Linear Logic: Essential Nets," Research Report, 2008.

[4] G. Gonthier, M. Abadi, and J.-J. Lévy, "Linear logic without boxes," in *LICS*, 1992, pp. 223–234.

[5] I. Mackie and J. S. Pinto, "Encoding linear logic with interaction combinators," *Inf. Comput.*, vol. 176, no. 2, pp. 153–186, 2002.

[6] B. Accattoli and S. Guerrini, "Jumping boxes," in *CSL*, 2009, pp. 55–70.

[7] V. Danos and L. Regnier, "Proof-nets and the Hilbert space," in *Advances in Linear Logic*. Cambridge University Press, 1995, pp. 307–328.

[8] S. Guerrini, S. Martini, and A. Masini, "Coherence for sharing proof nets," in *RTA*, 1996, pp. 215–229.

[9] V. Danos and L. Regnier, "Reversible, irreversible and optimal lambda-machines," *Theor. Comput. Sci.*, vol. 227, no. 1-2, pp. 79–97, 1999.

[10] J.-Y. Girard, "Light linear logic," *Inf. Comput.*, vol. 143, no. 2, pp. 175–204, 1998.

[11] V. Danos and J.-B. Joinet, "Linear logic and elementary time," *Inf. Comput.*, vol. 183, no. 1, pp. 123–137, 2003.

[12] A. Asperti, "Linear logic, comonads and optimal reduction," *Fundam. Inform.*, vol. 22, no. 1/2, pp. 3–22, 1995.

[13] P.-A. Melliès, "Functorial boxes in string diagrams," in *CSL*, 2006, pp. 1–30.

[14] L. Tortora de Falco, "The additive mutilboxes," *Ann. Pure Appl. Logic*, vol. 120, no. 1-3, pp. 65–102, 2003.

[15] R. Di Cosmo, D. Kesner, and E. Polonovski, "Proof nets and explicit substitutions," *Math. Str. in Comput. Sci.*, vol. 13, no. 3, pp. 409–450, 2003.

[16] A. Asperti, V. Danos, C. Laneve, and L. Regnier, "Paths in the lambda-calculus," in *LICS*, 1994, pp. 426–436.

[17] B. Accattoli and D. Kesner, "The structural λ-calculus," in *CSL*, 2010, pp. 381–395.

[18] ——, "Preservation of strong normalisation modulo permutations for the structural λ-calculus," *Logical Methods in Computer Science*, vol. 8, no. 1, 2012.

[19] B. Accattoli, "An abstract factorization theorem for explicit substitutions," in *RTA*, 2012, pp. 6–21.

[20] B. Accattoli and D. Kesner, "The permutative λ-calculus," in *LPAR*, 2012, pp. 23–36.

[21] B. Accattoli and U. Dal Lago, "On the invariance of the unitary cost model for head reduction," in *RTA*, 2012, pp. 22–37.

[22] B. Accattoli and L. Paolini, "Call-by-value solvability, revisited," in *FLOPS*, 2012, pp. 4–16.

[23] O. Laurent, "Étude de la polarisation en logique," Thèse de Doctorat, Université Aix-Marseille II, Mar. 2002.

[24] ——, "Polarized proof-nets and $\lambda\mu$-calculus," *Theor. Comput. Sci.*, vol. 290, no. 1, pp. 161–188, 2003.

[25] J.-Y. Girard, "A new constructive logic: Classical logic," *Math. Str. in Comput. Sci.*, vol. 1, no. 3, pp. 255–296, 1991.

[26] O. Laurent, "Polarized proof-nets: proof-nets for LC (extended abstract)," in *TLCA*, 1999, pp. 213–227.

[27] ——, "Polarized games," *Ann. Pure Appl. Logic*, vol. 130, no. 1-3, pp. 79–123, 2004.

[28] ——, "Syntax vs. semantics: A polarized approach," *Theor. Comput. Sci.*, vol. 343, no. 1-2, pp. 177–206, 2005.

[29] P.-A. Melliès and N. Tabareau, "Resource modalities in tensor logic," *Ann. Pure Appl. Logic*, vol. 161, no. 5, pp. 632–653, 2010.

[30] O. Laurent, M. Quatrini, and L. Tortora de Falco, "Polarized and focalized linear and classical proofs," *Ann. Pure Appl. Logic*, vol. 134, no. 2-3, pp. 217–264, 2005.

[31] C. Liang and D. Miller, "Focusing and polarization in linear, intuitionistic, and classical logics," *Theor. Comput. Sci.*, vol. 410, no. 46, pp. 4747–4768, 2009.

[32] J.-Y. Girard, "Locus solum: From the rules of logic to the logic of rules," *Math. Str. in Comput. Sci.*, vol. 11, no. 3, pp. 301–506, 2001.

[33] M. Basaldella and C. Faggian, "Ludics with repetitions (exponentials, interactive types and completeness)," *LMCS*, vol. 7, no. 2, 2011.

[34] G. Munch-Maccagnoni, "Focalisation and classical realisability," in *CSL*, 2009, pp. 409–423.

[35] K. Chaudhuri, "Compact proof certificates for linear logic," in *CPP*, 2012, pp. 208–223.

[36] O. Laurent and L. Regnier, "About translations of classical logic into polarized linear logic," in *LICS*, 2003, pp. 11–20.

[37] N. Zeilberger, "Polarity and the logic of delimited continuations," in *LICS*, 2010, pp. 219–227.

[38] K. Honda and O. Laurent, "An exact correspondence between a typed pi-calculus and polarised proof-nets," *Theor. Comput. Sci.*, vol. 411, no. 22-24, pp. 2223–2238, 2010.

[39] J.-Y. Girard, "Quantifiers in Linear Logic II," Université Paris VII, Paris, Prépublications de l'Équipe de Logique 19, 1991.

[40] P. Boudes, "Thick subtrees, games and experiments," in *TLCA*, 2009, pp. 65–79.

[41] P. Tranquilli, "Confluence of pure differential nets with promotion," in *CSL*, 2009, pp. 500–514.

[42] R. Di Cosmo and S. Guerrini, "Strong normalization of proof nets modulo structural congruences," in *RTA*, 1999, pp. 75–89.

[43] D. Kesner and S. Lengrand, "Resource operators for lambda-calculus," *Inf. Comput.*, vol. 205, no. 4, pp. 419–473, 2007.

[44] A. S. Murawski and C.-H. L. Ong, "Dominator trees and fast verification of proof nets," in *LICS*, 2000, pp. 181–191.

[45] A. S. Murawski, "On semantic and type-theoretic aspects of polynomial-time computability," PhD thesis, University of Oxford, 2001.

[46] S. Guerrini, "Proof nets and the lambda-calculus," in *Linear Logic in Computer Science*. Cambridge University Press, 2004, pp. 65–118.

[47] P. Di Giamberardino and C. Faggian, "Proof nets sequentialisation in multiplicative Linear Logic," *Ann. Pure Appl. Logic*, vol. 155, no. 3, pp. 173–182, 2008.

[48] P. Di Giamberardino, "Jump from parallel to sequential proofs: exponentials," 2011, to appear in the special number *Di?erential Linear Logic, Nets, and other quantitative approaches to Proof-Theory* of *MSCS*.

[49] P.-L. Curien and C. Faggian, "L-nets, strategies and proof-nets," in *CSL*, 2005, pp. 167–183.

[50] O. Laurent, "Polarized proof-nets: Proof-nets for lc," in *TLCA*, 1999, pp. 213–227.

[51] M. Pagani and P. Tranquilli, "The conservation theorem for differential nets," 2009, accepted for publication in Math. Str. in Comput. Sci.

[52] B. Accattoli, "Jumping around the box: graphical and operational studies on λ-calculus and linear logic," PhD thesis, *La Sapienza* University of Rome, february 2011.

[53] G. P. Huet, "Confluent reductions: Abstract properties and applications to term rewriting systems: Abstract properties and applications to term rewriting systems," *J. ACM*, vol. 27, no. 4, pp. 797–821, 1980.

[54] Terese, *Term Rewriting Systems*, ser. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2003, vol. 55.

[55] Y. Lafont, "Towards an algebraic theory of boolean circuits," *J. Pure Appl. Alg.*, vol. 184, pp. 257–310, 2003.

[56] Y. Guiraud and P. Malbos, "Higher-dimensional categories with finite derivation type," *Theor. Appl. Cat.*, vol. 22, pp. 420–478, 2009.

[57] S. Mimram, "Computing critical pairs in 2-dimensional rewriting systems," in *RTA*, 2010, pp. 227–242.

[58] M. Pagani and L. Tortora de Falco, "Strong normalization property for second order linear logic," *Theor. Comput. Sci.*, vol. 411, no. 2, pp. 410–444, 2010.