Confluence of Pure Differential Nets with Promotion

Paolo Tranquilli

Laboratoire PPS – Université Paris Diderot - Paris 7
Case 7014 – 75205 Paris – France
`ptranqui@pps.jussieu.fr`

Abstract. We study the confluence of Ehrhard and Regnier's differential nets with exponential promotion, in a pure setting. Confluence fails with promotion and codereliction in absence of associativity of (co)contractions. We thus introduce it as a necessary equivalence, together with other optional ones. We then prove that pure differential nets are Church-Rosser modulo such equivalences. This result generalizes to linear logic regular proof nets, where the same notion of equivalence was already studied in the literature, but only with respect to the problem of normalization in a typed setting. Our proof uses a result of finiteness of developments, which in this setting is given by strong normalization when blocking a suitable notion of "new" cuts.

1 Introduction

The inception of Linear Logic (LL, [1]) in the 80's has reinforced the bridge between logic and computer science already established by the Curry-Howard correspondence years before. LL is in fact a refinement of intuitionistic and classical logic brought forth by a fine semantical analysis. One of its main features is the introduction of two dual modalities, the *exponentials* ! and ?, regulating the use of structural rules (*weakening* and *contraction*), which on the program side correspond to erasure and duplication of resources.

This endeavour, among other things, led the way to a new, parallel syntax of proofs, *proof nets*. These are the syntax of choice for LL, especially when considering cut elimination. In fact one of the main advances of LL over classical logic is that, though preserving an involutive negation (and therefore two-sided sequents), it also preserves properties of intuitionistic logic lacking in the classical framework. One of these, central to our work, is confluence of cut elimination, i.e. the independence of the result of the cut elimination procedure with respect to the actual cuts one decides to reduce.

A further semantical analysis led by Ehrhard [2] has recently provided LL with new models based on topological vector spaces where we can take the derivative of an object. The efforts of the same author and Regnier have permitted to lift such operations to syntax, giving rise to Differential Linear Logic (DiLL, [3]), and their syntax, *differential nets*. Three new rules are introduced to handle the !-modality (*coweakening*, *cocontraction* and *codereliction*) which are duals to the LL rules handling ?. In the proofs as programs paradigm, codereliction allows to introduce depletable resources, which may be asked for many times but may be used just one time, nondeterministically choosing which query they satisfy. This feature configures differential nets as a promising logical framework to extend the Curry-Howard correspondence to nondeterminism and concurrency (see [4]).

Actually, [3] gives the syntax for the promotion free fragment of DiLL only, giving rise to *differential interaction nets*, a nondeterministic example of Lafont's interaction nets [5]. By modelling nondeterminism by formal sums confluence remains an important property, which is however straightforward in an interaction net paradigm, where no reduction can change the other ones. Here we will extend such property to the whole of differential nets. Promotion in proof nets is handled by *boxes*, synchronized areas of proofs enabling to mark what is to be erased or duplicated. Boxes break the interaction net paradigm: there are cuts (the *commutative* ones) which can be changed by other reductions, so confluence is definitely more delicate.

Part of a previous work of ours [6] was focused on proving confluence for the intuitionistic fragment, which used the recursive types needed to translate λ-calculus. There we observed that confluence fails without keeping into account some semantically grounded equivalences, namely associativity of contractions and cocontractions. A fully quotienting syntax as the one used in [7] for LL is seemingly out of reach in DiLL. Our solution in [6] was employing generalized (co)contraction cells in the style of [8], and some additional reductions.

Here we generalize the result in three ways. By concentrating on the computational contents rather than the logical one, we consider *pure* nets, where types (i.e. formulae) play no role whatsoever, not even recursive ones. Furthermore, the needed equivalences are settled to the maximum extent by means of... equivalences on nets. We thus generalize the equivalences and reductions of [9], providing as a byproduct the first proof of confluence[1] for such LL proof nets with equivalences in the completely pure case, as previous works concentrated on normalization in the typed one. Finally, we are able to introduce one more equivalence potentially giving the right to always consider boxes without sums inside (the *bang sum* equivalence).

This result has several ramifications. As is evident in [10], this is the first step in proving strong normalization in the typed case[2]. Furthermore, as can be deducted from [9], this can be the ground for new work on calculi with explicit substitutions: whether by extending some results to untyped calculi; or by considering explicit substitutions for nondeterministic calculi akin to Boudol's λ-calculus with resources (see [6]).

Our technique, reminiscent of the work done on LL in [10], uses a finite development theorem used to prove a strong confluence property of a suitable notion of parallel reduction.

1.1   Rewriting Theory Modulo Equivalence

The aim of this section is making the reader acquainted with the notion of rewriting modulo equivalence, to the extent needed for our purposes. We refer to [11, Section 14.3] for more in-depth details and proofs.

Let $(S, \to)$ be an abstract reduction system and let $\sim$ be an equivalence relation on S. As usual, $\overset{=}{\to}$ and $\overset{*}{\to}$ denote the reflexive and reflexive-transitive closures of $\to$ respectively. Take a symmetric relation $\vdash$ such that $\vdash^* = \sim$, possibly $\sim$ itself. Let $s \overset{\vee}{=} t$ (t and s are **joinable modulo** $\sim$) if $s \overset{*}{\to}\sim\overset{*}{\leftarrow} t$. We say then that $\to$ is

---

[1] To be precise, the stronger result of being Church-Rosser modulo (see Section 1.1).

[2] Actually the subject of a future submission by the author and Pagani.

- locally confluent modulo $\sim$ if $\leftarrow\rightarrow \;\subseteq\; \underline{\vee}$;
- confluent modulo $\sim$ if $\overset{*}{\leftarrow}\sim\overset{*}{\rightarrow} \;\subseteq\; \underline{\vee}$;
- locally coherent with $\vdash$ if $\vdash\rightarrow \;\subseteq\; \underline{\vee}$;
- Church-Rosser modulo $\sim$ (or $\mathsf{CR}\sim$) if $\approx \;\subseteq\; \underline{\vee}$, where $\approx := (\rightarrow \cup \leftarrow \cup \sim)^*$;
- strongly normalizing modulo $\sim$ (or $\mathsf{SN}\sim$) if $\Rightarrow$ is SN, where[3] $\Rightarrow := \sim\rightarrow\sim$;
- strongly Church-Rosser modulo $\sim$ if $\sim\overset{=}{\leftarrow}\overset{=}{\rightarrow}\sim \;\subseteq\; \overset{=}{\rightarrow}\sim\overset{=}{\leftarrow}$;

The last definition is our terminology, while the rest follows [11]. Being Church-Rosser modulo $\sim$ is the most important property of all those concerning confluence. In particular it implies the unique normal form modulo $\sim$ property, ($\approx\; =\; \sim$ on normal forms), which again implies that in order to compute the normal form one can use just regular reductions, without ever be forced to $\sim$-convert in order to get the result[4]. Contrary to what happens in regular reduction, $\mathsf{CR}\sim$ is strictly stronger than plain confluence in absence of WN [11, Remarks 14.3.6, Exercise 14.3.7]. Following are some important lemmas: the first is a generalization of Newman's Lemma, the last is a trivial result we did not find in the literature which we will need in our proof.

Lemma 1 (Huet). *If $\rightarrow$ is $\mathsf{SN}\sim$, locally confluent modulo $\sim$ and locally coherent with $\vdash$, then it is $\mathsf{CR}\sim$.*

Lemma 2 (van Oostrom). $\rightarrow$ *is $\mathsf{CR}\sim$ iff $\overset{*}{\rightarrow}$ is strongly $\mathsf{CR}\sim$.*

Lemma 3. *If $\rightarrow$ is strongly $\mathsf{CR}\sim$, then it is $\mathsf{CR}\sim$.*

Proof. Straightforward induction: to show that $\sim\overset{*}{\leftarrow}\overset{*}{\rightarrow}\sim$ is joinable, we proceed by induction first on one side, then on the other.

2 The System

A **net** is intuitively a network of **cells** linked by **wires** connecting their **ports**. A little more formally, a net π is given by the following data.

- A set $\mathsf{p}(\pi)$ of ports.
- A set $\mathsf{c}(\pi)$ of cells; to each cell c is assigned a **symbol** $\sigma(c)$ in a given alphabet, a port in $\mathsf{p}(\pi)$ called **principal**, and a number of other, **auxiliary** ones. How the latter are treated distinguishes between two kinds of cells: in non commutative ones, auxiliary ports are a finite sequence, in **commutative** ones they form a finite set. Every port in $\mathsf{p}(\pi)$ can be associated with at most one cell; a port associated with a cell is called **connected**, otherwise it is **free**. Free ports (also called conclusions) are denoted by $\mathsf{fp}(\pi)$. The number of auxiliary ports is determined by the symbol $\sigma(c)$.
- A set $\mathsf{w}(\pi)$ of wires, which can be either unordered pairs $\{p, q\}$ of ports, or **deadlocks**, i.e. wires not connecting any port (intuitively short circuited wires). Each port is in exactly one wire. A **directed wire** is an ordered pair (p, q) such that $\{p, q\}$ is a wire. **Terminal wires** are the directed ones going to the free ports.

---

[3] Here like in the rest of the paper, := means "defined as".

[4] This also means there is never the need to perform conversion steps in order to *ready* some redexes, i.e. make them visible.
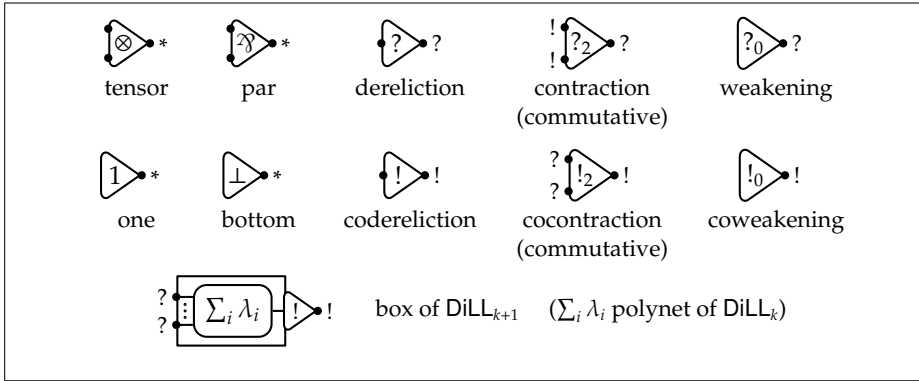
Fig. 1. The cells of differential nets. The labels in $\{!, ?, *\}$ assigned to ports will be needed only later (see page 507).

An **elementary path** in π is one in the graph trivially obtained by taking cells and free ports as nodes and directed wires as edges, which moreover does not intersect itself[5]. A **polynet** is a formal sum of nets, or equivalently a multiset of nets, all sharing the same free ports. At times we distinguish nets (thus singletons) from polynets by calling them **simple**.

2.1 Statics

DiLL_0 nets and polynets are built from all the symbols in Figure 1 but the box one. These are exactly the differential interaction nets presented in [3]. For the moment let us ignore the labels we assign to the ports in the figure, which will be needed only later (see page 507). As usual, the apex of the cell represents the principal port, while the auxiliary ones are depicted on the opposite side.

In order to add boxes, one proceeds by induction, by considering them as cells having a whole polynet as symbol. Let DiLL_{k+1} nets and polynets be the ones built from all the cells of Figure 1 where for each box its symbol is a polynet π in DiLL_k and there is a bijection between its ports and $\mathsf{fp}(\pi)$. The symbol $\sigma(B)$ of a box B is also called its **contents**. We will denote by $!\pi$ a generic box having π as contents. A DiLL polynet π is one of DiLL_k for any k; if such k is minimal, we say that k is the depth of π (in fact, the maximal number of nested boxes). A port is **active** if it is either a principal one, or an auxiliary one of a box. A wire linking two active ports is a **cut**.

Figures 5 and 6 will show examples of differential nets. The explicit marking of ports is dropped as they can always be identified with the extremities of wires.

Let $\mathsf{p}_!(\pi)$, $\mathsf{fp}_!(\pi)$ $\mathsf{c}_!(\pi)$ and $\mathsf{w}_!(\pi)$ be the set of all occurrences of ports, free ports, cells and wires respectively occurring in π, including in all the contents of the boxes in π. We can slice those sets by depth, so we will denote by $\mathsf{p}_i(\pi)$, $\mathsf{fp}_i(\pi)$ $\mathsf{c}_i(\pi)$ and $\mathsf{w}_i(\pi)$ the

---

[5] Technically, one prohibits the repetition of unoriented wires and that three ports of the same cell be crossed by the path.

corresponding elements of the nets contained in i nested boxes, where i is called the depth of the element in π[6].

Correctness Criterion. As usual, the nets blindly built with the cells available are not in general "correct", where the word can take the meaning of unsequentializable in sequent calculus, or having deranged computational behaviour. Since [12] one of the most used correctness criteria for proof nets is that of switching acyclicity. Given a DiLL net, a **switching path** is an elementary one which does not traverse two auxiliary ports of any $\mathfrak{N}$ or contraction cell (does not bounce "above" it). A DiLL polynet is called a DiLL **proof net** (or differential proof net) if it is switching acyclic, i.e. it has no deadlocks nor switching cycles, and inductively all box contents are also switching acyclic. From now on we will deal almost only with proof nets.

2.2 Dynamics

As with various calculi, the reduction of differential nets can be defined as the context closure of a set of reduction rules, presented as pairs of redexes and contracta. A **linear context** $\delta[\ ]$ is a simple net δ together with a subset H_δ of its free ports (the **hole** of $\delta[\ ]$). It is linear as it is not a sum and the hole is not inside a box. Given a simple net λ and a bijection σ between H_δ and $\mathsf{fp}(\lambda)$, the plugging $\delta[\lambda]$ of a simple net λ in the hole of $\delta[\ ]$ amounts to identifying the ports according to σ and welding the wires that come together in this way[7], as shown in Figure 2. This definition is then extended by linearity when we plug a polynet, by setting $\delta[\sum_i \lambda_i] := \sum_i \delta[\lambda_i]$.
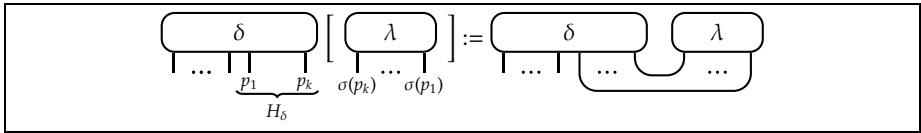

Fig. 2. Plugging of a net in a context

Finally, **contexts** generalize the concept in the following way. A linear context $\delta[\ ]$ is a context; furthermore if $\omega[\ ]$ is a context then $\delta[\omega[\ ]]$ for $\delta[\ ]$ linear[8], $\omega[\ ] + \pi$ for π polynet and $!\omega[\ ]$ (i.e. a box containing a context) are also contexts. Plugging is easily extended to all contexts. The **context closure** $\tilde{R}$ of a relation R is then defined by $\pi \tilde{R} \sigma$ iff $\pi = \omega[\lambda]$, $\sigma = \omega[\mu]$ and $\lambda R \mu$.

We are now able to define multiplicative reduction $\overset{\mathsf{m}}{\to}$ and the exponential one $\overset{\mathsf{e}}{\to}$ by context closure of the rules of Figure 3, which are pairs consisting of a simple net (the **redex**) and a polynet (the **contractum**). Each redex here is identified by a unique cut. The union $\overset{\mathsf{me}}{\to}$ of the two reduction is the cut elimination of DiLL.

---

[6] All of this can be defined more formally by an inductive definition. Nevertheless we leave it to the reader as an easy exercise.

[7] For the quite delicate technical details the reader is referred to [13].

[8] Composition of contexts should be defined, but it is trivial once plain plugging is defined.
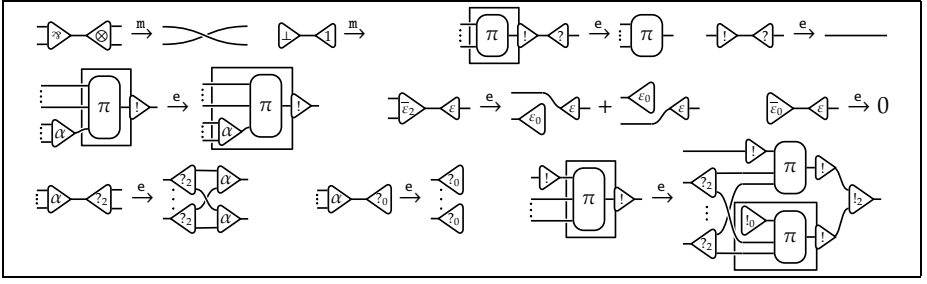
Fig. 3. The multiplicative and exponential reduction rules of DiLL. ε and $\overline{\varepsilon}$ denote either ? and ! or vice versa. α denotes any symbol among $!_2$, $!_0$ or a box symbol π. In particular weakening against coweakening reduces to the empty net. We make implicit use of the rule for context plugging of sums: the π inside boxes is a polynet. For example the dereliction on box rule may introduce sums.
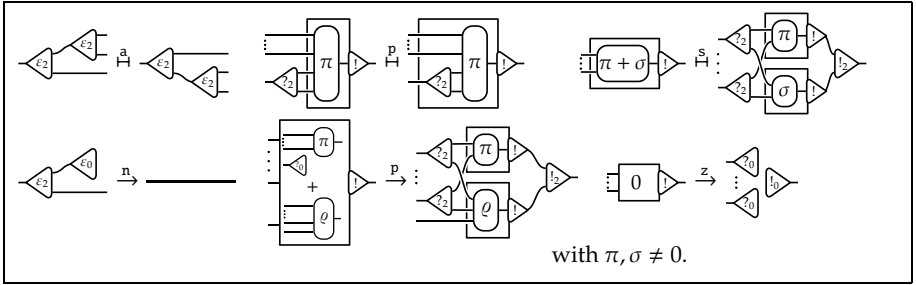

Fig. 4. Top: the rules for associative equivalence $\overset{a}{\backsim}$, the push one $\overset{p}{\backsim}$ and the bang sum one $\overset{s}{\backsim}$; $\vdash$ denotes a one-step conversion. Bottom: the rules for neutral reduction $\overset{n}{\rightarrow}$, the pull one $\overset{p}{\rightarrow}$ and the bang zero one $\overset{z}{\rightarrow}$. The condition $\pi, \sigma \neq 0$ applies to all rules.

2.3 Equivalences and Canonical Reductions

As we will show as a remark at page 506, the reductions just presented fail to give a confluent system: we cannot ignore associativity of (co)contractions and neutrality of (co)weakening over (co)contraction. This prompts us to introduce the former as an equivalence and the latter as a reduction. As we need anyway to consider reduction modulo an equivalence, we also study other equivalences (backed by semantical and observational equivalence) which are optional though must be taken together. Each equivalence is accompanied by a reduction which in a sense settles a zeroary case of the equivalence. Reversing each of these gives unwanted looping reductions. The **associative**, **push** and **bang sum** equivalences, together with the **neutral**, **pull** and **bang zero** reductions (which *do not* reduce cuts), are shown in Figure 4. The $\pi, \sigma \neq 0$ condition is needed, lest one would be able to spawn trees of contractions from nothing, giving looping reductions.
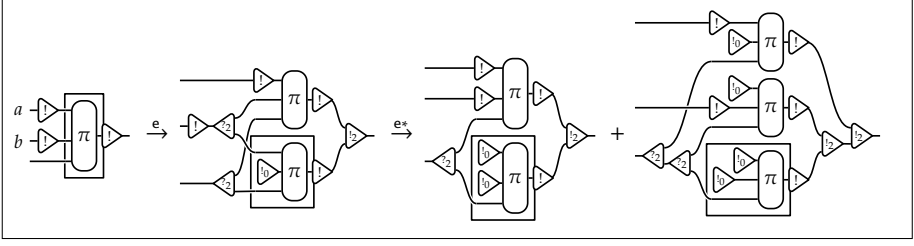
Fig. 5. Reduction of a box with two coderelictions on it. Starting with codereliction b swaps the two linear copies of $!\pi$ and therefore both the cocontraction and contraction trees in the last addend.

The push equivalence[9] has already been studied in the literature on proof nets and explicit substitutions [8,9]. The pull reduction may seem somewhat complicated, however it is a generalization of the reduction pulling out weakenings from boxes [9]. The usual reduction can be reobtained when having $\varrho = 0$, which by means of a z-reduction and some n ones gives the expected result. Such form (which in fact contains a sort of on-the-fly s-conversion) is required in order to get local coherence[10].

The part about sums inside boxes was already known to be valid semantically and observationally: we give here some syntactic ground to using it. From the point of view of semantics it is interesting to note that it implements the well known exponential isomorphism $!A \otimes !B \cong !(A \& B)$ from linear logic (see [2]).

From now on $\sim$ and $\xrightarrow{c}$ (**canonical** reduction) will denote either $\xrightarrow{a}$ and $\xrightarrow{n}$ or the union of the a, p and s conversions and the npz-reduction respectively. By checking the cases not already proved in the literature, one gets the following.

Proposition 4 (stability of correctness). *If π is switching acyclic and $\pi \xrightarrow{mec} \pi'$ or $\pi \sim \pi'$ then π' is switching acyclic also.*

Examples and remarks. Figure 5 gives the reason to employ associative equivalence, showing the reduction of the coderelictions on box critical peak, which cannot be joined with regular reductions. One reduction only is shown, as the other is symmetric. Other critical peaks due to the codereliction on box rule (namely when against dereliction and contraction) show that also the n-reductions cannot be left out. One can already see two big differences with respect to LL and the work done with it in [10]: firstly, sums may arise even without the "logical" step of dereliction on box; moreover, the codereliction on box rule, which reduces a commutative cut, changes the possible cuts on *all* other cuts of the box. These problems prevent an immediate adaptation of the measures used in [10]. The nets in Figure 6 are examples already known in LL showing issues about correctness and types.

---

[9] Though an equivalence is not directed, the name comes from [14] and [6] where it was a reduction. We felt like keeping it for its good pairing with the pull reduction.

[10] The problem arises in a p-equivalence and n-reduction critical peak, as the latter may eliminate a contraction of *one* of the addends in the box.
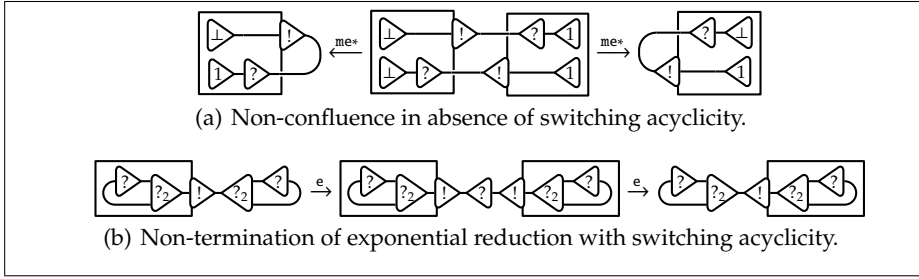
(a) Non-confluence in absence of switching acyclicity.

(b) Non-termination of exponential reduction with switching acyclicity.

Fig. 6. Issues with confluence. Figure 6(a) shows the need for correctness. The example shown is even simply typed. Figure 6(b) shows how in the pure case even the exponential reduction alone is not terminating.
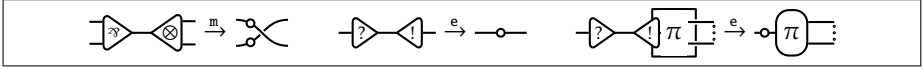
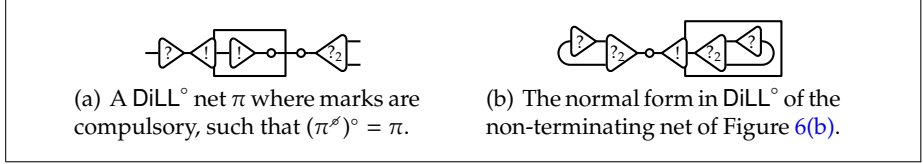## 3   The Finite Development Theorem

In [15], Danos proved the counterpart of the finite development theorem for MELL, and Pagani and Tortora de Falco did the same for the whole of second order LL in [10]. In this setting the actual definition of what a "new" redex is gets more technical.

3.1   Marking New Cuts

We define a notion of new and old cuts, by leaving a mark on the new ones. **Marks** are cells of a new symbol with two ports and no reduction rule, graphically depicted by little circles. Its main purpose is to block reductions and equivalences (for example a mark between two contractions blocks the a-conversion).

   Ideally, these marks are placed during reduction to block "new" wires. By new we mean two kinds of wires: those that in a typed setting would decrease the logical complexity of the cut formula, and those that before the reduction were exponential clashes. The latter are peculiar to a truly untyped setting, and are brought by the opening box and neutral reductions, which erase an exponential port. For example, if we erase marks from the net shown in Figure 8, and we fire the dereliction against box redex we end up with a valid multiplicative cut which was a clash before. Rather than lock this special kind of "new" wires during reduction, we can lock all potentially dangerous clashes since the start, as markings will prevent new clashes from arising. We thus need to define what an exponential clash is.

   Let τ be the partial function from $\mathsf{p}_!(\pi)$ to the labels $\{\,!, ?, *\,\}$ thus defined. On ports of cells it gives the values already shown in Figure 1; on the ports of marks it is undefined; for $p \in \mathsf{fp}_!(\pi)$ we set $\tau(p) = ?$ if p is over an auxiliary port of a box, we leave it undefined otherwise. τ provides for a sort of pre-typing. A directed wire (p, q) is called a **!-wire** (resp. **?-wire**) if $\tau(p) = ?$ and $\tau(q) = !$ (resp. vice versa), where however we let at most one of the two be undefined. In any case ! and ?-wires are called **exponential** (which applies to undirected wires also). An **exponential clash** (simply clash from now

Fig. 7. The modified reduction rules of DiLL°

(a) A DiLL° net π where marks are compulsory, such that $(\pi^{\not{e}})° = \pi$.

(b) The normal form in DiLL° of the non-terminating net of Figure 6(b).

Fig. 8. Examples for DiLL°

on) is a wire $\{p, q\}$ such that one of $\tau(p)$, $\tau(q)$ is ! (resp. ?) and the other is defined but not ? (resp. !)[11].

DiLL° is the system given by polynets with marks and without clashes, and by changing some rules to introduce the mark as depicted in Figure 7. It is immediate to see that the absence of clashes is preserved by reduction, as the new wires which could bring close unmatched ports are interrupted by marks. From the point of view of DiLL, clash-freeness imposes just that some marks be added: given any DiLL polynet π, we define the injection $\pi°$ in DiLL° by placing a mark interrupting each clash. Conversely, DiLL° can be clearly surjected on DiLL by erasing all marks. We call this surjection $\pi^{\not{e}}$. The net π in Figure 8(a) is an example of DiLL° net enjoying $(\pi^{\not{e}})° = \pi$. On the other hand, if σ is the net in Figure 6(b), then $\sigma° = \sigma$ and it is strongly normalizing to the net in Figure 8(b).
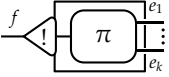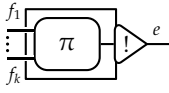
3.2 Measuring Exponential Reduction

Ideally, we may regard exponential reduction as a procedure that "slides" cells along exponential paths in the net, with ! and ? cells sliding in opposite direction. We thus assign to each cut a natural number, indicating how far the two cells around it are from the end of the path they are sliding on. After a reduction however many cuts may have arisen. So we will employ the multiset of the weights of the cuts and the multiset order[12]. Global additive duplication poses another problem. In [6] we settled it by employing multisets of multisets. Here however we estimate how many addends can sprout during reduction, so we can use this value and count each cut as many times as there can be addends containing it. We will also need to get an estimate of the number of copies (both regular and linear) of a box.

---

[11] More intuitively: !-wires and ?-wires are those that in a typing attempt would get an outermost ! or ? respectively, while clashes would give a failure to unify an outermost exponential modality.

[12] Multisets (here presented as $[a_1, \ldots, a_k]$ with additive notation) over a well founded set are well ordered by the transitive closure of $A + [a] > A + B$ with $\forall b \in B : b < a$.

Table 1. Rules for the ?-weight (top), the ! one (middle) and the spread sp (bottom). In the spread formula (which ranges over derelictions and coderelictions at depth 0) we use the notation $?(c)$ and $!(c)$ for the corresponding measure on the wire from the principal port of c.

$$?(e) = ?(e_1) + ?(e_2); \qquad\qquad ?(e) = ?(f);$$

$$?(e_i) = \begin{cases} ?(f)(1 + \sum_{j=1}^{k} !(e_j)) & \text{if } \pi = 0, \\ ?(f)(1 + \sum_{j=1}^{k} !(e_j)) \sum_{\lambda \in \pi} \mathsf{sp}(\lambda)?(e_i^\lambda) & \text{otherwise;} \end{cases}$$

otherwise : $?(e)$ is a variable if e is terminal at depth 0, $?(e) = 1$ otherwise.

$$!(e) = !(e_1) + !(e_2); \qquad\qquad !(e) = !(f);$$

$$!(e) = \begin{cases} 1 + \sum_{j=1}^{k} !(f_j) & \text{if } \pi = 0, \\ (1 + \sum_{j=1}^{k} !(f_j)) \sum_{\lambda \in \pi} \mathsf{sp}(\lambda) & \text{otherwise.} \end{cases}$$

$$!(e) = !(f) \qquad \begin{array}{l} \text{if } p \in \mathsf{fp}_0(\sigma(B)) \text{ for a box } B, p \text{ is above an auxiliary port,} \\ \text{and } f \text{ is the wire corresponding to } p \text{ outside the box,} \end{array}$$

otherwise : $!(e)$ is a variable if e is terminal at depth 0, $!(e) = 1$ otherwise.

$$\mathsf{sp}(\lambda) = \prod_{\substack{c \in \mathsf{c}_0(\lambda) \\ \sigma(c)=?}} !(c) \cdot \prod_{\substack{c \in \mathsf{c}_0(\lambda) \\ \sigma(c)=!}} ?(c)$$

Exponential Paths. An **!-path** (resp. **?-path**) is an elementary path made only of !-wires (resp. ?-wires), not traversing any mark, dereliction or codereliction (though it may end on them). In either case, the path is called **exponential**. All cells internal to an exponential path must necessarily be contractions, cocontractions or boxes. The main technical advantage of DiLL° over DiLL is that in it no reduction can open new exponential paths.

Next we define by mutual induction three basic measures on which we will base the measure of the whole net. Two of them, the ?-**weight** $?(e)$ and the !-weight $!(e)$, are on wires. The third, the **spread** $\mathsf{sp}(\lambda)$, is defined on simple subnets of the given net[13]. For the purpose of working modularly with the measures, we introduce *variables* on the terminal wires over $\mathsf{fp}_0(\pi)$. We will thus consider *variables* $!(d)$ (resp. $?(d)$) with d a terminal wire.

Weighting wires and estimating addends. Table 1 provides the laws for $?(e)$ (resp. $!(e)$), giving them depending on an adjacent cell. By the absence of clashes there is no

---

[13] Without getting into details, a subnet of π is a properly formed net given by a subset of cells and wires of π, all taken from the same depth (but then including all the elements contained in its boxes).

ambiguity in the definitions. By e^λ we denote the wire corresponding to e inside a box, in the net λ of the box contents. At the bottom we also show the law for the spread. Notice that all measures are polynomials with natural coefficients. Notice also that there is a circular dependency between the three measures, so the next lemma is not trivial.

Lemma 5. *Given a* DiLL° *proof net* π, $?(e)$, $!(e)$ *and* $\mathrm{sp}(\lambda)$ *are defined for all* $e \in w_!(\pi)$ *and all* λ *simple subnets of* π *at any depth.*

Proof (sketch). One proceeds by a primary induction on the depth: supposing all the (polynomial) measures have been defined inside all boxes, one can

- define $!(e)$ by induction on the maximal length of $?$-paths starting from e (instantiating the variables of the $?$-conclusions inside boxes in the process);
- only then, define $?(e)$ by induction on the maximal length of $!$-paths (relying also on the measures inside boxes just instantiated);
- finally define the spread from the two.   □

Weighting nets and polynets. The **weight** $|e|$ of a wire is $?(e) + !(e)$. Let $!cw_0(\lambda)$ (resp. $b_0(\lambda)$) be the set of exponential cuts (resp. boxes) at depth 0 of a simple net λ. Let us fix a polynet π, and let $c(B)$ (the **count** of the box B) denote $?(e)(1 + \sum_j !(f_j))$ with e and f_j the wires on the principal and the auxiliary ports of B respectively. Then for each sum (i.e. multiset) ρ of subnets of π we define by induction on their depth the following multiset (λ will denote a generic simple subnet):

$$\|\rho\| := \sum_{\lambda \in \rho} \mathrm{sp}(\lambda)\|\lambda\|, \quad \|\lambda\| := [\, |e| \mid e \in !cw_0(\lambda) \,] + \sum_{B \in b_0(\lambda)} c(B)\|\sigma(B)\|,$$

Finally, the polynomial measure of the whole net (a special case of the measure already given) can be instantiated with 1 for all variables to get an actual number.

Notice that this measure depends monotonously from the weight of each part of the net. This intuition will be given a solid ground by Lemma 7.

Intuitive ideas of the measures. As already hinted at the begin of this section, these measures should help to estimate how far each cut has to go in both directions to arrive at a "dead end" (for DiLL°, the logical rules), and how many times each cut should be accounted for.

Morally $!(e)$ measures the size of the tree of cocontractions above e (which is invariant under associativity). The most important feature is that it counts all the cocorelictions linked to e. On boxes we count

- the $!$-weight on the auxiliary ports because the cocoreliction against box rule creates a contraction and a cocoreliction; plus one to count the box itself, especially if it has no auxiliary ports;
- multiplied by the spread of the contents in order to be invariant by s-conversion, and keep such invariants even if the sum inside... spreads.

Dually $?(e)$ measures the size of the tree of contractions above e. The rule when e is on an auxiliary port of a box B contains:
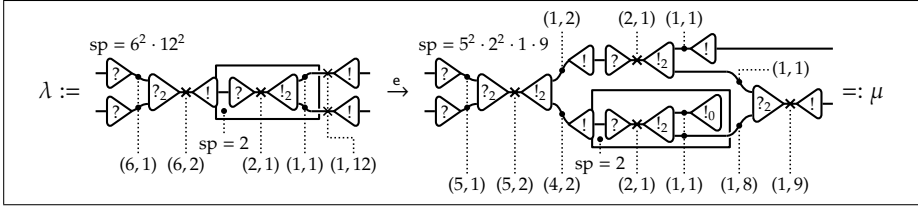
Fig. 9. An example of calculated measures. Each relevant wire e is labelled by the pair $(!(e), ?(e))$. Cuts are specially marked.

- $?(f)$ because the contractions on the principal port of B may shift to auxiliary ports during reduction;
- the sum of !-weights of the auxiliary wires because codereliction against box creates contractions; plus 1 to provide something to decrease when a cut enters a box (box against box and those similar);
- the ?-measures inside because either by opening the box or by p-conversion the contraction trees inside can pour outside; summed, to respect both p-conversion and s-conversion; this sum is weighted with the spread to prevent a reduction generating a sum inside from increasing such weight.

As already hinted, $\mathrm{sp}(\lambda)$ estimates how many addends may have a reduct of λ. This is achieved by morally multiplying all the possible number of choices potentially to be done in λ. Now sums arise

- on (co)dereliction against co(co)ntraction reductions, so the size of the tree of co(co)ntractions on the principal port of a (co)dereliction should estimate what choices that (co)dereliction may do;
- on (co)dereliction against box rules, when the box contains an actual sum; however the spread of a box contents are already accounted for in both the !-weight and the ?-weight.

Finally the cuts inside a box B count $c(B)$ times as this number estimates how many regular and linear copies of its contents may be done, and all cuts count $\mathrm{sp}(\lambda)$ times to account for additive duplication.

Before sketching the proofs, we show in Figure 9 an example of reduction step where we have calculated (in the way indicated by Lemma 5) all the relevant measures of the two nets. It turns out that $\|\lambda\| = 5184(12[3]+[8, 13, 13])$, while $\|\mu\| = 900(9[3]+[7, 10])$, which is indeed lower (though in a quite coarse way).

Replacement and Modularity Lemmas. In the following, we will consider the extensional (i.e. pointwise) order $\leq$ on non-zero values for all the polynomials.

For different simple nets λ, μ, we distinguish the weights calculated on one or the other by putting them as superscripts, as in $?^\lambda(e)$. Suppose λ and μ are two nets with identified terminal wires C. We say that λ **can replace** μ if for each terminal wire d we

have that $!^{\lambda}(d) \leq !^{\mu}(d)$ and $?^{\lambda}(d) \leq ?^{\mu}(d)$ (one of the comparisons may be a trivial one among the same variables). An induction on the context reveals the following lemma.

Lemma 6 (replacement). *Suppose λ can replace μ, $\omega[\ ]$ is a linear context with $\omega[\lambda]$ and $\omega[\mu]$ proof nets. Then for each e wire in the context ω, $?^{\omega[\mu]}(e) \leq ?^{\omega[\lambda]}(e)$ and $!^{\omega[\mu]}(e) \leq !^{\omega[\lambda]}(e)$.*

In the following the weight $|D|$ of a set of wires D is the multiset of the weights of its wires. A terminal wire is **dormant** if it connects an active port or two free ones. Dormant wires are those that can become cuts when glued in a context. The proof of the following lemma, which we omit, is an induction on the depth of the hole in the context.

Lemma 7 (modularity). *Take λ and $\mu_1, \ldots, \mu_n$ simple nets and $\omega[\ ]$ a context such that $\omega[\lambda]$ and $\omega[\mu_i]$ are all DiLL° pure proof nets. Suppose moreover that:*

- *for every i we have that μ_i can replace λ;*
- *$\sum_i \mathrm{sp}(\mu_i) \leq \mathrm{sp}(\lambda)$;*
- *if $n = 0$, then $\|\lambda\| > [\ ]$, otherwise for every i we have $\|\mu_i\| + |D_i|^{\mu_i} < \|\lambda\|$, (resp. $\leq$) where D_i is the set of active wires in μ_i that are not dormant in λ.*

Then we have the pointwise inequality $\|\omega[\sum_i \mu_i]\| < \|\omega[\lambda]\|$ (resp. $\leq$).

Thanks to modularity, the following result is up to mechanical checks which we omit altogether.

Lemma 8. *$\|\,.\,\|$ has the following properties.*

- *if $\pi \xrightarrow{\mathrm{e}} \pi'$ then $\|\pi'\| < \|\pi\|$;*
- *if $\pi \xrightarrow{\mathrm{mc}} \pi'$ then $\|\pi'\| \leq \|\pi\|$;*
- *if $\pi \sim \pi'$ then $\|\pi'\| = \|\pi\|$.*

Theorem 9 (finite developments). *Reduction on DiLL° is SN.*

Proof (sketch). Only m and c remain to be settled. For all reductions, the pair given by $(\|\pi\|, \#_{\mathrm{m}}(\pi) + \#_{\mathrm{c}}(\pi))$ strictly decreases for lexicographic ordering, where $\#_{\mathrm{m}}$ just counts the multiplicative cells in π, and $\#_{\mathrm{c}}$ weights coweakenings, weakenings and boxes containing 0 in the following inductive way:

$$\#_{\mathrm{c}}(\pi) := 1 + \#_{!_0}(\pi) + \#_{?_0}(\pi) + \sum_{B \in \mathsf{b}_0(\pi)} (1 + \deg(B))\,\#_{\mathrm{c}}(\sigma(B))$$

where $\#_{!_0}$ and $\#_{?_0}$ count coweakenings and weakenings at depth 0, and the degree $\deg(B)$ is the number of ports of B. $\qquad\qquad\square$

4   Proving Confluence

Recall that $\sim$ and c may be a-equivalence and n-reduction, or full asp-equivalence and nzp-reduction. Some of the diagrams show we cannot separate s-equivalence from the p one (and their associated reductions). Checking all local confluence and local coherence diagrams as indicated by Lemma 1, gives the following proposition, which then finally leads the way to the main theorem of the work.

**Proposition 10.** *Reduction in* DiLL° *is* CR~*, and so are* m *and* ec *alone.*

Main Theorem. *Reduction of* DiLL *pure proof nets is* CR~*, and so are* m *and* ec *alone.*

Proof. Using DiLL° we define a parallel reduction $\twoheadrightarrow$. Let $\pi \twoheadrightarrow \sigma$ iff $\pi° \xrightarrow{\text{mec}} \varrho$ in DiLL° and $\sigma = \varrho^{\varnothing}$. Then

- $\xrightarrow{\text{mec}} \subseteq \twoheadrightarrow \subseteq \xrightarrow{\text{mec}*}$, so that $\twoheadrightarrow^* = \xrightarrow{\text{mec}*}$ (notice $\pi°$ cannot block any reduction);
- $\twoheadrightarrow$ is strongly CR~ because $\xrightarrow{\text{mec}*}$ is so in DiLL°: if $\pi \twoheadrightarrow \sigma_1^{\varnothing}, \sigma_2^{\varnothing}$ with $\pi° \xrightarrow{\text{mec}} \sigma_1, \sigma_2$, then $\sigma_1, \sigma_2 \xrightarrow{\text{mec}*} \rho$, and then $(\sigma_i^{\varnothing})° \xrightarrow{\text{mec}*} \rho$, because $(\sigma_i^{\varnothing})°$ has less marks than σ_i as the latter is clash-free. In the end $\sigma_1^{\varnothing}, \sigma_2^{\varnothing} \twoheadrightarrow \rho^{\varnothing}$.

Then we conclude, as $\twoheadrightarrow$ is CR~ by Lemma 3 ($\twoheadrightarrow$ is reflexive as $(\pi°)^{\varnothing} = \pi$), which means that $\twoheadrightarrow^* = \xrightarrow{\text{mec}*}$ is strongly CR~, which in turn by Lemma 2 gives Church-Rosser modulo ~ for the ordinary reduction[14]. It is not hard to give parallel reductions for the ec and m ones and do the same. □

A Conclusion: The Case for MELL. Our system of reductions and equivalences bears close resemblance to the one developed for MELL in [9]. In fact, stripping DiLL of all its differential features, the only difference is the absence in [9] of anything related to the p-reduction. In MELL the p-reduction is given by simply pulling out a weakening out of a box, which in DiLL can be done by a concatenation of p, z and n reduction steps. In [16], the author calls such a variant a *total* p-reduction, which was here omitted because of its redundancy in DiLL. First, we argue that without such a step the CR~ property is broken, as shown by the following coherence critical peak, leading to two normal forms which are not directly equivalent[15]:
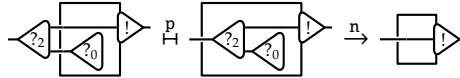
Then a direct consequence of the Main Theorem is the following, which may prove useful in the study of calculi with explicit substitutions.

Theorem 11. MELL *pure proof nets, with* a *and* p *equivalences together with* n *and total* p *reduction is* CR~.

References

1. Girard, J.Y.: Linear logic. Theor. Comput. Sci. 50, 1–102 (1987)
2. Ehrhard, T.: Finiteness spaces. Math. Structures Comput. Sci. 15(4), 615–646 (2005)
3. Ehrhard, T., Regnier, L.: Differential interaction nets. Theor. Comput. Sci. 364(2), 166–195 (2006)

---

[14] Notice that we cannot infer CR~ of $\twoheadrightarrow$ directly from the same property in DiLL°, as chained parallel reductions are not necessarily in DiLL°.

[15] In fact, even the confluence modulo property does not hold, though probably there is confluence for $\sim\rightarrow\sim$.

 4. Ehrhard, T., Laurent, O.: Interpreting a finitary pi-calculus in differential interaction nets. In: Caires, L., Vasconcelos, V.T. (eds.) CONCUR 2007. LNCS, vol. 4703, pp. 333–348. Springer, Heidelberg (2007)
 5. Lafont, Y.: Interaction nets. In: POPL 1990, pp. 95–108. ACM, New York (1990)
 6. Tranquilli, P.: Intuitionistic differential nets and lambda calculus. To appear on Theor. Comput. Sci. (2008)
 7. Regnier, L.: Lambda-Calcul et Réseaux. Thèse de doctorat, Université Paris 7 (1992)
 8. Di Cosmo, R., Guerrini, S.: Strong normalization of proof nets modulo structural congruences. In: Narendran, P., Rusinowitch, M. (eds.) RTA 1999. LNCS, vol. 1631, pp. 75–89. Springer, Heidelberg (1999)
 9. Di Cosmo, R., Kesner, D., Polonovski, E.: Proof nets and explicit substitutions. Math. Structures Comput. Sci. 13(3), 409–450 (2003)
10. Pagani, M., Tortora de Falco, L.: Strong normalization property for second order linear logic. To appear on Theor. Comput. Sci. (2008)
11. Terese: Term Rewriting Systems. Cambridge Tracts in Theoretical Computer Science, vol. 55. Cambridge University Press, Cambridge (2003)
12. Danos, V., Regnier, L.: The structure of multiplicatives. Archive for Mathematical Logic 28, 181–203 (1989)
13. Vaux, L.: λ-calcul différentiel et logique classique: interactions calculatoires. Thèse de doctorat, Université de la Méditerranée (2007)
14. Di Cosmo, R., Kesner, D.: Strong normalization of explicit substitutions via cut elimination in proof nets. In: LICS, p. 35. IEEE Computer Society Press, Los Alamitos (1997)
15. Danos, V.: La Logique Linéaire appliquée à l'étude de divers processus de normalisation (principalement du λ-calcul). Thèse de doctorat, Université Paris 7 (1990)
16. Tranquilli, P.: Nets between determinism and nondeterminism. Ph.D. thesis, Università Roma Tre/Université Paris Diderot, Paris 7 (April 2009)