

Realizability Proof for Normalization of Full Differential Linear Logic

Stéphane Gimenez

Laboratoire PPS, Université Paris-Diderot – Paris 7

Abstract. Realizability methods allowed to prove normalization results on many typed calculi. *Girard* adapted these methods to systems of nets and managed to prove normalization of second order *Linear Logic* [4]. Our contribution is to provide an extension of this proof that embrace *Full Differential Linear Logic* (a logic that can describe both single-use resources and inexhaustible resources). Anchored within the realizability framework our proof is modular enough so that further extensions (to second order, to additive constructs or to any other independent feature that can be dealt with using realizability) come for free.

Keywords: Linear Logic, Proof Nets, Differential Linear Logic, Differential Interaction Nets, Realizability, Weak Normalization.

Introduction

It happens that the three differential constructs of *Differential Linear Logic* (also abbreviated as *DiLL*), namely *co-weakening*, *co-contraction* and *co-dereliction*, and the original *promotion* construct from *Linear Logic* (*LL*) can be used altogether in a system that has been called *Full Differential Linear Logic* (*Full-DiLL*). In particular, this system embeds *Differential λ -calculus*, which was the first avatar of the differential paradigm introduced by *Ehrhard* and *Regnier* [1,3]. A similar system has been studied within an intuitionist setting by *Tranquilli* [9], and a combinatorial proof of its weak normalization has already been provided by *Pagani* [7]. We provide in this paper a new proof of its weak normalization, using a reducibility technique which is of great importance because it is the only known method to prove such a result in presence of second order quantifiers.

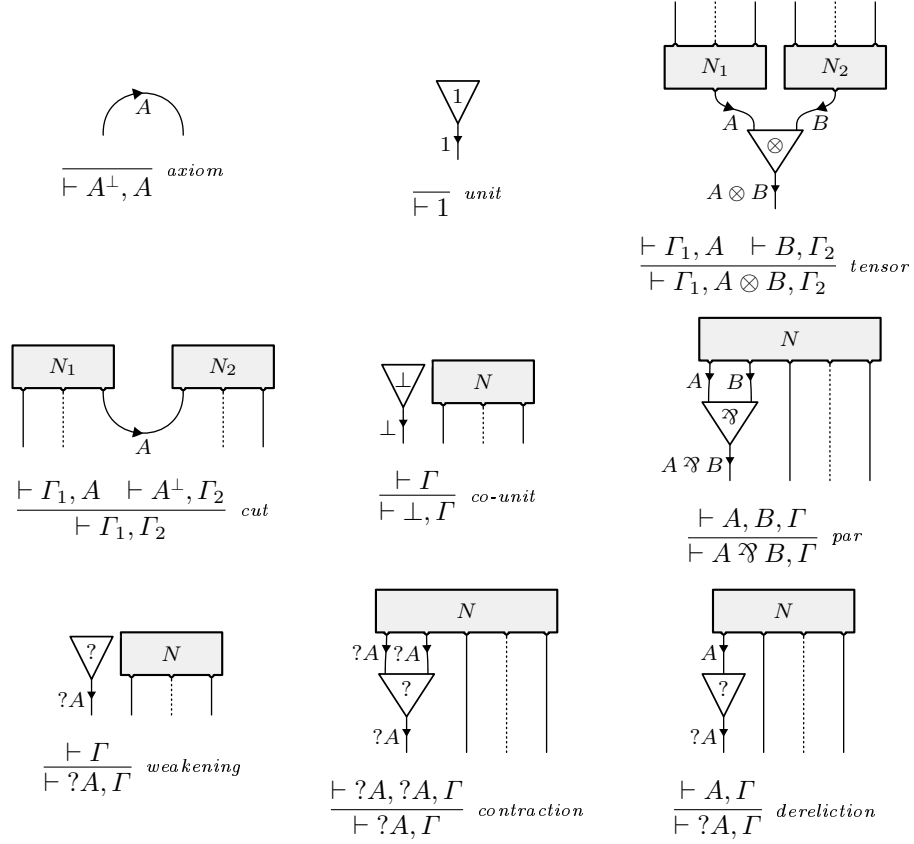
Plan A brief presentation of *Full-DiLL* is given in Section 1. Realizability tools used by *Girard* are introduced in Section 2 and the proof he provided for weak normalization of *LL* is recalled in Section 3. Weak normalization of *Full-DiLL* will be addressed in Section 4.

1 Full Differential Linear Logic

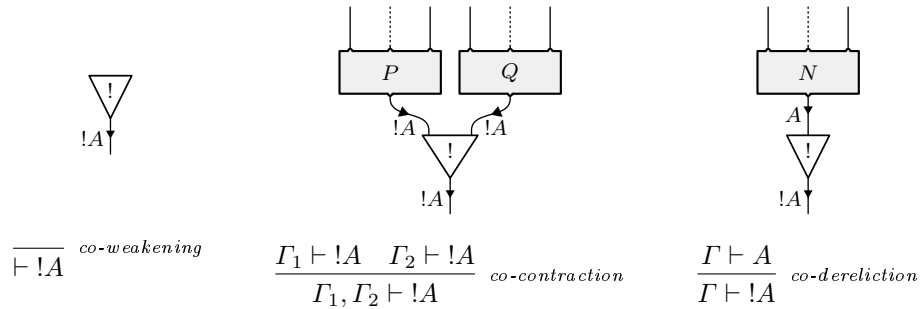
1.1 Syntax

In this paper, we will depict proofs of *LL* and *Full-DiLL* graphically using the *interaction nets* formalism [5], which is well suited to multiplicative constructions [6], and is the usual way to represent differential constructions, see [2].

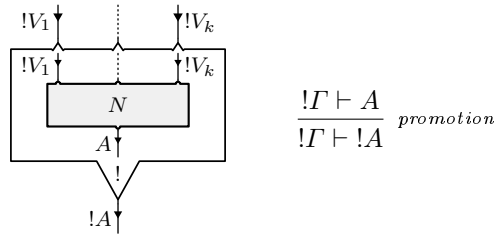
Our study will be restricted to *multiplicative* and *exponential* fragments of these logics (whose constructions are recalled below), because an extension of our result to *additive* and *second order* fragments can be extracted from Girard's proof [4] without any significant changes, and it would otherwise be cumbersome.



Specifically in the case of *Full-DiLL*, we add the three differential constructs which are represented like this:



Intuitively, links labeled with exponential modalities like $!A$ can be thought as communication channels between resource producers and resource consumers. A *co-weakening* provides an empty bag of resources, a *co-dereliction* provides a bag with a single resource and *co-contractions* allow to merge such bags. As for *promotions* (bags providing inexhaustible resources), they will be displayed as *boxes*, similarly to what is done in the standard *Proof Net* formalism.



Informally, a (simple) net is just a set of *nodes* and *free ports* linked by *wires* such that some typing constraints are satisfied. In their graphical representation, wires are labeled with the type of the destination port (according to the chosen orientation) which has to be orthogonal to the type of the source port. A net is said to have *interface* $\vdash A_1, \dots, A_n$ when its free ports have types A_1, \dots, A_n . Sequent rules can then be thought as valid steps to build nets. In fact, despite the syntactical freedom suggested by their graphical representations, most of the nets we consider will be valid in this sense.

Definition 1 (Sequentializable nets). *A net that is the direct translation of a proof from usual linear logic will be called an LL net. If the use of formal sums and sequent rules for co-weakening, co-dereliction, and co-contraction is also allowed, the net will be called a Full-DiLL net.*

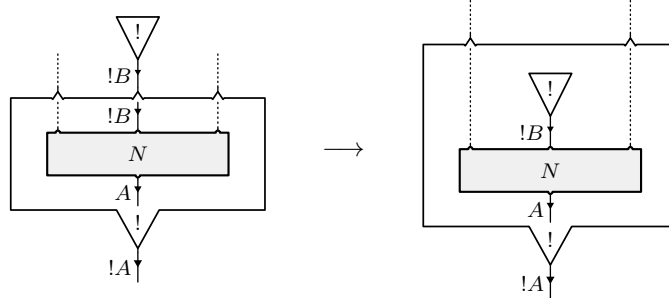
Inside a differential setting, we should indeed precise that all nets we consider are generalized to formal sums of simple nets that share the same interface. This generalization is needed to define their dynamics. Such sums distribute over any context but *promotion* which is the only construction that shall not be considered linear: a box is a special node which is itself parametrized by a formal sum of simple nets, and this sum will not distribute outside the scope of the box (unless it is open).

1.2 Dynamics

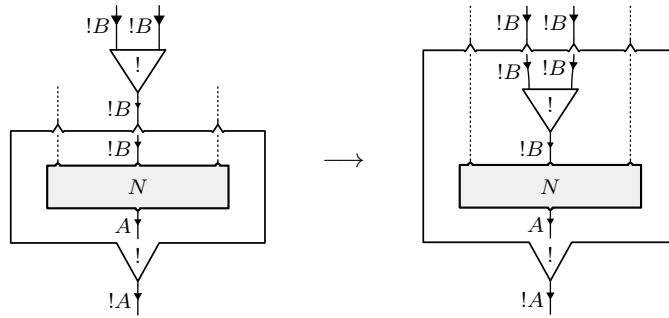
Written in a net formalism, this full system can be endowed with a dynamic similar to *cut-elimination*, by means of a reduction relation \longrightarrow that is contextual, preserves validity, and combines:

- reduction rules of standard *Proof Nets*, see [4].
- reduction rules of *Differential Interaction Nets*, see [2].
- and the following set of three rules, which tells how *promotions* interact with differential constructions.

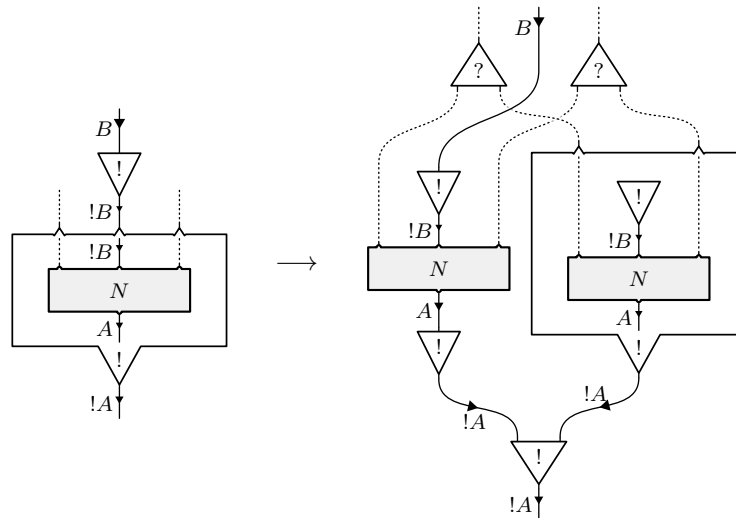
A *co-weakening* in front of a *box input* enters the box.



A *co-contraction* in front of a *box input* also enters the box.



When a *co-dereliction* encounters a *box input*, their interaction is called *chain rule* (the reason being that somehow it plays the same role as the one with the same name which is more widely known in mathematics):

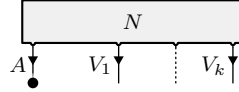


2 Realizability Tools

In this section, we recall what was Girard's approach to realizability, rephrasing it slightly to match with interaction nets syntax.

2.1 Observable and duality

Definition 2 (Pointed net). A pointed net is a net with a distinguished port (represented graphically by a dot).

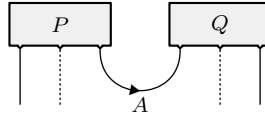


Such a net is said to have interface $\vdash A; V_1, \dots, V_k$. And \mathcal{N}_A denote the set of pointed nets whose distinguished port is of type A . We will also use \mathcal{N}_A^\dagger to denote the subset of \mathcal{N}_A whose elements can be promoted, that is, nets with interface $\vdash A; ?V_1, \dots, ?V_k$.

A pointed axiom of type A will be denoted by $Axiom_A$:



Also, when $P \in \mathcal{N}_A$ and $Q \in \mathcal{N}_{A^\perp}$, we use $Cut_A(P, Q)$ to denote the cut between nets P and Q along their distinguished ports:



Let \mathcal{O} be a chosen set of nets called *observable*, we define the binary relation \star on pointed nets such that $P \star Q$ holds if and only if $P \in \mathcal{N}_A$ and $Q \in \mathcal{N}_{A^\perp}$ and we have $Cut_A(P, Q) \in \mathcal{O}$.

Definition 3 (Duality). If \mathcal{P} is a set of pointed nets of type A , its dual is defined as:

$$\mathcal{P}^* := \{Q \mid \forall P \in \mathcal{P}, P \star Q\}$$

Property 1. For any set of pointed nets $\mathcal{P} \subseteq \mathcal{N}_A$ and $\mathcal{Q} \subseteq \mathcal{N}_A$, the following properties hold:

$$\mathcal{P} \subseteq \mathcal{Q} \implies \mathcal{Q}^* \subseteq \mathcal{P}^* \qquad \mathcal{P} \subseteq \mathcal{P}^{**} \qquad \mathcal{P}^{***} = \mathcal{P}^*$$

Proof. The two first assertions may be deduced without difficulties from the definition of duality. As for the third one, $\mathcal{P}^{***} \subseteq \mathcal{P}^*$ is obtained substituting \mathcal{P} by \mathcal{P}^* in the second one, and $\mathcal{P}^{***} \supseteq \mathcal{P}^*$ is obtained substituting \mathcal{Q} by \mathcal{P}^{**} in the first one.

Exponentials The behavior $\llbracket !A \rrbracket \subseteq \mathcal{N}_{!A}$ is defined from $\llbracket A \rrbracket \subseteq \mathcal{N}_A$ as follows:

$$\llbracket !A \rrbracket := \left\{ \left(\begin{array}{c} \text{---} \text{---} \text{---} \\ \vdots \\ \text{---} \text{---} \text{---} \\ \boxed{P} \\ \text{---} \text{---} \text{---} \\ \downarrow \\ A \\ \downarrow \\ !A \\ \bullet \end{array} \right) \mid P \in \llbracket A \rrbracket \cap \mathcal{N}_A^\dagger \right\}^{**}$$

And $\llbracket ?A \rrbracket \subseteq \mathcal{N}_{?A}$ is defined by:

$$\llbracket ?A \rrbracket := \llbracket !A^\perp \rrbracket^*$$

3 Weak Normalisation of Linear Logic

We now chose for \mathcal{O} the set of all weakly normalizing nets. Then, when $P \in \mathcal{N}_A$ and $Q \in \mathcal{N}_{A^\perp}$, $P \star Q$ means that $Cut_A(P, Q)$ reduces to a normal form. Also, $Axiom_A$ being obviously a weakly normalizing net, we can deduce from the definition of duality that $\mathcal{O}_{A^\perp}^* \subseteq \mathcal{O}_A$.

3.1 Reducibility Candidates in LL

Definition 5 (Reducibility candidate). A reducibility candidate is a behavior \mathcal{P} such that:

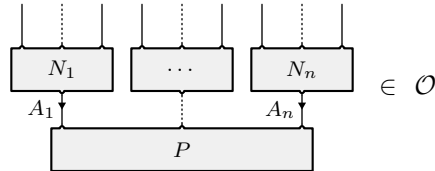
$$\mathcal{O}_{A^\perp}^* \subseteq \mathcal{P} = \mathcal{P}^{**} \subseteq \mathcal{O}_A$$

Lemma 1. The set of weakly normalizing nets being chosen as observable, for every type A , behavior $\llbracket A \rrbracket$ previously defined is a reducibility candidate.

Proof. The required inclusions hold for base types. Those spread inductively to behaviors built with positive connectors (i.e. 1 , \otimes , $!$), because the various constructions involved in their definition (including double dual closures) preserve weak normalization. Building behaviors for types with negative connectors (i.e. \perp , \wp , $?$) involves duality, but according to the first statement in Property 1, the sets obtained satisfy the same inclusions.

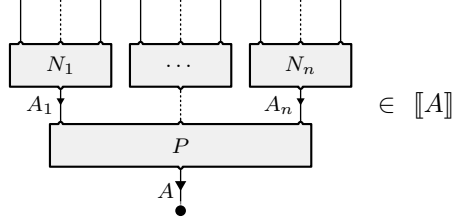
3.2 Reducibility

Definition 6 (Net reducibility). A net P with interface $\vdash A_1^\perp, \dots, A_n^\perp$ is reducible when for every family of nets $(N_i)_{i \in [1, n]}$ such that $N_i \in \llbracket A_i \rrbracket$, we have:



The pictured net will also be denoted shortly as $Cuts(P; N_1, \dots, N_n)$, sometimes abbreviated further using vectorial notation as $Cuts(P; \mathbf{N})$.

Definition 7 (Pointed net reducibility). A pointed net P with interface $\vdash A; A_1^\perp, \dots, A_n^\perp$ is reducible when for every family of nets $(N_i)_{i \in [1, n]}$ such that $N_i \in \llbracket A_i \rrbracket$, we have:



This last net will be denoted using the same notation $Cuts(P; N_1, \dots, N_n)$. Notice that when P is a pointed net, no net is plugged on the distinguished port.

Property 3. Reducibility of a net N is equivalent to reducibility of any pointed net obtained from N choosing one of its ports as distinguished.

The weak normalization theorem for *Linear Logic* will follow quickly from the next lemma.

Lemma 2. Every LL net is reducible.

Proof. By induction on the net N we consider:

(Axiom) We must show that for all $N_1 \in \llbracket A \rrbracket$ and $N_2 \in \llbracket A^\perp \rrbracket$ the net $N' = Cuts(Axiom_A; N_1, N_2)$ is normalizing. This net is simply $Cut_A(N_1, N_2)$, which normalizes because $\llbracket A^\perp \rrbracket = \llbracket A \rrbracket^*$, and therefore $N_1 \star N_2$.

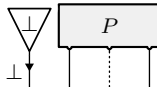
(Cut) We must show that for all reducible $R_1 \in \llbracket A \rrbracket$ and $R_2 \in \llbracket A^\perp \rrbracket$, the net $N' = Cuts(Cut_A(R_1, R_2); \mathbf{N})$ normalizes. Quantification over pointed nets N_i is universal, they are supposed to be in their respective candidates: $N_i \in \llbracket A_i \rrbracket$ (this kind of quantification will be left implicit in the rest of the proof). In this case, since $Cuts(R_1; \mathbf{N}) \in \llbracket A \rrbracket$ and $Cuts(R_2; \mathbf{N}) \in \llbracket A^\perp \rrbracket$ the normalization of N' follows.

(Unit) When N is of this form:



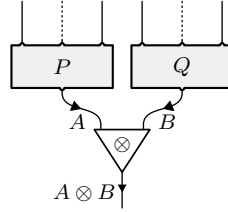
The associated pointed net will be denoted as $Unit_\bullet$. Using Property 3, reducibility of N can be written $Unit_\bullet \in \llbracket 1 \rrbracket$, but this holds by definition.

(Co-unit) When N is of the following form, and assuming reducibility of P by induction hypothesis:



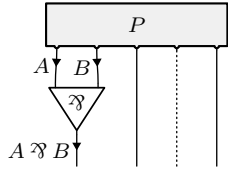
We use N^\bullet to denote the associated pointed net, the reducibility of N can be written $Cuts(N^\bullet; \mathbf{N}) \in \llbracket \perp \rrbracket$. Given that $\llbracket \perp \rrbracket = \{Unit_\bullet\}^*$, this assertion holds when the net $Cuts(N; Unit_\bullet, \mathbf{N})$ normalizes. This net reduces to $Cuts(P; \mathbf{N})$ which normalizes by reducibility of P .

(Tensor) When N is of the following form, and assuming reducibility of P and Q as induction hypothesis:



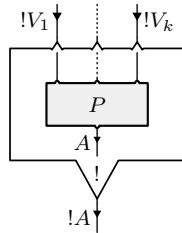
We use $Tensor_\bullet(P, Q)$ to denote the associated pointed net. Reducibility of N can be written $Cuts(Tensor_\bullet(P, Q), \mathbf{N}) \in \llbracket A \otimes B \rrbracket$, and holds by definition of $\llbracket A \otimes B \rrbracket$ because $Cuts(P; \mathbf{N}) \in \llbracket A \rrbracket$ and $Cuts(Q; \mathbf{N}) \in \llbracket B \rrbracket$ by reducibility of P and Q .

(Par) When N is of the following form, and assuming reducibility of P as induction hypothesis:

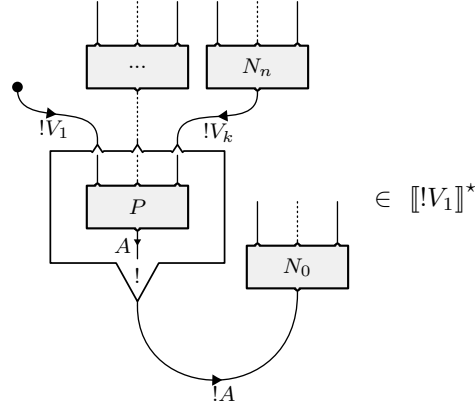


Reducibility of N can be written $Cuts(N^\bullet; \mathbf{N}) \in \llbracket A \circ B \rrbracket$. Given that $\llbracket A \circ B \rrbracket = \{Tensor_\bullet(R_1, R_2) \mid R_1 \in \llbracket A^\perp \rrbracket, R_2 \in \llbracket B^\perp \rrbracket\}^*$, this assertion holds when for all $R_1 \in \llbracket A^\perp \rrbracket$ and $R_2 \in \llbracket B^\perp \rrbracket$, the net $Cuts(N; Tensor_\bullet(R_1, R_2), \mathbf{N})$ normalizes. But, the latter reduces to $Cuts(P; R_1, R_2, \mathbf{N})$ which normalizes by reducibility of P .

(Promotion) When N is of the following form, and assuming reducibility of P as induction hypothesis:

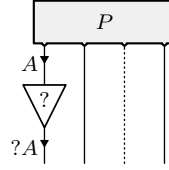


This net is also denoted as $Prom(P)$. Its reducibility can be written: for all $N_0 \in \llbracket !A \rrbracket^*$, for all $N_i \in \llbracket !V_i \rrbracket$, $Cuts(Prom(P); N_0, \mathbf{N}) \in \mathcal{O}$. But this holds as soon as for all $N_0 \in \llbracket !A \rrbracket^*$ and for all $N_i \in \llbracket !V_i \rrbracket$ ($i > 1$) we have:



Given that $[[!V_i]]^* = \{ Prom_{\bullet}(R) \mid R \in [[V_i]] \}^*$, we now need to show that $Cuts(Prom(P); N_0, \mathbf{N}) \in \mathcal{O}$, but only for nets $N_1 \in \{ Prom_{\bullet}(R) \mid R \in [[V_1]] \}$. Let $N_1 = Prom_{\bullet}(R_1)$ be a net from this set (that is $R_1 \in [[V_1]]$). Let us then consider our net pointed on the second *box input*, and by iteration of this method, our problem boils down to the case where all nets N_i are such that $N_i = Prom_{\bullet}(R_i)$ for some $R_i \in [[V_i]]$. Finally, we just need to show $Cuts(Prom_{\bullet}(P); Prom_{\bullet}(R_i)) \in [[!A]]$. This net reduces by exponential commutation to $Prom_{\bullet}(P')$ where $P' = Cuts(P; Prom_{\bullet}(R_i))$. But P being reducible by induction hypothesis, since $R_i \in [[V_i]]$, we obtain $P' \in [[A]]$. Therefore, $Prom_{\bullet}(P') \in [[!A]]$.

(Dereliction) When N is of the following form, and assuming reducibility of P as induction hypothesis:



Reducibility of N can be written $Cuts(N^{\bullet}; \mathbf{N}) \in [[?A]]$. Given that $[[?A]] = \{ Prom_{\bullet}(R) \mid R \in [[A^{\perp}]] \}^*$, this assertion holds as soon as for all $R \in [[A^{\perp}]]$, the net $Cuts(N; Prom_{\bullet}(R), \mathbf{N})$ normalizes. But this net reduces to $Cuts(P; R, \mathbf{N})$, which normalizes by reducibility of P .

(Weakening) Idem, the net reduces to $Cuts(P; \mathbf{N})$ plus some *weakenings*.

(Contraction) Idem, the net reduces to $Cuts(P; Prom_{\bullet}(R), Prom_{\bullet}(R), \mathbf{N})$ plus some *contractions*.

Theorem 1. *Reduction of LL nets is weakly normalizing.*

Proof. It is a corollary of Lemma 2: every net N is reducible, and we can instantiate the N_i by axioms in the definition of reducibility.

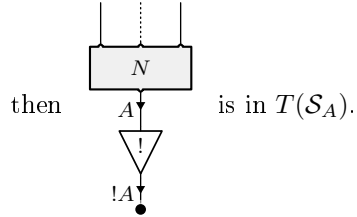
4 Weak Normalisation of Full Differential Linear Logic


We now come to the actual contribution of this paper and extend the proof of the previous theorem to differential constructions.

4.1 Behaviors and Reducibility Candidates in *Full-DiLL*

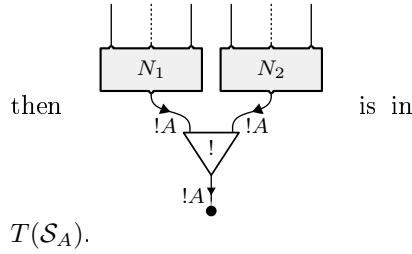
In presence of differential constructions, behaviors for exponential types must be built differently. First, for every set $\mathcal{S}_A \subseteq \mathcal{N}_A$, we define inductively a set $T(\mathcal{S}_A)$ as the smallest subset of $\mathcal{N}_{!A}$ such that:

– if $N \in \mathcal{S}_A$,

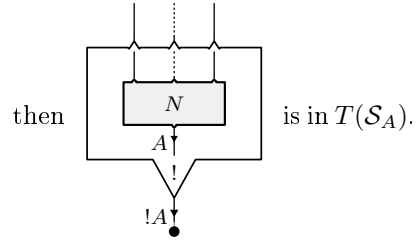



–  is in $T(\mathcal{S}_A)$.

– if $N_1 \in T(\mathcal{S}_A)$ and $N_2 \in T(\mathcal{S}_A)$,

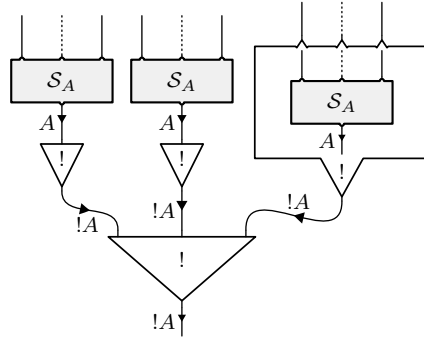


– if $N \in \mathcal{S}_A \cap \mathcal{N}_A^\dagger$,



– the axiom  is in $T(\mathcal{S}_A)$.

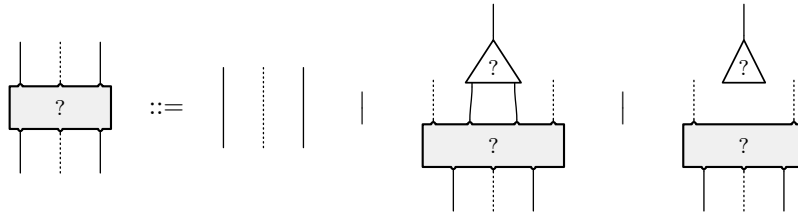
The set $T(\mathcal{S}_A)$ contains nets alike to the one given below, where some *co-derelections* and *promotions* are found behind a *co-contraction* tree of arbitrary arity. Informally, one could say that going from \mathcal{S}_A to $T(\mathcal{S}_A)$ consists in applying a macro-construction constituted of several exponential operators, but restricted to a single exponential layer.



Let $\llbracket !A \rrbracket := (T(\llbracket A \rrbracket))^{**}$ be the new inductive way to define exponential behaviors (and reducibility candidates) in *Full-DiLL*. It replaces the one given for *LL* in 2.2. Notice that in a differential setting, nets belonging to all those sets we consider are generalized nets (formal sums of simple nets).

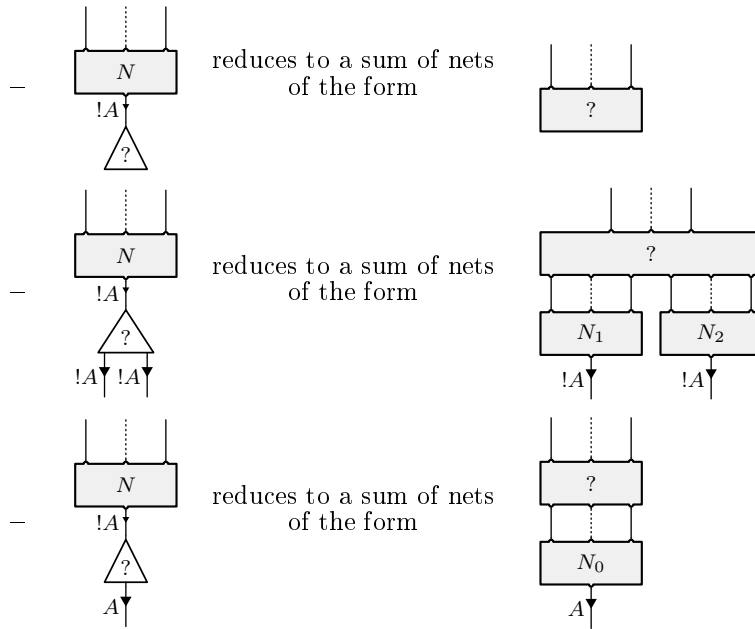
4.2 Reduction of exponential macro-constructions

We use rectangular cells labeled $\langle ? \rangle$ to represent trees of *weakening* and *contraction* nodes directed upwards:



In this definition, some links might not be connected to any nodes. Those links are not required to have exponential types.

Property 4. For every $N \in T(\mathcal{S}_A)$,



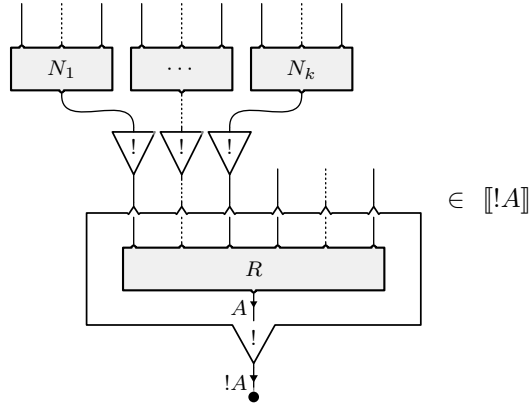
where $N_1, N_2 \in T(\mathcal{S}_A)$ and $N_0 \in \mathcal{S}_A$ (those nets might be different for each member of the sums obtained).

Proof. A safe method is to prove all the three cases simultaneously by induction on N .

4.3 Reducibility

Net reducibility is still expressed by the two properties that were given in Definition 6 and Definition 7. Those definitions would be slightly more complex if second order was explicitly considered (details can be found in [4]). However, this would not interfere with our following developments. As expected, a difficulty to prove normalization of our system comes from the *chain rule* given in 1.2.

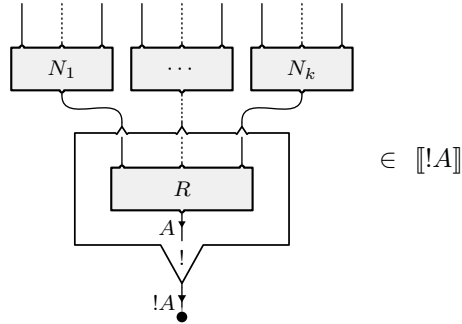
Lemma 3. *Take any reducible net R with interface $\vdash A; ?V_1^\perp, \dots, ?V_k^\perp, ?U_1, \dots, ?U_l$ and let $N_1 \in \llbracket V_1 \rrbracket, \dots, N_k \in \llbracket V_k \rrbracket$, we have:*



Proof. Recursively on k . No difficulty when $k = 0$. When $k > 0$, we assume that the property holds for any $k' < k$ (generalized recurrence hypothesis). Applying the *chain rule* to deal with the interaction between one *co-dereliction* and the box, and then solving the interactions between generated *contractions* and remaining *co-derelictions*, bring us to a sum of several nets for which the recursion hypothesis applies (the number of *co-derelictions* was diminished strictly below k). Thanks to this hypothesis, one can show that each net of the sum is in $\llbracket !A \rrbracket$.

This statement can be generalized to settings where arbitrary exponential constructs face the inputs of a *promotion* box.

Lemma 4. *Let R be a reducible net with interface $\vdash A; ?V_1^\perp, \dots, ?V_k^\perp$ and let $N_1 \in T(\llbracket V_1 \rrbracket), \dots, N_k \in T(\llbracket V_k \rrbracket)$, we have:*



Proof. Again, we will reduce this net. Though, to avoid complications we need to use a specific strategy. Two cases can occur: **(i)** every N_i is a *co-derelection* or an *axiom*. **(ii)** one of the N_i is not a *co-derelection* or an *axiom*. The first case is dealt with using Lemma 3. The second case reduces to the first one using exponential commutations: a *co-weakening* or an other *box* will enter the main box in one step, and similarly a *co-contraction* will, but in several steps (the operation needs to be iterated for both nets behind it). Since by global hypothesis all those constructs are in their respective candidates, we can show that the net obtained inside the box after reduction remains reducible, and we actually fall back to the first case.

The inductive definition of $T(\mathcal{S}_A)$ has been chosen so as to obtain this lemma. It is essential in the proof of the following theorem which is the main result of this paper. The fact that we used a specific reduction strategy in the proof of this lemma means that some more work would be needed if we were to prove strong normalisation of the system, as it was done by *Pagani and Tortora de Falco* for *LL* in [8]. It is presently not clear whether this would be the only point that would raise difficulties.

Theorem 2. *Every Full-DiLL net is reducible.*

Proof. By induction on the net N :

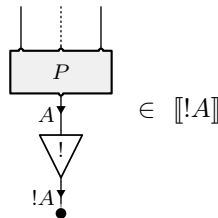
(Axiom, Cut and Multiplicatives) Same as in the proof of Lemma 2.

(Co-weakening) When N is a simple *co-weakening*:



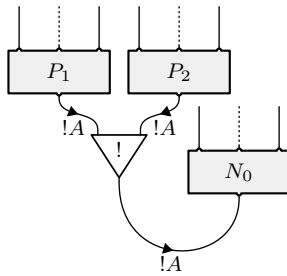
N is part of the syntax $T(\llbracket A \rrbracket)$ and therefore it belongs to $\llbracket !A \rrbracket$ by definition.

(Co-derelection) When N is a *co-derelection* construct, assuming $P \in \llbracket A \rrbracket$ by induction hypothesis, we obtain again:

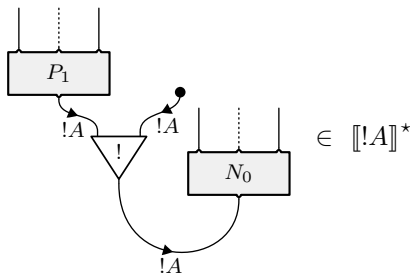


as a direct consequence of the fact that this net belongs to $T(\llbracket A \rrbracket)$.

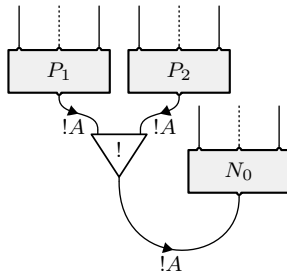
(Co-contraction) The *co-contraction* raises an issue similar to the one of *promotion* in the proof of Theorem 2. To deal with this case, it is sufficient to show that when $P_1 \in \llbracket !A \rrbracket$, $P_2 \in \llbracket !A \rrbracket$ and $N_0 \in \llbracket !A \rrbracket^*$ the following net normalizes:



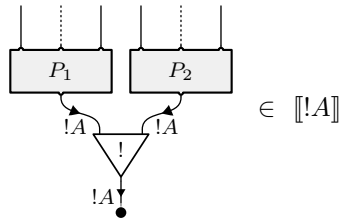
In other words, we have to show that:



That is to say, by definition of $[[!A]]$, that the following net normalizes for all $P_2 \in T([[A]])$:

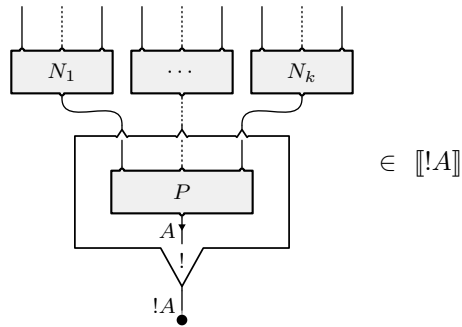


By the same argument, we need to prove this only for $P_1 \in T([[A]])$. Finally, we are reduced to showing that for all $P_1 \in T([[A]])$ and $P_2 \in T([[A]])$:



and this is a direct consequence of the definition of $[[!A]]$.

(Promotion) When P is reducible and $N_1 \in [[!V_1]], \dots, N_k \in [[!V_k]]$, we want:



Always using the same argument, everything boils down to cases where $N_1 \in T(\llbracket V_1 \rrbracket)$, \dots , $N_k \in T(\llbracket V_k \rrbracket)$, and Lemma 4 can be applied.

(Weakening, Dereliction, Contraction) Nets built with these three constructions interact with exponential macro-constructions in a normalizing way. Property 4 is used to deal with these cases.

(Sums of nets) Let J be a finite set, and for every $j \in J$, let P_j be a reducible net with interface $\vdash A_1^\perp, \dots, A_n^\perp$. We want to show that $\sum_{j \in J} P_j$ is also reducible. In fact, we know that for every family $N_i \in \llbracket A_i \rrbracket$, the net $Cuts(\sum_{j \in J} P_j; \mathbf{N}) = \sum_{j \in J} Cuts(P_j; \mathbf{N})$ normalizes because for every $j \in J$, by reducibility of P_j , we know that $Cuts(P_j; \mathbf{N})$ normalizes.

Acknowledgments I want to thank *Thomas Ehrhard* for his constant support, *Christine Tasson* who also helped in proof-reading this paper, and *Paolo Tranquilli* for the fruitful discussions we had some time ago.

References

1. Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. In *TCS*, volume 309, pages 1–41. Elsevier Science Publishers Ltd., 2004.
2. Thomas Ehrhard and Laurent Regnier. Differential interaction nets. In *TCS*, volume 364, pages 166–195. Elsevier Science Publishers Ltd., 2006.
3. Thomas Ehrhard and Laurent Regnier. Uniformity and the Taylor expansion of ordinary lambda-terms. In *TCS*, volume 403, pages 347–372. Elsevier Science Publishers Ltd., 2008.
4. Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
5. Yves Lafont. Interaction nets. In *Principles of Programming Languages*, pages 95–108. ACM, 1990.
6. Yves Lafont. From proof nets to interaction nets. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pages 225–247. Cambridge University Press, 1995.
7. Michele Pagani. The Cut-Elimination Theorem for Differential Nets with Boxes. In Pierre-Louis Curien, editor, *TLCA*, LNCS, pages 219–233. Springer, 2009.
8. Michele Pagani and Lorenzo Tortora de Falco. Strong Normalization Property for Second Order Linear Logic. *Theoretical Computer Science*, 411(2):410–444, 2010.
9. Paolo Tranquilli. Intuitionistic differential nets and lambda-calculus. *Theoretical Computer Science*, 412(20):1979–1997, 2011.