On the dependencies of logical rules

Marc Bagnol[1], Amina Doumane[2], and Alexis Saurin[2] *

[1] IML, Université d'Aix-Marseille, bagnol@iml.univ-mrs.fr
[2] PPS, CNRS, Université Paris Diderot & INRIA,
{amina.doumane,alexis.saurin}@pps.univ-paris-diderot.fr

Abstract. Many correctness criteria have been proposed since linear logic was introduced and it is not clear how they relate to each other. In this paper, we study proof-nets and their correctness criteria from the perspective of *dependency*, as introduced by Mogbil and Jacobé de Naurois. We introduce a new correctness criterion, called DepGraph, and show that together with Danos' contractibility criterion and Mogbil and Naurois criterion, they form the three faces of a notion of dependency which is crucial for correctness of proof-structures. Finally, we study the logical meaning of the dependency relation and show that it allows to recover and characterize some constraints on the ordering of inferences which are implicit in the proof-net.

Keywords: Linear logic, MLL, Proof nets, Correctness criterion, Contractibility, Mogbil-Naurois Criterion, Permutability of inferences

1 Introduction

The benefits of Curry-Howard. Since the discovery of Curry-Howard correspondence [2], that is of the deep connections between logical proofs and computer programs, programming language theory and proof theory have been tightly intertwined.

Among the numerous and fruitful back-and-forths between proofs and programs, linear logic [3] certainly has to stand as exemplary. Girard indeed introduced system F [4] while working on second-order arithmetics. This system, independently rediscovered by Reynolds [5], features a polymorphic λ-calculus (or second-order λ-calculus).

About fifteen years later, Girard established the coherent semantics of system F [6] which elaborates on Berry stable semantics [7]. Coherent spaces led to a decomposition of the implication connective which, in turn, is the cornerstone of linear logic [3]: $A \Rightarrow B = \,!\,A \multimap B$. This linear decomposition, observed in the semantics of system F, turns to be syntactically reflected in a very well-behaved proof system. A new field of thoughts was opened where linear logic concepts radically renewed the traditional viewpoints on proofs and programs.

* An extended version with supporting proofs can be found in [1] at http://www.pps.univ-paris-diderot.fr/~saurin/Publi/DepGraphLong.pdf.

On the logical side, proof nets [3,8,9] and interactive semantics (geometry of interaction [10], game semantics [11], ludics [12]) were the first and probably most striking novelties of linear logic. More recently, polarized logics [13], light logics [14] and differential logic [15] turn out to impact programming theory and provide new proof-theoretical methods to analyze side-effects, computational complexity or non-determinism.

On the programming side, linearity impacted several problems: optimality and sharing [16,17], computational interpretation of classical logic [18,19,13], type system providing information about the computational complexity of the programs [20], connections between focalization and logical languages [21,22], logical interpretation of pattern-matching [23,24], and more recently non-determinism and probabilistic programming [25].

From linear logic to proof nets. With the above mentioned linear decomposition, entered into the picture exponential connectives (!, ?), which control the usage of hypothesis (via restrictions on the use of contraction and weakening structural rules). An essential consequence of the fine-grained control on structural rules is that various inferences for conjunction (resp. disjunction) which are usually provably equivalent in classical logic, are no more inter-derivable in the linear setting. Conjunction (resp. disjunction) is thus split into two connectives. Logical constants for validity and absurdity are split as well. Those new logical connectives can be structured in a multiplicative group ($\parr$, $\otimes$, 1, $\bot$) and an additive group ($\oplus$, &, $\top$, 0). Linearity allows to retain an involutive negation ($A^{\bot\bot} = A$) as in classical logic while constructive principles still holds for some connectives. De Morgan laws thus allow to restrict the use of negation to atomic formulas only and to consider sequents where all formulas are to the right of the sequent, reducing the size (and to some extent the redundancy) of the proof systems. Last, linear logic comes with a strong notion of fragments: one can consider the fragment restricted to multiplicative connectives (with or without logical constants), called MLL, the fragment for only multiplicative and exponential connectives, called MELL...

Proof nets deserve a special treatment in that they probably constitute the most original innovations of linear logic. They constitute a graphical notation for proofs, resulting in very canonical proof objects [3] in which cut-elimination is very elegant, and simple.

The beauty of proof-nets is especially striking in the multiplicative fragment with no logical constant (also said unit-free multiplicative logic) to which the rest of the paper will be dedicated. Actually, it is more a matter of simplifying the presentation than a true restriction of our results since our results can straightforwardly be adapted to MELL, thus capturing typed lambda-calculus.

Proof-nets and logical correctness. By representing of proofs as graphs instead of more usual sequential notations, we step back from the intuition of what a proof

[3] Contrarily to sequent proofs which contain a lot of irrelevant information on the ordering of inferences which has no logical justification but the sequential nature of sequent proofs which structure proofs as trees.

is. (Typically a reasoning which is conducted progressively, in an ordered way, from its starting point to its conclusion, structured with intermediate lemmas and where each step shall be logically valid.)

A consequence of proof-nets being non-sequential is that, if such a proof-object is logically flawed (that is, if it contains some paralogism), the logical errors may not be easily detected, contrarily to most of the deductive systems (Hilbert systems, natural deduction, sequent calculus, deep inference, ...) where logical correctness is purely local. One will actually speak of proof structure for an object that looks like a proof net but that may not be logically correct.

This actually is the price to pay for the benefits of a graphical syntax and the well-behaved cut-elimination of proof-nets: moving from inductive objects (sequent calculus proof trees for instance) to more geometrical objects (proof structures) where the characteristic properties are no more local but global, as connectedness or acyclicity.

Still, proofs should primarily (and even before being computational objects) be objects which support an assertion and allow to transmit. Considered along these lines, the correctness problem can be rephrased as: if one communicates a proof structure R, one must have the means to check that the structure is logically sound and does not contain erroneous reasonings [4]. Potentially, the receiver of the proof structure will be willing to have a certificate ensuring him that the proof structure is indeed correct. The simplest idea is of course to transmit the sequentialized proof of R but this naive approach soon reveals its limits: not only because it rests on sequent calculus, but also because the proof structure that we communicate will typically result from cut-elimination processes and that is we do not have a sequentialized proof to use.

One is thus willing to have conditions on proof structures, of a graphical and geometric nature, which ensure the correctness of proof structures, that is which discriminate the proof structures which come from sequent proof from those who don't, those which are logically valid from those that may contain logical mistakes.

Here is the true problem of correctness of proof nets, to which this paper is a contribution. We propose a new correctness criterion which is particularly simple and allows to tell whether a proof structure is a desequentialized sequent proof. Our criterion simplifies that of Mogbil and Jacobé de Naurois and completely abstracts from switchings.

Several correctness criteria have been introduced in the literature. Among the best-known criteria, one can refer to the original long-trip criterion (LT) [3], Danos-Regnier criterion (DR) [9], counter-proofs criterion (CP) [26,27], contractibility (C) [28], graph-parsing criterion (GP) [29,30], Dominator Tree (DT) [31] and more recently Mogbil-Naurois criterion (MN) [32].

On the structure and efficiency of correctness criteria. Danos-Regnier criterion is certainly the most standard correctness criterion for its beauty and because it is convenient to reason about. However, it is not efficient in order to effectively

[4] Or, through the Curry-Howard looking-glass, that the program is well-typed...

test the correctness of a proof structure since one shall check the connectedness and acyclicity of 2^n correction graphs, where n is the number of $⅋$ nodes in the structure. One can hardly hope to improve this bound by restricting the criterion to a well-chosen subset of switchings: one can indeed show that there exist proof structures having an arbitrary large number of $⅋$ of which all correction graphs but one are connected and acyclic (see [1]), unless one can use an additional information on the topology of the net. (For instance, under planarity assumptions one can restrict to considering only two switchings for deciding the correctness of a proof structure [33].)

Other criteria improve on Danos-Regnier bound: for instance, the contractibility criterion [28] has a quadratic complexity while the parsing criterion [30] can be implemented in linear time. Mogbil and Jacobé de Naurois characterized the space complexity of the correctness problem [32].

From the point of view of their structure, some criteria, such as the parsing or contractibility criteria, are directly related to the sequentialization process of a proof net into a sequent proof. Other criteria, such as Danos-Regnier, use the notion of switchings, which induce subgraphs of the proof structure on which a certain property is tested. From this point of view, MN-criterion is slightly odd since it uses an arbitrarily picked switching in order to build the *dependency graph*.

Considering the above mentioned remark on the fact that incorrect proof structure may have an arbitrarily large proportion of connected and acyclic correction graphs, it is surprising that a single switching can ensure correctness. This is the starting point of our work: we wanted to understand what was the true use of picking a switching in MN-criterion for deciding correctness. In the rest of the paper, we will show that the switching can be completely forgotten and that one can reason directly on the proof structure.

Relating correctness criteria. Actually, correctness criteria usually provide us with some specific viewpoints on the proof-theoretical or computational properties of proofs. For instance, they can (i) provide precise means to sequentialize a proof-net into a sequent proof, or (ii) tell us about the complexity of the correctness problem, or even (iii) say something about the structure of proofs.

Although correctness of proof-nets is now well-studied and understood, the question of comparing and relating those criteria attracted much less attention.

Contributions of the paper. The present paper is a contribution in this direction: we investigate a notion of dependency between inferences of a proof structure and use it to compare three correctness criteria (C, MN and DepGraph, a new criterion we introduce here) showing that they constitute three faces of this dependency relation.

We reformulate Contractibility in a big-step version from which arises the notion of dependency that one finds in MN criterion. This leads us to introduce a new criterion, DepGraph. We then show that these three criteria, arising from the notion of dependency, meet the three categories given above: we show that Contractibility gives actually a sequentialization of a proof-net, MN is a criterion with efficiency purposes and DepGraph emphasizes the structural properties of logic since (i) it deals separately with positive and negative inferences, suggestion possible connections with focusing, (ii) it is switching-independent, contrarily to MN, (iv) it makes use of a well-known necessary condition for correction following from Euler-Poincaré property [27] and finally (iv) we use its notion of dependency in order to characterize constraints on the order of introduction of inferences which are shared by all sequentializations of a given proof-net.

We focus on multiplicative and unit-free linear logic. Rather than a restriction of the results, this is a matter of presentation: DepGraph criterion can easily be extended to MELL, thus capturing typed lambda-calculus

Organization of the paper. In Section 2, we recall the basics of proof nets and correctness criteria and dedicate Section 3 to analyzing and comparing the three criteria mentioned above by (i) showing how contractibility is related with sequentialization, (ii) formulating a big-step notion of contractibility, (iii) justifying the occurrence of a dependency relation in proof-nets, (iv) introducing a new correctness criterion, DepGraph and (iv) comparing DepGraph with MN-criterion. We finally focus in Section 4 on the logical meaning of dependency graphs. Due to lack of space, proofs are omitted but can be found in an extended version with supporting proofs and more material, available online in [1].

2 Correctness problem of proof structures in linear logic

2.1 Linear logic and proof nets

MLL. In this paper, we will deal with multiplicative linear logic (MLL), which is a fragment of linear logic. MLL formulas are built from the following grammar:

$$\boxed{A, B := X \mid X^{\perp} \mid A \otimes B \mid A \parr B \quad (X \in \mathcal{V})}$$

MLL is usually presented via a sequent calculus: an MLL sequent is a finite unordered list of MLL formulas, written $\vdash \Gamma$ and a proof is a tree with nodes labelled by $(ax), (cut), (\otimes), (\parr)$ and edges are labelled by sequents as follows:

Identity Group:

$$\frac{}{\vdash A, A^{\perp}}\,(ax) \qquad \frac{\vdash A, \Gamma \quad \vdash A^{\perp}, \Delta}{\vdash \Gamma, \Delta}\,(cut)$$

Multiplicative Group:

$$\frac{\vdash A, \Gamma \quad \vdash B, \Delta}{\vdash A \otimes B, \Gamma, \Delta}\,(\otimes) \qquad \frac{\vdash A, B, \Gamma}{\vdash A \parr B, \Gamma}\,(\parr)$$

Sequent calculus induces a sometimes irrelevant order between inferences. This is evidence by possible permutations between inferences of a sequent proof. We recall those permutations in figure **??**

Notice that cut elimination in MLL terminates but it is not confluent. Indeed, two cut elimination paths of a proof can lead to two cut-free proofs, which are possibly not equal, but which are equivalent modulo the previous permutations.

$$\dfrac{\dfrac{\vdash A, C, \Gamma \qquad \vdash D, \Delta}{\vdash A, C \otimes D, \Gamma, \Delta}\,(\otimes) \qquad \vdash B, \Sigma}{\vdash A \otimes B, C \otimes D, \Gamma, \Delta, \Sigma}\,(\otimes) \quad \leftrightarrow \quad \dfrac{\dfrac{\vdash A, C, \Gamma \qquad \vdash B, \Sigma}{\vdash A \otimes B, C, \Gamma, \Sigma}\,(\otimes) \qquad \vdash D, \Delta}{\vdash A \otimes B, C \otimes D, \Gamma, \Delta, \Sigma}\,(\otimes)$$

$$\dfrac{\dfrac{\vdash A, B, C, D, \Gamma}{\vdash A \parr B, C, D, \Gamma}\,(\parr)}{\vdash A \parr B, C \parr D, \Gamma}\,(\parr) \quad \leftrightarrow \quad \dfrac{\dfrac{\vdash A, B, C, D, \Gamma}{\vdash A, B, C \parr D, \Gamma}\,(\parr)}{\vdash A \parr B, C \parr D, \Gamma}\,(\parr)$$

$$\dfrac{\vdash A, \Gamma \qquad \dfrac{\vdash B, C, D, \Delta}{\vdash B, C \parr D, \Delta}\,(\parr)}{\vdash A \otimes B, C \parr D, \Gamma, \Delta}\,(\otimes) \quad \leftrightarrow \quad \dfrac{\dfrac{\vdash A, \Gamma \qquad \vdash B, C, D, \Delta}{\vdash A \otimes B, C, D, \Gamma, \Delta}\,(\otimes)}{\vdash A \otimes B, C \parr D, \Gamma, \Delta}\,(\parr)$$

Fig. 1: Key cases of inference permutations in the sequent calculus.

Proof structures and proof nets. Proof nets are canonical representations of MLL sequent proofs quotienting them by the previous permutation rules, resulting in a confluent cut-elimination and other very good properties. Proof structures allow to present MLL proofs in a non-sequential way and therefore those objects are not inductively presented anymore which makes the checking of the logical correctness of those object challenging, calling for correctness criteria.

In the following, we shall consider only cut-free proof structures. Indeed, cut behaves exactly as $\otimes$ from the view point of correctness and therefore introduces no difficulty nor interesting aspects in our developments.

Definition 1 (Proof structure). *A proof structure is a finite undirected graph where vertices are labelled by the names of MLL inference rules or the special label* **c** *(for the conclusions of the proof) and edges are labelled with formulas of MLL. Moreover, edges which are adjacent to a vertice are partitioned into premises and conclusions according to the following rules:*

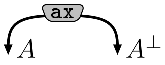
- *Nodes of label $\otimes$ (resp. $\parr$) have two premises and one conclusion. If A is the label of the first premise and B the label of the second premise, then the conclusion is labelled $A \otimes B$ (resp. $A \parr B$);*
- *Nodes of label* **ax** *have no premise and two conclusions. If the label of the first conclusion is A, the label of the second conclusion is $A^\perp$;*
- *Nodes of label* **cut** *have two premises and no conclusion. If the first premise is labelled A, the second premises is labelled $A^\perp$;*
- *Nodes labelled* **c** *have one premise and no conclusion*[5]*.*
- *Every edge is premise of one of its endpoints and conclusion of the other.*

To any MLL proof π, one associates a proof structure $[\pi]$. Any proof-structure which is image of a sequent proof in this way will be called a *proof net.*

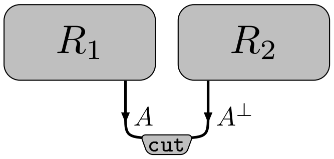
[5] We shall often omit those nodes in the graphical representation of nets: they will be depicted as pending edges.

Definition 2 (Proof net). *The proof structure corresponding to an MLL proof π is defined by induction on the structure of π and by case on the last inference rule of π:*

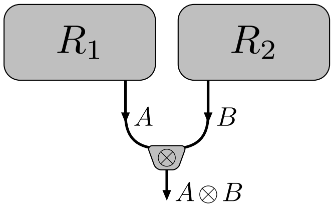
- ***ax*** *rule : the proof net corresponding to $\vdash A, A^{\perp}$ is the graph containing a single node ax with out-going arrows labelled with A and $A^{\perp}$ which connect the* ***ax*** *node with two conclusion nodes:*



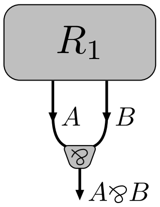
- ***cut*** *rule : if R_1 and R_2 are the proof nets associated with the two subproofs rooted in the premises of the* ***cut*** *rule, the proof net associated with the complete proof is obtained by adding a node labelled* ***cut*** *between the edges corresponding to occurrences of the cut formulae:*



- *$\otimes$ rule : if R_1 and R_2 are proof nets associated with the subproofs rooted in the premises of the $\otimes$ rule, the proof net associated with the complete proof is obtained by adding a $\otimes$ node connected to the edges labelled with the formulas involved in the inference rule and by connecting the outgoing edge to a conclusion node:*



- *$\parr$ rule : if R_1 is the proof net associated to the subproof rooted in the premise of the rule, the proof-net associated with the complete proof is obtained by adding a $\parr$ node connected to the edges labelled with the formulas involved in the inference rule and by connecting the outgoing edge to a conclusion node:*



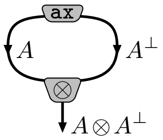


Fig. 2: A proof structure which is not a proof net.

Remark 1. In the graphical representation of proof nets, we put arrows on edges to represent the information on premise/conclusion, but we consider the graph as undirected, in particular with respect to any notion such as paths, cycles, ...

2.2 Correctness criteria

The graph in figure 2 is indeed a proof structure but it cannot be associated with a MLL proof. A proof structure therefore does not necessarily correspond to a sequent calculus proof; such a proof structure is called non-sequentializable. There is a number of methods to distinguish sequentializable proof structures – proof nets – from non sequentializable ones; such methods are called *correctness criteria.*

Several correctness criteria have been introduced in the literature. In the rest of this section, we shall present Danos-Regnier (DR) criterion which is one of the most popular criteria; then we present Contractibility and Mogbil-Naurois (MN) criterion which we will compare in the next section.

2.3 Danos-Regnier Criterion

Definition 3 (Switching). *A* **switching** *of a proof structure R is the choice, for every $\parr$ node of the graph, of one of its premises. More formally, a switching of R is a map from the $\parr$ nodes of R to $\{l, r\}$.*

Given a switching s of a proof structure R, a $\parr$ node n will be said to be switched to the right (resp. to the left) if the right premise (resp. left) has been selected, that is if $s(n) = l$ (resp. r).

Definition 4 (Correction graph). *A* **Correction graph** *$\mathcal{S}(R)$ of a proof structure R and a $\mathcal{S}$ of R is the undirected graph defined as:*

1. *its nodes are those of R;*
2. *its edges are (undirected versions) of R edges where the edge of R corresponding to the left (resp. right) premise of a $\parr$ node n is an edge $\mathcal{S}(R)$ if, and only if, $\mathcal{S}(n) = r$ (resp. l);*
3. *labels of $\mathcal{S}(R)$ are those of the corresponding edge in R.*

Definition 5 (Danos-Regnier criterion (DR)). *A proof structure satisfies the Danos-Regnier criterion if every correction graph is connected and acyclic; in that case, it is said to be DR-correct.*

Note that the statement of Danos-Regnier criterion does not refer explicitly to sequentializability of a proof structure in a sequent calculus proof.

Theorem 1. *A proof structure is a proof net if, and only if, it is DR-correct.*

Proof. See [9]. □

2.4 Contractibility

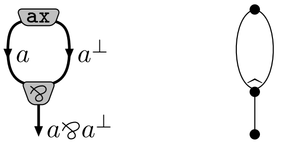
Contractibility criterion expresses a topological property of the proof structure, more precisely of an underlying graph structure, the paired graph which contains just enough information to distinguish premises of a $\parr$ from the other edges.

Definition 6 (Paired graph). *A* **paired graph** *is given by a graph $G = (V, E)$ together with a set $P(G)$ of* **paired edges***, that are undirected pairs of edges which share at least one endpoint. A node which is the endpoint of paired edges is called a* **paired node***, the other endpoints are called* **the premises***.*

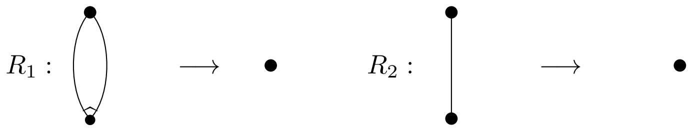
Remark 2. Note that a paired node may have 0, 1 or 2 premises.

Definition 7 ($C(R)$). *To a proof structure R, one associates a paired graph, written $C(R)$, which is simply R together with the set of paired edges given as the set of pairs of edges which are premises of a $\parr$ node.*

Example. *We show below the unique proof net $R_{a \parr a^\perp}$ for the sequent $\vdash a \parr a^\perp$ and the paired graph $C(R_{a \parr a^\perp})$ which is associated to $R_{a \parr a^\perp}$ (paired edges are distinguished by a $\frown$):*



Definition 8 (Contraction rules). *One defines two graph-rewriting rules on paired graphs as follows (note that in both rules the two nodes shall be distinct):*



Definition 9 (Contractibility). *A proof structure R is* **contractible** *if*

$$C(R) \to^* \bullet.$$

Contractibility characterizes proof nets, it provides a correctness criterion:

Theorem 2. *A proof structure is a proof net if, and only if, it is contractible.*

Proof. See [28]. □

2.5 Mogbil-Naurois criterion

We shall first present Mogbil-Naurois criterion, one of the most recent correctness criteria which characterized the space-complexity of the correctness problem.

Definition 10 (Elementary path). *A path in a undirected graph is* **elementary** *when it does not enter twice the same edge.*

Definition 11 (Dependency graph of a correction graph). *Given R a proof structure and $\mathcal{S}$ a switching of R, the dependency graph of $\mathcal{S}(R)$, written $D(\mathcal{S}, R)$ is an oriented graph (V, E) defined as follows:*

- *The set of nodes V consists in the set of conclusions of $\parr$ nodes of R together with an additional node s.*
- *Let x be a $\parr$ node in R, x_r (resp. x_l) its right (resp. left) premise in R.*
  - *There is an edge $(s \to x)$ in E if there exists an elementary path $x_l, \ldots, x_r$ in $\mathcal{S}(R)$ which goes through no $\parr$ node.*
  - *Let y be another $\parr$ node in R. There is an edge $(y \to x)$ if there exists an elementary path $x_l, \ldots, x_r$ in $\mathcal{S}(R)$ containing y.*

Definition 12 ($SDAG$ graphs). *A graph G is $SDAG$ if:*

- *it is acyclic and*
- *it contains a node s, the source node, such that all nodes of G are accessible from s.*

Definition 13 (Mogbil-Naurois Criterion). *A proof structure satisfies the Mogbil-Naurois criterion (MN) if there exists a switching $\mathcal{S}$ such that:*

- *$D(\mathcal{S}, R)$ is $SDAG$ of source s;*
- *$\mathcal{S}$ is connected and acyclic.*

Such a proof structure is said **MN-correct***.*

Theorem 3. *A proof structure is a proof net if, and only if, it is MN-correct.*

Proof. See [32]. □

One notices that dependency graphs are defined on correction graphs and thus they depend on the switching. Compared to Danos-Regnier, the use of switchings in (MN)-criterion is quite odd: it only requires to analyze one switching and the corresponding correction graph. Moreover, the choice of this switching is itself arbitrary. It is therefore natural to wonder what is the exact role of this switching: is it really necessary? We answer this question in the following by going back to the origin of the idea of dependence, which was already present in the contractibility criterion as we shall see in section 3. From that point, we state a dependency-graph based criterion which does not rely on any switching.

3 On the three faces of contractibility

Despite the wide diversity of correctness criteria, their relationship remains poorly studied in the literature. In this section, we shall investigate the connections between three criteria: Mogbil-Naurois, Contractibility and DepGraph which is a new criterion that we introduce in the remainder.

3.1 Contractibilty and sequentialization

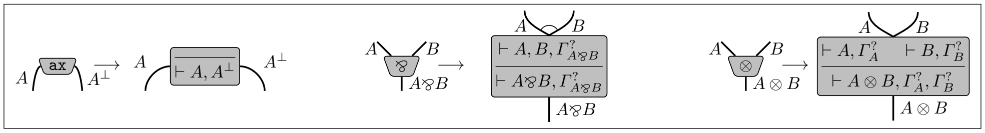
Before relating contractibility with the other two criteria, we make clear that it gives a genuine sequentialization. To do this, we simply label nodes of the paired graph of the proof structures with open proofs containing context variables. These open proofs correspond to partial sequentializations, which become larger and larger as contraction progresses, until reaching a full MLL proof. More precisely, these open proofs are constructed on sequents with context variables, generated by the following syntax (F is a formula and $\Gamma^?$ is a context variable):

$$S := \quad \emptyset \quad | \quad S, F \quad | \quad S, \Gamma^?$$

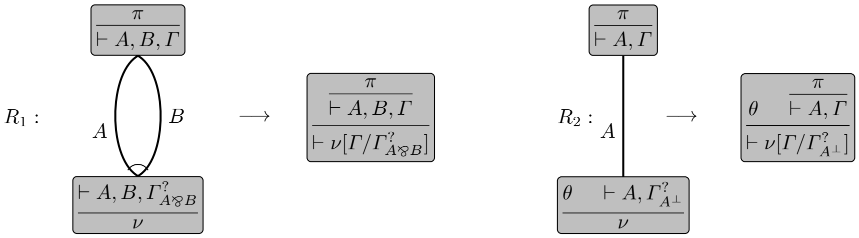
We consider these sequents up to commutativity. Open proofs are constructed by the following syntax:

$$\boxed{\dfrac{}{\vdash A \parr A^\perp}\,(ax) \qquad \vdash S \qquad \dfrac{\vdash S_1, A \qquad \vdash S_2, B}{\vdash S_1, S_2, A \otimes B}\,(\otimes) \qquad \dfrac{\vdash S, A, B}{\vdash S, A \parr B}\,(\parr)}$$

Given a proof structure R, the labelled paired graph $C_l(R)$ is obtained by applying the following rules:

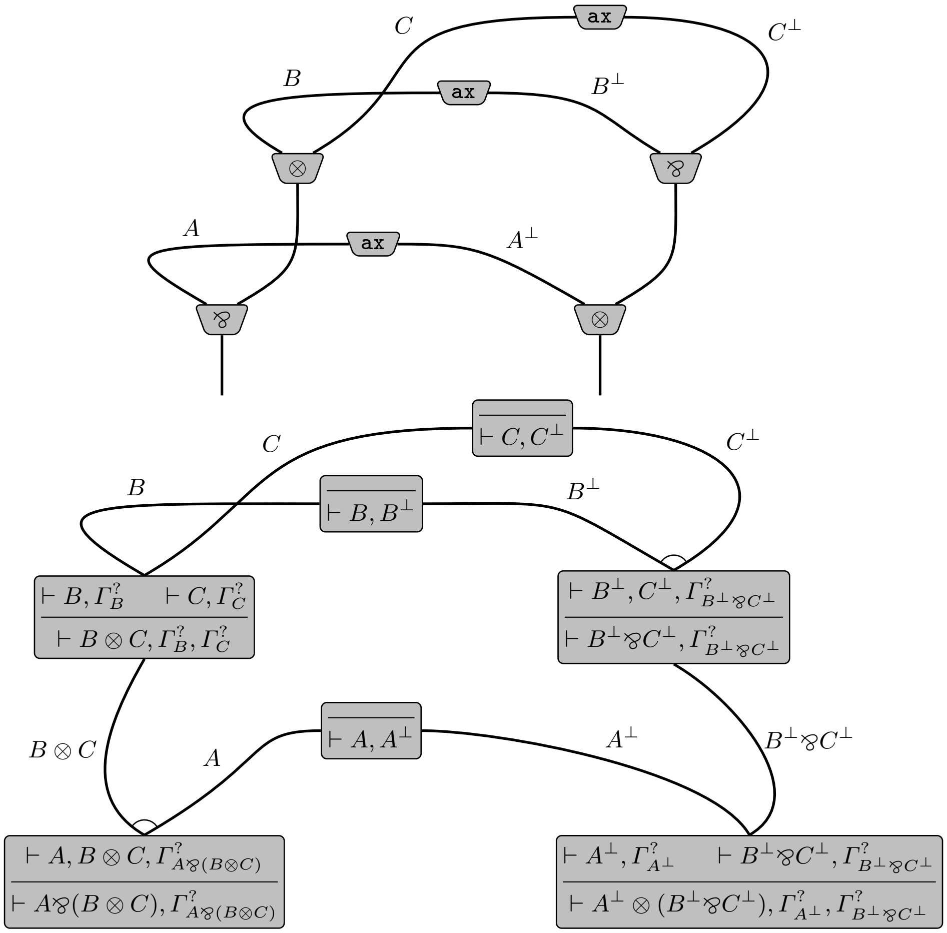


Labeled contractibility rules become:



When a proof structure is actually a proof net, the node at which its paired graph contracts is labeled by one of its sequentializations. For example, if we consider the proof structure and its paired graph given in figure 3, the contraction steps (figures 4 to 9) lead to a sequentialization of this proof net. This is formally expressed in the following proposition:

Proposition 1. *Let R be a proof structure. If $C(R)$ contracts (by rules R_1 and R_2) to a point, then by following the same contraction path, $C_l(R)$ contracts to a point whose label is a sequentialization of R.*



C
ax
$C^\perp$
B
ax
$B^\perp$
$\otimes$
$\parr$
A
ax
$A^\perp$
$\parr$
$\otimes$
C
$\vdash C, C^\perp$
$C^\perp$
B
$\vdash B, B^\perp$
$B^\perp$
$\vdash B, \Gamma^?_B \quad \vdash C, \Gamma^?_C$ / $\vdash B \otimes C, \Gamma^?_B, \Gamma^?_C$
$\vdash B^\perp, C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$ / $\vdash B^\perp \parr C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$B \otimes C$
A
$\vdash A, A^\perp$
$A^\perp$
$B^\perp \parr C^\perp$
$\vdash A, B \otimes C, \Gamma^?_{A \parr (B \otimes C)}$ / $\vdash A \parr (B \otimes C), \Gamma^?_{A \parr (B \otimes C)}$
$\vdash A^\perp, \Gamma^?_{A^\perp} \quad \vdash B^\perp \parr C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$ / $\vdash A^\perp \otimes (B^\perp \parr C^\perp), \Gamma^?_{A^\perp}, \Gamma^?_{B^\perp \parr C^\perp}$

Fig. 3: A PROOF NET AND ITS LABELED PAIRED GRAPH.

Notice that two different contraction paths may lead to different sequentializations of a proof net.

3.2 Big-step contractibility

We reformulate Contractibility in a big-step fashion to highlight the intrinsic notion of dependency present in this criterion.

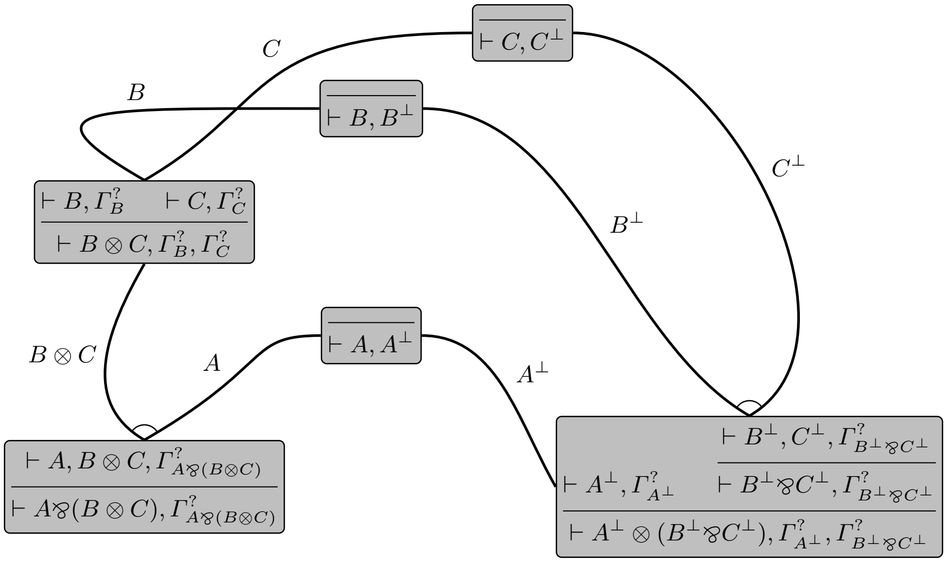

C
$\vdash C, C^\perp$
B
$\vdash B, B^\perp$
$C^\perp$
$\vdash B, \Gamma^?_B \quad \vdash C, \Gamma^?_C$
$\vdash B \otimes C, \Gamma^?_B, \Gamma^?_C$
$B^\perp$
$B \otimes C$
A
$\vdash A, A^\perp$
$A^\perp$
$\vdash A, B \otimes C, \Gamma^?_{A \parr (B \otimes C)}$
$\vdash A \parr (B \otimes C), \Gamma^?_{A \parr (B \otimes C)}$
$\vdash B^\perp, C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$\vdash A^\perp, \Gamma^?_{A^\perp} \quad \vdash B^\perp \parr C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$\vdash A^\perp \otimes (B^\perp \parr C^\perp), \Gamma^?_{A^\perp}, \Gamma^?_{B^\perp \parr C^\perp}$

Fig. 4: First step of labeled contraction

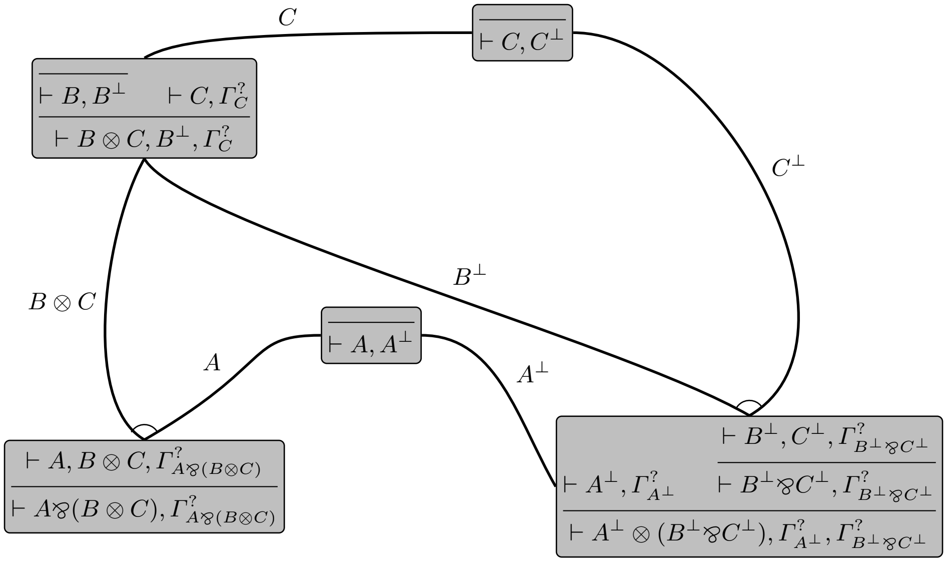

C
$\vdash C, C^\perp$
$\vdash B, B^\perp \quad \vdash C, \Gamma^?_C$
$\vdash B \otimes C, B^\perp, \Gamma^?_C$
$C^\perp$
$B^\perp$
$B \otimes C$
A
$\vdash A, A^\perp$
$A^\perp$
$\vdash A, B \otimes C, \Gamma^?_{A \parr (B \otimes C)}$
$\vdash A \parr (B \otimes C), \Gamma^?_{A \parr (B \otimes C)}$
$\vdash B^\perp, C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$\vdash A^\perp, \Gamma^?_{A^\perp} \quad \vdash B^\perp \parr C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$\vdash A^\perp \otimes (B^\perp \parr C^\perp), \Gamma^?_{A^\perp}, \Gamma^?_{B^\perp \parr C^\perp}$

Fig. 5: Second step of labeled contraction

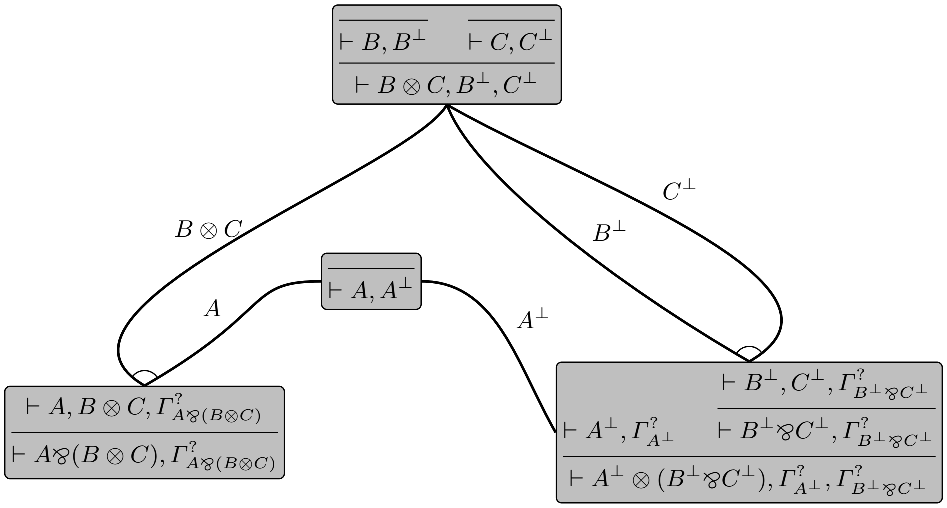

$\vdash B, B^\perp$
$\vdash C, C^\perp$
$\vdash B \otimes C, B^\perp, C^\perp$
$B \otimes C$
$C^\perp$
$B^\perp$
$\vdash A, A^\perp$
A
$A^\perp$
$\vdash A, B \otimes C, \Gamma^?_{A \parr (B \otimes C)}$
$\vdash A \parr (B \otimes C), \Gamma^?_{A \parr (B \otimes C)}$
$\vdash B^\perp, C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$\vdash A^\perp, \Gamma^?_{A^\perp}$
$\vdash B^\perp \parr C^\perp, \Gamma^?_{B^\perp \parr C^\perp}$
$\vdash A^\perp \otimes (B^\perp \parr C^\perp), \Gamma^?_{A^\perp}, \Gamma^?_{B^\perp \parr C^\perp}$

Fig. 6: Third step of labeled contraction

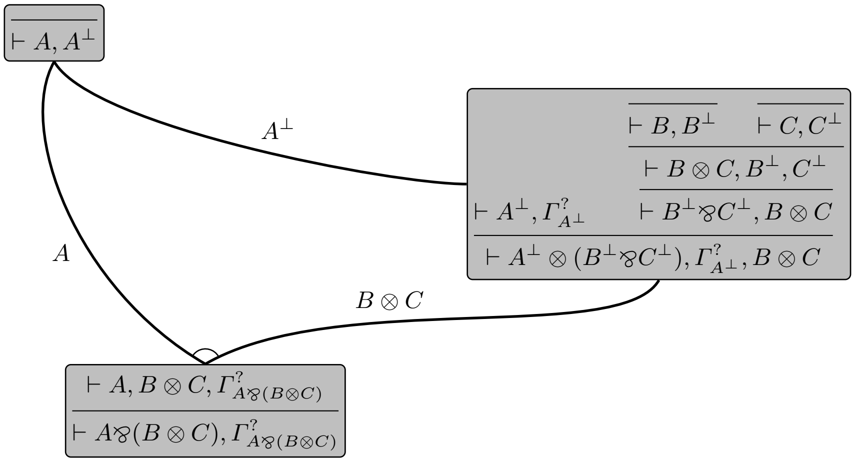

$\vdash A, A^\perp$
$A^\perp$
$\vdash B, B^\perp$
$\vdash C, C^\perp$
$\vdash B \otimes C, B^\perp, C^\perp$
$\vdash A^\perp, \Gamma^?_{A^\perp}$
$\vdash B^\perp \parr C^\perp, B \otimes C$
$\vdash A^\perp \otimes (B^\perp \parr C^\perp), \Gamma^?_{A^\perp}, B \otimes C$
A
$B \otimes C$
$\vdash A, B \otimes C, \Gamma^?_{A \parr (B \otimes C)}$
$\vdash A \parr (B \otimes C), \Gamma^?_{A \parr (B \otimes C)}$

Fig. 7: Fourth step of labeled contraction

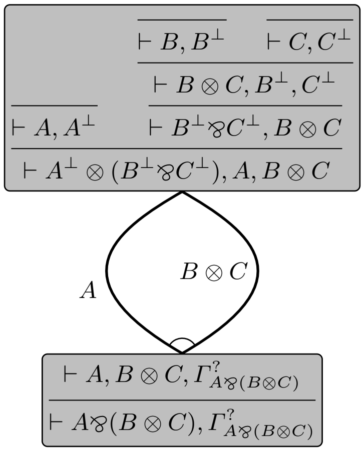

$\vdash B, B^{\perp}$
$\vdash C, C^{\perp}$
$\vdash B \otimes C, B^{\perp}, C^{\perp}$
$\vdash A, A^{\perp}$
$\vdash B^{\perp} \parr C^{\perp}, B \otimes C$
$\vdash A^{\perp} \otimes (B^{\perp} \parr C^{\perp}), A, B \otimes C$
A
$B \otimes C$
$\vdash A, B \otimes C, \Gamma^{?}_{A \parr (B \otimes C)}$
$\vdash A \parr (B \otimes C), \Gamma^{?}_{A \parr (B \otimes C)}$

Fig. 8: Fifth step of labeled contraction

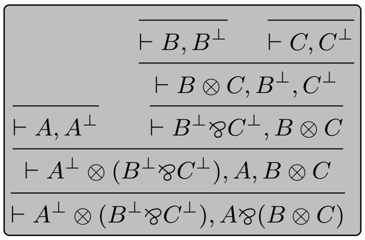
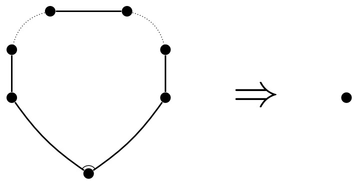

$\vdash B, B^{\perp}$
$\vdash C, C^{\perp}$
$\vdash B \otimes C, B^{\perp}, C^{\perp}$
$\vdash A, A^{\perp}$
$\vdash B^{\perp} \parr C^{\perp}, B \otimes C$
$\vdash A^{\perp} \otimes (B^{\perp} \parr C^{\perp}), A, B \otimes C$
$\vdash A^{\perp} \otimes (B^{\perp} \parr C^{\perp}), A \parr (B \otimes C)$

Fig. 9: Sixth and last step of labeled contraction

One defines a new graph-rewriting rule R as follows:

Definition 14 (Big-step Contraction R). *An elementary cycle can be contracted to a point if it contains exactly two paired edges that are paired together that are adjacent in the cycle.*



This new notion of contractibility is easily seen to correspond to usual contractibility and thus induces a correctness criterion expressed as:

Theorem 4. *A proof structure is a proof net if, and only if, contraction R can be applied until:*

- *no paired edges are left and*
- *it leads to a tree of unpaired edges.*

3.3 Towards dependency graphs

This version of contractibility criterion induces a natural dependency relation between the $\parr$ nodes of the proof structure: when the premises of a $\parr$ node are connected by a path that does not go through any premise of an other $\parr$ node (see figure 10), one can contract directly this path; these are the nodes connected at the source in the dependency graph of MN-criterion. When, on the contrary, the path from the premises of a $\parr$ node ($\parr_1$) goes through one of the premises of another $\parr$ node ($\parr_2$) (see figure 10), we say that $\parr_1$ depends on $\parr_2$ because $\parr_1$ can only be contracted if $\parr_2$ is contracted before. In this way, we can construct a dependency graph which looks like the dependency graph of MN criterion,

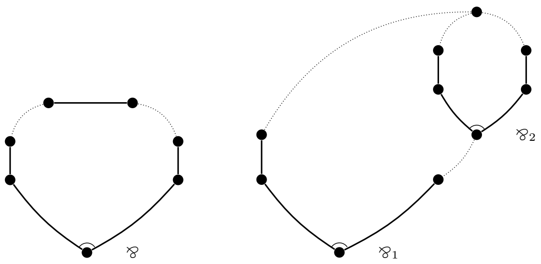


Fig. 10: Various dependency configurations.

but this one is built directly on the proof structure rather than on a correction graph. The first condition of big-step contractibilty criterion says simply that this graph is SDAG. We will see how to transform the second condition in order to get a full correction criterion. Before moving to the study of this new criterion, let us simply remark that one can actually define a dependency relation between $\parr$ nodes of a proof structure R and any set of nodes of R as follows:

Definition 15 (Dependency graph of a proof structure, relatively to a set of nodes). *Let R be a proof structure and N a set of nodes of R. We denote by P the set of $\parr$ nodes of R. The dependency graph of R relatively to N, $D_N(R)$, is the oriented graph (V, E) defined as follows:*

- *$V = N \cup P \cup \{s\}$ where s is an additional node.*
- *Let p be an element of P.*
  - *There is an edge $(s \to p)$ in E if the premises of p are connected by an elementary path in R which goes through no $\parr$ node.*
  - *Let q be an element of V. There is an edge $(q \to p)$ if the premises of p are connected by an elementary path containing q which does not go successively through the two premises of a $\parr$ node.*

Remark 3. The intuition underlying this extended notion of dependency graph is that in big-step contractibility, the contraction of the paired graph depends not only on depdencies between paired edges, but also on the fact that the $\otimes$ nodes on the cycles actually can be contracted to a point (with no loop), thus making a $\parr$ node depend on a $\otimes$ node.

Notation. *The previous definition has two natural instances: when we take N to be the set of the $\parr$ nodes of a proof structure N, $D_N(R)$ is a graph which expresses the dependency relation between $\parr$ nodes only. We note it by $D_{\parr}(R)$. When N is taken to be the set of all $\parr$ and $\otimes$ nodes of a proof structure, $D_N(R)$ is a graph which expresses the dependency relations between the $\parr$ nodes and the other $\parr$ and $\otimes$ nodes. We denote it by $D_{\parr,\otimes}(R)$.*
In the following we shall consider only $D_{\parr}(R)$ until section 4 where $D_{\parr,\otimes}(R)$ will be considered. When there is no ambiguity will shall omit the subscript.

3.4 DepGraph criterion

As said before, the first condition of big-step contractibility expresses that $D_{\parr}(R)$ is SDAG: the existence of a contractibility sequence ensures that there is some $\parr$ node having a cycle that does not contain any paired edges which is the condition to be connected to the source, while the acyclicity condition ensures that we will always find a $\parr$ node with a cycle that can be contracted.

To get a full correction criterion, we will make use of a graph-theoretic property called Euler-Poincaré lemma, as suggested by Girard in [27].

Definition 16. *Let G be an undirected graph and n, e be its numbers of nodes and edges. We set $\chi_G = n - e$ and call this quantity the* **characteristic** *of G.*

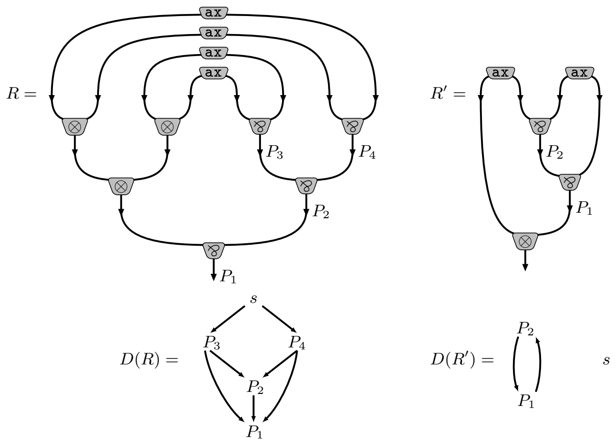


Fig. 11: Examples of dependency graphs.

Theorem 5 (Euler-Poincaré Lemma). *Let G be an undirected acyclic graph and c_G be its number of connected components. The following equality holds:*

$$\chi_G = c_G.$$

Proof. By induction on the number of edges in the graph.

If the graph contains no edge, there are as many connected components as there are nodes in the graph ($e = 0$ and $n = c$) and the equality is trivially satisfied.

Assume the equality holds for acyclic graphs with k edges and let G be an acyclic graph with $k + 1$ edges. By removing one edge e from G, obtaining the acyclic graph with k edges G_e, we remark that the number of connected components of G_e has increased by one while its number of edges has decreased by one: $\chi_G + 1 = \chi_{G'}$ and $c_G + 1 = c_{G'}$. By induction hypothesis $\chi_{G'} = c_{G'}$ and thus $\chi_G = c_G$. □

Proposition 2. *For every correction graph G of a proof net, one has $\chi_G = 1$.*

Proof. Immediate by (DR). □

Proposition 3. *Every correction graph G of a proof structure R satisfies:*

$$\chi_G = \#ax - \#\otimes.$$

Proof. The result follows immediately from a simple counting:

- $n = \#ax + \#\otimes + \#\parr + \#concl;$

- To count the edges, let us remark that every edge enters exactly one node of the correction graph, from which we have:

$$a = 0 \times \#ax + 2 \times \#\otimes + \#\parr + \#concl.$$

□

Putting the two previous propositions together, a sequentializable proof structure must have one more axiom link than it has tensor links: $\#ax - \#\otimes = 1$.

Remark 4. When a structure contains cuts, one has $\chi_G = \#ax - \#\otimes - \#cut$ for every correction graph G. The condition above becomes $\#ax - \#\otimes - \#cut = 1$.

We can finally state our new criterion, *DepGraph*:

Definition 17 (DepGraph criterion). *A proof structure R is $D_\parr$-correct (or satisfies DepGraph criterion) if*

- *$D_\parr(R)$ is SDAG;*
- *R is connected;*
- *$\#ax - \#\otimes = 1$*

Theorem 6. *A proof structure is a proof net if, and only if, it is $D_\parr$-correct.*

In the following, we will show the two sides of this equivalence.

Theorem 7. *Every proof net R is $D_\parr$-correct.*

We show that if π is a sequentialization of the considered net R, every dependence present in the dependency graph is also present in the order of introduction graph, more precisely:

Definition 18 (Order of introduction). *Let π be an MLL proof. For every $\parr$ or $\otimes$ rule $\mathbf{r}_F$ introducing formula F, we note π_F the sub-tree of π rooted in the premise of $\mathbf{r}_F$. We define a partial order on the formulas introduced by $\parr$ or $\otimes$ inferences in π, that will be noted $<_\pi$ as follows:*

$$F <_\pi G \text{ if } \mathbf{r}_F \in \pi_G$$

It formalizes the relation "to be introduced above".

The graph of this relation is noted $O^-(\pi)$ and one defines $O(\pi)$ as $O^-(\pi)$ augmented by adding a vertice s and, for all vertice e in $O^-(\pi)$, an edge $s \to e$.

Proposition 4. *Let π be an MLL proof and R the corresponding proof net. Then:*

$$D_\parr(R) \subseteq O(\pi)$$

Proof. Is a trivial corollary of proposition 5, since $D_\parr(R) \subseteq D_{\parr,\otimes}(R)$. □

Lemma 1. *Let π be an MLL proof. $O(\pi)$ is acyclic.*

Proof. Immediate (for antisymmetry: a sequent proof π induces a partial order on its inferences of which $O(\pi)$ is a sub-relation). □

Lemma 2. *Let G be a directed and acyclic graph. If every node of G has a parent but for one, written s, then G is $SDAG$ of source s.*

Proof. This is a basic result of graph theory. □

We now move to the proof of theorem 7:

Proof. Let R be a proof net.

- Since R is a proof net, there exists a proof π which is a sequentialization of R. By theorem 4, we know that $D(R) \subseteq O(\pi)$. Since $O(\pi)$ is acyclic, by lemma 1, we conclude that $D(R)$ is also acyclic. On the other hand, since R is a proof net, (DR) ensures that every correction graph is connected. Let G be a correction graph of R. The two premises of each $\parr$ node are connected by a path in G, which happens to be an s-path in R. It follows that in $D(R)$, every node but for s has a parent. Lemma 2 ensures that $D(R)$ is $SDAG$.
- A proof net is obviously connected.
- The third condition of DepGraph follows from propositions 2 and 3.

□

Theorem 8. *If a proof structure is $D_{\parr}$-correct, then it is a proof net.*

We will need a notion of dependency graphs defined on paired graphs.

Definition 19 **(s-path in a paired graph).** *A path in a paired graph is called an s-path if it is elementary and if it does not go successively through two edges which are paired together.*

Notation. *If p is such an s-path, we also write $p[e_1, \dots, e_n]$ or $p[E]$ to indicate the paired edges it contains ($\{e_1, \dots, e_n\}$ and E being the paired edges occurring in p).*

Definition 20 **(Dependency graph of a paired graph).** *Let G be a paired graph. The dependency graph, $D(G)$, of G is the oriented graph (V, E) defined as follows:*

- *The set of nodes V consists in the set $P(G)$ of paired edges of G together with an additional node s.*
- *Let $e = (e_1, e_2)$ be an element of $P(G)$.*
  - *There is an edge $(s \to e)$ in E if none of e_1, e_2 are loops and the premises of the common endpoint are connected by an s-path in G which goes through no paired edge.*
  - *Let $f = (f_1, f_2)$ be another element of $P(G)$. There is an edge $(f \to e)$ if the premise(s) of a common endpoint of e_1, e_2 are connected by an s-path containing f_1 or f_2.*

Remark 5. Notice that:

- In both cases of the previous definition, when e_1 and e_2 have both of their endpoints in common (let n_1, n_2 be those endpoints), there is an ambiguity on the definition of which premises we consider: n_1 has $\{n_2\}$ as set of premises and conversely n_2 has $\{n_1\}$ as set of premises. Anyway, in the case of an edge $(s \to e)$ there is an empty s-path connecting either premise to itself and in the other case we only ask that there is an s-path via f in either connecting either of the sets of premises.
- In the case of an edge $(s \to e)$, if there is an edge $(s \to e)$, that means that there is a s-path of the form $p[\emptyset]$ between the premises of common endpoints.
- In the case of an edge $(f \to e)$, if there is an edge $(f \to e)$, that means that there is a s-path of the form $p[\{y_i\} \cup F]$ between the premises of common endpoints.

Remark 6. Notice that, for any proof struture R, its dependency graph $D_{\parr}(R)$ is exactly the dependency graph of its paired graph $D(C(R))$.

We will prove that a $D_{\parr}$-correct proof structure satisfies contractibility criterion, from which the result follows. We need the following lemma:

Lemma 3. *Contractibility steps preserve:*

1. *connectedness of the contracted graph;*
2. *the number $n - e_{npe} - e_{pe}$ (with n the number of nodes of the graph, e_{npe} the number of non-paired edges and e_{pe} the number of paired edges).*

Proof. Both rules of contractibility preserves connectedness and it is easily checked that they also preserve $n - e_{npe} - e_{pe}$. □

Lemma 4. *Let G be a paired graph and G' be such that $G \to^\star G'$. Let $\{b_1, \ldots, b_m\}$ be the paired edges which have been contracted and let x and y be two nodes in G. If x and y are connected by an s-path $p[a_1, \ldots, a_n]$ in G, then:*

- *either x and y have been merged;*
- *or there exists an s-path $q[\{a_1, \ldots, a_n\} \setminus \{b_1, \ldots, b_m\}]$ connecting x and y in G'.*

Conversely, every s-path of G' is the residual of an s-path in G.

Proof. The result is proved by considering the different possible configurations for one step of contraction of edges in G and has no difficulty. Extension to $\to^\star$ is straightforward. □

Let us go back to the proof of the theorem:

Proof. Let R be a proof structure and $G = C(R)$ its paired graph. We shall build a sequence of contractions $c_1, \ldots, c_n$ such that, noting $G := G_0 \to^\star_{c_1} G_1 \cdots \to^\star_{c_n} G_n$ we have the following conditions:

- the dependency graph of G_{i+1}, $D(G_{i+1})$ is obtained from $D(G_i)$, by removing one node v which was directly connected to the source s and by linking the source to the nodes which were only accessible from v. It is easy to see that this transformation preserves SDAG-ness.
- $D(G_n)$ contains a single node s (and thus contains no paired node).

We proceed by a decreasing induction on the number of nodes in the graph and suppose that we have already built the i^{th} graph G_i. Let $v = (e_1, e_2)$ be a node directly connected to the source in $D(G_i)$, the dependency graph of G_i. There exists a path $p[\emptyset]$ connecting both premises of v. By contracting all the edges of $p[\emptyset]$ (by R_2), the two premises of v are merged and r_1 can be applied on the paired edges v. Let c_{i+1} be this sequence of contractions and G_{i+1} the graph obtained from G_i by applying c_{i+1}.

Let us analyze the consequence of c_{i+1} on the s-paths of G_i: since the only paired edges that have been contracted in c_{i+1} is v, the lemma 4 ensures that for any paired node x, is $p[E]$ is an s-path connecting two premises of x, then there is an s-path $q[E \setminus \{e_1, e_2\}]$ in G_{i+1}.

There are two possible cases:

- If x had v as sole parent in $D(G_i)$, it means that all s-paths connecting premises of x in G_i (we know that there is at least one such s-path) are of the form $p[e_1]$ or $p[e_2]$. Those paths become s-path $q[\emptyset]$ in G_{i+1}. Then x is connected to the source in $D(G_{i+1})$.
- If x had several parents $\{n_1, \ldots, n_k\}$ in $D(G_i)$, then its parents become $\{n_1, \ldots, n_k\} \setminus \{v\}$ in $D(G_{i+1})$.

$D(G_{i+1})$ contains the same nodes as $D(G_i)$ but for v and its edges are the same but for the nodes which had v as single parent and are now connected to the source.

Since the dependency graphs are SDAG at every step, there is always a node which is directly connected to the source. This ensures that the procedure can be applied as long as there are paired nodes. Finally, we obtain G_n which contains no paired nodes. By applying all possible R_2 reductions from G_n, we have a graph G'.

G' is connected (by lemma 3.1) and it cannot contain more than one node (since it is R_2-normal).

By lemma 3.2, $n - e_{npe} - e_{pe}$ has remained the same during the contraction steps and it is therefore equal to 1 (by $D_{\otimes}$-correctness of G) for G'. G' has no paired node and thus $e_{pe} = 0$ and it contains a single node, so $e_{npe} = 0$ as well and G' contains no edge at all.

We conclude that G is contractible. □

3.5 Comparing the two notions of dependency graphs

Example on figure 12 shows that Mogbil-Naurois dependency graphs are switching-dependent. We will show that, for proof nets, they are almost invariant: the transitive closure of the dependency graphs induced different switchings are all equal and are equal to the transitive closure of the dependency graph we introduced in the previous section.

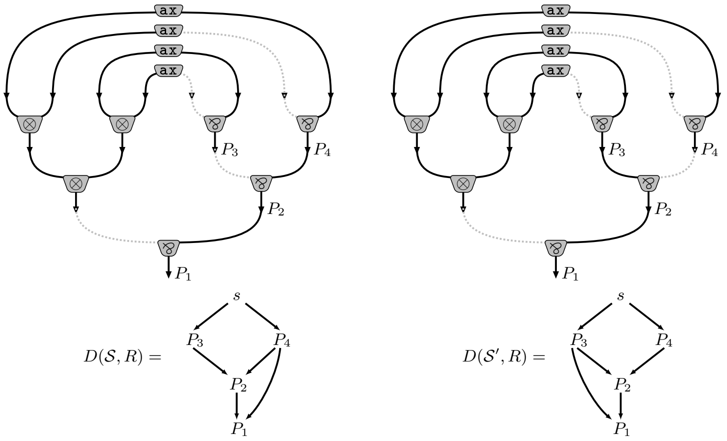


Fig. 12: Switchings $\mathcal{S}$ $\mathcal{S}'$ of net R, the associated dependency graphs.

Notations. *If $\mathcal{S}$ is a switching for a proof structure R and a a $\parr$-link in R, we note $\mathcal{S}_a$ the switching $\mathcal{S}$ in which we have toggled the switching for a.*

Given a graph D, D^* is its transitive closure.

Lemma 5. *Let z and a be two $\parr$ links of a proof net R and $\mathcal{S}$ be a switching.*

- *if $(z \to a) \in D(\mathcal{S}, R)$, then $(z \to a) \in D(\mathcal{S}_a, R)$*
- *if $(a \to z) \in D(\mathcal{S}, R)$, then $(a \to z) \in D(\mathcal{S}_a, R)$*

Proof. The first point follows from the fact that elementary paths between premises of a are not modified by the toggling of a.

As for the second point: were it not the case, it would result in an elementary path between premises Z_l and Z_r of z which does not go through a in $\mathcal{S}_a(R)$; this path would still be in $\mathcal{S}(R)$, but by hypothesis we already have in $\mathcal{S}(R)$ an elementary path between Z_l and Z_r which passes through a, those two distinct paths between Z_g and Z_d in $\mathcal{S}(R)$ would contradict acyclicity. □

Theorem 9. *Let R be a proof net and $\mathcal{S}, \mathcal{S}'$ be switchings of R. Then we have*

$$D(\mathcal{S}, R)^* = D(\mathcal{S}', R)^*.$$

Remark 7. The proof relies strongly on the fact that in a connected acyclic graph, there always exists a single elementary path between two nodes. The result would not hold if the structure were not correct.

Proof. We show that for all switching $\mathcal{S}$ and $\parr$-link a of the structure, we have $D(\mathcal{S},R) \subseteq D(\mathcal{S}_a,R)^*$. By symmetry and since $\mathcal{S}' = \mathcal{S}_{a_1\ldots a_n}$ for a certain sequence $a_1 \ldots a_n$ of $\parr$-links, it is enough.

Let R be a DR-correct proof structure, $\mathcal{S}$ of switching of R, x and a two $\parr$-links of R. Let also y be such that $(y \to x) \in D(\mathcal{S},R)$. We note X_l and X_r the two premises of x.

If y is s, source of $D(\mathcal{S},R)$, then $(a \to x) \notin D(\mathcal{S},R)$ by definition. The elementary path between X_l and X_r is not affected when toggling a and contains no $\parr$-link, so $(s \to x) \in D(\mathcal{S}_a,R)$.

Let us assume $y \neq s$, that is y is a $\parr$-link.

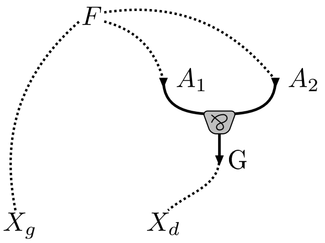
If $(a \to x) \notin D(\mathcal{S},R)$, the elementary path between X_l and X_r is not affected when toggling a and still contains y so that $(y \to x) \in D(\mathcal{S}_a,R)$.

If $(a \to x) \in D(\mathcal{S},R)$ and $(y \to a) \in D(\mathcal{S},R)$, lemma 5 implies that $(y \to a) \in D(\mathcal{S}_a,R)$ and $(a \to x) \in D(\mathcal{S}_a,R)$, so that $(y \to x) \in D(\mathcal{S}_a,R)^*$.

Finally, if $(a \to x) \in D(\mathcal{S},R)$ and $(y \to a) \notin D(\mathcal{S},R)$, let us note

- A_1 the premise of a which is contained in the elementary path between X_l and X_r in $\mathcal{S}(R)$;
- A_2 the premise of a which is contained in the elementary path between X_l and X_r in $\mathcal{S}_a(R)$.

Let us also note F and G the branching points of these paths (one of them, say G, is the conclusion of a), as shown on the following picture:



(where the dotted lines represent elementary paths which are present in both $\mathcal{S}(R)$ and $\mathcal{S}_a(R)$)

Since, by hypotheses $(y \to a) \notin D(\mathcal{S},R)$, y can be neither in the elementary path connecting F to A_1, nor in that connecting F to A_2 in $\mathcal{S}(R)$. Moreover $(y \to x) \in D(\mathcal{S},R)$, so that y must be in the elementary path between X_l and X_r in $\mathcal{S}(R)$: it must be either in the path between F and X_l or in that between G and X_r.

Those two paths are still present in $\mathcal{S}_a(R)$, and as a conclusion $(y \to x) \in D(\mathcal{S}_a,R)$. □

Finally, we have:

Theorem 10. *Let R be a proof net and $\mathcal{S}$ a switching for R. Then: $D_{\parr}(R)^\star = D(\mathcal{S},R)^\star$.*

Proof. Indeed, given a switching $\mathcal{S}_0$, theorem 9 gives:

$$D(\mathcal{S}_0, R) \subseteq D_{\parr}(R) = \bigcup_{\mathcal{S} \text{ switching of } R} D(\mathcal{S}, R) \subseteq \bigcup_{\mathcal{S} \text{ switching of } R} D(\mathcal{S}, R)^{\star} = D(\mathcal{S}_0, R)^{\star}$$

From this follows that $D(\mathcal{S}_0, R)^{\star} \subseteq D_{\parr}(R)^{\star} = D(\mathcal{S}, R)^{\star\star} = D(\mathcal{S}_0, R)^{\star}$ and the expected equality. □

4 On the order of introduction of connectives in sequentializations

In this section, we will investigate the logical meaning of dependency graphs introduced for DepGraph criterion. A crucial step in proving that a proof net satisfies DepGraph was to show that if π is a sequentialization of the considered net R, every dependence present in the dependency graph is also present in the order of introduction graph.

As a consequence, $D_{\parr}(R) \subseteq O(R) := \cap_{\pi,[\pi]=R} O(\pi)$ where $O(R)$ can be seen as the essence of the sequentalizations of R. It is natural to wonder whether this inclusion can be sharpened in a characterization of $O(R)$ relying on our notion of dependency. Actually, $D_{\parr}$ expresses only the relationship betweep $\parr$ nodes, and it is not enough to characterize $O(R)$. We will use instead the dependency graph $D_{\parr,\otimes}(R)$ to take in acount also the dependency relation between $\parr$ and $\otimes$ nodes.

Definition 21 (Subformula graph of a proof net). *Let R be a proof net. The subformula graph of R, $SF(R)$, is the directed graph (V, E) defined as follows:*

- *$V = P \cup T$ where P and T are respectively the set of $\parr$ nodes and $\otimes$ nodes.*
- *Let n and m be two elements of V. There is an edge $(m \rightarrow n)$ in E if the formula of the conclusion of m is a subformula of the formula of the conclusion of n.*

Theorem 11. *Let R be a proof net. Then $(D_{\parr,\otimes}(R) \cup SF(R))^{\star} = O(R)$.*

Proposition 5. *Let π be an MLL proof and R its desequentialization. Then:*

$$D_{\parr,\otimes}(R) \subseteq O(\pi).$$

Proof. By induction on the structure of π.

- The base case of the axiom rule is trivial since $D_{\parr,\otimes}(R) = O(\pi) = \{s\}$.
- Assume proof π of $\vdash \Gamma, A \parr B$ has been built from a proof ν of $\vdash \Gamma, A, B$. Note R_π and R_ν the proof nets corresponding respectively to π and ν. Since $A \parr B$ is below all formulas introduced by a $\parr$ rule in π and since the order on all other formulas is the same in π and in ν, the graph $O(\pi)$ is obtained from $O(\nu)$, by adding a node $A \parr B$ together with edges from the nodes of $O(\pi)$ to $A \parr B$.

On the other hand, $D_{⅋,\otimes}(R_\pi)$ is obtained from graph $D_{⅋,\otimes}(R_\nu)$ by adding a node $A⅋B$ together with some edges leaving nodes of $D_{⅋,\otimes}(R_\nu)$ and pointing to $A⅋B$. Indeed, one cannot create new edges between nodes of $D_{⅋,\otimes}(R_\nu)$ since it would mean that there exists an elementary path between the corresponding ⅋ nodes in π which was not present in ν. Such a path would certainly go successively through the left and right premises of node $A⅋B$, which would be absurd. Moreover, there exists no edge from $A⅋B$ towards nodes of ν since, here again, an elementary path justifying such an edge would consecutively visit both premises of node $A⅋B$ which is absurd.

By induction hypothesis, one has $D_{⅋,\otimes}(R_\nu) \subseteq O(\nu)$. By the preceding remarks, one concludes that $D_{⅋,\otimes}(R_\pi) \subseteq O(\pi)$.

- Assume proof π of $\vdash \Gamma, A \otimes B$ has been obtained from proof π_1 of $\vdash \Gamma, A$ and proof π_2 of $\vdash \Delta, B$. One shall note R_π, R_{π_1} and R_{π_2} the proof nets corresponding to π, π_1 and π_2.
  It is easy to see that $O(\pi)$ is obtained from the union of $O(\pi_1)$ and $O(\pi_2)$, by adding a node $A \otimes B$ together with edges from the nodes of $O(\pi_1)$ and $O(\pi_2)$ to $A \otimes B$.

  Moreover, $D_{⅋,\otimes}(R_\pi)$ is the union of $D_{⅋,\otimes}(R_{\pi_1})$ and $D_{⅋,\otimes}(R_{\pi_2})$ together with a node $A \otimes B$. Indeed, if there is an elementary path between the premisses of a ⅋ node, it will necessarily contain a cycle since it must visit twice node $A \otimes B$. Again, this is absurd.

  By induction hypothesis, one has $D_{⅋,\otimes}(R_{\pi_1}) \subseteq O(\pi_1)$ and $D_{⅋,\otimes}(R_{\pi_2}) \subseteq O(\pi_2)$. By the above considerations, one can conclude that $D_{⅋,\otimes}(R_\pi) \subseteq O(\pi)$.

□

Corollary 1. *Let R be a proof net. One has $(D_{⅋,\otimes}(R) \cup SF(R))^\star \subseteq O(R)$.*

Proof. By applying lemma 5, and by using the fact that for every proof net R and every MLL proof π such that $[\pi] = R$ one has $SF(R) \subseteq O(\pi)$ and $O(\pi)$ is closed by transitivity, we obtain that $(D_{⅋,\otimes}(R) \cup SF(R))^\star \subseteq \cap_{\pi,[\pi]=R} O(\pi)$. □

To prove the other inclusion, we will use the fact that two MLL proofs have the same proof net if and only if they are obtained from each other by the permutation rules introduced in section 2.1. More precisely, we will make use of the two following lemmas, proven in the long version of this paper:

Lemma 6. *Let π be an MLL proof and $\mathbf{r}_1$, $\mathbf{r}_2$ two consecutive rules in π introducing the formulas F_1 and F_2, such that $\mathbf{r}_1$ is above $\mathbf{r}_2$. If $\mathbf{r}_1$, $\mathbf{r}_2$ are not permutable, then:*

$$(F_1 \to F_2) \in O([\pi])$$

Definition 22. *Let π be an MLL proof and $\mathbf{r}_1$,$\mathbf{r}_2$ two inference rules in π introducing the formulas F_1 and F_2, such that $\mathbf{r}_{F_1}$ is above $\mathbf{r}_{F_2}$. We note by $]\mathbf{r}_2, \mathbf{r}_1[$ the sequence bottom-up of inference rules lying between $\mathbf{r}_1$ and $\mathbf{r}_2$ in the* *branch of π connecting $\mathbf{r}_1$ and $\mathbf{r}_2$. The distance between $\mathbf{r}_1$, $\mathbf{r}_2$ in π, noted by $d_\pi(\mathbf{r}_1, \mathbf{r}_2)$, is the length of $]\mathbf{r}_2, \mathbf{r}_1[$. The minimal distance between $\mathbf{r}_1$, $\mathbf{r}_2$, noted by $d_m(\mathbf{r}_1, \mathbf{r}_2)$, is defined by: $d_m(\mathbf{r}_1, \mathbf{r}_2) = inf_{\nu,[\nu]=[\pi]} d_\nu(\mathbf{r}_1, \mathbf{r}_2)$.*

Lemma 7. *Let π be an MLL proof and $\mathbf{r}_1$, $\mathbf{r}_2$ two inference rules in π introducing the formulas F_1 and F_2, such that $\mathbf{r}_1$ is above $\mathbf{r}_2$. We assume that $d_\pi(\mathbf{r}_1, \mathbf{r}_2) = d_m(\mathbf{r}_1, \mathbf{r}_2)$. If r is an inference rule introducing a formula F in π such that $\mathbf{r} \in]\mathbf{r}_2, \mathbf{r}_1[$, then $(F_1 \to F) \in O([\pi])$.*

Proof. By induction on $d_\pi(\mathbf{r}_1, \mathbf{r})$.

- If $d_\pi(\mathbf{r}_1, \mathbf{r}) = 0$, one has that $\mathbf{r}$ is just below $\mathbf{r}_1$. $\mathbf{r}_1$ and $\mathbf{r}$ are not permutable, if they were, this would contradict the minimality of π. By applying lemma 6, one concludes that $(F_1 \to F) \in O([\pi])$.
- Suppose that $d_\pi(\mathbf{r}_1, \mathbf{r}) > 0$. One can permute r with the derivations above it until an inference rule $\mathbf{r}'$ introducing a fomula F' which does not permute with it. One has $\mathbf{r}' \in [\mathbf{r}_1, \mathbf{r}[$ otherwise the minimality of π would fail. By applying lemma 6 to $\mathbf{r}$ and $\mathbf{r}'$ one has $(F' \to F) \in O([\pi])$. Moroever, one has $d_\pi(\mathbf{r}_1, \mathbf{r}') < d_\pi(\mathbf{r}_1, \mathbf{r})$, we can apply the induction hypothesis on $\mathbf{r}_1$ and $\mathbf{r}'$ to get that $(F_1 \to F') \in O([\pi])$. By transitivity of $O([\pi])$ we get that $(F_1 \to F) \in O([\pi])$.

□

We can finally prove the following (see [1] for a detailed proof):

Proposition 6. *Let R be a proof net. One has $O(R) \subseteq (D_{\parr,\otimes}(R) \cup SF(R))^\star$.*

Proof. We show by induction on $d_m(\mathbf{r}_1, \mathbf{r}_2)$ that for any two inference rules r_1 and r_2 introducing respectively the formulas F_1 and F_2, one has: $(F_1 \to F_2) \in O(R) \Rightarrow (F_1 \to F_2) \in (D_{\parr,\otimes}(R) \cup SF(R))^\star$.

- Suppose that $d_m(\mathbf{r}_1, \mathbf{r}_2) = 0$ and let π an MLL proof such that $d_m(\mathbf{r}_1, \mathbf{r}_2) = d_\pi(\mathbf{r}_1, \mathbf{r}_2)$. If $(F_1 \to F_2) \in O(R)$ then $\mathbf{r}_1$ is above $\mathbf{r}_2$ in π. If $\mathbf{r}_1$ and $\mathbf{r}_2$ were permutable in π, this permutation would lead to an MLL proof ν where $\mathbf{r}_2$ is above $\mathbf{r}_1$ which contradicts $(F_1 \to F_2) \in O(R)$. There are only two reasons why $\mathbf{r}_1$ and $\mathbf{r}_2$ are not permutable, we will show that $(F_1 \to F_2) \in (D_{\parr,\otimes}(R) \cup SF(R))^\star$ in both cases:
  - F_1 is a sub-formula of F_2: one has $(F_1 \to F_2) \in SF(R)$ and then $(F_1 \to F_2) \in (D_{\parr,\otimes}(R) \cup SF(R))^\star$.
  - F_1 is of the form $A \otimes B$, F_2 is of the form $C \parr D$ and π_{F_2} -the sub-tree of π rooted in $\mathbf{r}_2$- is of the form:

    $$\dfrac{\dfrac{\dfrac{\nu_1}{\vdash A, C, \Gamma} \qquad \dfrac{\nu_2}{\vdash B, D, \Delta}}{\vdash A \otimes B, C, D, \Gamma, \Delta}\ (\mathbf{r}_1)}{\vdash A \otimes B, C \parr D, \Gamma, \Delta}\ (\mathbf{r}_2)$$

    Let π_{F_1} the sub-tree of π rooted in $\mathbf{r}_1$ and R_1 be the proof net of π_{F_1}. The formulas C and D are conclusions of the proof net R_1. By DR-criterion on R_1 one has that C and D are connected by an elementary path wich does not go successively through the two premises of a $\parr$ node. Moroever, this elementary path goes necessarily throught $A \otimes B$ since R_1 is obtained by linking $[\nu_1]$ and $[\nu_2]$ via $A \otimes B$. Since C and D are the premisses of the $\parr$ node $C \parr D$ in π, we conclude that $(F_1 \rightarrow F_2) \in (D_{\parr,\otimes}(R) \cup SF(R))^{\star}$.

- Suppose that $d_m(\mathbf{r}_1, \mathbf{r}_2) > 0$ and let π an MLL proof such that $d_m(\mathbf{r}_1, \mathbf{r}_2) = d_\pi(\mathbf{r}_1, \mathbf{r}_2)$. If $(F_1 \rightarrow F_2) \in O(R)$ then $\mathbf{r}_1$ is above $\mathbf{r}_2$ in π and the sequence considered bottom-up of inferences between $\mathbf{r}_1$ and $\mathbf{r}_2$ in π, $]\mathbf{r}_2, \mathbf{r}_1[$, is of the form $]\mathbf{r}_2, \mathbf{r}_1[= \mathbf{r},]\mathbf{r}, \mathbf{r}_1[$ where $\mathbf{r}$ is an inference rule introducing the formula F. If $\mathbf{r}$ and $\mathbf{r}_2$ were permutable, this would contradict the minimality of π. By applying lemma 6 we have $(\mathbf{r} \rightarrow \mathbf{r}_2) \in O(R)$. Moreover, one has obviously $d_m(\mathbf{r}, \mathbf{r}_2) = 0$. We can now apply the induction hypothesis to $\mathbf{r}$ and $\mathbf{r}_2$ to conclude that $(F \rightarrow F_2) \in (D_{\parr,\otimes}(R) \cup SF(R))^{\star}$. Furthermore, by applying lemma 7, one has $(F_1 \rightarrow F) \in O([\pi])$. As one has obviously $d_m(\mathbf{r}_1, \mathbf{r}) < d_m(\mathbf{r}_1, \mathbf{r}_2)$, we finally conclude, by induction hypothesis, that $(F \rightarrow F_2) \in (D_{\parr,\otimes}(R) \cup SF(R))^{\star}$.

□

5 Conclusion

Comparing correctness criteria. We have seen that Contractibility, Mogbil and Naurois' criterion and DepGraph are three faces of the same notion, dependency. More precisely, those three criteria can been understood as different concrete implementations of a *proto-criterion* related with dependency relation, along the different points of view developed in the introduction: we showed that (i) Contractibility gives actually a sequentialization of a proof-net from which arises dependency, (ii) MN is a criterion with efficiency purposes (working on the generalized dependency graph, it is not clear how to stay in NL since it requires to remember which premise of a $\parr$ node has been visited, thus justifying the seemingly odd choice of a switching) while (iii) DepGraph emphasizes structural properties of logic by clearly separating conditions on $\parr$ inferences from other inferences and by unveiling the meaning of its dependency graph which correponds (when considered together with the subformula relation) to the order of introduction of inferences common to all sequentializations of a given proof-net.

This last point actually evidences an intresting fact. While they are completely parallel proof objects, proof-nets contain enough logical dependency to allow for the retrieval of inherently sequential information. Indeed, by computing the dependency relation we can extract the true logical causality of sequential proofs.

Future works. The present work suggests two main directions for future works:

- The separation between positive and negative inferences which is the cornerstone of DepGraph criterion suggests connections with focusing. While proof-nets and focalized proofs are the results of diverging choices of proof-theoretical design (parallelism versus hypersequentiality), this suggests that they actually may be different aspects of the same phenomenon as already advocated in the study of multi-focusing [34].

- Another direction concerns the development and the validation of our comparative study of proof-nets. Indeed, the prototypical classification we suggested is mainly built on empirical considerations and we plan to investigate it more systematically in the future, in particular by considering connections with other criteria which seems to be related with the notion of dependency such as Di Giamberardino and Faggian's work on jumps [35], or Murawski and Ong's work on dominator's trees [31].

References

1. Bagnol, M., Doumane, A., Saurin, A.: long version of this submission. available at http://www.pps.univ-paris-diderot.fr/~saurin/Publi/DepGraphLong.pdf
2. Howard, W.A.: The formulae-as-type notion of construction, 1969. In Seldin, J.P., Hindley, R., eds.: To H. B. Curry: Essays in Combinatory Logic, Lambda Calculus, and Formalism. Academic Press, New York (1980) 479–490
3. Girard, J.Y.: Linear logic. Theoretical Computer Science **50**(1) (1987) 1 – 101
4. Girard, J.Y.: Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur. Thèse de doctorat d'etat, Université Paris VII (1972)
5. Reynolds, J.C.: Towards a theory of type structure. In: Symposium on Programming. Volume 19 of Lecture Notes in Computer Science., Springer (1974) 408–423
6. Girard, J.Y.: The system F of variable types, fifteen years later. Theoretical Computer Science **45** (1986) 159–192
7. Berry, G.: Stable models of typed lambda-calculi. (1978) 72–89
8. Girard, J.Y.: Proof-nets: the parallel syntax for proof-theory. In Ursini, A., Agliano, P., eds.: Logic and Algebra. Volume 180 of Lecture Notes In Pure and Applied Mathematics., New York, Marcel Dekker (1996) 97–124
9. Danos, V., Regnier, L.: The structure of multiplicatives. **28** (1989) 181–203
10. Girard, J.Y.: Towards a geometry of interaction. In: Categories in Computer Science and Logic, Providence, AMS (1989) 69–108 Proceedings of Symposia in Pure Mathematics n°92.
11. Hyland, M., Ong, L.: On full abstraction for PCF. Information and Computation **163**(2) (December 2000) 285–408
12. Girard, J.Y.: Locus solum. **11** (2001) 301–506
13. Laurent, O.: Étude de la polarisation en logique. PhD thesis (March 2002)
14. Girard, J.Y.: Light linear logic. Information and Computation **143**(2) (June 1998) 175–204
15. Ehrhard, T., Regnier, L.: Differential interaction nets. Theor. Comput. Sci. **364**(2) (2006) 166–195
16. Gonthier, G., Abadi, M., Lévy, J.J.: The geometry of optimal lambda reduction. In: Proceedings of the 19^{th} Annual ACM Symposium on Principles of Programming Languages. (1992) 15–26
17. Lafont, Y.: Interaction nets. In: POPL90, San Francisco, California, ACM Press (1990) 95–108

18. Girard, J.Y.: A new constructive logic: classical logic. **1**(3) (1991) 255–296
19. Danos, V., Joinet, J.B., Schellinx, H.: A new deconstructive logic: Linear logic. **62**(3) (1997) 755–807
20. Lafont, Y.: Soft linear logic and polynomial time. TCS **318**(1–2) (June 2004) 163–180
21. Andreoli, J.M.: Proposition pour une synthèse des paradigmes de la programmation logique et de la programmation par objets. Thèse de doctorat, Université Paris VI (June 1990)
22. Miller, D.: Overview of linear logic programming. In Ehrhard, T., Girard, J.Y., Ruet, P., Scott, P., eds.: Linear Logic in Computer Science. Volume 316 of London Mathematical Society Lecture Note. Cambridge University Press (2004) 119–150
23. Zeilberger, N.: Focusing and higher-order abstract syntax. In: POPL, ACM (2008) 359–369
24. Curien, P.L., Munch-Maccagnoni, G.: The duality of computation under focus. In: IFIP TCS. Volume 323., Springer (2010) 165–181
25. Ehrhard, T., Tasson, C., Pagani, M.: Probabilistic coherence spaces are fully abstract for probabilistic pcf. In: Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. POPL '14, New York, NY, USA, ACM (2014) 309–320
26. Curien, P.L.: Introduction to linear logic and ludics, part ii (2006)
27. Girard, J.Y.: Le Point Aveugle: Cours de logique. Tome 1, Vers la perfection; Tome 2, Vers l'imperfection. Hermann (2006)
28. Danos, V.: Une application de la logique linéaire à l'étude des processus de normalisation (principalement du λ-calcul). Thèse de doctorat, Université Denis Diderot, Paris 7 (1990)
29. Guerrini, S.: Correctness of multiplicative proof nets is linear. In: LICS, IEEE Computer Society (1999) 454–463
30. Guerrini, S.: A linear algorithm for MLL proof net correctness and sequentialization. Theor. Comput. Sci. **412**(20) (2011) 1958–1978
31. Murawski, A.S., Ong, C.H.L.: Fast verification of mll proof nets via imll. ACM Trans. Comput. Logic **7**(3) (July 2006) 473–498
32. de Naurois, P.J., Mogbil, V.: Correctness of linear logic proof structures is NL-complete. Theor. Comput. Sci. **412**(20) (2011) 1941–1957
33. Nagayama, M., Okada, M.: A new correctness criterion for the proof nets of non-commutative multiplicative linear logics. J. Symb. Log. **66**(4) (2001) 1524–1542
34. Chaudhuri, K., Miller, D., Saurin, A.: Canonical sequent proofs via multi-focusing. In: Fifth IFIP International Conference On Theoretical Computer Science - TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, September 7-10, 2008, Milano, Italy. Volume 273 of IFIP., Springer (2008) 383–396
35. Giamberardino, P.D., Faggian, C.: Jump from parallel to sequential proofs: Multiplicatives. In: Computer Science Logic, 20th International Workshop, CSL 2006, 15th Annual Conference of the EACSL, Szeged, Hungary, September 25-29, 2006, Proceedings. (2006) 319–333