Correctness of Multiplicative (and Exponential) Proof Structures is *NL*-Complete

Paulin Jacobé de Naurois* and Virgile Mogbil*

LIPN – UMR7030, CNRS – Université Paris 13,
99 av. J-B Clément, F–93430 Villetaneuse, France
denaurois@lipn.univ-paris13.fr
virgile.mogbil@lipn.univ-paris13.fr

Abstract. We provide a new correctness criterion for unit-free MLL proof structures and MELL proof structures with units. We prove that deciding the correctness of a MLL and of a MELL proof structure is *NL*-complete. We also prove that deciding the correctness of an intuitionistic multiplicative essential net is *NL*-complete.

Introduction

The *proof nets* [Gir87,DR89] of Linear logic (LL) are a parallel syntax for logical proofs without all the bureaucracy of sequent calculus. They are a non-sequential graph-theoretic representation of proofs, where the order in which some rules are used in a sequent calculus derivation, when irrelevant, is neglected. The unit-free multiplicative proof nets are inductively defined from sequent calculus rules of unit-free Multiplicative Linear Logic (MLL). A *proof structure* is freely built on the same syntax as proof nets, without any reference to a sequent calculus derivation.
In LL we are mainly interested in the following decision problems: Deciding the provability of a given formula, which gives the expressiveness of the logic; deciding if two given proofs reduce to the same normal form, i.e. the cut-elimination problem which corresponds to program equivalence using the Curry-Howard isomorphism; and deciding the correctness of a given proof structure, i.e. whether it comes from a sequent calculus derivation. For this last decision problem, one uses a *correctness criterion* to distinguish proof nets among proof structures. We recall the following main results [Kan92,Mai] and we complete (in bold) the correctness cases:

fragment		decision problem		
	units	provability	cut-elimination	correctness
MLL	no	*NP*-complete	*P*-complete	**NL-complete**
MELL	yes	open	(at most non-elementary)	

* Work supported by project NO-CoST (ANR)

Correctness is equivalent to provability for unit only MLL because proof nets are formulae syntactic trees. However it is not so obvious for the propositional case as one can observe following the long story of correctness criteria:

- Long-trip [Gir87] is based on travels and was the first one.
- Acyclic-Connected [DR89] is a condition is based on switchings i.e. the choice of one premise for each $\parr$ connective. The condition is that all the associated graphs are trees. A naive implementation of Acyclic-Connected uses exponential time.
- Contractibility [Dan90] is done in quadratic time by repeating two graph rewriting rules until one obtains a simple node.
- Graph Parsing [Laf95] is a strategy for Contractibility which is implemented in linear time as a sort of unification [Gue99].
- Dominator Tree [MO00,MO06] is a linear time correctness criterion for essential nets, to which proof structures correctness reduces in linear time.
- Ribbon [Mel04] is a topological condition requiring homeomorphism to the disk.

For other fragments of Linear Logic, some of these criteria apply or are extended as for MELL[1] [Dan90,GM01].
A feature of these criteria is that they successively lower the complexity of sequential, deterministic algorithms deciding correctness for MLL. Switching from proof structures to paired graphs, that is undirected graphs with a distinguished set of edges, we give a new correctness criterion for MLL and more generally for MELL. This new correctness criterion gives us a lower bound for the correctness decision problem for both MLL and MELL. This lower bound yields an exact characterization of the complexity of this problem, and induces naturally efficient parallel and randomized algorithms for it.
The paper is organized as follows: we recall preliminary definitions and results in linear logic and complexity theory in Section 1. Section 2 is devoted to the exposition of a new correctness criterion for unit-free MLL and MELL with units. We prove its *NL*-completeness in Section 3, and the *NL*-completeness of the criterion for IMLL in Section 4.

1 Background

1.1 MLL and Proof Structures

Roman capitals A, B stand for MLL formulae, which are given by the following grammar, where $\otimes$ and $\parr$ are duals for the negation $^\perp$, accordingly to De Morgan laws:

$$F ::= A \mid A^\perp \mid F \otimes F \mid F \parr F$$

[1] As usual M, A and E denote respectively for Multiplicative, Additive and Exponential fragments of LL

Greek capitals Γ, Δ stand for sequents, which are multiset of formulae, so that exchange is implicit. The MLL sequent calculus is given by the following rules:

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

Definition 1. *A* MLL proof structure *is a finite directed acyclic graph (DAG)*[2] *whose nodes, called* links*, are defined together with an arity and a coarity, i.e. a given number of incident edges called the* premises *of the node and a given number of emergent edges called the* conclusions *of the node. Moreover the proof structure edges are labelled by formulae and every edge is conclusion of exactly one link and premise of at most one link. The links of are the following:*

nodes		*ax*		*cut*		$\otimes$		$\parr$	
arity	*edge labels*	*0*	$\emptyset$	*2*	$A, A^{\perp}$	*2*	A, B	*2*	A, B
coarity	*edge labels*	*2*	$A, A^{\perp}$	*0*	$\emptyset$	*1*	$A \otimes B$	*1*	$A \parr B$

We allow edges with a source but no target (i.e pending or dandling edges), they are called the conclusions *of the proof-structure.*
A MLL proof net *is a* MLL *proof structure inductively defined as follows:*

- *an ax-link is a proof net with conclusions* $A, A^{\perp}$*,*
- *if* P *is a proof net with conclusions* Γ, A, B *then* P *extended with a* $\parr$*-link of premises* A *and* B *is a proof net with conclusions* $\Gamma, A \parr B$*.*
- *if* P_1 *and* P_2 *are disjoint proof nets with respective conclusions* Γ, A *and* Δ, B *then* $P_1 \cup P_2$ *extended with a* $\otimes$*-link of premises* A *and* B *is a proof net with conclusions* $\Gamma, A \otimes B, \Delta$*.*

It follows from the definition that MLL proof structures and proof nets have a non-empty set of conclusions, which corresponds to a MLL sequent. The inductive definition of MLL proof nets corresponds to a graph theoretic abstraction of the derivation rules of MLL; any proof net is sequentializable, i.e. corresponds to a MLL derivation: given a proof net P of conclusion Γ, there exists a sequent calculus proof of $\vdash \Gamma$ which infers P.

Definition 2. *A* paired graph *is an undirected graph* $G = (V, E)$ *with a set of* pairs $C(G) \subseteq E \times E$ *which are pairwise disjoint couples of edges with the same target, called a* pair-node*, and two (possibly distinct) sources called the* premise-nodes*.*
A switching S *of* G *is the choice of an edge for every pair of* $C(G)$*. With each switching* S *is associated a subgraph* $S(G)$ *of* G*: for every pair of* $C(G)$*, erase the edges which are not selected by* S*. When* S *selects the (abusively speaking) left edge of each pair,* $S(G)$ *is denoted as* $G[\forall \mapsto \setminus \cdot]$*. Also,* $G[\forall \mapsto \cdot.]$ *stands for* $G \setminus \{e, e' |\ (e, e') \in C(G)\}$*.*

[2] For convenience the edges are oriented up-down, so we do not mention the orientation

Let $R = (V, E)$ be a MLL proof structure. To R, we naturally associate the paired graph $G_R = (V, E')$ where E' is the set of non-pending edges of E and $C(G_R)$ contains the premises of each $\parr$-link of R (Figure 1). For a pair of edges $(v, x), (w, x)$, we adopt the representation of Figure 1, where the two edges of the pair are joined by an arc.

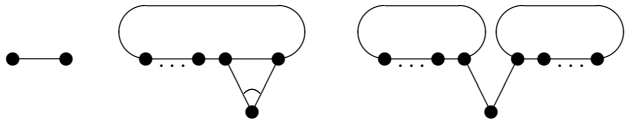

Fig. 1. Paired graph constructors associated to MLL proof nets: ax-link, $\parr$-link and $\otimes$-link.

We define the following graph rewriting rules $\rightarrow_c$ of Figure 2 on paired graphs where all the nodes are distinct and rule $\rightarrow_{R_2}$ applies only for a non-pair edge:

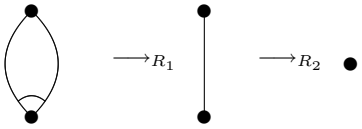

Fig. 2. Contraction rules $\rightarrow_c$

We denote by $G \rightarrow_c \bullet$ the fact that G contracts to a single vertex with no edge.

Definition 3. *A* MLL *proof structure R is* DR-correct *if for all switching S of G_R, the graph $S(G_R)$ is acyclic and connected.*
A MLL *proof structure R is* contractile *if $G_R \rightarrow_c^* \bullet$.*

Theorem 1. *[DR89,Dan90] A* MLL *proof structure R is a* MLL *proof net iff R is DR-correct iff R is contractile*[3]*.*

We define the following decision problem MLL-CORR:
GIVEN: A MLL proof structure R
PROBLEM: Is R a MLL proof net?

1.2 MELL and Proof Structures

The definition of MELL formulae follows that of MLL formulae in Section 1.1, with ! and ? duals for the negation $^\perp$, as well as the neutral elements 1 and $\perp$:

$$\text{MELL:} \qquad F ::= A \mid A^\perp \mid F \otimes F \mid F \parr F \mid !F \mid ?F \mid 1 \mid \perp$$

The MELL sequent calculus contains the rules of the MLL sequent calculus, as well as the following rules:

$$\frac{\vdash \Gamma}{\vdash \Gamma, \perp}\ \perp \qquad \frac{}{\vdash 1}\ 1 \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?A}\ ?W \qquad \frac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A}\ ?C \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?A}\ ?D \qquad \frac{\vdash ?\Gamma, A}{\vdash ?\Gamma, !A}\ !P$$

[3] The criteria in [DR89,Dan90] are expressed for switchings and contraction rules for proof structures only. The equivalence with Definition 3 is left to the reader.

Definition 4. MELL structures *are defined similarly to* MLL *proof structures (Definition 1), with the additional links, where the* $?W$*-link subsumes both* $?W$ *and* $\bot$*rules:*

nodes		1		*?W*		*?C*		*?D*		*!P*	
arity	*edge labels*	*0*	$\emptyset$	*0*	$\emptyset$	*2*	$?A, ?A$	*1*	A	*1*	A
coarity	*edge labels*	*1*	1	*1*	$\bot$ *or* $?A$	*1*	$?A$	*1*	$?A$	*1*	$!A$

Definition 5. *An* exponential box *is a* MELL *structure whose conclusions are all ?-formulae but one, its* principal door*, which is conclusion of a* $!P$*-link. Similarly, a* weakening box *is a* MELL *structure with a distinguished conclusion, its* principal door*, which is conclusion of a* $?W$*-link. A* box *is either an exponential or a weakening box.*

Definition 6. *A* MELL proof structure (R, B) *is given by a* MELL *structure* R *and a box mapping* B*, which associates to any link* l *of* R *a box* b_l *or* R*. Moreover, boxes may nest but may not partially overlap, and a unique exponential (respectively weakening) box is associated to each* $!P$*-(resp.* $?W$*-)link. By convention, when a link belongs to several boxes, the mapping returns the innermost box to which it belongs, otherwise it returns* R*.*

It follows from the definition that, for any $!P$ (respectively $?W$)-link, the box mapping associates the exponential (resp. weakening) box to which it naturally corresponds. The whole proof structure R is treated as a particular box, and is associated to all links that do not belong to any exponential or weakening box. Let (R, B) be a MELL proof structure, with boxes $b_1, \ldots, b_n$. Let $b_0 = R$. We define as follows the family $\mathcal{G}_{(R,B)} = \{G^i_{(R,B)}\}_{i=0\ldots n}$ of paired-graphs:

- $G^i_{(R,B)}$ contains a node l for every link l of $R \setminus \{?W\text{-links}\}$ with $B(l) = b_i$, and an edge (l, l') for all links l, l' of $R \setminus \{?W\text{-links}\}$ with $B(l) = B(l') = b_i$. $C(G^i_{(R,B)})$ contains the premises of each $\parr$-link and $?C$-link l of R with $B(l) = b_i$.
- Assume b_j is an outermost box included in b_i. A node $\overline{b_j} \in G^i_{(R,B)}$ is associated to b_j, and an edge $(\overline{b_j}, l) \in G^i_{(R,B)}$ for all link l conclusion of a link in b_j and such that $B(l) = b_i$.

Essentially, $G^i_{(R,B)}$ is the paired graph corresponding to the box b_i, where all inner boxes are considered contracted to a single node. Moreover, for the sake of connectivity, the $?W$-link (if there is any) corresponding to b_i is removed.

Definition 7. *A* MELL *proof structure* (R, B) *with boxes* $b_1, \ldots, b_n$ *is* contractile *if* $\forall i \in \{0, \ldots, n\}$*,* $G^i_{(R,B)} \rightarrow_c^* \bullet$*.*

As for MLL, one may inductively define particular MELL proof structures, called MELL proof nets, which exactly correspond to derivations in MELL sequent calculus. Theorem 2 allows to distinguish MELL proof nets among proof structures:

Theorem 2. *[GM01] A* MELL *proof structure* (R, B) *is a* MELL *proof net iff* (R, B) *is contractile*[4]*.*

[4] The criterion in [GM01] uses contraction rules for MELL proof structures only. As for Theorem 1, the proof of the equivalence with Definition 7 is left to the reader.

We define the following decision problem MELL-CORR:
GIVEN: A MELL proof structure (R, B)
PROBLEM: Is (R, B) a MELL proof net?

1.3 Intuitionistic Multiplicative Linear Logic and Essential Nets

The intuitionistic fragment of MLL (IMLL) is the $(\otimes, \multimap)$-fragment of Linear logic, where linear implication is no more defined by $A \multimap B = A^{\perp} \parr B$ but is a connective. The sequent calculus corresponds to the MLL sequent calculus but with two-sided sequents (using linear negation) and at most one (distinguished) formula on the right.
For this sub-logic, Lamarche has proposed a version of proof structures called *essential nets*. They are built on the links given in Figure 3 where, to each $\parr^+$-labelled node, one associates a negatively-labelled node (left premise) called the sink of p. They also have a distinguished link called the *root*.

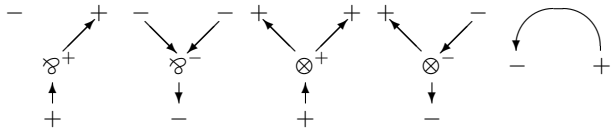

Fig. 3. Essential net links: $\parr^+$, $\parr^-$, $\otimes^+$, $\otimes^-$ and Axiom

Definition 8. *An essential net of a linearly balanced*[5] IMLL *sequent is L-correct if it is acyclic, every node is reachable from the root, and every $\parr^+$-node p satisfies the L-condition: every path from the root that reaches the sink of p passes through p.*

Lamarche has shown that the essential net of an IMLL sequent denotes a IMLL sequent derivation if and only if it is L-correct. For additional information on essential net correction, including translation to proof structures, we refer the reader to [MO06].
We define the following decision problem IMLL-CORR:
GIVEN: A multiplicative essential net N of a linearly balanced IMLL sequent
PROBLEM: Is N correct?

1.4 Complexity Classes and Related Problems

We will mention several major complexity classes below P, some of which having natural complete problems that we will use in this paper. Let us briefly recall some basic definitions and results.

Definition 9. *Complexity classes:*

[5] I.e. every atom that occurs in the sequent does so exactly twice, once positively and once negatively

- *AC^0 (respectively AC^1) is the class of problems solvable by a uniform family of circuits of constant (resp. logarithmic) depth and polynomial size, with NOT gates and AND, OR gates of unbounded fan-in.*
- *L is the class of problems solvable by a deterministic Turing machine which only uses a logarithmic working space.*
- *NL (respectively coNL) is the class of problems solvable by a non-deterministic Turing machine which only uses a logarithmic working space, such that:*
  1. *If the answer is "yes," at least one (resp. all) computation path accepts.*
  2. *If the answer is "no," all (resp. at least one) computation paths reject.*

Theorem 3. *[Imm88,Sze87] NL = coNL.*

The following inclusion results are also well known:

$$AC^0 \subset L \subset NL \subset AC^1 \subset P, \tag{1}$$

where it remains unknown whether any of these inclusions is strict.
It is important to note that Theorems 6 and 7 give *NL*-completeness results under constant-depth (actually AC^0) reductions. From (1) above, it should be clear to the reader that the reductions lie indeed in a class small enough for being relevant. For a good exposition of constant-depth reducibility, see [CSV84].
In the sequel, we will often use the notion of a *path* in a directed -or undirected- graph. A path is a sequence of vertices such that there is an edge between any two consecutive vertices in the path. A path will be called *elementary* when any node occurs at most once in the path.
Let us now list some graph-theoretic problems that will be used in this paper.

Is Tree (IT): Given an undirected graph $G = (V, E)$, is it a tree?
IT is *L*-complete under constant-depth reductions [JLM97].

Source-Target Connectivity (STCONN): Given a directed graph $G = (V, E)$ and two vertices s and t, is there a path from s to t in G ?
STCONN is *NL*-complete under constant-depth reductions [JLL76].

Undirected Source-Target Connectivity (USTCONN):
Given an undirected graph $G = (V, E)$ and two vertices s and t, do s and t belong to the same connected component of G ?
USTCONN is *L*-complete under constant-depth reductions [Rei05].

Universal Source DAG (SDAG):
Given a directed graph $G = (V, E)$, is it acyclic and does there exist a source node s such that there is a path from s to each vertex ?

Proposition 1. SDAG $\in$ *NL*.

Proof. Given $G = (V, E)$ a directed graph, its acyclicity can be expressed as follows:

$$\forall (x, y) \in V^2 : \neg \text{STCONN}(G, x, y) \vee \neg \text{STCONN}(G, y, x).$$

Since *NL* = *coNL* (Theorem 3) and STCONN $\in$ *NL*, acyclicity is clearly in *NL*. Checking whether each vertex can be reached from a vertex s can also be done with STCONN subroutines, therefore SDAG is in *NL*. □

Proposition 2. SDAG *is coNL-hard under constant-depth reductions.*

Proof. Let $\mathcal{L}$ be any language in *coNL*. $\mathcal{L}$ is then decided by a non-deterministic Turing machine M in space less than $k\log(n)$ on inputs of size n, for some $k \geq 0$. Let $\mathcal{C}_n$ be the set of configurations of M of size less or equal to $k\log(n)$, and define $T = |\mathcal{C}_n|$. Clearly, $T \leq n^k$ is an upper bound for the computation time of M on inputs of size n. Without loss of generality, we assume that every configuration of M has at least one outgoing transition, possibly towards itself, and that the result of the computation is given by the state reached by M after exactly T computation steps. A configuration is thus either accepting or rejecting.
Let us consider the following directed graph $G_n = (V_n, E_n)$, where:

$V_n = \bigcup_{c\in\mathcal{C}_n, t\in[0,T]}\{(c,t)\} \cup \{c_A\} \cup \{c_R\} \cup \{s\}$.
For $(c,t), (c',t+1) \in V_n$, $((c',t+1) \to (c,t)) \in E_n$ if and only if $c \to c'$ is a transition of M.
For $c \in \mathcal{C}_n$, $(c_A \to (c,T)) \in E_n$ iff c is an accepting configuration of M.
For $c \in \mathcal{C}_n$, $(c_R \to (c,T)) \in E_n$ iff c is a rejecting configuration of M.
$(s \to c_A) \in E_n$, $(s \to c_R) \in E_n$.

A path $(c_1,t_1) \to \cdot\cdot \to (c_k,t_k)$ in G_n follows by construction a sequence $t_1,\ldots,t_k$ that is strictly decreasing. Since there is no edge $(c,t) \to c_A$, $(c,t) \to c_R$ nor $(c,t) \to s$, it is then clear that G_n is acyclic.
Moreover, since every configuration of M has at least one outgoing transition, every vertex $(c,t), t < T$ in G_n has at least one parent node $(c',t+1)$. By induction on t, it follows that every vertex in G_n is reachable from s. Therefore G_n satisfies SDAG.
Let x be an input of size n to M. An initial configuration $c_x \in \mathcal{C}_n$ of M is naturally associated to this input x. Consider now the directed graph $H_n^x = G_n \cup \{(c_x,0) \to c_R\}$.
Then, H_n^x satisfies SDAG if and only if $x \in \mathcal{L}$. Indeed, by Definition 9, $x \in \mathcal{L}$ if and only if there exists no computation path $c_x \to \cdot\cdot \to c_r$ of length T in M, where c_r is a rejecting configuration. By construction of G_n, such a path corresponds to a path $(c_r,T) \to \cdot\cdot \to (c_x,0)$ in G_n. Then $x \in \mathcal{L}$ if and only if there exists no path $c_R \to \cdot\cdot \to (c_x,0)$ in G_n, if and only if H_n^x is acyclic. Since G_n satisfies SDAG, it follows that H_n^x satisfies SDAG if and only if $x \in \mathcal{L}$.
Moreover, it is well known that the configuration graph of a Turing machine can be computed with constant-depth circuits. Computing H_n^x from the configuration graph of M requires only purely local rewriting rules, that can all be performed in parallel. Therefore, H_n^x can also be computed with constant-depth circuits. □

Propositions 1 and 2, and Theorem 3 yield the following result:

Theorem 4. SDAG *is NL-complete under constant-depth reductions.*

2 New Correctness Criteria for MLL and MELL

For a given proof net, the following notion of dependency graph provides a partial order among its ⅋-nodes corresponding to some valid contraction sequences accordingly to rule R_1.

Definition 10. *Let G be a paired graph. The* dependency graph $D(G)$ *of G is the directed graph (V_G, E_G) defined as follows:*

- $V_G = \{v \mid v \text{ is a pair-node in } G\} \cup \{s\}$.
- *Let x be a pair-node in G, with premise-nodes x_l and x_r. The edge $(s \to x)$ is in E_G if and only if:*
  1. *There exists an elementary path $p_x = x_l, \ldots, x_r$ in $G[\forall \mapsto \backslash\cdot]$,*
  2. *$x \notin p_x$, and for all pair-node y in G, $y \notin p_x$.*
- *Let x be a pair-node in G, with premise-nodes x_l and x_r, and let $y \neq x$ be another pair-node in G. The edge $(y \to x)$ is in E_G if and only if:*
  1. *There exists an elementary path $p_x = x_l, \ldots, x_r$ in $G[\forall \mapsto \backslash\cdot]$,*
  2. *$x \notin p_x$, and for every elementary path $p_x = x_l, \ldots, x_r$ in $G[\forall \mapsto \backslash\cdot]$ with $x \notin p_x$, $y \in p_x$.*

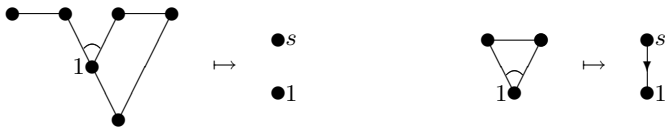

Fig. 4. Dependency graph examples: source is needed!

Remark 1. Other definitions of dependency graph are possible:
- One may choose to consider not only elementary paths, but *any* path, which yields a stronger contraction order. Surprisingly enough, Theorem 6 holds also for this relaxed version of dependency.
- One may also consider the dependency graph only for a paired graph G where $G[\forall \mapsto \backslash\cdot]$ is a tree. In that case, if there exists an elementary path $p_x = x_l, \ldots, x_r$ which does not contain x, this elementary path is unique. The results of Section 3 hold also for this version of dependecy, yet Lemma 4 does not rely on [Rei05], but rather on the fact that checking reachability in a forest is L-complete [CM87].

Lemma 1. *Let G and H be paired graphs, with $G \to_c H$. Then, $G[\forall \mapsto \backslash\cdot] \to_c^* H[\forall \mapsto \backslash\cdot]$, and $G[\forall \mapsto \backslash\cdot]$ is a tree if and only if $H[\forall \mapsto \backslash\cdot]$ is a tree.*

Proof. If $G \to_{R_1} H$ denote by v the redex pair-node in G, with premise w. The reduced pattern in H is the non-pair edge (v, w), therefore $G[\forall \mapsto \backslash\cdot] = H[\forall \mapsto \backslash\cdot]$. If $G \to_{R_2} H$, it is clear that $G[\forall \mapsto \backslash\cdot] \to_{R_2} H[\forall \mapsto \backslash\cdot]$ with the same redex. It is also clear that rule $\to_{R_2}$ preserves connectivity and acyclicity. □

Lemma 2. *If $G \to_c^* \bullet$ then $D(G)$ satisfies* SDAG.

Proof. Since $\bullet[\forall \mapsto \backslash\cdot]$ is a tree, by Lemma 1 so is $G[\forall \mapsto \backslash\cdot]$. Therefore, for any pair-node x with premise-nodes x_l and x_r in G, there exists a unique elementary path $p_x = x_l - \cdots - x_r$ in $G[\forall \mapsto \backslash\cdot]$. It follows by construction of $D(G)$ that x has at least one parent node in $D(G)$. Moreover, a path $x \to \cdot\cdot \to y$ in $D(G)$ induces by construction an elementary path $x_l - \cdots - y$ in $G[\forall \mapsto \backslash\cdot]$. Therefore a cycle $x \to \cdot\cdot \to y, y \to \cdot\cdot \to x$ in $D(G)$ induces a cycle $x_l - \cdots - y, y_l - \cdots - x$ in $G[\forall \mapsto \backslash\cdot]$. Since $G[\forall \mapsto \backslash\cdot]$ is a tree, $D(G)$ is acyclic. Since every vertex of $D(G)$ but s has at least one parent node and $D(G)$ is acyclic, $D(G)$ satisfies SDAG. □

Lemma 3. *Let G be a paired graph such that $G[\forall \mapsto \backslash\cdot]$ is a tree. If the dependency graph $D(G)$ of G satisfies* SDAG *then $G \to_c^* \bullet$.*

Proof. let $d(v)$, the depth of a pair-node $v \in G$, be the length of the longest path from the source s of $D(G)$ to the vertex $v \in D(G)$. Assume that $D(G)$ satisfies SDAG, and let $X^d = \{x \text{ pair-node in } G | d(x) = d\}$ and $Y^d = \cup_{d' \leqslant d} X^{d'}$. By induction on the depth we prove that there exists a sequence of contractions $\mathcal{C}_d$ such that $G \to^{\mathcal{C}_d} G^d$ satisfies:

$$\text{Each pair-node } y \in G \text{ s.t. } d(y) \leqslant d \text{ is contracted in } G^d. \tag{2}$$

The proof by induction is as follows:

- For $d = 1$, let $x \in X^1$, with premise-nodes x_l and x_r. By definition of X^1, there exists an elementary path $p_x = x_l - \cdots - x_r$ in $G[\forall \mapsto \backslash\cdot]$ such that $x \notin p_x$ and for any pair-node y in $G[\forall \mapsto \backslash\cdot]$, $y \notin p_x$. The same holds for the path $p_x = x_l - \cdots - x_r$ in G, with respect to any pair-node $y \in G$.
  Let $\mathcal{E}_x^1 = \{e \text{ edge of } p_x \mid x \in X^1\}$. The set of contractions $\mathcal{R}_x^1 = \{e \to_c \bullet \mid e \in \mathcal{E}_x^1\}$ contracts the edges of p_x, and let $\mathcal{R}^1 = \cup_{x \in X^1} \mathcal{R}_x^1$. Clearly, $x_l = x_r \neq x$ in the contracted paired graph obtained from G by $\mathcal{R}_x^1$. Since $x \notin p_y$ for any $y \in X^1$, the same holds for the paired graph obtained from G by $\mathcal{R}^1$.
  Let $\mathcal{C}_1$ be the sequence $\mathcal{R}^1$, followed by the set of contraction rules of the pair-nodes $x \in X^1$. Define G^1 such that $G \to^{\mathcal{C}_1} G^1$. It is clear that G^1 satisfies (2).
- Assume by induction that there exists a sequence of contractions $\mathcal{C}_d$ such that $G \to^{\mathcal{C}_d} G^d$ satisfies (2).
  Let $x \in X^{d+1}$, with premise-nodes x_l and x_r.
  Since $G \to^{\mathcal{C}_d} G^d$ and $G[\forall \mapsto \backslash\cdot]$ is a tree, Lemma 1 applies:

$$G[\forall \mapsto \backslash\cdot] \to^{\mathcal{C}'_d} G^d[\forall \mapsto \backslash\cdot], \text{ and } G^d[\forall \mapsto \backslash\cdot] \text{ is a tree.} \tag{3}$$

  By definition of X^{d+1}, there exists an elementary path $p_x = x_l - \cdots - x_r$ in $G[\forall \mapsto \backslash\cdot]$ such that $x \notin p_x$ and, for every pair-node $y \in G$ of depth $d(y) > d$, $y \notin p_x$.
  Define p_x^d such that $p_x \to^{\mathcal{C}'_d} p_x^d$. By (3), p_x^d is an elementary path in $G^d[\forall \mapsto \backslash\cdot]$ such that $x \notin p_x^d$ and, for every pair-node $y \in G^d[\forall \mapsto \backslash\cdot]$ of depth $d(y) > d$, $y \notin p_x^d$. The same holds for p_x^d in G^d, with respect to any pair-node $y \in G^d$, since, by induction, for any pair-node $y \in G^d$, $d(y) > d$.
  Let $\mathcal{E}_x^{d+1} = \{e \text{ edge of } p_x \mid x \in X^{d+1}\}$. The set of contractions $\mathcal{R}_x^{d+1} = \{e \to_c \bullet \mid e \in \mathcal{E}_x^{d+1}\}$ contracts the edges of p_x^d, and let $\mathcal{R}^{d+1} = \cup_{x \in X^{d+1}} \mathcal{R}_x^{d+1}$. Clearly, $x_l = x_r \neq x$ in the contracted paired graph obtained from G by $\mathcal{R}_x^{d+1}$. Since $x \notin p_y$ for any $y \in X^{d+1}$, the same holds for the contracted paired graph obtained from G by $\mathcal{R}^{d+1}$.
  Let $\mathcal{C}_{d+1}$ be the sequence $\mathcal{C}_d$, followed by $\mathcal{R}_{d+1}$, and followed by the set of contraction rules of the pair-nodes $x \in X^{d+1}$. Define G^{d+1} such that $G \to^{\mathcal{C}_{d+1}} G^{d+1}$. G^{d+1} satisfies (2).

Since $D(G)$ satisfies SDAG, the maximal depth $m = max\{d(x)|x \in D(G)\}$ is well-defined and every pair-node x of G belongs to X^m. Therefore, $G \rightarrow^{\mathcal{C}_m} G^m$ and G^m satisfies (2). Since $G[\forall \mapsto \downharpoonleft \cdot]$ is a tree, by Lemma 1 so is $G^m[\forall \mapsto \downharpoonleft \cdot] = G^m$. It follows that $G \rightarrow_c^* \bullet$. □

Lemmas 2 and 3 and Theorems 1 and 2 imply the Theorem:

Theorem 5 (Correctness Criteria).
A MLL *proof structure R is a* MLL *proof net if and only if:*

1. *$D(G_R)$ satisfies* SDAG*, and*
2. *$G_R[\forall \mapsto \downharpoonleft \cdot]$ is a tree.*

A MELL *proof structure (R, B) with boxes $b_1, \ldots, b_n$ is a* MELL *proof net if and only if:*

1. *$\forall i \in \{0, \ldots, n\}$, $D(G^i_{(R,B)})$ satisfies* SDAG*, and*
2. *$\forall i \in \{0, \ldots, n\}$, $G^i_{(R,B)}[\forall \mapsto \downharpoonleft \cdot]$ is a tree.*

3 *NL*-Completeness of the Criteria for MLL and MELL

Let DepGRAPH be the function: $G \mapsto D(G)$, which associates its dependency graph to a paired graph G.

Lemma 4. DepGRAPH $\in$ *FL*.

Proof. The following functions can easily be computed in *FL*:

- $G, x \in G \mapsto (G[\forall \mapsto \downharpoonleft \cdot]) \setminus \{x\}$
- $G, x \in G \mapsto (G[\forall \mapsto \cdot\!\cdot]) \setminus \{x\}$
- $G, x \in G, y \in G \mapsto (G[\forall \mapsto \downharpoonleft \cdot]) \setminus \{x, y\}$

Consider now the following algorithm for DepGRAPH:

INPUT (G)
FOR ALL x pair-node in G, with premise-nodes x_l and x_r DO
  IF USTCONN$((G[\forall \mapsto \cdot\!\cdot]) \setminus \{x\}, x_l, x_r)$ THEN OUTPUT $(s \rightarrow x) \in D(G)$
FOR ALL (x pair-node in G, with premise-nodes x_l and x_r, y pair-node in G) DO
  IF $\neg$USTCONN$((G[\forall \mapsto \downharpoonleft \cdot]) \setminus \{x, y\}, x_l, x_r)$
  AND USTCONN$((G[\forall \mapsto \downharpoonleft \cdot]) \setminus \{x\}, x_l, x_r)$ THEN
    OUTPUT $(y \rightarrow \{x\}) \in D(G)$.

- USTCONN$((G[\forall \mapsto \cdot\!\cdot]) \setminus \{x\}, x_l, x_r)$ tests whether there exists an elementary path $p_x = x_l - \cdots - x_r$ such that $x \notin p_x$ and, for all pair-node y in G, $y \notin p_x$.
- $\neg$ USTCONN$((G[\forall \mapsto \cdot\!\downharpoonleft]) \setminus \{x, y\}, x_l, x_r)$ tests whether any elementary path $p_x = x_l - \cdots - x_r$ such that $x \notin p_x$ contains y.
- USTCONN$((G[\forall \mapsto \cdot\!\downharpoonleft]) \setminus \{x\}, x_l, x_r)$ tests whether there exists a path $p_x = x_l - \cdots - x_r$ in G' such that $x \notin p_x$. From the previous point, if such a path p_x exists, $y \in p_x$.

It follows that this algorithm computes DepGRAPH. Since USTCONN $\in L$, this algorithm belongs to FL^L (the class of functions computable in logarithmic space, with oracles in L). Since $FL^L = FL$, DepGRAPH $\in$ *FL*. □

Proposition 3. *$MELL - \text{CORR} \in NL$.*

Proof. Let (R, B) be a MELL-proof structure with boxes $b_1, \ldots, b_n$. Each function $(R, B), i \in \{0, \ldots, n\} \mapsto G^i_{(R,B)}$ can be easily be computed in *FL*. Checking that $G^i_{(R,B)}[\forall \mapsto \nwarrow]$ is a tree is doable in L since $\text{IT} \in L$. Checking that $D(G^i_{(R,B)})$ satisfies SDAG can be done in *NL*, by composing the function DEPGRAPH in *FL* (Lemma 4) with an *NL* algorithm for SDAG (Theorem 4). Since the number of paired graphs $G^i_{(R,B)}$ is linearly bounded, it suffices to sequentially perform these tasks for $i = 0, \ldots, n$, with a counter i of logarithmic size. □

Note that the previous best algorithms [Laf95,Gue99] are not likely to be implemented in logarithmic space, since they require on-line modification of the structure they manipulate. The purpose of our criterion of Theorem 5 is precisely that it allows a space-efficient implementation.

Proposition 4. *MLL-CORR is NL-hard under constant-depth reductions.*

Proof. We actually reduce SDAG to MLL-CORR. Let G be a directed graph, and consider the proof structure S_G defined as follows (see Figure 5), and let G_{S_G} be its associated paired graph:

1. To any vertex v of G, we associate a $\otimes$-link $\overline{v}$ with parent links $\overline{v}_{in}$ and $\overline{v}_{out}$.
2. If there are $i > 0$ in-going edges to v, $\overline{v}_{in}$ is a $\parr$-link of arity i, with parent links $\overline{v}^1_{in}, \ldots, \overline{v}^i_{in}$. If v has no in-going edge, $\overline{v}_{in}$ is one conclusion of an axiom-link Ax^v_{in}, the other conclusion of Ax^v_{in} being a conclusion of S^x_n.
3. If there are $j > 0$ outgoing edges from v, $\overline{v}_{out}$ is a $\otimes$-link of arity j, with parent links $\overline{v}^1_{out}, \ldots, \overline{v}^j_{out}$. If v has no outgoing edge, $\overline{v}_{out}$ is one conclusion of an axiom-link Ax^v_{out}, the other conclusion of Ax^v_{out} being a conclusion of S^x_n.
4. To an edge $v \to w$ of G, we associate an axiom-link $Ax^{(v \to w)}$ with conclusions $Ax^{(v \to w)}_{in}$ and $Ax^{(v \to w)}_{out}$. Moreover, if $v \to w$ is the k^{th} outgoing edge from v, $Ax^{(v \to w)}_{in}$ is $\overline{v}^k_{out}$. If $v \to w$ is the l^{th} in-going edge to w, $Ax^{(v \to w)}_{out}$ is $\overline{w}^l_{in}$.

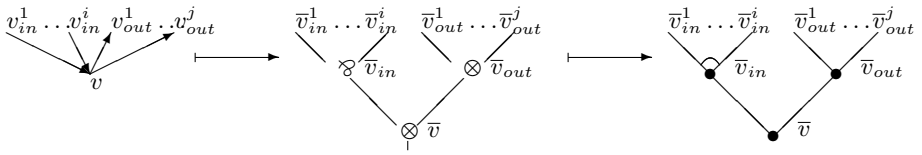

Fig. 5. Construction of S_G and G_{S_G}

It is quite clear that this reduction can be computed by constant-depth circuits. We now claim that S_G is correct if and only if G satisfies SDAG.
Assume G contains a cycle. There exists then an elementary path $p = x_1 \to \cdot\cdot \to x_l$, with $x_l \to x_1 \in G$. Then, for any edge $x_t \to x_{t+1} \in p$, there exists a switching of the pair-node $\overline{x_{t+1}}_{in}$ in G_{S_G}, which connects $\overline{x_t}$ and $\overline{x_{t+1}}$. Similarly

for the edge $x_l \to x_1 \in G$. Since p is elementary, these pair-nodes are all different; therefore there exists cyclic switching of G_{S_G} and S_G is not correct.
It is clear that if G is acyclic, it has at least one node of arity 0. Moreover, if G is acyclic and has only one node of arity 0, a proof by induction shows that G satisfies SDAG.
Assume therefore that G is acyclic and has at least two nodes, r and s, of arity 0. Let S' be any switching of G_{S_G}, and assume that there exists an elementary path p from $\overline{r}$ to $\overline{s}$ in S'. Let $p' = \overline{r}, \overline{x_1}, \ldots, \overline{x_k}, \overline{s}$ be the sequence of non pair-nodes of p corresponding to vertices of G. p' follows by construction edges of G, accordingly to their orientation or not. Since r and s have arity 0, there exist three nodes $\overline{x_t}, \overline{x_{t+1}}, \overline{x_{t+2}}$ in p' such that $(x_t \to x_{t+1})$ and $(x_{t+2} \to x_{t+1})$ are edges of G. By construction of G_{S_G}, $\overline{x_t}$ and $\overline{x_{t+2}}$ are then premise-nodes of the same pair-node $\overline{x_{t+1}}_{in}$ in G_{S_G}, which contradicts that p is a path in S'. Therefore, S' is not connected, and S_G is not correct.
Assume now that G satisfies SDAG and let $d(v)$, the depth of a vertex v of G, be the length of the longest path from the source s of G to v. Denote by G^d the subgraph of G consisting only in the vertices of depth less than d, and by $G_{S_G^d}$ the corresponding paired graph. It is easy to see that the rules of Figure 1 can be turned in a n-ary version, and that $G_{S_G^{d+1}}$ can be obtained from $G_{S_G^d}$ by these n-ary rules. By induction on d, it follows that S_G is correct. □

Since MLL is a subsystem of MELL, Propositions 3 and 4 immediately yield the following result:

Theorem 6. *MLL-*CORR *and MELL-*CORR *are NL-complete under constant-depth reductions.*

4 *NL*-Completeness of the Criterion for IMLL

Proposition 5. $IMLL - \text{CORR} \in NL$.

Proof. For a given essential net N, denote by $r(N)$ its root. For a given $\invamp^+$-link x in N, denote by $s(x)$ its sink. Consider now the following algorithm for IMLL-CORR:

INPUT (N)
IF $\neg$SDAG$(N, r(N))$ THEN REJECT
FOR ALL x $\invamp^+$-link in N DO
    IF STCONN$(N \setminus \{x\}, r(N), s(x))$ THEN REJECT
ELSE ACCEPT.

This algorithm checks that N satisfies SDAG, and that the L-condition applies. Since SDAG $\in NL$ and STCONN $\in NL$, and $NL = coNL$ (Theorem 3), this algorithm belongs clearly to NL. □

Proposition 6. *IMLL-*CORR *is NL-hard under constant-depth reductions.*

Proof. We actually reduce SDAG to IMLL-CORR. Let G be a directed graph, and consider the essential net N_G defined as follows (see Figure 6):

1. To any vertex v of G of arity $i > 0$, we associate a $\otimes^-$-node $\overline{v}$, with parent node (right premise) $\overline{v}_{in}$ of polarity $-$ and child node (left premise) $\overline{v}_{out}$ of polarity $+$. To v of arity 0 we associate a node $\overline{v} = \overline{v}_{out}$ of polarity $+$.
2. If there are $i > 0$ in-going edges to v , $\overline{v}_{in}$ is a $\parr^-$-node of arity i, with parent nodes $\overline{v}^1_{in}, \ldots, \overline{v}^i_{in}$ of polarity $-$.
3. If there are $j > 0$ outgoing edges from v, $\overline{v}_{out}$ is a $\otimes^+$-node of arity j, with child nodes $\overline{v}^1_{out}, \ldots, \overline{v}^j_{out}$ of polarity $+$.
4. If there is no outgoing edge from v, $\overline{v}_{out}$ is a $\parr^+$-node with child node (right premise) $\overline{v}^1_{out}$ of polarity $+$, and sink node (left premise) $\overline{v}^{sink}_{out}$, of polarity $-$. There is moreover an axiom-edge $\overline{v}^1_{out} \to \overline{v}^{sink}_{out}$.
5. Let $v \to w$ be an edge of G. Assume $v \to w$ is the k^{th} outgoing edge from v, and the l^{th} in-going edge to w. To $v \to w$, we associate an axiom-edge from $\overline{v}^k_{out}$ of polarity $+$ to $\overline{w}^l_{in}$ of polarity $-$.

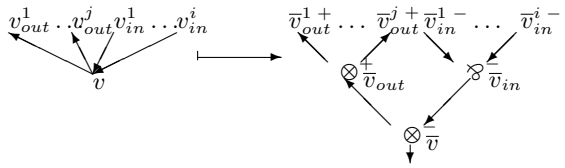

Fig. 6. Construction of N_G

It is clear that the reduction is constant-depth. Since the only $\parr^+$-links of N_G correspond to leaves of G, N_G satisfies the L-condition by construction. Therefore, it is L-correct if and only if G satisfies SDAG. □

Propositions 5 and 6 immediately yield the following result:

Theorem 7. *IMLL-*CORR *is NL-complete under constant-depth reductions.*

Note that [MO06] provides a linear-time reduction from MLL-CORR to IMLL-CORR, which yields a linear-time algorithm for MLL-CORR. This reduction actually occurs to be linear-space, and cannot be used for directly proving Proposition 6.

Conclusion and Acknowledgments

Deciding the correctness of unit-free MLL proof structures, MELL proof structures, and unit-free IMLL essential nets where problems known to be decidable in deterministic, sequential linear time. We have shown their *NL*-completeness, thus establishing that it would be most unlikely to find better sequential deterministic algorithms. As a byproduct, we obtain efficient parallel algorithms for both problems, namely AC^1 algorithms. Moreover, since $NL = RL = ZPL$, we also naturally obtain Monte-Carlo and Las-Vegas logarithmic space random algorithms, by simply using random walks for our graph reachability procedures. It remains to be checked whether our approach can be extended to MALL.

We are grateful to Harry Mairson for raising the question of the exact complexity of the correctness problems, and to the members of the No-Cost project for

useful discussions and comments. We also thank the anonymous referees for their comments.

References

[CM87] Stephen A. Cook and Pierre McKenzie. Problems complete for deterministic logarithmic space. *J. Algorithms*, 8(3):385–394, 1987.

[CSV84] Ashok K. Chandra, Larry J. Stockmeyer, and Uzi Vishkin. Constant depth reducibility. *SIAM J. Comput.*, 13(2):423–439, 1984.

[Dan90] Vincent Danos. *Une application de la logique linéaire à l'étude des processus de normalisation (principalement de λ-calcul).* PhD thesis, Université Denis Diderot, Paris 7, 1990.

[DR89] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28(3):181–203, 1989.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–102, 1987.

[GM01] Stefano Guerrini and Andrea Masini. Parsing MELL proof nets. *Theoretical Computer Science*, 254(1–2):317–335, 2001.

[Gue99] Stefano Guerrini. Correctness of multiplicative proof nets is linear. In *Proc. of the annual Symp. on Logic in Computer Science (LICS'99)*, pages 454–463. IEEE Computer Society Press, 1999.

[Imm88] Neil Immerman. Nondeterministic space is closed under complementation. *SIAM J. Comput.*, 17(5):935–938, 1988.

[JLL76] Neil D. Jones, Y. Edmund Lien, and William T. Laaser. New problems complete for nondeterministic logspace. *Mathematical Syst. Theory*, 10:1–17, 1976.

[JLM97] B. Jenner, K.-J. Lange, and P. McKenzie. Tree isomorphism and some other complete problems for deterministic logspace. DIRO 1059, Univ. de Montréal, 1997.

[Kan92] Max I. Kanovich. Horn programming in linear logic is NP-complete. In *Proc. of the annual Symp. on Logic in Computer Science (LICS'92)*, pages 200–210. IEEE Computer Society Press, 1992.

[Laf95] Yves Lafont. From proof-nets to interaction nets. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, volume 222, pages 225–247. Cambridge University Press, 1995.

[Mai] Harry G. Mairson. Normalization bounds for multiplicative linear logic are axiom-sensitive. Presentation at GEOCAL'06 Workshop of Implicit Computational Complexity. Slides available at http://www-lipn.univ-paris13.fr/∼baillot/GEOCAL06/SLIDES/Mairson.pdf.

[Mel04] Paul-André Melliès. *A topological correctness criterion for non-commutative logic*, volume 316 of *London Mathematical Society Lecture Notes Series*. Cambridge University Press, 2004.

[MO00] Andrzej Murawski and Luke Ong. Dominator trees and fast verification of proof nets. In *Proc. of the annual Symp. on Logic in Computer Science (LICS'00)*, pages 181–191. IEEE Computer Society Press, 2000.

[MO06] Andrzej Murawski and Luke Ong. Fast verification of MLL proof nets via IMLL. *ACM Trans. Comput. Logic*, 7(3):473–498, 2006.

[Rei05] Omer Reingold. Undirected st-connectivity in log-space. In Harold N. Gabow and Ronald Fagin, editors, *STOC*, pages 376–385. ACM, 2005.

[Sze87] Róbert Szelepcsényi. The method of forcing for nondeterministic automata. *Bulletin of the EATCS*, 33:96–99, 1987.