The proof equivalence problem for multiplicative linear logic is PSPACE-complete v0.3

Willem Heijltjes and Robin Houston · September 13, 2013

Abstract

MLL proof equivalence is the problem of deciding whether two proofs are related by a series of rule permutations. Previous work has shown the problem to be equivalent to a rewiring problem on proof nets, which are not canonical for full MLL due to the presence of the two units. Drawing from recent work on reconfiguration problems, in this paper it is shown that MLL proof equivalence is PSPACE-complete, using a reduction from Nondeterministic Constraint Logic.

$$\frac{\Gamma}{\Gamma,\bot}\,\bot \qquad \frac{}{1}\,1 \qquad \frac{\Gamma,A,B}{\Gamma,A\parr B}\,\parr \qquad \frac{\Gamma,A \qquad \Delta,B}{\Gamma,\Delta,A\otimes B}\,\otimes$$

Figure 1: Inference rules for unit-only MLL

$$\dfrac{\dfrac{\Gamma}{\Gamma,\bot^a}\,\bot}{\Gamma,\bot^a,\bot^b}\,\bot \sim \dfrac{\dfrac{\Gamma}{\Gamma,\bot^b}\,\bot}{\Gamma,\bot^a,\bot^b}\,\bot \qquad \dfrac{\dfrac{\Gamma,A,B}{\Gamma,A\parr B}\,\parr}{\Gamma,A\parr B,\bot}\,\bot \sim \dfrac{\dfrac{\Gamma,A,B}{\Gamma,A,B,\bot}\,\bot}{\Gamma,A\parr B,\bot}\,\parr$$

$$\dfrac{\dfrac{\Gamma,A \qquad \Delta,B}{\Gamma,\Delta,A\otimes B}\,\otimes}{\Gamma,\Delta,A\otimes B,\bot}\,\bot \sim \dfrac{\dfrac{\Gamma,A}{\Gamma,A,\bot}\,\bot \qquad \Delta,B}{\Gamma,\Delta,A\otimes B,\bot}\,\otimes \sim \dfrac{\Gamma,A \qquad \dfrac{\Delta,B}{\Delta,B,\bot}\,\bot}{\Gamma,\Delta,A\otimes B,\bot}\,\otimes$$

$$\dfrac{\dfrac{\Gamma,A,B,C,D}{\Gamma,A\parr B,C,D}\,\parr}{\Gamma,A\parr B,C\parr D}\,\parr \sim \dfrac{\dfrac{\Gamma,A,B,C,D}{\Gamma,A,B,C\parr D}\,\parr}{\Gamma,A\parr B,C\parr D}\,\parr$$

$$\dfrac{\dfrac{\Gamma,A \qquad \Delta,B,C,D}{\Gamma,\Delta,A\otimes B,C,D}\,\otimes}{\Gamma,\Delta,A\otimes B,C\parr D}\,\parr \sim \dfrac{\Gamma,A \qquad \dfrac{\Delta,B,C,D}{\Delta,B,C\parr D}\,\parr}{\Gamma,\Delta,A\otimes B,C\parr D}\,\otimes$$

$$\dfrac{\Gamma,A \qquad \dfrac{\Delta,B,C \qquad \Lambda,D}{\Delta,\Lambda,B,C\otimes D}\,\otimes}{\Gamma,\Delta,\Lambda,A\otimes B,C\otimes D}\,\otimes \sim \dfrac{\dfrac{\Gamma,A \qquad \Delta,B,C}{\Gamma,\Delta,A\otimes B,C}\,\otimes \qquad \Lambda,D}{\Gamma,\Delta,\Lambda,A\otimes B,C\otimes D}\,\otimes$$

Figure 2: Permutations

1 MLL

The formulae of unit-only multiplicative linear logic are given by the following grammar.

$$A,B,C := \bot \mid 1 \mid A\parr B \mid A\otimes B$$

The connectives $\otimes$ and $\parr$ will be considered up to associativity, and *duality* $A^\star$ is via DeMorgan. A *sequent* Γ,Δ will be a multiset of formulae. Within a sequent, connectives and units will be *named* with distinct elements from an arbitrary set of names N, e.g. $1^a \parr^b 1^c, \bot^d \otimes^e \bot^f$. This allows to 1) avoid using the notion of *occurrence*, and instead refer to subformulae by the name of their root connective, as e.g. A^b, 2) distinguish the two proofs of the above sequent while using standard multiset sequents, and 3) easily extract proof nets, as graphs using the names of connectives as vertices. Names will mostly be left implicit.

Proofs are constructed from the inference rules in Figure 1. The names of connectives are preserved through inferences. Only cut-free proofs are considered, and no cut-rule is added. *Permutations* of inference rules are displayed in Figure 2; the symmetric variants of the last two permutations, *par-tensor* and *tensor-tensor*, have been omitted.

Definition 1. *Equivalence* of proofs in (cut-free, unit-only) multiplicative linear logic ($\sim$) is the congruence generated by the permutations given in Figure 2. *MLL proof equivalence* is the problem of deciding whether two given proofs are equivalent.

The motivation to consider proofs up to equivalence is three-fold. Firstly, there is the strong intuition that the order of permutable inferences does not contribute to the essential content of the proof. Secondly, a technical motivation is that cut-elimination in MLL incorporates permutation steps, and composition via cut-elimination is only associative up to permutations. Thirdly, equivalent proofs are identified in natural models of multiplicative linear logic such as coherence spaces, and in the categorical semantics of MLL, $\star$-autonomous categories.

In one of several possible definitions, a *$\star$-autonomous category* (Barr, 1979) is a symmetric monoidal category $(\mathcal{C},\otimes,1)$ with:

- a *duality*, a contravariant functor $-^\star$ such that $A\cong A^{\star\star}$, and

- *closure*, an adjunction $- \otimes B \dashv (B \otimes -^\star)^\star$ for any object B,

satisfying natural coherence conditions. The category with as objects unit-only MLL-formulae and as morphisms $A \to B$ the equivalence classes of proofs of $A^\star \parr B$, denoted MLL$(\varnothing)$, is a $\star$-autonomous category. The present formulation of formulae induces two forms of *strictness*, instances where isomorphisms of the definition are identities: DeMorgan duality means $A = A^{\star\star}$, while associativity is an identity by decree. Modulo strictness, MLL$(\varnothing)$ is the *free* $\star$-autonomous category over the empty category $\varnothing$. This means that *any* $\star$-autonomous category is a model of the logic, and that MLL proof equivalence is the *word problem* for $\star$-autonomous categories, the problem of deciding when two representations of morphisms denote the same morphism.

1.1 Proof nets

A partial solution to the MLL proof equivalence problem is provided by proof nets.

Definition 2. For a sequent Γ,

- a *linking* ℓ is a function from the names of $\bot$-subformulae to the names of 1-subformulae,
- a *switching graph* for ℓ is an undirected graph over the names of Γ, with for every subformula $A^a \otimes^c B^b$ the edges $a - c$ and $b - c$, for every subformula $A^a \parr^c B^b$ either the edge $a - c$ or the edge $b - c$, and for every subformula $\bot^a$ the edge $a - \ell(a)$,
- a *proof net* ℓ or (Γ, ℓ) is a linking ℓ such that every switching graph is acyclic and connected.

An edge $a - \ell(a)$ in a proof net or switching graph is a *link* or *jump*.

Definition 3. A *permutation* between proof nets is the redirection of exactly one link. *Equivalence* $(\sim)$ of proof nets over a sequent Γ is the congruence generated by permutations.

There is no canonical interpretation of a proof as a proof net, since the introduction rule for $\bot$ in proofs joins a $\bot$-formula to a sequent, rather than a formula.

Definition 4. The relation $(\Mapsto)$ interprets a proof Π for a sequent Γ by a linking ℓ as follows: $\Pi \Mapsto \ell$ if for each $\bot^a$ in Γ, if Δ is the context of the inference introducing $\bot^a$, as illustrated below, then $\ell(a)$ is the name of some 1 in Δ.

$$\frac{\Delta}{\Delta, \bot^a}\,\bot$$

Proposition 5 (Danos and Regnier, 1989). *For a proof Π with conclusion Γ, if $\Pi \Mapsto \ell$ then ℓ is a proof net for Γ. For a net ℓ for Γ, there is a proof Π of Γ such that $\Pi \Mapsto \ell$ (*sequentialisation*).*

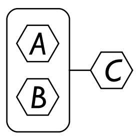
Proof nets are canonical representations of proofs in the absence of units: they factor out the permutations among tensor- and par-inferences, which are the last three permutations in Figure 2. Equivalence of proof nets is generated by the remaining equations, the permutations on $\bot$-introduction.

Proposition 6 (Hughes, 2012). *For proofs Π, Π' and proof nets ℓ, ℓ' such that $\Pi \Mapsto \ell$ and $\Pi' \Mapsto \ell'$, $\Pi \sim \Pi'$ if and only if $\ell \sim \ell'$.*

MLL proof equivalence is the problem of deciding equivalence of proof nets.

1.2 Notation

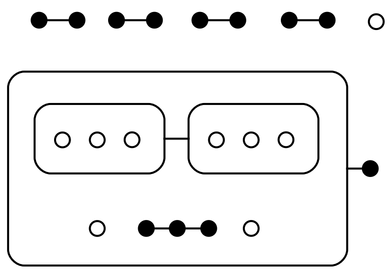
We will use a concise diagrammatic notation for sequents and proof nets. The units 1 and $\bot$ are represented by a circle $\circ$ and a disc $\bullet$ respectively. A tensor is represented by a line connecting both subformulae, and a par by juxtaposition: if A and B are represented by $\langle A \rangle$ and $\langle B \rangle$, then $A \parr B$ is $\langle A \rangle\ \ \langle B \rangle$ and $A \otimes B$ is $\langle A \rangle\!-\!\langle B \rangle$. A tensor of multiple elements is denoted by stringing them together in a line, so $A \otimes B \otimes C$ is $\langle A \rangle\!-\!\langle B \rangle\!-\!\langle C \rangle$. Boxes play the role of parentheses around par-formulae, so $(A \parr B) \otimes C$ is drawn as



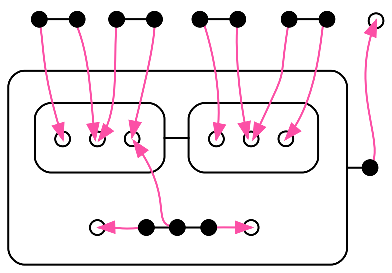
For example, this sequent

$$\vdash \bot \otimes \bot, \bot \otimes \bot, \bot \otimes \bot, \bot \otimes \bot, 1, \{[(1 \parr 1 \parr 1) \otimes (1 \parr 1 \parr 1)] \parr 1 \parr (\bot \otimes \bot \otimes \bot) \parr 1\} \otimes \bot$$

could be drawn like this:



We represent a proof net by drawing an arrow from each • to some ∘. For example, one proof net on the above sequent is



2 Equivalence in the absence of ⅋

Let a *1-alternation* sequent be one over formulae of the form 1 or $\bot \otimes \ldots \otimes \bot$, where the number of $\bot$-subformulae is at least 2. Such a sequent is inhabited exactly when the number of formulae in the sequent is one greater than the total number of $\bot$-subformulae it contains. An inhabited 1-alternation sequent with only one tensor-formula, i.e. a sequent of the form $1, \ldots, 1, \bot \otimes \ldots \otimes \bot$ with n $\bot$-subformulae and n 1-subformulae, will admit $n!$ different proof nets, each with n links. Since no link can re-attach, its equivalence classes are singletons.

Proposition 7. *For a 1-alternation sequent with at least two tensor-formulae there are at most two equivalence classes of proof nets.*

Proof. It will be shown by induction on the number of $\bot$-formulae in Γ that every proof net for Γ belongs to one of two equivalence classes. For the base case, the smallest inhabited sequent with two tensor-formulae is the following.

$$1, 1, 1, \bot \otimes \bot, \bot\otimes, \bot$$

It has two equivalence classes, of 12 proof nets each. Apart from listing these exhaustively, this can also be shown by using the proof of the inductive step, below, to reduce the base case to that of the sequent $1, 1, \bot \otimes \bot$, which has two singleton equivalence classes.

For the inductive step, let Γ be the following sequent.

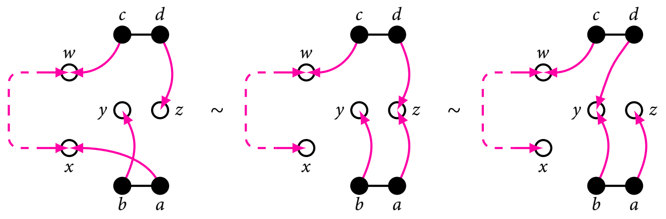
$$\Delta, A \otimes \bot^a, 1^z$$

There are two cases: 1) where A is a tensor-formula, and 2) where A is $\bot$ and where, for the induction hypothesis to apply, Δ contains at least two tensor-formulae. For both cases, it will be shown that any net ℓ for Γ is equivalent to a net ℓ' where $\bot^a$ connects to 1^z, and is the only link to do so. This reduces equivalence on Γ to equivalence on Δ, A in case 1, and on Δ in case 2, so that the induction hypothesis applies.

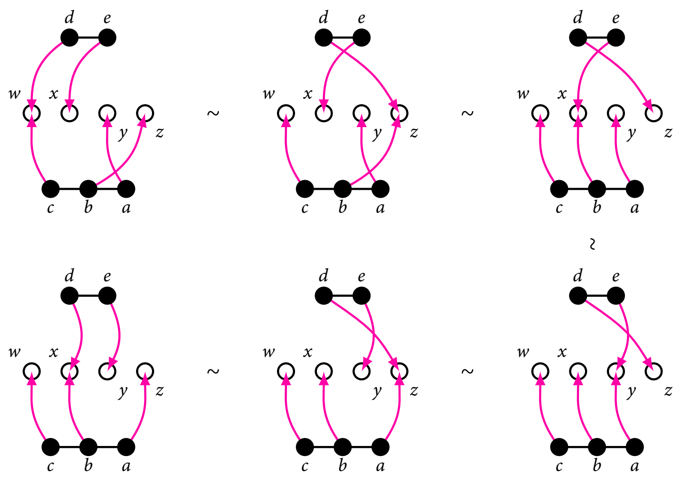
In constructing ℓ', since the proof net must be connected, there is a path from a to z. We will consider only the jumps on this path, not any other edges. There are four cases.

i. The path consists of exactly the jump $a - z$. Let $b - y$ be a jump from a $\bot^b$ in A; then ℓ' is obtained from ℓ by changing: $\ell'(e) = y$ for every $\bot^e$ such that $\ell(e) = z$ and $e \neq a$.

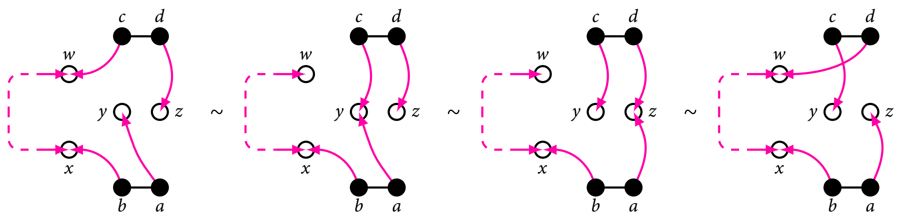
ii. The path starts with the jump $a - x$ (and $x \neq z$). Let $b - y$ be a jump from a $\bot^b$ in A, and let the path end with the jumps $w - c$ and $d - z$, where $\bot^c$ and $\bot^d$ are in the same tensor-formula B. Then ℓ' is obtained from ℓ by changing: $\ell'(a) = z$, and $\ell'(e) = w$ for every $\bot^e$ such that $\ell(e) = z$ and $e \neq a$, including d. These changes are illustrated as permutations below.



iii. The path consists of exactly one jump $b - z$ from a $\bot^b$ in A (and $b \neq a$). Let $\ell(a) = y$. Choose a jump $c - w$ from $\bot^c$ in A such that there is another $d - w$ from $\bot^d$ in a formula B (not excluding the possibilities $c = b$ and $c = a$). Let $e - x$ be a further jump from $\bot^e$ in B. Then ℓ' is obtained from ℓ by changing: $\ell'(a) = z$, $\ell'(d) = x$, $\ell'(e) = y$, and $\ell'(f) = x$ for each f (including b) such that $\ell(f) = z$ and $f \neq a$. These changes are exhibited as a series of permutations below, from top left to bottom left (note that the jump from d moves twice).



iv. The path starts with a jump $b - x$ from a $\bot^b$ in A (and $b \neq a$, $x \neq z$). Let the path end with the jumps $w - c$ and $d - z$, and let $\ell(a) = y$. Then ℓ' is obtained from ℓ by changing: $\ell'(c) = y$, $\ell'(a) = z$, and $\ell'(e) = w$ for every $\bot^e$ such that $e \neq a$ and $\ell(e) = z$, including d. This is illustrated below.



□

Proposition 8. *For a proof net for a 1-alternation sequent containing a link $\bot^a - 1^b$ and a formula 1^c, the following are equivalent.*

- *The edge $a - b$ can be permuted to $a - c$.*
- *There is a path from b to c not passing through a.*
- *The path from a to c starts with the jump $a - b$.*

If a link $a - b$ may be reconnected as $a - c$ it is said that *a may connect to c*. By the above proposition, it is immediate that if a and b may both connect to c, then after actually reconnecting $a - c$, still b may connect to c.

Consider the following naming scheme for the units in a 1-alternation sequent Γ with tensor-formulae $A_1, \ldots, A_n$.

- One 1 in Γ is named $*$, and the remaining ones with the numbers $n + 1, \ldots, m$.
- A $\bot$-formula in A_i is named by a pair (i, k), where $k = *$ for the first $\bot$-formula in each A_i, and for the remaining $\bot$-formulae in all A_i, each k is a distinct number in $n + 1, \ldots, m$.

The naming scheme suggests a linking for Γ, defined by $\ell(i, k) = k$; i.e the first $\bot$ in each tensor-formula connects to 1^*, while other $\bot$-subformulae connect uniquely to the remaining 1-subformulae.

A net for Γ is interpreted as a combinatorial permutation (an automorphism on $\{1, \ldots, m\}$) as follows.

Definition 9. To a proof net ℓ for a 1-alternation sequent Γ named as above, associate the *permutation* $p_\ell : \{1,\dots,m\} \to \{1,\dots,m\}$ given by:

$$p_\ell(k) = \begin{cases} i & \text{if } (i,k) \text{ may connect to } *\text{; and} \\ \ell(i,k) & \text{otherwise.} \end{cases}$$

The *parity* of ℓ is the parity of its permutation.

To see that p_ℓ is injective, consider the following.

- The domains of i and $\ell(i,k)$, respectively $1,\dots,n$ and $n+1,\dots,m$, are disjoint.
- Exactly one $\bot$-formula in each A_i may connect to $*$ because of connectedness and acyclicity, since if a $\bot$-formula may connect to $*$ it has a path to $*$ (Proposition 8).
- If two $\bot$-formulae have the same target, which means they are in different tensor-formulae, at least one may connect to $*$ via the other tensor-formula, which must have a path to $*$ by the above.

Proposition 10. *A permutation on a net ℓ preserves its parity.*

Proof. Let ℓ be a net for Γ, with Γ named as above, and let the link $(i,k) - x$ in ℓ re-attach as $(i,k) - y$, forming ℓ'. There are two cases, depending on whether (i,k) may connect to $*$. If so, using Proposition 8, the re-wiring preserves which $\bot$-formulae may connect to $*$, since for any path to $*$ via $(i,k) - x$ in ℓ there is a path to $*$ via $(i,k) - y$. Then the permutation of ℓ' is that of ℓ.

If (i,k) may not connect to $*$, let the path from x to y run via the following $\bot$- and 1-vertices.

$$x = x_1, (i_1, j_1), (i_1, k_1), x_2, (i_2, j_2), \dots, (i_n, k_n), x_{n+1} = y$$

Note that the $\bot$-formulae (i_a, j_a) may connect to $*$. On the relevant domain, this gives the following permutation for ℓ.

$$\begin{pmatrix} j_1 & \dots & j_n & k & k_1 & \dots & k_n \\ i_1 & \dots & i_n & x_1 & x_2 & \dots & x_{n+1} \end{pmatrix}$$

In ℓ', since $\ell'(i,k) = y$, the $\bot$-formulae that may connect to $*$ are the (i_a, k_a). The permutation $p_{\ell'}$ is the following.

$$\begin{pmatrix} j_1 & \dots & j_n & k & k_1 & \dots & k_n \\ x_1 & \dots & x_n & x_{n+1} & i_1 & \dots & i_n \end{pmatrix}$$

The parity of both permutations is the same if and only if the relative permutation, below, is even.

$$\begin{pmatrix} i_1 & \dots & i_n & x_1 & x_2 & \dots & x_{n+1} \\ x_1 & \dots & x_n & x_{n+1} & i_1 & \dots & i_n \end{pmatrix}$$

This is the case, as it is obtained by the exchange of x_a and i_a for each $a \leq n$, and subsequently the exchange of x_{n+1} and each i_a in turn. $\square$

Theorem 11. *MLL proof equivalence in the absence of $\parr$ is linear-time decidable.*

Proof. For a sequent with 1 tensor-formula, the problem is syntactic equality. For a sequent with 2 or more tensor-formulae, by Propositions 7 and 10 the equivalence of two nets is determined by their relative parity. Following Definition 9 the parity of a net can be read off in a single traversal of the net. This yields a linear-time algorithm. $\square$

References

Michael Barr. **-Autonomous categories*, volume 752 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, Heidelberg, New York, 1979.

Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28:181–203, 1989.

Dominic J.D. Hughes. Simple multiplicative proof nets with units. *Annals of Pure and Applied Logic*, 2012. arXiv:math.LO/0507003.