A linear logical view of linear type isomorphisms

Roberto Di Cosmo
LIENS and DMI
Ecole Normale Supérieure
45 rue d'Ulm
75005 Paris, France
dicosmo@dmi.ens.fr

Vincent Balat
LIENS and DMI
Ecole Normale Supérieure
45 rue d'Ulm
75005 Paris, France
balat@dmi.ens.fr

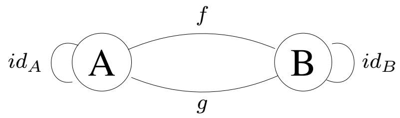
March 21, 1999

Abstract

The notion of isomorphisms of types has many theoretical as well as practical consequences, and isomorphisms of types have been investigated at length over the past years. Isomorphisms in weak system (like linear lambda calculus) have recently been investigated due to their practical interest in library search. In theis paper we give a remarkably simple and elegant characterization of linearly isomorphisms in the setting of Multiplicative Linear Logic (MLL), by making an essential use of the correctness criterion for Proof Nets due to Girard.

1 Introduction and Survey

The interest in building models satisfying specific isomorphisms of types (or domain equations) is a long standing one, as it is a crucial problem in the denotational semantics of programming languages. In the 1980s, though, some interest started to develop around the dual problem of finding the domain equations (type isomorphisms) that must hold in *every* model of a given language. Alternatively, one could say that we are looking for those objects that can be encoded into one-another by means of conversion functions f and g *without* loss of information, i.e. such that the following diagram commutes



The seminal paper by Bruce and Longo [3] addressed the case of pure first and second order typed λ-calculus with essentially model-theoretic motivations, but due to the connections between typed λ-calculus, cartesian closed categories, proof theory and functional programming, the notion of *isomorphism of types* showed up as a central idea that translates easily in each of those different but related settings. In the framework of category theory, Soloviev already studied the problem of characterizing types (objects) that are isomorphic in every cartesian closed category, providing a model theoretic proof of completeness for the theory we will see later on (actually [18] is based on techniques used originally in [12], while another different proof can be found in [13]). A treatment of this same problem by means of purely syntactic methods for a λ-calculus extended with surjective pairing and unit type was developed in [2], where the relations between these settings, category theory and proof theory, originally suggested by Mints, have been studied, and pursued further on in [9]. Finally, [5] provides a complete characterization of valid isomorphisms of types for second order λ-calculus with surjective pairing and terminal object type, that includes all the previously studied systems (see table 1).

$$(\mathbf{swap})\quad A \to (B \to C) = B \to (A \to C) \quad \big\}\, Th^1$$

$$\left.\left.\left.\begin{array}{l}
\left.\begin{array}{ll}
1. & A \times B = B \times A \\
2. & A \times (B \times C) = (A \times B) \times C \\
3. & (A \times B) \to C = A \to (B \to C) \\
4. & A \to (B \times C) = (A \to B) \times (A \to C) \\
5. & A \times \mathbf{T} = A \\
6. & A \to \mathbf{T} = \mathbf{T} \\
7. & \mathbf{T} \to A = A
\end{array}\right\} Th^1_{\times T} \\
\\
\left.\begin{array}{ll}
8. & \forall X.\forall Y.A = \forall Y.\forall X.A \\
9. & \forall X.A = \forall Y.A[Y/X] \quad \text{[1]} \\
10. & \forall X.(A \to B) = A \to \forall X.B \quad \text{[2]}
\end{array}\right\} + \mathbf{swap} = Th^2 \\
\\
\begin{array}{ll}
11. & \forall X.A \times B = \forall X.A \times \forall X.B \\
12. & \forall X.\mathbf{T} = \mathbf{T}
\end{array}
\end{array}\right\} Th^2_{\times T} \\
\\
\mathbf{split} \quad \forall X.A \times B = \forall X.\forall Y.A \times (B[Y/X])
\end{array}\right\} - 10, 11 = Th^{ML}\right.$$

A, B, C stand for any type, while **T** is a constant for the terminal type `unit`.
Axiom **swap** in Th^1 is derivable in $Th^1_{\times T}$ via 1 and 4.

Table 1: Theories of isomorphisms for some type lambda calculi.

These results have found their applications in the area of Functional Programming, where they provide a means to search functions by type (see [15, 16, 14, 17, 7, 6, 8]) and to match modules by specifications [1]. Also, they are used in proof assistant to find proofs in libraries up to irrelevant syntactical details [4].

Linear isomorphisms of types Recently, some weaker variants of isomorphism of types are also of practical interest, linear isomorphism of types in particular (e.g. , for library search in databases), which correspond to the isomorphism of objects in Simmetric Monoidal Categories, and can be also described as the isomorphism of types in the system of lambda calculus which corresponds to intuitionistic multiplicative linear logic. (A description of this system can be found in [Soloviev93]).

In [Soloviev93] it was shown that the axiom system consisting of the axioms Swap, 1, 2, 3, 5, and 7 defines an equivalence relation on types that coincides with the relation of linear isomorphism of types.

[1]Provided X is free for Y in A, and $Y \notin FTV(A)$

[2]Provided $X \notin FTV(A)$

In this paper, instead of studying linear isomorphisms of intuitionistic formulae, as is done when considering directly linear λ terms, we focus on linear isomorphisms of types *inside* the multiplicative fragment (MLL) of Linear Logic [10], which is the natural settings for investigating the effect of linearity. We should stress that isomorphisms in MLL are not the same as in linear lambda calculus: MLL is a richer system, allowing formulae, and proofs, like $A^{\perp}\otimes A$ that have no correspondence in linear lambda calculus, so we are investigating a finer world. But a particularly nice result of this change of point of view is that the axioms for linear isomorphisms are reduced to a remarkably simple form, manely, ot associativity and commutativity of the logical connectives tensor and par of MLL, plus the obvious axioms for the identities.

For example, when interpreting as usual linear implication $A \multimap B$ as $A^{\perp}\invamp B$, the isomorphisms $\mathbf{T} \to A = A$ becomes the simple identity isomorphism $\perp\invamp A = A$. Similarly, currying ($(A \times B) \to C = A \to (B \to C)$) becomes just associativity of the par connective $(A^{\perp}\invamp B^{\perp})\invamp C = A^{\perp}(\invamp B^{\perp}\invamp C)$ and Swap ($A \to (B \to C) = B \to (A \to C)$) becomes just $A^{\perp}(\invamp B^{\perp}\invamp C) = B^{\perp}(\invamp A^{\perp}\invamp C)$, which is a consequence of associativity and commutativity of par.

Formally, isomorphisms of formulae in MLL is defined as follows:

Definition 1.1 (Linear isomorphism) *Two formulae A and B are* isomorphic *iff*

- *A and B are linearly equivalent, i.e. $\vdash A^{\perp}, B$ and $\vdash B^{\perp}, A$*
- *there exists proofs of $\vdash A^{\perp}, B$ and $\vdash B^{\perp}, A$ that reduce, when composed by a cut rule to obtain a proof of $\vdash A^{\perp}, A$ (resp. $\vdash B^{\perp}, B$)), to the expansion of the axiom $\vdash A^{\perp}, A$ (resp $\vdash B^{\perp}, B$) after cut elimination*[3].

We show in this paper that two formulae A and B are linearly isomorphic if and only if $AC(\otimes,\invamp)\vdash A = B$ in the case of MLL without units, and $ACI(\otimes,\invamp)\vdash A = B$ in the case of MLL with units, where $AC(\otimes,\invamp)$ and $ACI(\otimes,\invamp)$ are defined in the following way:

Definition 1.2 ($AC(\otimes,\invamp)$) *Let $AC(\otimes,\invamp)$ denote the set constituted by the four following equations:*

$$X\otimes Y = Y\otimes X$$

$$X\invamp Y = Y\invamp X$$

$$(X\invamp Y)\invamp Z = X\invamp(Y\invamp Z)$$

$$(X\otimes Y)\otimes Z = X\otimes(Y\otimes Z)$$

$AC(\otimes,\invamp)\vdash A = B$ means that $A = B$ belongs to the equationnal theory generated by $AC(\otimes,\invamp)$ over the set of linear logic fomulae; in other words, it means that the formulae are equal modulo associativity and commutativity of $\invamp$ and $\otimes$.

Definition 1.3 ($ACI(\otimes,\invamp)$) *Let $ACI(\otimes,\invamp)$ denote the set constituted by the equations of $AC(\otimes,\invamp)$ plus:*

$$X\otimes 1 = X$$

[3]That is to say, to the proof of $\vdash A^{\perp}, A$ obtained by allowing only atomic axioms.

$$X ⅋ \bot = X$$

$ACI(\otimes, ⅋) \vdash A = B$ *means that* $A = B$ *belongs to the equationnal theory generated by* $ACI(\otimes, ⅋)$.

As always, in the investigaiton of theories of isomorphic objects, the soundness part is easy to prove:

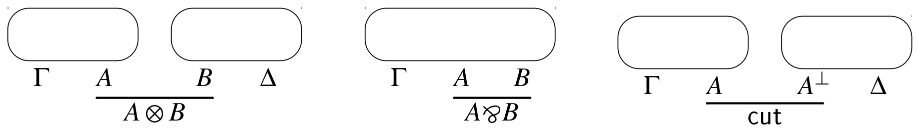
Theorem 1.4 (Isos soundness) *If* $ACI(\otimes, ⅋) \vdash A = B$, *then* A *and* B *are linearly isomorphic*

Proof. By exhibiting the simple nets for the axioms and showing context closure. □

All the difficulty really lies in the proof of the other implication, completeness: the rest of the article focuses on its proof. To do that, we use in an essential way the proof-nets of Girard.

Let's now recall some preliminary definitions from linear logic (bu we refer to [10] for a complete introduction):

Definition 1.5 (proof nets) *A proof net is a structure inductively obtained starting from* axiom links $\overset{\text{ax}}{\overline{A \quad A^\perp}}$ *via the three construction rules*



Definition 1.6 (simple nets) *A proof net is simple if it contains only atomic axiom links (i.e. axiom links involving only atomic formulae).*

It is quite simple to show that

Proposition 1.7 (η-expansion of proof nets) *For any (possibly not simple) proof net S, there is a simple proof net with the same conclusions, which we call $\eta(S)$.*

Proof. This is done by iterating a simple procedure of $\eta - expansion$ of non atomic axiom links, which can be replaced by two axiom links plus one par and one times link. □

This means that, as far as provability is concerned, one can restrict our attention to simple nets. We will show that also as far as isomorphism is concerned we can make this assumption.

We first characterize isomorphic formulae in MLL without units, by reducing isomorphism to the existence of two particular proof-nets called *simple bipartite proof-nets*. Then we show that we can restrict to *non-ambiguous formulae* (that is, formulae in which atoms occurs at most once positive and at most once negative). This characterization in term of nets allows to prove completeness of $AC(\otimes, ⅋)$ for MLL (without units) by a simple induction on the size of the proof-net.

In the presence of units, we first simplify the formulae by removing all the nodes of the shape $\bot ⅋ A$ and $1 \otimes A$. Then we remark that isomorphisms for simplified formulae are very similar to the case without units. By showing a remarkable property of proof-nets for simplified formulae, we can indeed reduce the completeness proof to the case without units.

The paper is organized as follows: reduction to simple bipartite proof-nets and non-ambiguous formulae will be detailed in sections 2 and 3. Then we will show the final result in section 4 before extending it to the case with units in section 5.

2 Reduction to simple bipartite proof nets

First of all, we formalize the reduction to simple nets hinted at in the introduction.

Definition 2.1 (tree of a formula, identity simple net) *A cut-free simple proof net S proving A is actually composed of the* tree *of A, (named $T(A)$), and a set of axiom links over atomic formulae. We call* identity simple net *of A the simple cut-free proof net obtained by a full η-expansion of the (generally not simple) net $\overline{A \qquad A^{\perp}}$. This net is made up of $T(A)$, de $T(A^{\perp})$ and a set of axiom links that connect atoms in $T(A)$ with atoms in $T(A^{\perp})$.*
Notice that, in simple nets, the identity axiom for A is interpreted by the identity simple net of A.

A first remark, which is important for a simple treatment of linear isomorphisms, is that we can focus, w.l.o.g., on witnesses of isomorphisms which are simple proof nets.

Lemma 2.2 (Simple vs. non-simple nets) *If a (non-simple) net S reduces via cut-elimination to S', then the simple net $\eta(S)$ reduces to $\eta(S')$.*

Proof. It is sufficient to show that if $S \triangleright_1 S'$ (one step reduction), then $\eta(S) \triangleright^* \eta(S')$ (arbitrary long reduction). If the redex R reduced in S does not contain any axiom link, then exactly the same redex appears in $\eta(S)$ that can therefore be reduced in one step to $\eta(S')$. Otherwise R contains an axiom link, that may be non atomic (if the axiom link is atomic, then the considered redex is exactly the same in S and $\eta(S)$ and the property is obvious). If F is non atomic and $\underline{F \quad F^{\perp}}$ is the cut link of R, let n be the number of atoms of F (counted with multiplicity). Then $n-1$ is the number of connectives in F, and $\eta(S)$ can be reduced to $\eta(S')$ in $2n-1$ steps ($n-1$ steps to propagate the cut link to atomic formulae, and n steps to reduce every atomic axiom link produced this way). □

Theorem 2.3 (Reduction to simple proof nets) *Two formulae A and B are* isomorphic *iff there are two simple nets S with conclusions $A^{\perp}, B$ and S' with conclusions $B^{\perp}, A$ that when composed using a cut rule over B (resp. A) yield after cut elimination the identity simple net of A (resp. B).*

Proof. The 'if' direction is trivial, since a proof net represents a proof and cut elimination in proof nets correspond to cut elimination over proofs.
For the 'only if' direction, take the two proofs giving the isomorphism and build the associated proof nets S and S'. These nets have as conclusions $A^{\perp}, B$ (resp. $B^{\perp}, A$), and we know that after composing them via cut over B (resp. A) and performing cut elimination, one obtains the axiom net of A (resp. B).Now take the full η-expansions of S and S' as the required simple nets: by lemma 2.2, they reduce by composition over B (resp. A) to the identity simple net of A (resp. B). □

We will show now that if two formulae are isomorphic then the isomorphism can be given by means of proof nets whose structure is particularly simple.

Definition 2.4 (bipartite simple proof-nets) *A cut-free simple proof net is* bipartite *if it has exactly two conclusions A and B, and it consists of $T(A)$, $T(B)$ and a set of axiom links connecting atoms of A to atoms of B, but not atoms of A between them or atoms of B between them (an example is shown in figure 1).*

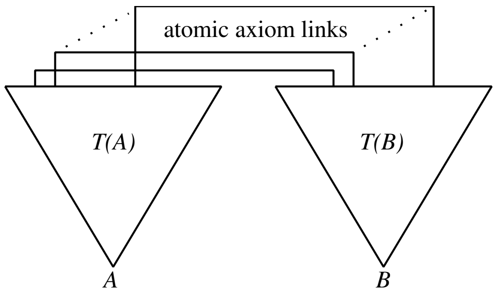


Figure 1: The shape of a bipartite proof net.

Lemma 2.5 (cuts and trees) *Let S be a simple net (not a proof net) without conclusions built out of just $T(A)$ and $T(A^{\perp})$, with no axiom link, and the cut $\underline{A \quad A^{\perp}}$. Then cut-elimination on S yields as a result just a set of atomic cut links $\underline{p_i \quad p_i^{\perp}}$ between atoms of A and atoms of $A^{\perp}$.*

Proof. This is a simple induction on the size of the net. □

Theorem 2.6 (Isomorphisms are bipartite) *Let S be a cut-free simple proof net with conclusions $A^{\perp}$ and B, and S' be a cut-free proof net with conclusions $B^{\perp}$ and A. If their composition by cut gives respectively the identity simple net of A and B, then S and S' are bipartite.*

Proof. We actually show the contrapositive: if S or S' is not bipartite, then their composition by cut is not bipartite, hence is not an identity proof net. By symetry, we can assume w.l.o.g. that S is not bipartite, and contains an axiom link between two atoms of $A^{\perp}$. We claim that the composition of S and S' by cut over B is not bipartite.
Since S and S' are cut-free, their composition only contains a single cut link (between B and $B^{\perp}$). Since S and S' are simple, every axiom link of their composition is atomic. Hence every (atomic) axiom link of the net $\underline{S \quad S'}$ that is reduced by cut elimination is connected to an atom of B or $B^{\perp}$. In particular, the axiom link of S only connected to atoms of A is *not reduced*. This proves that cut elimination in $\underline{S \quad S'}$ does not lead to a bipartite net.
As a consequence, the theorem holds. □

3 Reduction to non-ambiguous formulae

To prove the correctness theorem, we first show that one can restrict our study to non-ambiguous formulae, i.e. to formulae where each atom appears only once positive and one negated.

Definition 3.1 (non-ambiguous formulae) *We say that a formula A is* non-ambiguous *if each atom in A occurs at most once positive and at most once negative. For example, $A \otimes B$ and $A \otimes A^{\perp}$ are non-ambiguous, while $A \otimes A$ is not.*

As expected, a non-ambiguous formula can only be isomorphic to another non-ambiguous formula.

Lemma 3.2 (non-ambiguous isomorphic formulae) *Let A and B be isomorphic formulae, and α a renaming such that $\alpha(A)$ is non-ambiguous, then $\alpha(B^{\perp})$ is non-ambiguous.*

Proof. If A and B are isomorphic formulae, then (by theorem 2.6) there exist a bipartite proof net with conclusions A, $B^{\perp}$. Since α is a renaming, there also exists a *bipartite proof net with conclusions* $\alpha(A)$, $\alpha(B^{\perp})$. Then, $\alpha(A)$ and $\alpha(B^{\perp})$ have exactly the same atoms. And since $\alpha(A)$ is non-ambiguous, $\alpha(B^{\perp})$ is also non-ambiguous. $\square$

We can now prove that isomoprhism is invariant by renaming.

Definition 3.3 (renaming) *An application α from the set of occurrences of atoms in a proof net R to a set of atoms is a* renaming *if $\alpha(R)$ (the net obtained by substitution of each occurrence p of an atom of R by $\alpha(p)$) is a correct proof net.*

Note that if R is simple, the definition of α only on the occurrences of atoms in *axiom links* is sufficient to define α on every occurrence in R. If R is simple and bipartite, then the definition of α only on the occurrences of atoms in *one conclusion* of R is sufficient to define α on every occurrence in R. Note also that, if the conclusions of R are ambiguous formulae, then two different occurrences of the same atom can be renamed differently.

Theorem 3.4 (renaming preserve isomorphisms) *If A and B are linearly isomorphic, let R and R' be the associated simple proof nets (wih conclusions $A^{\perp}$, B and $B^{\perp}$, A respectively). If α is a renaming of (the occurrences of) the atoms of R, then there exists α', a renaming of the atoms of R' such that $\alpha'(A)$ and $\alpha(B)$ are isomorphic, i.e.:*

- *$\alpha'(R')$ is a correct proof net.*
- $\alpha(A^{\perp}) \equiv (\alpha'(A))^{\perp}$
- $\alpha'(B^{\perp}) \equiv (\alpha(B))^{\perp}$
- *The composition of $\alpha(R)$ and $\alpha'(R')$ by cut over $\alpha(B)$ (resp. $\alpha'(A)$) gives the identity simple net of $\alpha'(A)$ (resp. $\alpha(B)$).*

Proof. We first have to define α'. Since (by theorem 2.6) R' is bipartite, it is sufficient to define α' only on the occurrences of $B^{\perp}$, i.e. to define $\alpha'(B^{\perp})$. So one can define: $\alpha'(B^{\perp}) \equiv (\alpha(B))^{\perp}$. Then the composition of $\alpha(R)$ and $\alpha'(R')$ by cut over B is a correct proof net. And since reduction of proof nets does not depend on labels, this composition reduces to an identity net with conclusions $\alpha(A^{\perp})$ and $\alpha'(A)$. An easy induction (on the number of connectives of $\alpha(A)$) show that η-reduction in this net gives an axiom link. One then has $\alpha(A^{\perp}) \equiv (\alpha'(A))^{\perp}$.
But then, the composition of $\alpha'(R')$ with $\alpha(R)$ by cut over $\alpha'(A)$ is a correct proof net, that reduces to an identity net (since reduction of proof nets does not depend on labels) that is the identity simple net of $\alpha(B)$.
Hence, $\alpha'(A)$ and $\alpha(B)$ are isomorphic. $\square$

We are finally in a position to show that we can restrict attention to non-ambiguous formulae.

Lemma 3.5 (ambiguous isomorphic formulae) *Let A and B be isomorphic formulae such that A is ambiguous, then there exists a substitution σ and formulae A' and B'* non-ambiguous *such that:*

- A' and B' are isomorphic formulae.

- $A' \equiv \sigma(A)$.
- $B' \equiv \sigma(B)$.

Proof.
Let R and R' be bipartite proof nets with conclusions $B^\perp$, A and $A^\perp$, B respectively associated to the isomorphism between A and B. Since it is sufficient here to define α only on occurrences of atoms of A, one can define a renaming α such that $\alpha(A)$ has only distinct atoms (i.e. no atom of $\alpha(A)$ occurs twice in $\alpha(A)$, even once positively and once negatively). In particular, $\alpha(A)$ is non-ambiguous. Then, theorem 3.4 gives an algorithm for defining a renaming α' such that $\alpha(A)$ and $\alpha'(B)$ are isomorphic: in particular $\alpha'(A^\perp) \equiv (\alpha(A))^\perp$ and $\alpha(B^\perp) \equiv (\alpha'(B))^\perp$.

Let $A' \equiv \alpha(A)$ and $B' \equiv \alpha'(B)$. By theorem 3.4, A' and B' are isomorphic and $\alpha(R)$ has conclusions A' and $B'^\perp$.

On $\alpha(R)$ one can define a renaming α^{-1} such that $\alpha^{-1}(A') \equiv A$, and hence $\alpha^{-1}(B'^\perp) \equiv B^\perp$.

Since $\alpha(R)$ is bipartite, it is equivalent to define α^{-1} on occurrences of R, or only on occurrences of atoms of A'. But since all atoms of A' are distincts, two distinct occurrences of atoms of A' correspond to distinct atoms of A'. One can then define a substitution σ on atoms of A' by: $\sigma(p) \equiv \alpha^{-1}(Occ(p))$ where $Occ(p)$ is the *single* occurrence of the atom p in A'.

Thus, $R \equiv \alpha^{-1}(\alpha(R)) \equiv \sigma(\alpha(R))$: in particular $\sigma(A') \equiv A$ and $\sigma(B'^\perp) \equiv \sigma(\alpha(B^\perp)) \equiv \alpha^{-1}(\alpha(B^\perp)) \equiv B^\perp$, so $\sigma(B') \equiv B$.

Finally, A' and B' are isomorphic non-ambiguous formulae such that $\sigma(A') \equiv A$ and $\sigma(B') \equiv B$. □

Corollary 3.6 (reduction to non-ambiguous formulae) *The set of couples of isomorphic formulae is the set of instances (by a substitution on atoms) of couples of isomorphic* non-ambiguous formulae.

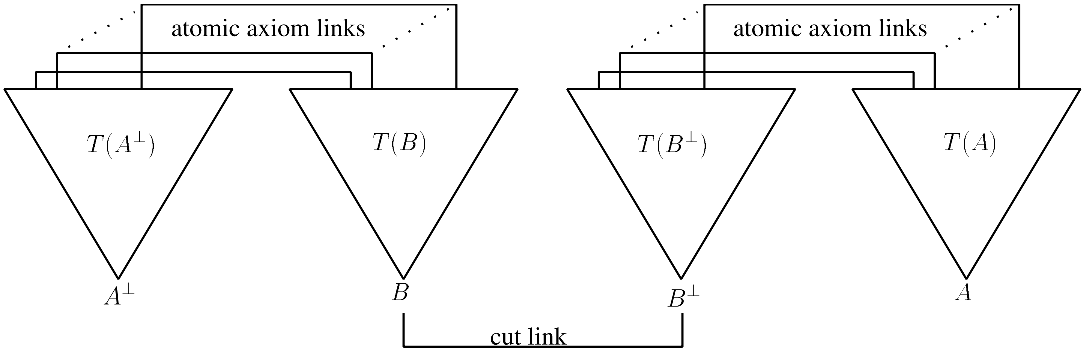
Proof. We show each inclusion separately.
Let A and B be two isomorphic formulae. By lemma 3.5, A and B are instances of two non-ambiguous isomorphic formulae A' and B' (eventually $A \equiv A'$ and $B \equiv B'$).

Conversly let C and D be isomorphic formulae and σ be a substitution on atoms of C (and then also on atoms of D). Let R and R' be two bipartite proof nets associated to C and D. The substitution σ defines on R a renaming α (any substitution can be seen as a renaming). Let α' be the renaming defined on R', associated to α as in theorem 3.4. Since $\sigma(C^\perp) \equiv \alpha(C^\perp) \equiv (\alpha'(C))^\perp$, α' is also the renaming induced by σ on R'. By theorem 3.4, $\alpha(C)$ and $\alpha(D)$ are isomorphic. Hence $\sigma(C)$ and $\sigma(D)$ are isomorphic. □

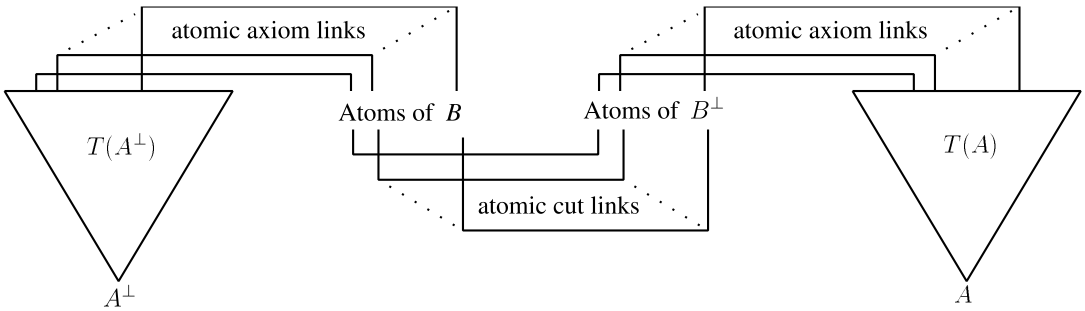
Hence, in what follows, we can focus only on non-ambiguous formulae.

We are now able to show that for non-ambiguous formulae the very existence of bipartite simple nets implies isomorphism.

Theorem 3.7 (bipartite simple nets for non-ambiguous formulae) *Let S be a bipartite simple proof net over $A^\perp$ and B, and S' a bipartite simple proof net over $B^\perp$ and A. Then their composition by cut over B reduces to the identity simple net of A (resp. their composition by cut over A reduces to the identity simple net over B).*



Proof. Consider the composition of S and S' by cut over B (see figure).By lemma 2.5, cut elimination in the subnet $\underline{T(B)\quad T(B^{\perp})}$ gives a set of atomic cut links between atoms of B and atoms of $B^{\perp}$. Since S ans S' are bipartite, each such *atomic cut link* is connected to an *atomic axiom link* between an atom of $A^{\perp}$ and an atom of B, and to an *atomic axiom link* between an atom of $B^{\perp}$ and an atom of A. Now, the net only contains atomic redex composed of cut and axiom links. The reduction of these redex gives the identity tree of A (since there is no axiom link connecting atoms of A —resp. $(^{\perp}A)$ between them). □



Theorems 3.7 and 2.6 have the following fundamental consequence.

Corollary 3.8 *Two linear non-ambiguous formulae A and B are isomorphic iff and only if there exists simple bipartite proof nets having conclusions $A^{\perp}, B$, and $B^{\perp}, A$.*

4 Completeness for isomorphisms in *MLL*

Using the result of the above section, and the following simple lemma, we can prove our main result, i.e. completeness of $AC(\otimes, \parr)$ for MLL without constants.

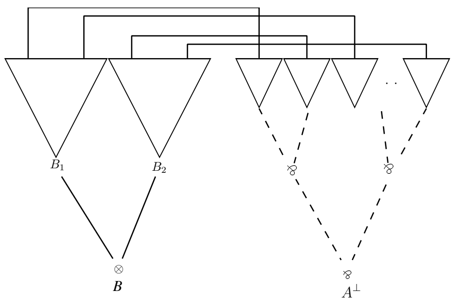
Lemma 4.1 (isomorphic formulae) *If A and B are linearly isomorphic, then they are both, either $\parr$-formulae, or $\otimes$-formulae.*

Proof. Actually, one $\parr$-formula and one $\otimes$-formula can not be isomorphic. If $A_1 \parr A_2$ and $A_3 \otimes A_4$ were isomorphic, then $(A_1 \parr A_2)^{\perp} \parr (A_3 \otimes A_4)$ would be provable. But there is no proof net with conclusion $(A_1^{\perp} \otimes A_2^{\perp}), (A_3 \otimes A_4)$. Effectively, such a net does *not* have a splitting (terminal) tensor, since removing one of both terminal tensor links does not give two diconnected graphs (by bipartiteness). Hence two isomorphic formulae must be both, either $\parr$-formulae, or $\otimes$-formulae. $\square$

Theorem 4.2 (Isos completeness) *If A and B are linearly isomorphic, then:*
$AC(\otimes, \parr) \vdash A = B$.

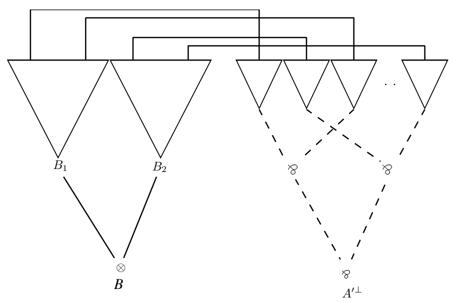
Proof. By induction on the size of the simple bipartite proof net, given by the isomorphism, with conclusions $A^\perp, B$.
If A and B are atomic, then the property is obvious.
Else the formulae $A^\perp$ and B are *both* non atomic (since the net is bipartite, they must contain the same number of atoms). Moreover, (by lemma 4.1) since A and B are isomorphic, one of the formulae $A^\perp, B$ is a $\parr$-formlua and the other one a $\otimes$-formula. One can assume, w.l.o.g., that B is a $\otimes$-formula, and $A^\perp$ a $\parr$-formula.

Now, removing all dangling $\parr$ nodes in the proof net with conclusions $A^\perp$, B gives a correct proofs net with conclusions of the form $A_1^\perp, \ldots, A_k^\perp, B_1 \otimes B_2$. If one of the formulae $A_1^\perp, \ldots, A_k^\perp$ contains a tensor node, removing it does not lead to two disconnected graphs, since (by bipartiteness) every atom of $A_1^\perp, \ldots, A_k^\perp$ is connected to the formula $B_1 \otimes B_2$.

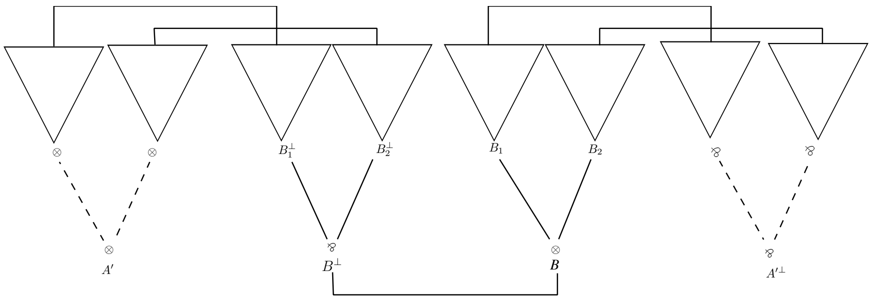


The splitting tensor node (that must exist due to Girard's correctness criterion) is necessarily the $B_1 \otimes B_2$ node. Removing it yields, due to the correctness criterion, two disconnected proof nets, which are still simple since the axiom links were not modified.

We can recover from these nets two bipartite simple proof nets (a bipartite proof net must have exacly two conclusions) by adding $\parr$-links under the A_i (because of $AC(\otimes, \parr)$, the order of the A_i does not matter). This constructs two formulae ${A'_1}^\perp$ (connected with B_1) and ${A'_2}^\perp$ (connected with B_2). The nets that we obtain contain at least one link less than the original net, and thus, are strictly smaller. It only remains to verify that B_1 and A'_1 are isomorphic, and that B_2 and A'_2 are isomorphic, so that we can apply the induction hypothesis.



To do that, we use the fact that the two initial formulae are isomorphic. If we put the initial ⅋ and $\otimes$ back, we obtain two formulae A' and B, that are isomorphic (by theorem 1.4). There exists a simple bipartite proof-nets with conclusions A' and $B^{\perp}$. Since the formulae are non-ambiguous, we can extract from this proof-net to simple bipartite proof-nets; one with conclusions A'_1 and $B_1^{\perp}$, the other with conclusions A'_2 and $B_2^{\perp}$. Hence, by theorem 3.7, A'_1 and B_1 are isomorphic, and A'_2 and B_2 are isomorphic.



By induction hypothesis, $AC(\otimes, ⅋) \vdash A'_1 = B_1$ and $AC(\otimes, ⅋) \vdash A'_2 = B_2$, and thus $AC(\otimes, ⅋) \vdash A' = B$. We can conclude using the fact that $AC(\otimes, ⅋) \vdash A = A'$. □

5 Handling the units

We have shown above soundness and completeness result for the theory of isomorphisms given in the introduction w.r.t. *provable* isomorphisms in *MLL*. This essentially corresponds to isomorphisms in all *-autonomous categories, which is a superset of all Symmetric Monoidal Closed Categories (SMCC's) *without units*. Nevertheless, if we want to get an interesting result also in terms of models, and handle then also SMCC's in their full form, we need to be able to add units to our treatment.

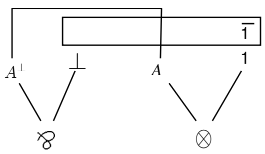
5.1 Expansions of axioms with units: identity simple nets revisited

In the presence of the units $\mathbf{1}$ and especially $\perp$, proof nets in general get more involved, as $\perp$ forces the introduction of the notion of *box* for which we refer the interested reader to [10], where a detailed explanation is presented.

For our purpose, it will suffice to recall here the proof-net formation rules for the units:

$$\frac{}{\mathbf{1}} \qquad\qquad \frac{\boxed{\begin{array}{c}\vdots\\ \overline{\Gamma}\end{array}}}{\Gamma \quad \perp}$$

Now, the expansion of an axiom can contain boxes, if the axiom formula involves units; for example, the axiom $\vdash (A \otimes \mathbf{1})^{\perp}, (A \otimes \mathbf{1})$ gets fully η-expanded into:



5.2 Reduction of isomorphisms to simplified formulae

First notice that a formula of the shape $1 \otimes A$ is always isomorphic to A, and $\perp \parr A$ is isomorphic to A.

Definition 5.1 (simplified formulae) *A formula is called* simplified *if it has no subformula of the shape $1 \otimes A$ or $\perp \parr A$ (where A is any formula). To each formula A, we associate the formula $s(A)$ obtained by rewriting it using the system:*

$$1 \otimes A \rightarrow A$$

$$A \otimes 1 \rightarrow A$$

$$\perp \parr A \rightarrow A$$

$$A \parr \perp \rightarrow A$$

We can restrict our attention to simplified formulae using the following theorem:

Theorem 5.2 *Two formulae A and B are linearly isomorphic if and only if $s(A)$ and $s(B)$ are linearly isomorphic.*

Proof. We first show that $1 \otimes A$ is isomorphic to A, and $\perp \parr A$ is isomorphic to A, and conclude using the fact that linear isomorphism is a congruence. □

5.3 Completeness with units

For simplified formulae, the proof of completeness is very similar than in the case without units. We just have to extent the definition of bipartite simple proof-nets:

Definition 5.3 (bipartite simple proof-nets) *A cut-free simple proof-net is* bipartite *if it has exactly two conclusions A and B, and it consists of*

- *$T(A)$, $T(B)$*
- *a set of axiom links connecting atoms of A to atoms of B, but not atoms of A between them or atoms of B between them*
- *and boxes with at least one conclusion in $T(A)$ and at least one conclusion in $T(B)$.*

Theorem 5.4 *Let A and B be two simplified formulae. Let S be a cut-free simple proof net with conclusions $A^\perp$ and B, and S' be a cut-free proof net with conclusions $B^\perp$ and A. All the boxes in S and S' are boxes containing only the constant* 1.

Proof. In this proof, let n_1^X and $n_\perp^X$ denote respectively the number of 1 and the number of $\perp$ in the proof-net X.

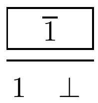
- The case where A or B is a constant is obvious.
- Otherwise, all the occurrences of the constant 1 in the two proof-nets are one of the two sub-terms of a $\parr$-link. The only way to have these two sub-terms connected (which is necessary to make a proof-net) is to put the 1 in a box. Each of these boxes correspond to a distinct $\perp$. From there we deduce that $n_\perp^S \geq n_1^S$, and $n_\perp^{S'} \geq n_1^{S'}$.

- But $n_{\perp}^{S} = n_{1}^{S'}$, $n_{\perp}^{S'} = n_{1}^{S}$. Hence $n_{\perp}^{S} = n_{1}^{S}$. Thus there is no box with somewhat else than an 1 in S. Idem for S'.

$\square$

It is easy to show that boxes of that kind behave like axiom links for cut-elimination. It suffices to remark that the only possible case of cut-elimination involving boxes in such a proof-net is the following one:

that reduces to



Thus units can be viewed exactly as atoms in this case, and we can proceed exactly as in the case without units to proove that

Theorem 5.5 *If A and B are linearly isomorphic, then:*
$AC(\otimes, \parr) \vdash s(A) = s(B)$.

Hence

Theorem 5.6 (Isos completeness with units) *If A and B are linearly isomorphic, then:*
$ACI(\otimes, \parr) \vdash A = B$.

6 Conclusions

We have shown that in MLL the only isomorphisms of formulae are given by the most intuitive axioms, namely associativity and commutativity, only. Besides the interest of the result on its own, this gives a very elegant symmetrical interpretation of linear isomorphism in linear lambda calculus, and provides a justification of the fact, observed several times in the past, that currying in functional programming correspond to "a sort of" associativity (this happens in the implementation of abstract machines, as well as in the coding of lambda terms into Berry's CDS). Our result confirms once more that Linear Logic is a loking glass through which fundamental properties of functional computation appear symetrized and simplified. It should also be remarked that the axiom links from Linear Logic play a similar role to the formula links originally introduced by Lambek in his study of SMC objects [11], and that proof nets were really the missing tool to understand linearity. In proving the result, we used essentially the topological properties of the proof nets of linear logic, which simplified enormoulsy our task (for example, the reduction to non ambiguous formulae when working directly with lambda terms is far more complex than here, where the axiom links allows us to give an elegant proof).

References

[1] M.-V. Aponte and R. Di Cosmo. Type isomorphisms for module signatures. In *Programming Languages Implementation and Logic Programming (PLILP)*, volume 1140 of *Lecture Notes in Computer Science*, pages 334–346. Springer-Verlag, 1996.

[2] K. Bruce, R. Di Cosmo, and G. Longo. Provable isomorphisms of types. *Mathematical Structures in Computer Science*, 2(2):231–247, 1992.

[3] K. Bruce and G. Longo. Provable isomorphisms and domain equations in models of typed languages. *ACM Symposium on Theory of Computing (STOC 85)*, May 1985.

[4] D. Delahaye, R. Di Cosmo, and B. Werner. Recherche dans une bibliothèque de preuves Coq en utilisant le type et modulo isomorphismes. In *PRC/GDR de programmation, Pôle Preuves et Spécifications Algébriques*, November 1997.

[5] R. Di Cosmo. Invertibility of terms and valid isomorphisms. a proof theoretic study on second order λ-calculus with surjective pairing and terminal object. Technical Report 91-10, LIENS - Ecole Normale Supérieure, 1991.

[6] R. Di Cosmo. Type isomorphisms in a type assignment framework. In *19th Ann. ACM Symp. on Principles of Programming Languages (POPL)*, pages 200–210. ACM, 1992.

[7] R. Di Cosmo. Deciding type isomorphisms in a type assignment framework. *Journal of Functional Programming*, 3(3):485–525, 1993. Special Issue on ML.

[8] R. Di Cosmo. *Isomorphisms of types: from λ-calculus to information retrieval and language design*. Birkhauser, 1995. ISBN-0-8176-3763-X.

[9] R. Di Cosmo and G. Longo. Constuctively equivalent propositions and isomorphisms of objects (or terms as natural transformations). In Moschovakis, editor, *Logic from Computer Science*, volume 21 of *Mathematical Sciences Research Institute Publications*, pages 73–94. Springer Verlag, Berkeley, 1991.

[10] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50, 1987.

[11] S. Mac Lane and G. M. Kelly. Coherence in Closed Categories. *Journal of Pure and Applied Algebra*, 1(1):97–140, 1971.

[12] C. F. Martin. Axiomatic bases for equational theories of natural numbers. *Notices of the Am. Math. Soc.*, 19(7):778, 1972.

[13] L. Meertens and A. Siebes. Universal type isomorphisms in cartesian closed categories. Centrum voor Wiskunde en Informatica, Amsterdam, the Netherlands. E-mail: lambert,arno@cwi.nl, 1990.

[14] M. Rittri. Retrieving library identifiers by equational matching of types in 10th Int. Conf. on Automated Deduction. *Lecture Notes in Computer Science*, 449, July 1990.

[15] M. Rittri. *Searching program libraries by type and proving compiler correctness by bisimulation*. PhD thesis, University of Göteborg, Göteborg, Sweden, 1990.

[16] M. Rittri. Using types as search keys in function libraries. *Journal of Functional Programming*, 1(1):71–89, 1991.

[17] C. Runciman and I. Toyn. Retrieving re-usable software components by polymorphic type. *Journal of Functional Programming*, 1(2):191–211, 1991.

[18] S. V. Soloviev. The category of finite sets and cartesian closed categories. *Journal of Soviet Mathematics*, 22(3):1387–1400, 1983.