Multiple Conclusion Linear Logic: Cut Elimination and more

Harley Eades III and Valeria de Paiva

Abstract

Full Intuitionistic Linear Logic (FILL) was first introduced by Hyland and de Paiva as one of the results of their investigation into a categorical understanding of Gödel's dialectica interpretation. FILL went against current beliefs that it was not possible to incorporate all of the linear connectives, e.g. tensor, par, and implication, into an intuitionistic linear logic. They showed that it is natural to support all of the connectives given sequents that have multiple hypotheses and multiple conclusions. To enforce intuitionism de Paiva's original formalization of FILL used the well-known Dragalin restriction, forcing the implication right rule to have only a single conclusion in its premise, but Schellinx showed that this results in a failure of cut-elimination. To overcome this failure Hyland and de Paiva introduced a term assignment for FILL that eliminated the need for the strong restriction. The main idea was to first relax the restriction by assigning variables to each hypothesis and terms to each conclusion. Then when introducing an implication on the right enforcing that the variable annotating the hypothesis being discharged is only free in the term annotating the conclusion of the implication. Bierman showed that this formalization of FILL still did not enjoy cut-elimination, because of a flaw in the left rule for par. However, Bellin proposed an alternate left rule for par and gave an indirect proof showing that cut-elimination is restored. In this note we adopt Bellin's proposed rule and give a direct proof of cut-elimination. Following the proof of cut-elimination we show that a categorical model of FILL in the basic dialectica category is also a LNL model of Benton and a full tensor model of Melliès' and Tabareau's tensorial logic. Lastly, we give a double-negation translation of linear logic into FILL that explicitly uses par in addition to tensor.

1 Introduction

A commonly held belief during the early history of linear logic was that the linear-connective par could not be incorporated into an intuitionistic linear logic. This belief was challenged when de Paiva gave a categorical understanding of Gödel's dialectica interpretation in terms of dialectica categories [8, 7]. Upon setting out on her investigation she initially believed that dialectica categories would end up being a model of intuitionistic logic, but to her surprise they are actually models of intuitionistic linear logic, containing the linear connectives: tensor, implication, additives, and exponentials. She then improved her models to capture both FILL and CLL. Furthermore, unlike other models at that time the units did not collapse into a single object.

Armed with this semantic insight de Paiva gave the first formalization of Full Intuitionistic Linear Logic (FILL) [7]. FILL is a sequent calculus with multiple conclusions in addition to multiple hypotheses. Logics of this type go back to Gentzen's work on the sequent calculi LK and LJ, and Maehara's work on LJ' [13, 21]. The sequents in these types of logics usually have the form $\Gamma \vdash \Delta$ where Γ and Δ are multisets of formulas. Sequents such as these are read as "the conjunction of the formulas in Γ imply the disjunction of the formulas in Δ". For a brief, but more complete history of logics with multiple conclusions see the introduction to [9].

Gentzen showed that to obtain intuitionistic logic one could start with the logic LK and then place a cardinality restriction on the right-hand side of sequents, however, this is not the only means of enforcing intuitionism. Maehara showed that in the propositional case one could simply place the cardinality restriction on the premise of the implication right rule, and leave all of the other rules of LK unrestricted. This restriction is sometimes called the Dragalin restriction, as it appeared in his AMS textbook [10]. The classical implication right rule has the form:

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta}\ \text{IMPR}$$

By placing the Dragalin restriction on the previous rule we obtain:

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ \text{IMPR}$$

de Paiva's first formalization of FILL used the Dragalin restriction, see [7] p. 58, but Schellinx showed that this restriction has the unfortunate consequence of breaking cut-elimination [19].

Later, Hyland and de Paiva gave an alternative formalization of FILL with the intention of regaining cut-elimination [11]. This new formalization lifted the Dragalin restriction by decorating sequents with a term assignment. Hypotheses were assigned variables, and the conclusions were assigned terms. Then using these terms one can track the use of hypotheses throughout a derivation. They proposed a new implication right rule:

$$\frac{\Gamma, x : A \vdash t : B, \Delta \qquad x \notin \mathsf{FV}(\Delta)}{\Gamma \vdash \lambda x.t : A \multimap B, \Delta}\ \text{IMPR}$$

Intuitionism is enforced in this rule by requiring that the variable being dischanged, x, is potentially free in only one term annotating a conclusion. Unfortunately, this formalization did not enjoy cut-elimination either.

Bierman was able to give a counterexample to cut-elimination [4]. As Bierman explains the problem was with the left rule for par. The original rule was as follows:

$$\frac{\Gamma, x : A \vdash \Delta \qquad \Gamma', y : B \vdash \Delta'}{\Gamma, \Gamma', z : A \parr B \vdash \mathsf{let}\, z\, \mathsf{be}\, (x \parr -)\, \mathsf{in}\, \Delta \mid \mathsf{let}\, z\, \mathsf{be}\, (- \parr y)\, \mathsf{in}\, \Delta'}\ \text{PARL}$$

In this rule the pattern variables x and y are bound in each term of Δ and Δ' respectively. Notice that the variable z becomes free in every term in Δ and Δ'. Bierman showed that this rule mixed with the restriction on implication right prevents the usual cut-elimination step that commutes cut with the left rule for par. The main idea behind the counterexample is that in the derivation before commuting the cut it is possible to discharge z using implication right, but after the cut is commuted past the left rule for par, the variable z becomes free in more than one conclusion, and thus, can no longer be discharged.

In the conclusion of Bierman's note he gives an alternate left rule for par that he attributes to Bellin. This new left-rule is as follows:

$$\frac{\Gamma, x : A \vdash \Delta \quad \Gamma', y : B \vdash \Delta'}{\Gamma, \Gamma', z : A \parr B \vdash \text{let-pat}\, z\, (x \parr -)\, \Delta \mid \text{let-pat}\, z\, (- \parr y)\, \Delta'}\ \text{ParL}$$

In this rule let-pat $z\,(x \parr -)\, t$ and let-pat $z\,(- \parr y)\, t'$ only let-bind z in t or t' if $x \in FV(t)$ or $y \in FV(t')$. Otherwise the terms are left unaltered. Bellin showed that by adopting this rule cut-elimination can be proven by reduction to the cut-elimination procedure for proof nets for multiplicative linear logic with the mix rule [1]. However, this is an indirect proof that requires the adoption of proof nets.

Contributions. In this paper our main contribution is to give a direct proof of cut-elimination for FILL with Bellin's proposed par-left rule (Section 3). A direct proof accomplishes two goals: the first is to complete the picture of FILL Hyland and de Paiva started, and the second is to view a direct proof of cut-elimination as a means of checking the correctness of the formulation of FILL given here. The latter point is important for future work. Following the proof of cut-elimination we show that the categorical model of FILL called $\mathsf{Dial}_2(\mathsf{Sets})$, the basic dialectica category, is also a linear/non-linear model of Benton (Section 4) and a full tensor model of Melliès' and Tabareau's tensor logic (Section 5). Finally, we give a double-negation translation of multi-conclusion linear logic into FILL (Seciton 5.1). Due to the complexities of working in $\mathsf{Dial}_2(\mathsf{Sets})$ we have formalized all of the constructions and proofs used in Section 4 and Section 5 in the Agda proof assistant[1].

[1]The Agda development can be found at `https://github.com/heades/cut-fill-agda`.

Related Work. The first formalization of FILL with cut-elimination was due to Braüner and de Paiva [5]. Their formalization can be seen as a linear version of LK with a sophisticated meta-level dependency tracking system. A proof of a FILL sequent in their formalization amounts to a classical derivation, π, invariant in what they call the FILL property:

- The hypothesis discharged by an application of the implication right rule in π is a dependency of the conclusion of the implication being introduced.

They were able to show that their formalization is sound, complete, and enjoys cut-elimination. In favor of the term assignment formalization given here over Braüner and de Paiva's formalization is that the dependency tracking system complicates both the definition of the logic and its use. However, one might conjecture that their system is more fundamental and hence more generalizable. It might be possible to prove cut-elimination of the term assignment formalization of FILL relative to Braüner and de Paiva's dependency tracking system by erasing the terms on conclusions and then tracking which variable is free in which conclusion. However, as we stated above a direct proof is more desirable than a relative one.

de Paiva and Pereira used annotations on the sequents of LK to arrive at full intuitionistic logic (FIL) with multiple conclusion that enjoys cut-elimination [9]. They annotate hypothesis with natural number indices, and conclusions with finite sets of indices. The sets of indices on conclusions correspond to the collection of the hypotheses that the conclusion depends on. Then they have a similar property to that of Braüner and de Paiva's formalization. In fact, the dependency tracking system is very similar to this formalization, but the dependency tracking has been collapsed into the object language instead of being at the meta-level.

Clouston et al. give both a deep inference calculus and a display calculus for FILL that admits cut-elimination [6]. Both of these systems are refinements of a larger one called bi-intuitionistic linear logic (BiLL). This logic contains every logical connective of FILL with the addition of the exclusion (or subtraction) connective. This connective can be defined categorically as the left-adjoint to par. Thus, exclusion is the dual to implication. A positive aspect to this work is that the resulting systems are annotation free, but at a price of obscurity. Deep inference and display calculi are harder to understand, and their system requires FILL to be defined as a refinement of a system with additional connectives. We show in this paper that such a refinement is unnecessary. In addition, a term assignment system is closer to traditional logic than deep inference and display calculi, and it is closer, through the lens of the Curry-Howard-Lambek correspondence, to a type theoretic understanding of FILL.

2 Full Intuitionistic Linear Logic (FILL)

In this section we give a brief description of FILL. We first give the syntax of formulas, patterns, terms, and contexts. Following the syntax we define several meta-functions that will be used when defining the inference rules of the logic.

Definition 1. *The syntax for FILL is as follows:*

$$\begin{array}{rl}
\textit{(Formulas)} & A, B, C, D, E ::= \top \mid \perp \mid A \multimap B \mid A \otimes B \mid A \parr B \\
\textit{(Patterns)} & p ::= * \mid - \mid x \mid p_1 \otimes p_2 \mid p_1 \parr p_2 \\
\textit{(Terms)} & t, e ::= x \mid * \mid \circ \mid t_1 \otimes t_2 \mid t_1 \parr t_2 \mid \lambda x.t \mid \mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in}\, e \mid t_1\, t_2 \\
\textit{(Left Contexts)} & \Gamma ::= \cdot \mid x : A \mid \Gamma_1, \Gamma_2 \\
\textit{(Right Contexts)} & \Delta ::= \cdot \mid t : A \mid \Delta_1, \Delta_2
\end{array}$$

The formulas of FILL are standard, but we denote the unit of tensor as $\top$ and the unit of par as $\perp$. Patterns are used to distinguish between the various let-expressions for tensor, par, and their units. There are three different let-expressions:

Tensor:	Par:	Tensor Unit:
$\mathsf{let}\, t\, \mathsf{be}\, p_1 \otimes p_2\, \mathsf{in}\, e$	$\mathsf{let}\, t\, \mathsf{be}\, p_1 \parr p_2\, \mathsf{in}\, e$	$\mathsf{let}\, t\, \mathsf{be}\, *\, \mathsf{in}\, e$

$$\frac{}{x : A \vdash x : A}\ \textsc{Ax} \qquad \frac{\Gamma \vdash t : A \mid \Delta \quad \Gamma', y : A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta \mid [t/y]\Delta'}\ \textsc{Cut} \qquad \frac{\Gamma \vdash \Delta}{\Gamma, x : \top \vdash \mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in}\, \Delta}\ \textsc{Tl} \qquad \frac{}{\cdot \vdash * : \top}\ \textsc{Tr}$$

$$\frac{\Gamma, x : A, y : B \vdash \Delta}{\Gamma, z : A \otimes B \vdash \mathsf{let}\, z\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, \Delta}\ \textsc{Tenl} \qquad \frac{\Gamma \vdash e : A \mid \Delta \quad \Gamma' \vdash f : B \mid \Delta'}{\Gamma, \Gamma' \vdash e \otimes f : A \otimes B \mid \Delta \mid \Delta'}\ \textsc{Tenr} \qquad \frac{}{x : \perp \vdash \cdot}\ \textsc{Pl} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \circ : \perp \mid \Delta}\ \textsc{Pr}$$

$$\frac{\Gamma, x : A \vdash \Delta \quad \Gamma', y : B \vdash \Delta'}{\Gamma, \Gamma', z : A \parr B \vdash \text{let-pat}\, z\, (x \parr -)\, \Delta \mid \text{let-pat}\, z\, (- \parr y)\, \Delta'}\ \textsc{Parl} \qquad \frac{\Gamma \vdash \Delta \mid e : A \mid f : B \mid \Delta'}{\Gamma \vdash \Delta \mid e \parr f : A \parr B \mid \Delta'}\ \textsc{Parr}$$

$$\frac{\Gamma \vdash e : A \mid \Delta \quad \Gamma', x : B \vdash \Delta'}{\Gamma, y : A \multimap B, \Gamma' \vdash \Delta \mid [y\; e/x]\Delta'}\ \textsc{Impl} \qquad \frac{\Gamma, x : A \vdash e : B \mid \Delta \quad x \notin \mathsf{FV}(\Delta)}{\Gamma \vdash \lambda x.e : A \multimap B \mid \Delta}\ \textsc{Impr} \qquad \frac{\Gamma, x : A, y : B \vdash \Delta}{\Gamma, y : B, x : A \vdash \Delta}\ \textsc{Exl}$$

$$\frac{\Gamma \vdash \Delta_1 \mid t_1 : A \mid t_2 : B \mid \Delta_2}{\Gamma \vdash \Delta_1 \mid t_2 : B \mid t_1 : A \mid \Delta_2}\ \textsc{Exr}$$

Figure 1: Inference rules for FILL

In addition, each of these will have their own equational rules, see Figure 2. The role each term plays in the overall logic will become clear after we introduce the inference rules.

At this point we introduce some syntax and meta-level functions that will be used in the definition of the inference rules for FILL. Left contexts are multisets of formulas labeled with a variable, and right contexts are multisets of formulas labeled with a term. We will often write $\Delta_1 \mid \Delta_2$ as syntactic sugar for Δ_1, Δ_2. The former should be read as "Δ_1 or Δ_2." We denote the usual capture-avoiding substitution by $[t/x]t'$, and its straightforward extension to right contexts as $[t/x]\Delta$. Similarly, we find it convenient to be able to do this style of extension for the let-binding as well.

Definition 2. *We extend let-binding terms to right contexts as follows:*

$$\begin{array}{l}
\mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in} \cdot = \cdot \\
\mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in}\, (t' : A) = (\mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in}\, t') : A \\
\mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in}\, (\Delta_1 \mid \Delta_2) = (\mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in}\, \Delta_1) \mid (\mathsf{let}\, t\, \mathsf{be}\, p\, \mathsf{in}\, \Delta_2)
\end{array}$$

Lastly, we denote the usual function that computes the set of free variables in a term by $\mathsf{FV}(t)$, and its straightforward extension to right contexts as $\mathsf{FV}(\Delta)$.

The inference rules for FILL are defined in Figure 1. The Parl rule depends on the function let-pat $z\, p\, \Delta$ which we define next.

Definition 3. *The function* let-pat $z\, p\, t$ *is defined as follows:*

$$\begin{array}{ccc}
\text{let-pat}\, z\, (x \parr -)\, t = t & \text{let-pat}\, z\, (- \parr y)\, t = t & \text{let-pat}\, z\, p\, t = \mathsf{let}\, z\, \mathsf{be}\, p\, \mathsf{in}\, t \\
\textit{where } x \notin \mathsf{FV}(t) & \textit{where } y \notin \mathsf{FV}(t) &
\end{array}$$

It is straightforward to extend the previous definition to right-contexts, and we denote this extension by let-pat $z\, p\, \Delta$*.*

The motivation behind this function is that it only binds the pattern variables in $x \parr -$ and $- \parr y$ if and only if those pattern variables are free in the body of the let. This overcomes the counterexample given by Bierman in [4].

The terms of FILL are equipped with an equivalence relation defined in Figure 2. There are a number of α, β, and η like rules as well as several rules we call naturality rules. These rules are similar to the rules presented in [11].

3 Cut-elimination

FILL can be viewed from two different angles: i. as an intuitionistic linear logic with par, or ii. as a restricted form of classical linear logic. Thus, to prove cut-elimination of FILL one only needs to start with

$$\frac{y \notin \mathsf{FV}(t)}{t = [y/x]t}\ \textsc{Alpha} \qquad \frac{x \notin \mathsf{FV}(f)}{(\lambda x.f\ x) = f}\ \textsc{EtaFun} \qquad \frac{}{(\lambda x.e)\ e' = [e'/x]e}\ \textsc{BetaFun} \qquad \frac{}{\mathsf{let}\ *\ \mathsf{be}\ *\ \mathsf{in}\ e = e}\ \textsc{Eta1I}$$

$$\frac{y \notin \mathsf{FV}(f)}{\mathsf{let}\ y\ \mathsf{be}\ *\ \mathsf{in}\ f = f}\ \textsc{Eta2I} \qquad \frac{}{\mathsf{let}\ u\ \mathsf{be}\ *\ \mathsf{in}\ [*/z]f = [u/z]f}\ \textsc{BetaI} \qquad \frac{}{[\mathsf{let}\ u\ \mathsf{be}\ *\ \mathsf{in}\ e/y]f = \mathsf{let}\ u\ \mathsf{be}\ *\ \mathsf{in}\ [e/y]f}\ \textsc{NatI}$$

$$\frac{x, y \notin \mathsf{FV}(t)}{\mathsf{let}\ t'\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ t = t}\ \textsc{EtaTen} \qquad \frac{}{\mathsf{let}\ e \otimes t\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ u = [e/x, t/y]u}\ \textsc{Beta1Ten}$$

$$\frac{}{\mathsf{let}\ u\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ [x \otimes y/z]f = [u/z]f}\ \textsc{Beta2Ten} \qquad \frac{}{[\mathsf{let}\ u\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ g/w]f = \mathsf{let}\ u\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ [g/w]f}\ \textsc{NatTen}$$

$$\frac{}{u = \circ}\ \textsc{EtaParU} \qquad \frac{}{(\mathsf{let}\ u\ \mathsf{be}\ x \parr -\ \mathsf{in}\ x) \parr (\mathsf{let}\ u\ \mathsf{be}\ - \parr y\ \mathsf{in}\ y) = u}\ \textsc{EtaPar} \qquad \frac{}{\mathsf{let}\ u \parr t\ \mathsf{be}\ x \parr -\ \mathsf{in}\ e = [u/x]e}\ \textsc{Beta1Par}$$

$$\frac{}{\mathsf{let}\ u \parr t\ \mathsf{be}\ - \parr y\ \mathsf{in}\ e = [t/y]e}\ \textsc{Beta2Par} \qquad \frac{}{\mathsf{let}\ t\ \mathsf{be}\ x \parr -\ \mathsf{in}\ [u/x]f = [\mathsf{let}\ t\ \mathsf{be}\ x \parr -\ \mathsf{in}\ u/x]f}\ \textsc{Nat1Par}$$

$$\frac{}{\mathsf{let}\ t\ \mathsf{be}\ - \parr y\ \mathsf{in}\ [v/y]f = [\mathsf{let}\ t\ \mathsf{be}\ - \parr y\ \mathsf{in}\ v/y]f}\ \textsc{Nat2Par} \qquad \frac{t = t'}{\lambda x.t = \lambda x.t''}\ \textsc{Lam} \qquad \frac{t_1 = t_1'}{t_1\ t_2 = t_1'\ t_2}\ \textsc{App1}$$

$$\frac{t_2 = t_2'}{t_1\ t_2 = t_1\ t_2'}\ \textsc{App2} \qquad \frac{t_1 = t_1'}{t_1 \otimes t_2 = t_1' \otimes t_2}\ \textsc{Ten1} \qquad \frac{t_2 = t_2'}{t_1 \otimes t_2 = t_1 \otimes t_2'}\ \textsc{Ten2} \qquad \frac{t_1 = t_1'}{t_1 \parr t_2 = t_1' \parr t_2}\ \textsc{Par1}$$

$$\frac{t_2 = t_2'}{t_1 \parr t_2 = t_1 \parr t_2'}\ \textsc{Par2} \qquad \frac{t = t'}{\mathsf{let}\ t\ \mathsf{be}\ p\ \mathsf{in}\ e = \mathsf{let}\ t'\ \mathsf{be}\ p\ \mathsf{in}\ e}\ \textsc{Let1} \qquad \frac{e = e'}{\mathsf{let}\ t\ \mathsf{be}\ p\ \mathsf{in}\ e = \mathsf{let}\ t\ \mathsf{be}\ p\ \mathsf{in}\ e'}\ \textsc{Let2} \qquad \frac{}{t = t}\ \textsc{Refl}$$

$$\frac{t = t'}{t' = t}\ \textsc{Sym} \qquad \frac{t_1 = t_2 \quad t_2 = t_3}{t_1 = t_3}\ \textsc{Trans}$$

Figure 2: Equivalence on terms

the cut-elimination procedure for intuitionistic linear logic, and then dualize all of the steps in the procedure for tensor and its unit to obtain the steps for par and its unit. Similarly, one could just as easily start with the cut-elimination procedure for classical linear logic, and then apply the restriction on the implication right rule producing a cut-elimination procedure for FILL.

The major difference between proving cut-elimination of FILL from classical or intuitionistic linear logic is that we must prove an invariant across each step in the procedure. The invariant is that if a derivation π is transformed into a derivation π', then the terms in the conclusion of the final rule applied in π must be be transformable, when the equivalences defined in Figure 2 are taken as left-to-right rewrite rules, into the terms in the conclusion of the final rule applied in π'.

The cut elimination procedure requires the following two basic results:

Lemma 4 (Substitution Distribution)**.** *For any terms t, t_1, and t_2,* $[t_1/x][t_2/y]t = [[t_1/x]t_2/y][t_2/x]t$.

Proof. This proof holds by straightforward induction on the form of t. □

Lemma 5 (Let-pat Distribution)**.** *For any terms t, t_1, and t_2, and pattern p,*
let-pat $t\ p\ [t_1/y]t_2 = [\text{let-pat}\ t\ p\ t_1/y]t_2$.

Proof. This proof holds by case splitting over p, and then using the naturality equations for the respective pattern. □

We finally arrive at cut-elimination.

Theorem 6. *If $\Gamma \vdash t_1 : A_1, \ldots, t_i : A_i$ steps to $\Gamma \vdash t_1' : A_1, \ldots, t_i' : A_i$ using the cut-elimination procedure, then $t_j = t_j'$ for $1 \leq j \leq i$.*

Proof. The cut-elimination procedure given here is the standard cut-elimination procedure for classical linear logic except the cases involving the implication right rule have the FILL restriction. The structure of our procedure follows the structure of the procedure found in [14]. Throughout this proof we treat the equivalences defined in Figure 2 as left-to-right rewrite rules.

Case: commuting conversion cut vs cut (first case). The following proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_2, x : A, \Gamma_3 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_1, y : B, \Gamma_4 \vdash \Delta_2}}{\Gamma_1, \Gamma_2, x : A, \Gamma_3, \Gamma_4 \vdash \Delta_1 \mid [t_1/y]\Delta_2}\,\textsc{Cut}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, \Gamma_4 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x][t_1/y]\Delta_2}\,\textsc{Cut}$$

is transformed into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_2, x : A, \Gamma_3 \vdash t_1 : B \mid \Delta_1}}{\Gamma_2, \Gamma, \Gamma_3 \vdash [t/x]t_1 : B \mid [t/x]\Delta_1} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_1, y : B, \Gamma_4 \vdash \Delta_2}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, \Gamma_4 \vdash \Delta \mid [t/x]\Delta_1 \mid [[t/x]t_1/y]\Delta_2}\,\textsc{Cut}$$

First, if Δ_2 is empty, then all the terms in the conclusion of the previous two derivations are equivalent. So suppose $\Delta_2 = t_2 : C \mid \Delta_2'$. Then we know that the term $[t/x][t_1/y]t_2$ in the first derivation above is equivalent to $[[t/x]t_1/y][t/x]t_2$ by Lemma 4. Furthermore, by inspecting the first derivation we can see that $x \notin \mathsf{FV}(t_2)$, and thus, $[[t/x]t_1/y][t/x]t_2 = [[t/x]t_1/y]t_2$. This argument may be repeated for any term in Δ_2', and thus, we know $[t/x][t_1/y]\Delta_2 = [[t/x]t_1/y]\Delta_2$.

Case: commuting conversion cut vs. cut (second case). The second commuting conversion on cut begins with the proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma' \vdash t' : B \mid \Delta'} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2, y : B, \Gamma_3 \vdash \Delta_1}}{\Gamma_1, x : A, \Gamma_2, \Gamma', \Gamma_3 \vdash \Delta' \mid [t'/y]\Delta_1}\,\textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma', \Gamma_3 \vdash \Delta \mid [t/x]\Delta' \mid [t/x][t'/y]\Delta_1}\,\textsc{Cut}$$

is transformed into the following proof:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma' \vdash t' : B \mid \Delta'} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2, y : B, \Gamma_3 \vdash \Delta_1}}{\Gamma_1, \Gamma, \Gamma_2, y : B, \Gamma_3 \vdash \Delta \mid [t/x]\Delta_1}\,\textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma', \Gamma_3 \vdash \Delta' \mid [t'/y]\Delta \mid [t'/y][t/x]\Delta_1}\,\textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma', \Gamma_3 \vdash [t'/y]\Delta \mid \Delta' \mid [t'/y][t/x]\Delta_1}\,\textsc{Series of Exchanges}$$

We know $x, y \notin \mathsf{FV}(\Delta)$ by inspection of the first derivation, and so we know that $\Delta = [t'/y]\Delta$ and $\Delta' = [t/x]\Delta'$. Without loss of generality suppose $\Delta_1 = t_1 : C \mid \Delta_1'$. Then we know that $x, y \notin \mathsf{FV}(t)$ and $x, y \notin \mathsf{FV}(t')$. Thus, by this fact and Lemma 4, we know that $[t/x][t'/y]t_1 = [[t/x]t'/y][t/x]t_1 = [t'/y][t/x]t_1$. This argument can be repeated for any term in Δ_1', hence, $[t/x][t'/y]\Delta_1 = [t'/y][t/x]\Delta_1$.

Case: the η-expansion cases: tensor. The proof

$$\dfrac{}{x : A \otimes B \vdash x : A \otimes B}\ \textsc{Ax}$$

is transformed into the proof

$$\dfrac{\dfrac{}{y : A \vdash y : A}\ \textsc{Ax} \qquad \dfrac{}{z : B \vdash z : B}\ \textsc{Ax}}{\dfrac{y : A, z : B \vdash y \otimes z : A \otimes B}{x : A \otimes B \vdash \mathsf{let}\ x\ \mathsf{be}\ y \otimes z\ \mathsf{in}\ (y \otimes z) : A \otimes B}\ \textsc{Tl}}\ \textsc{Tr}$$

By the rule EQ_EtaTensor we know $\mathsf{let}\ x\ \mathsf{be}\ y \otimes z\ \mathsf{in}\ (y \otimes z) = x$.

Case: the η-expansion cases: par. The proof

$$\dfrac{}{x : A \parr B \vdash x : A \parr B}\ \textsc{Ax}$$

is transformed into the proof

$$\dfrac{\dfrac{\dfrac{}{y : A \vdash y : A}\ \textsc{Ax} \qquad \dfrac{}{z : B \vdash z : B}\ \textsc{Ax}}{x : A \parr B \vdash \mathsf{let}\ x\ \mathsf{be}\ (y \parr -)\ \mathsf{in}\ y : A \mid \mathsf{let}\ x\ \mathsf{be}\ (- \parr z)\ \mathsf{in}\ z : B}\ \textsc{ParL}}{x : A \parr B \vdash (\mathsf{let}\ x\ \mathsf{be}\ (y \parr -)\ \mathsf{in}\ y) \parr (\mathsf{let}\ x\ \mathsf{be}\ (- \parr z)\ \mathsf{in}\ z) : A \parr B}\ \textsc{ParR}$$

By rule Eq_EtaPar we know $((\mathsf{let}\ x\ \mathsf{be}\ (y \parr -)\ \mathsf{in}\ y) \parr (\mathsf{let}\ x\ \mathsf{be}\ (- \parr z)\ \mathsf{in}\ z)) = x$.

Case: the η-expansion cases: implication. The proof

$$\dfrac{}{x : A \multimap B \vdash x : A \multimap B}\ \textsc{Ax}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{}{y : A \vdash y : A}\ \textsc{Ax} \qquad \dfrac{}{z : B \vdash z : B}\ \textsc{Ax}}{y : A, x : A \multimap B \vdash x\ y : B}\ \textsc{ImpL}}{x : A \multimap B \vdash \lambda y.x\ y : A \multimap B}\ \textsc{ImpR}$$

All terms in the two derivations are equivalent, because $(\lambda y.x\ y) = x$ by the Eq_EtaFun rule.

Case: the η-expansion cases: tensor unit. The proof

$$\dfrac{}{x : \top \vdash x : \top}\ \textsc{Ax}$$

transforms into the proof

$$\dfrac{\dfrac{}{\cdot \vdash * : \top}\ \textsc{Ir}}{x : \top \vdash \mathsf{let}\ x\ \mathsf{be}\ *\ \mathsf{in}\ * : \top}\ \textsc{Il}$$

We know $x = \mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in}\, *$ by EQ_ETAI.

Case: the η-expansion cases: par unit. The proof

$$\dfrac{}{x :\perp\vdash x :\perp}\ \textsc{Ax}$$

transforms into the proof

$$\dfrac{\dfrac{}{x :\perp\vdash \cdot}\ \textsc{Pl}}{x :\perp\vdash \circ :\perp}\ \textsc{Pr}$$

We know $x = \circ$ by EQ_ETAPARU.

Case: the axiom steps: the axiom step. The proof

$$\dfrac{\dfrac{}{x : A \vdash x : A}\ \textsc{Ax} \qquad \dfrac{\pi \atop \vdots}{\Gamma_1, y : A, \Gamma_2 \vdash \Delta}}{\Gamma_1, x : A, \Gamma_2 \vdash [x/y]\Delta}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\pi \atop \vdots}{\Gamma_1, y : A, \Gamma_2 \vdash \Delta}$$

By EQ_ALPHA, we know, for any t in Δ, $t = [x/y]t$, and hence $\Delta = [x/y]\Delta$.

Case: the axiom steps: conclusion vs. axiom. The proof

$$\dfrac{\dfrac{\pi \atop \vdots}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{}{x : A \vdash x : A}\ \textsc{Ax}}{\Gamma \vdash \Delta \mid [t/x]x : A}\ \textsc{Cut}$$

transforms into

$$\dfrac{\dfrac{\pi \atop \vdots}{\Gamma \vdash t : A \mid \Delta}}{\Gamma \vdash \Delta \mid t : A}\ \textsc{Series of Exchanges}$$

By the definition of the substitution function we know $t = [t/x]x$.

Case: the exchange steps: conclusion vs. left-exchange (the first case). The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, y : B, \Gamma_2 \vdash \Delta'}}{\Gamma_1, y : B, x : A, \Gamma_2 \vdash \Delta'}\,\textsc{Exl}}{\Gamma_1, y : B, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta'}\,\textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, y : B, \Gamma_2 \vdash \Delta'}}{\Gamma_1, \Gamma, y : B, \Gamma_2 \vdash \Delta \mid [t/x]\Delta'}\,\textsc{Cut}}{\Gamma_1, y : B, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta'}\,\textsc{Series of Exchanges}$$

Clearly, all terms are equivalent.

Case: the exchange steps: conclusion vs. left-exchange (the second case). The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : B \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, y : B, \Gamma_2 \vdash \Delta'}}{\Gamma_1, y : B, x : A, \Gamma_2 \vdash \Delta'}\,\textsc{Exl}}{\Gamma_1, \Gamma, x : A, \Gamma_2 \vdash \Delta \mid [t/y]\Delta'}\,\textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : B \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, y : B, \Gamma_2 \vdash \Delta'}}{\Gamma_1, x : A, \Gamma, \Gamma_2 \vdash \Delta \mid [t/y]\Delta'}\,\textsc{Cut}}{\Gamma_1, \Gamma, x : A, \Gamma_2 \vdash \Delta \mid [t/y]\Delta'}\,\textsc{Series of Exchanges}$$

Clearly, all terms are equivalent.

Case: the exchange steps: conclusion vs. right-exchange The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 : B \mid t_2 : C \mid \Delta'}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_2 : C \mid t_1 : B \mid \Delta'}\,\textsc{Exr}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_2 : C \mid [t/x]t_1 : B \mid [t/x]\Delta'}\,\textsc{Cut}$$

transforms into this proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 : B \mid t_2 : C \mid \Delta'}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_1 : B \mid [t/x]t_2 : C \mid [t/x]\Delta'}\,\textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2 \vdash [t/x]\Delta_1 \mid [t/x]t_2 : C \mid [t/x]t_1 : B \mid [t/x]\Delta'}\,\textsc{Exr}$$

Clearly, all terms are equivalent.

Case: principal formula vs. principal formula: tensor. The proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma_1 \vdash t_1 : A \mid \Delta_1} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_2 \vdash t_2 : B \mid \Delta_2}}{\Gamma_1, \Gamma_2 \vdash t_1 \otimes t_2 : A \otimes B \mid \Delta_1 \mid \Delta_2}\,\textsc{Tr} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, x : A, y : B, \Gamma_4 \vdash \Delta_3}}{\Gamma_3, z : A \otimes B, \Gamma_4 \vdash \mathsf{let}\, z\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, \Delta_3}\,\textsc{Tl}}{\Gamma_3, \Gamma_1, \Gamma_2, \Gamma_4 \vdash \Delta_1 \mid \Delta_2 \mid [t_1 \otimes t_2/z](\mathsf{let}\, z\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, \Delta_3)}\,\textsc{Cut}$$

is transformed into the proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma_1 \vdash t_1 : A \mid \Delta_1} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_2 \vdash t_2 : B \mid \Delta_2} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, x : A, y : B, \Gamma_4 \vdash \Delta_3}}{\Gamma_3, x : A, \Gamma_2, \Gamma_4 \vdash \Delta_2 \mid [t_2/y]\Delta_3}\,\textsc{Cut}}{\Gamma_3, \Gamma_1, \Gamma_2, \Gamma_4 \vdash \Delta_1 \mid \Delta_2 \mid [t_1/x][t_2/y]\Delta_3}\,\textsc{Cut}$$

Without loss of generality suppose $\Delta_3 = t_3 : C, \Delta_3'$. We can see that $[t_1 \otimes t_2/z](\mathsf{let}\, z\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, t_3) = \mathsf{let}\, t_1 \otimes t_2\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, t_3$ by the definition of substitution, and by using the $\textsc{Eq\_Beta1Tensor}$ rule we obtain $\mathsf{let}\, t_1 \otimes t_2\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, t_3 = [t_1/x][t_2/y]t_3$. This argument can be repeated for any term in $[t_1 \otimes t_2/z](\mathsf{let}\, z\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, \Delta_3')$, and thus, $[t_1 \otimes t_2/z](\mathsf{let}\, z\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, \Delta_3) = [t_1/x][t_2/y]\Delta_3$.

Note that in the second derivation of the above transformation we first cut on B, and then A, but we could have cut on A first, and then B, but this would yeild equivalent derivations as above by using Lemma 4.

Case: principal formula vs. principal formula: par. The proof

$$\dfrac{\textsc{ParR}\,\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma_1 \vdash \Delta_1 \mid t_1 : A \mid t_2 : B \mid \Delta_2}}{\Gamma_1 \vdash \Delta_1 \mid t_1 \parr t_2 : A \parr B \mid \Delta_2} \quad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_2, x : A \vdash \Delta_3} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, y : B \vdash \Delta_4}}{\Gamma_2, \Gamma_3, z : A \parr B \vdash \text{let-pat}\, z\, (x \parr -)\, \Delta_3 \mid \text{let-pat}\, z\, (- \parr y)\, \Delta_4}\,\textsc{ParL}}{\Gamma_2, \Gamma_3, \Gamma_1 \vdash \Delta_1 \mid \Delta_2 \mid [t_1 \parr t_2/z](\text{let-pat}\, z\, (x \parr -)\, \Delta_3) \mid [t_1 \parr t_2/z](\text{let-pat}\, z\, (- \parr y)\, \Delta_4)}\,\textsc{Cut}$$

is transformed into the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma_1 \vdash \Delta_1 \mid t_1 : A \mid t_2 : B \mid \Delta_2} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, y : B \vdash \Delta_4}}{\Gamma_3, \Gamma_1 \vdash \Delta_1 \mid t_1 : A \mid \Delta_2 \mid [t_2/y]\Delta_4}\,\textsc{Cut} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_2, x : A \vdash \Delta_3}}{\Gamma_2, \Gamma_3, \Gamma_1 \vdash \Delta_1 \mid \Delta_2 \mid [t_2/y]\Delta_4 \mid [t_1/x]\Delta_3}\,\textsc{Cut}}{\Gamma_2, \Gamma_3, \Gamma_1 \vdash \Delta_1 \mid \Delta_2 \mid [t_1/x]\Delta_3 \mid [t_2/y]\Delta_4}\,\textsc{Series of Exchanges}$$

Without loss of generality consider the case when $\Delta_3 = t_3 : C_1 \mid \Delta_3'$ and $\Delta_4 = t_4 : C_2 \mid \Delta_4'$. First, $[t_1 \parr t_2/z](\text{let-pat}\, z\, (x \parr -)\, t_3) = \text{let-pat}\, (t_1 \parr t_2)\, (x \parr -)\, t_3$, and by $\textsc{Eq\_Beta1Par}$ we know $\text{let-pat}\, (t_1 \parr t_2)\, (x \parr -)\, t_3 = [t_1/x]t_3$ if $x \in \mathsf{FV}(t_3)$ or $\text{let-pat}\, (t_1 \parr t_2)\, (x \parr -)\, t_3 = t_3$ otherwise. In the latter case we can see that $t_3 = [t_1/x]t_3$, thus, in both cases $\text{let-pat}\, (t_1 \parr t_2)\, (x \parr -)\, t_3 = [t_1/x]t_3$. This argument can be repeated for any terms in Δ_3', and hence $[t_1 \parr t_2/z](\text{let-pat}\, z\, (x \parr -)\, \Delta_3) = \text{let-pat}\, (t_1 \parr t_2)\, (x \parr -)\, \Delta_3 = [t_1/x]\Delta_3$. We can apply a similar argument for
$[t_1 \parr t_2/z](\text{let-pat}\, z\, (- \parr y)\, t_4)$ and $[t_1 \parr t_2/z](\text{let-pat}\, z\, (- \parr y)\, \Delta_4)$.

Note that we could have first cut on A, and then on B in the second derivation, but we would have arrived at the same result just with potentially more exchanges on the right. Note that just as we mentioned about tensor we could have first cut on A, and then on B in the second derivation, but we would have arrived at the same result just with potentially more exchanges on the right.

Case: principal formula vs. principal formula: implication. The proof

$$\dfrac{\dfrac{\dfrac{\overset{\pi_1}{\vdots}}{\Gamma, x : A \vdash t : B \mid \Delta} \qquad x \notin \mathsf{FV}(\Delta)}{\Gamma \vdash \lambda x.t : A \multimap B \mid \Delta}\;\textsc{ImpR} \qquad \dfrac{\dfrac{\overset{\pi_2}{\vdots}}{\Gamma_1 \vdash t_1 : A \mid \Delta_1} \qquad \dfrac{\overset{\pi_3}{\vdots}}{\Gamma_2, y : B \vdash \Delta_2}}{\Gamma_1, z : A \multimap B, \Gamma_2 \vdash \Delta_1 \mid [z\, t_1/y]\Delta_2}\;\textsc{ImpL}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [\lambda x.t/z]\Delta_1 \mid [\lambda x.t/z][z\, t_1/y]\Delta_2}\;\textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\overset{\pi_2}{\vdots}}{\Gamma_1 \vdash t_1 : A \mid \Delta_1} \quad \dfrac{\overset{\pi_1}{\vdots}}{\Gamma, x : A \vdash t : B \mid \Delta} \quad x \notin \mathsf{FV}(\Delta)}{\Gamma, \Gamma_1 \vdash \Delta_1 \mid [t_1/x]t : B \mid [t_1/x]\Delta}\;\textsc{Cut} \qquad \dfrac{\overset{\pi_3}{\vdots}}{\Gamma_2, y : B \vdash \Delta_2}}{\Gamma_2, \Gamma, \Gamma_1 \vdash \Delta_1 \mid [t_1/x]\Delta \mid [[t_1/x]t/y]\Delta_2}\;\textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2 \vdash [t_1/x]\Delta \mid \Delta_1 \mid [[t_1/x]t/y]\Delta_2}\;\textsc{Series of Exchanges}$$

Without loss of generality consider the case when $\Delta_2 = t_2 : C \mid \Delta_2'$. First, by hypothesis we know $x \notin \mathsf{FV}(\Delta)$, and so we know $\Delta = [t_1/x]\Delta$. We can see that $[\lambda x.t/z][z\, t_1/y]t_2 = [(\lambda x.t)\, t_1/y]t_2 = [[t_1/x]t/y]t_2$ by using the congruence rules of equality and the rule $\textsc{Eq\_BetaFun}$. This argument can be repeated for any term in $[\lambda x.t/z][z\, t_1/y]\Delta_2'$, and so $[\lambda x.t/z][z\, t_1/y]\Delta_2 = [[t_1/x]t/y]\Delta_2$. Finally, by inspecting the previous derivations we can see that $z \notin \mathsf{FV}(\Delta_1)$, and thus, $\Delta_1 = [\lambda x.t/z]\Delta_1$.

Case: principal formula vs. principal formula: tensor unit. The proof

$$\dfrac{\dfrac{}{\cdot \vdash * : \top}\;\textsc{Ir} \qquad \dfrac{\dfrac{\overset{\pi}{\vdots}}{\Gamma \vdash \Delta}}{\Gamma, x : \top \vdash \mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in}\, \Delta}\;\textsc{Il}}{\Gamma \vdash [*/x](\mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in}\, \Delta)}\;\textsc{Cut}$$

is transformed into the proof

$$\dfrac{\overset{\pi}{\vdots}}{\Gamma \vdash \Delta}$$

Without loss of generality suppose $\Delta = t : A \mid \Delta'$. We can see that $[*/x](\mathsf{let}\, x\, \mathsf{be} * \mathsf{in}\, t) = \mathsf{let} * \mathsf{be} * \mathsf{in}\, t = t$ by the definition of substitution and the $\textsc{Eq\_EtaI}$ rule. This argument can be repeated for any term in $[*/x](\mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in}\, \Delta')$, and hence, $[*/x](\mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in}\, \Delta) = \Delta$.

Case: principal formula vs. principal formula: par unit. The proof

$$\dfrac{\dfrac{\dfrac{\overset{\pi}{\vdots}}{\Gamma \vdash \Delta}}{\Gamma \vdash \circ : \perp \mid \Delta}\;\textsc{Pr} \qquad \dfrac{}{x : \perp \vdash \cdot}\;\textsc{Pl}}{\Gamma \vdash \Delta \mid [\circ/x]\cdot}\;\textsc{Cut}$$

transforms into the proof

$$\dfrac{\begin{array}{c}\pi\\ \vdots\end{array}}{\Gamma \vdash \Delta}$$

Clearly, $[\circ/x]\cdot = \cdot$.

Case: secondary conclusion: left introduction of implication. The proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t_1 : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : B, \Gamma_2 \vdash t_2 : C \mid \Delta_2}}{\Gamma, y : A \multimap B, \Gamma_1, \Gamma_2 \vdash \Delta \mid [y\ t_1/x]t_2 : C \mid [y\ t_1/x]\Delta_2}\ \textsc{Impl} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, z : C, \Gamma_4 \vdash \Delta_3}}{\Gamma_3, \Gamma, y : A \multimap B, \Gamma_1, \Gamma_2, \Gamma_4 \vdash \Delta \mid [y\ t_1/x]\Delta_2 \mid [[y\ t_1/x]t_2/z]\Delta_3}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t_1 : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : B, \Gamma_2 \vdash t_2 : C \mid \Delta_2} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, z : C, \Gamma_4 \vdash \Delta_3}}{\Gamma_3, \Gamma_1, x : B, \Gamma_2, \Gamma_4 \vdash \Delta_2 \mid [t_2/z]\Delta_3}\ \textsc{Cut}}{\Gamma, y : A \multimap B, \Gamma_3, \Gamma_1, \Gamma_2, \Gamma_4 \vdash \Delta \mid [y\ t_1/x]\Delta_2 \mid [y\ t_1/x][t_2/z]\Delta_3}\ \textsc{Impl}}{\Gamma_3, \Gamma, y : A \multimap B, \Gamma_1, \Gamma_2, \Gamma_4 \vdash \Delta \mid [y\ t_1/x]\Delta_2 \mid [y\ t_1/x][t_2/z]\Delta_3}\ \textsc{Series of Exchanges}$$

This case holds because we can prove that $[y\ t_1/x][t_2/z]\Delta_3 = [[y\ t_1/x]t_2/z]\Delta_3$ by Lemma 4 and the fact that $x \notin \mathsf{FV}(\Delta_3)$.

Case: secondary conclusion: left introduction of exchange. The proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma, y : B, x : A, \Gamma' \vdash t : C \mid \Delta}}{\Gamma, x : A, y : B, \Gamma' \vdash t : C \mid \Delta}\ \textsc{Exl} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, z : C, \Gamma_2 \vdash \Delta_2}}{\Gamma_1, \Gamma, x : A, y : B, \Gamma', \Gamma_2 \vdash \Delta \mid [t/z]\Delta_2}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma, y : B, x : A, \Gamma' \vdash t : C \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, z : C, \Gamma_2 \vdash \Delta_2}}{\Gamma_1, \Gamma, y : B, x : A, \Gamma', \Gamma_2 \vdash \Delta \mid [t/z]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma, x : A, y : B, \Gamma', \Gamma_2 \vdash \Delta \mid [t/z]\Delta_2}\ \textsc{Exl}$$

Clearly, all terms are equivalent.

Case: secondary conclusion: left introduction of tensor. The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma, x : A, y : B \vdash t : C \mid \Delta\end{array}}{\Gamma, z : A \otimes B \vdash \mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ t : C \mid \mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ \Delta}\ \textsc{Tl} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, w : C, \Gamma_2 \vdash \Delta_2\end{array}}{\Gamma_1, \Gamma, z : A \otimes B, \Gamma_2 \vdash \mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ \Delta \mid [\mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ t/w]\Delta_2}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma, x : A, y : B \vdash t : C \mid \Delta\end{array} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, w : C, \Gamma_2 \vdash \Delta_2\end{array}}{\Gamma_1, \Gamma, x : A, y : B, \Gamma_2 \vdash \Delta \mid [t/w]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma, z : A \otimes B, \Gamma_2 \vdash \mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ \Delta \mid \mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ ([t/w]\Delta_2)}\ \textsc{Tl}$$

It suffices to show that $\mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ ([t/w]\Delta_2) = [\mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ t/w]\Delta_2$. This is a simple consequence of the rule Eq_NatTensor.

Case: secondary conclusion: left introduction of Par The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma, x : A \vdash \Delta\end{array} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma', y : B \vdash t' : C \mid \Delta'\end{array}}{\Gamma, \Gamma', z : A \parr B \vdash \text{let-pat}\ z\ (x \parr -)\ \Delta \mid \text{let-pat}\ z\ (- \parr y)\ t' : C \mid \text{let-pat}\ z\ (- \parr y)\ \Delta'}\ \textsc{Parl} \qquad \begin{array}{c}\pi_3\\ \vdots\\ \hline \Gamma_1, w : C, \Gamma_2 \vdash \Delta_2\end{array}}{\Gamma_1, \Gamma, \Gamma', z : A \parr B, \Gamma_2 \vdash \text{let-pat}\ z\ (x \parr -)\ \Delta \mid \text{let-pat}\ z\ (- \parr y)\ \Delta' \mid [\text{let-pat}\ z\ (- \parr y)\ t'/w]\Delta_2}\ \textsc{Cut}$$

is transformed into the proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma, x : A \vdash \Delta\end{array} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma', y : B \vdash t' : C \mid \Delta'\end{array} \qquad \begin{array}{c}\pi_3\\ \vdots\\ \hline \Gamma_1, w : C, \Gamma_2 \vdash \Delta_2\end{array}}{\Gamma_1, \Gamma', y : B, \Gamma_2 \vdash \Delta' \mid [t'/w]\Delta_2}\ \textsc{Cut}}{\Gamma, \Gamma_1, \Gamma', \Gamma_2, z : A \parr B \vdash \text{let-pat}\ z\ (x \parr -)\ \Delta \mid \text{let-pat}\ z\ (- \parr y)\ \Delta' \mid \text{let-pat}\ z\ (- \parr y)\ [t'/w]\Delta_2}\ \textsc{Parl}}{\Gamma_1, \Gamma, \Gamma', z : A \parr B, \Gamma_2 \vdash \text{let-pat}\ z\ (x \parr -)\ \Delta \mid \text{let-pat}\ z\ (- \parr y)\ \Delta' \mid \text{let-pat}\ z\ (- \parr y)\ [t'/w]\Delta_2}\ \textsc{Series of Exchanges}$$

It suffices to show that let-pat $z\ (- \parr y)\ [t'/w]\Delta_2 = [\text{let-pat}\ z\ (- \parr y)\ t'/w]\Delta_2$. This follows from the rule Eq_Nat2Par.

Case: secondary conclusion: left introduction of tensor unit. The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma \vdash t : C \mid \Delta\end{array}}{\Gamma, x : \top \vdash t : C \mid \Delta}\ \textsc{Il} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, w : C, \Gamma_2 \vdash \Delta_1\end{array}}{\Gamma_1, \Gamma, x : \top, \Gamma_2 \vdash \Delta \mid [t/w]\Delta_1}\ \textsc{Cut}$$

is transformed into the following:

$$
\begin{prooftree}
\AxiomC{π_1}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma \vdash t : C \mid \Delta$}
\AxiomC{π_2}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_1, w : C, \Gamma_2 \vdash \Delta_1$}
\RightLabel{\textsc{Cut}}
\BinaryInfC{$\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/w]\Delta_1$}
\RightLabel{\textsc{Il}}
\UnaryInfC{$\Gamma_1, \Gamma, \Gamma_2, x : \top \vdash \Delta \mid [t/w]\Delta_1$}
\RightLabel{\scriptsize\textsc{Series of Exchanges}}
\UnaryInfC{$\Gamma_1, \Gamma, x : \top, \Gamma_2 \vdash \Delta \mid [t/w]\Delta_1$}
\end{prooftree}
$$

Clearly, all terms are equivalent. Note that we do not give a case for secondary conclusion of the left introduction of par's unit, because it can only be introduced given an empty right context, and thus there is no cut formula.

Case: secondary hypothesis: left introduction of tensor. The proof

$$
\begin{prooftree}
\AxiomC{π_1}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma \vdash t : A \mid \Delta$}
\AxiomC{π_2}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_1, x : A, \Gamma_2, y : B, z : C, \Gamma_3 \vdash t_1 : D \mid \Delta_1$}
\RightLabel{\textsc{Tl}}
\UnaryInfC{$\Gamma_1, x : A, \Gamma_2, w : B \otimes C, \Gamma_3 \vdash \mathsf{let}\, w \,\mathsf{be}\, y \otimes z \,\mathsf{in}\, t_1 : D \mid \mathsf{let}\, w \,\mathsf{be}\, y \otimes z \,\mathsf{in}\, \Delta_1$}
\RightLabel{\textsc{Cut}}
\BinaryInfC{$\Gamma_1, \Gamma, \Gamma_2, w : B \otimes C, \Gamma_3 \vdash \Delta \mid [t/x](\mathsf{let}\, w \,\mathsf{be}\, y \otimes z \,\mathsf{in}\, t_1) : D \mid [t/x](\mathsf{let}\, w \,\mathsf{be}\, y \otimes z \,\mathsf{in}\, \Delta_1)$}
\end{prooftree}
$$

transforms into the proof

$$
\begin{prooftree}
\AxiomC{π_1}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma \vdash t : A \mid \Delta$}
\AxiomC{π_2}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_1, x : A, \Gamma_2, y : B, z : C, \Gamma_3 \vdash t_1 : D \mid \Delta_1$}
\RightLabel{\textsc{Cut}}
\BinaryInfC{$\Gamma_1, \Gamma, \Gamma_2, y : B, z : C, \Gamma_3 \vdash \Delta \mid [t/x]t_1 : D \mid [t/x]\Delta_1$}
\RightLabel{\textsc{Tl}}
\UnaryInfC{$\Gamma_1, \Gamma, \Gamma_2, w : B \otimes C, \Gamma_3 \vdash \mathsf{let}\, w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, \Delta \mid \mathsf{let}\, w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, [t/x]t_1 : D \mid \mathsf{let}\, w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, [t/x]\Delta_1$}
\end{prooftree}
$$

First, we can see by inspection of the previous derivations that $x, y \notin \mathsf{FV}(\Delta)$, thus, by using similar reasoning as above we can use the EtaTensor rule to obtain $\mathsf{let}\, w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, \Delta = \Delta$. It is a well-known property of substitution that $[t/x](\mathsf{let}\, w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, t_1) = \mathsf{let}\, [t/x]w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, [t/x]t_1 = \mathsf{let}\, w \,\mathsf{be}\, x \otimes y \,\mathsf{in}\, [t/x]t_1$.

Case: secondary hypothesis: right introduction of tensor (first case). The proof

$$
\begin{prooftree}
\AxiomC{π_1}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma \vdash t : A \mid \Delta$}
\AxiomC{π_2}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_1, x : A, \Gamma_2 \vdash t_1 : B \mid \Delta_1$}
\AxiomC{π_3}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_3 \vdash t_2 : C \mid \Delta_2$}
\RightLabel{\textsc{Tr}}
\BinaryInfC{$\Gamma_1, x : A, \Gamma_2, \Gamma_3 \vdash t_1 \otimes t_2 : B \otimes C \mid \Delta_1 \mid \Delta_2$}
\RightLabel{\textsc{Cut}}
\BinaryInfC{$\Gamma_1, \Gamma, \Gamma_2, \Gamma_3 \vdash \Delta \mid [t/x](t_1 \otimes t_2) : B \otimes C \mid [t/x]\Delta_1 \mid [t/x]\Delta_2$}
\end{prooftree}
$$

transforms into the proof

$$
\begin{prooftree}
\AxiomC{π_1}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma \vdash t : A \mid \Delta$}
\AxiomC{π_2}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_1, x : A, \Gamma_2 \vdash t_1 : B \mid \Delta_1$}
\RightLabel{\textsc{Cut}}
\BinaryInfC{$\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]t_1 : B \mid [t/x]\Delta_1$}
\AxiomC{π_3}
\noLine
\UnaryInfC{$\vdots$}
\UnaryInfC{$\Gamma_3 \vdash t_2 : C \mid \Delta_2$}
\RightLabel{\textsc{Tr}}
\BinaryInfC{$\Gamma_1, \Gamma, \Gamma_2, \Gamma_3 \vdash [t/x]t_1 \otimes t_2 : B \otimes C \mid \Delta \mid [t/x]\Delta_1 \mid \Delta_2$}
\RightLabel{\scriptsize\textsc{Series of Exchanges}}
\UnaryInfC{$\Gamma_1, \Gamma, \Gamma_2, \Gamma_3 \vdash \Delta \mid ([t/x]t_1) \otimes t_2 : B \otimes C \mid [t/x]\Delta_1 \mid \Delta_2$}
\end{prooftree}
$$

By inspection of the previous derivations we can see that $x \notin \mathsf{FV}(t_2)$ and $x \notin \mathsf{FV}(\Delta_2)$. Thus, $[t/x]\Delta_2 = \Delta_2$ and $[t/x](t_1 \otimes t_2) = ([t/x]t_1) \otimes ([t/x]t_2) = ([t/x]t_1) \otimes t_2$.

Case: secondary hypothesis: right introduction of tensor (second case). The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_2, x : A, \Gamma_3 \vdash t_2 : C \mid \Delta_2}}{\Gamma_1, \Gamma_2, x : A, \Gamma_3 \vdash t_1 \otimes t_2 : B \otimes C \mid \Delta_1 \mid \Delta_2}\ \textsc{Tr}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma_3 \vdash \Delta \mid [t/x](t_1 \otimes t_2) : B \otimes C \mid [t/x]\Delta_1 \mid [t/x]\Delta_2}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_2, x : A, \Gamma_3 \vdash t_2 : C \mid \Delta_2}}{\Gamma_2, \Gamma, \Gamma_3 \vdash \Delta \mid [t/x]t_2 : C \mid [t/x]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3 \vdash t_1 \otimes ([t/x]t_2) : B \otimes C \mid \Delta_1 \mid \Delta \mid [t/x]\Delta_2}\ \textsc{Tr}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma_3 \vdash \Delta \mid t_1 \otimes ([t/x]t_2) : B \otimes C \mid \Delta_1 \mid [t/x]\Delta_2}\ \textsc{Series of Exchanges}$$

This case is similar to the previous case.

Case: secondary hypothesis: right introduction of par. The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 : B \mid t_2 : C \mid \Delta_2}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 \parr t_2 : B \parr C \mid \Delta_2}\ \textsc{ParR}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x](t_1 \parr t_2) : B \parr C \mid [t/x]\Delta_2}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 : B \mid t_2 : C \mid \Delta_2}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_1 : B \mid [t/x]t_2 : C \mid [t/x]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_1 \parr [t/x]t_2 : B \parr C \mid [t/x]\Delta_2}\ \textsc{ParR}$$

Clearly, $[t/x](t_1 \parr t_2) = ([t/x]t_1) \parr [t/x]t_2$.

Case: secondary hypothesis: left introduction of par (first case). The proof

$$\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{array}{c}\pi_2\\ \vdots\end{array}}{\Gamma_1, x : A, \Gamma_2, y : B \vdash \Delta_1} \qquad \dfrac{\begin{array}{c}\pi_3\\ \vdots\end{array}}{\Gamma_3, z : C \vdash \Delta_2}}{\Gamma_1, x : A, \Gamma_2, \Gamma_3, w : B \parr C \vdash \text{let-pat}\, w\, (y \parr -)\, \Delta_1 \mid \text{let-pat}\, w\, (- \parr z)\, \Delta_2}\ \textsc{ParL}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma_3, w : B \parr C \vdash \Delta \mid [t/x](\text{let-pat}\, w\, (y \parr -)\, \Delta_1) \mid [t/x](\text{let-pat}\, w\, (- \parr z)\, \Delta_2)}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2, y : B \vdash \Delta_1}}{\Gamma_1, \Gamma, \Gamma_2, y : B \vdash \Delta \mid [t/x]\Delta_1}\,\textsc{Cut} \qquad \dfrac{\begin{matrix}\pi_3\\ \vdots\end{matrix}}{\Gamma_3, z : C \vdash \Delta_2}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma_3, w : B \parr C \vdash \text{let-pat}\, w\, (y \parr -)\, \Delta \mid \text{let-pat}\, w\, (y \parr -)\, [t/x]\Delta_1 \mid \text{let-pat}\, w\, (- \parr z)\, \Delta_2}\,\textsc{ParL}$$

First, by inspection of the previous proofs we can see that $y \notin \mathsf{FV}(\Delta)$ and $x \notin \mathsf{FV}(\Delta_2)$. Thus, $\text{let-pat}\, w\, (y \parr -)\, \Delta = \Delta$, and $[t/x](\text{let-pat}\, w\, (- \parr z)\, \Delta_2) = \text{let-pat}\, w\, (- \parr z)\, \Delta_2$. It suffices to show that $[t/x](\text{let-pat}\, w\, (y \parr -)\, \Delta_1) = \text{let-pat}\, w\, (y \parr -)\, [t/x]\Delta_1$ but this follows by distributing the substitution into the let-pat, and then simplifying using the fact that $w \neq x$.

Case: secondary hypothesis: left introduction of par (second case). The proof

$$\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, y : B \vdash \Delta_1} \qquad \dfrac{\begin{matrix}\pi_3\\ \vdots\end{matrix}}{\Gamma_2, x : A, \Gamma_3, z : C \vdash \Delta_2}}{\Gamma_1, \Gamma_2, x : A, \Gamma_3, w : B \parr C \vdash \text{let-pat}\, w\, (y \parr -)\, \Delta_1 \mid \text{let-pat}\, w\, (- \parr z)\, \Delta_2}\,\textsc{ParL}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, w : B \parr C \vdash \Delta \mid [t/x](\text{let-pat}\, w\, (y \parr -)\, \Delta_1) \mid [t/x](\text{let-pat}\, w\, (- \parr z)\, \Delta_2)}\,\textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, y : B \vdash \Delta_1} \qquad \dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{matrix}\pi_3\\ \vdots\end{matrix}}{\Gamma_2, x : A, \Gamma_3, z : C \vdash \Delta_2}}{\Gamma_2, \Gamma, \Gamma_3, z : C \vdash \Delta \mid [t/x]\Delta_2}\,\textsc{Cut}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, w : B \parr C \vdash \text{let-pat}\, w\, (y \parr -)\, \Delta_1 \mid \text{let-pat}\, w\, (- \parr z)\, \Delta \mid \text{let-pat}\, w\, (- \parr z)\, [t/x]\Delta_2}\,\textsc{ParL}$$

Similar to the previous case.

Case: secondary hypothesis: left introduction of implication (first case). The proof

$$\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\begin{matrix}\pi_3\\ \vdots\end{matrix}}{\Gamma_3, y : C \vdash \Delta_2}}{\Gamma_1, x : A, \Gamma_2, \Gamma_3, z : B \multimap C \vdash \Delta_1 \mid [z\, t_1/y]\Delta_2}\,\textsc{Impl}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma_3, z : B \multimap C \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x][z\, t_1/y]\Delta_2}\,\textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2 \vdash t_1 : B \mid \Delta_1}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]t_1 : B \mid [t/x]\Delta_1}\,\textsc{Cut} \qquad \dfrac{\begin{matrix}\pi_3\\ \vdots\end{matrix}}{\Gamma_3, y : C \vdash \Delta_2}}{\Gamma_1, \Gamma, \Gamma_2, \Gamma_3, z : B \multimap C \vdash \Delta \mid [t/x]\Delta_1 \mid [z\, ([t/x]t_1)/y]\Delta_2}\,\textsc{Impl}$$

By inspection of the above derivations we can see that $x \notin \mathsf{FV}(\Delta_2)$, and hence, by this fact and substitution distribution (Lemma 4) we know $[t/x][z\ t_1/y]\Delta_2 = [(([t/x]z)\ ([t/x]t_1)/y][t/x]\Delta_2 = [z\ ([t/x]t_1)/y]\Delta_2$.

Case: secondary hypothesis: left introduction of implication (second case). The proof

$$
\dfrac{\dfrac{\pi_1}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\pi_2}{\Gamma_1 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\pi_3}{\Gamma_2, x : A, \Gamma_3, y : C \vdash \Delta_2}}{\Gamma_1, \Gamma_2, x : A, \Gamma_3, z : B \multimap C \vdash \Delta_1 \mid [z\ t_1/y]\Delta_2}\ \textsc{Impl}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, z : B \multimap C \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x][z\ t_1/y]\Delta_2}\ \textsc{Cut}
$$

transforms into the proof

$$
\dfrac{\dfrac{\dfrac{\pi_2}{\Gamma_1 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\dfrac{\pi_1}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\pi_3}{\Gamma_2, x : A, \Gamma_3, y : C \vdash \Delta_2}}{\Gamma_2, \Gamma, \Gamma_3, y : C \vdash \Delta \mid [t/x]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, z : B \multimap C \vdash \Delta_1 \mid [z\ t_1/y]\Delta \mid [z\ t_1/y][t/x]\Delta_2}\ \textsc{Impl}}{\Gamma_1, \Gamma_2, \Gamma, \Gamma_3, z : B \multimap C \vdash [z\ t_1/y]\Delta \mid \Delta_1 \mid [z\ t_1/y][t/x]\Delta_2}\ \textsc{Series of Exchanges}
$$

By inspection of the above proofs we can see that $y \notin \mathsf{FV}(\Delta)$. Thus, $[z\ t_1/y]\Delta = \Delta$. The same can be said for the variable x and context Δ_1, and hence, $[t/x]\Delta_1 = \Delta_1$. Finally, by inspection of the above proofs $x \notin \mathsf{FV}(t_1)$ and so by substitution distribution (Lemma 4) we know $[t/x][z\ t_1/y]\Delta_2 = [z\ t_1/y][t/x]\Delta_2$.

Case: secondary hypothesis: left introduction of implication (third case). The proof

$$
\dfrac{\dfrac{\pi_1}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\pi_2}{\Gamma_1 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\pi_3}{\Gamma_2, y : C, \Gamma_3, x : A \vdash \Delta_2}}{\Gamma_1, \Gamma_2, z : B \multimap C, \Gamma_3, x : A \vdash \Delta_1 \mid [z\ t_1/y]\Delta_2}\ \textsc{Impl}}{\Gamma_1, \Gamma_2, z : B \multimap C, \Gamma_3, \Gamma \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x][z\ t_1/y]\Delta_2}\ \textsc{Cut}
$$

transforms into the proof

$$
\dfrac{\dfrac{\dfrac{\pi_2}{\Gamma_1 \vdash t_1 : B \mid \Delta_1} \qquad \dfrac{\dfrac{\pi_1}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\pi_3}{\Gamma_2, y : C, \Gamma_3, x : A \vdash \Delta_2}}{\Gamma_2, y : C, \Gamma_3, \Gamma \vdash \Delta \mid [t/x]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma_2, z : B \multimap C, \Gamma_3, \Gamma \vdash \Delta_1 \mid [z\ t_1/y]\Delta \mid [z\ t_1/y][t/x]\Delta_2}\ \textsc{Impl}}{\Gamma_1, \Gamma_2, z : B \multimap C, \Gamma_3, \Gamma \vdash [z\ t_1/y]\Delta \mid \Delta_1 \mid [z\ t_1/y][t/x]\Delta_2}\ \textsc{Series of Exchanges}
$$

Similar to the previous case.

Case: secondary hypothesis: right introduction of implication. The proof

$$\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2, y : B \vdash t_1 : C \mid \Delta_1} \quad y \notin \mathsf{FV}(\Delta_1)}{\Gamma_1, x : A, \Gamma_2 \vdash \lambda y.t_1 : B \multimap C \mid \Delta_1}\ \textsc{ImpR}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x](\lambda y.t_1) : B \multimap C \mid [t/x]\Delta_1}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2, y : B \vdash t_1 : C \mid \Delta_1}}{\Gamma_1, \Gamma, \Gamma_2, y : B \vdash \Delta \mid [t/x]t_1 : C \mid [t/x]\Delta_1}\ \textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid \lambda y.[t/x]t_1 : B \multimap C \mid [t/x]\Delta_1}\ \textsc{ImpR}$$

Clearly, $[t/x](\lambda y.t_1) = \lambda y.[t/x]t_1$.

Case: secondary hypothesis: left introduction of tensor unit. The proof

$$\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1}}{\Gamma_1, x : A, \Gamma_2, y : \top \vdash \mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, \Delta_1}\ \textsc{Il}}{\Gamma_1, \Gamma, \Gamma_2, y : \top \vdash \Delta \mid [t/x](\mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, \Delta_1)}\ \textsc{Cut}$$

transforms into the proof

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1}\ \textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2, y : \top \vdash \mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, \Delta \mid \mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, [t/x]\Delta_1}\ \textsc{Il}$$

It suffices to show that $\Delta = \mathsf{let}\, y\, \mathsf{be} * \mathsf{in}\, \Delta$ and $[t/x](\mathsf{let}\, y\, \mathsf{be} * \mathsf{in}\, \Delta_1) = \mathsf{let}\, y\, \mathsf{be} * \mathsf{in}\, [t/x]\Delta_1$. Without loss of generality suppose $\Delta = t : B, \Delta'$. We know that it must be the case that $y \notin \mathsf{FV}(t)$, and we know that $[y/z]t = t$ when $z \notin \mathsf{FV}(t)$. Then by $\textsc{Eq\_Eta2I}$ we have $t = \mathsf{let}\, y\, \mathsf{be} * \mathsf{in}\, t$. This argument can be repeated for any other term in Δ'. Thus, $\Delta = \mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, \Delta$. It is easy to see that $[t/x](\mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, \Delta_1) = \mathsf{let}\, y\, \mathsf{be}\, * \,\mathsf{in}\, [t/x]\Delta_1$ using the rule $\textsc{Eq\_NatI}$.

Case: secondary hypothesis: right introduction of par unit. The proof

$$\dfrac{\dfrac{\begin{matrix}\pi_1\\ \vdots\end{matrix}}{\Gamma \vdash t : A \mid \Delta} \qquad \dfrac{\dfrac{\begin{matrix}\pi_2\\ \vdots\end{matrix}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1}}{\Gamma_1, x : A, \Gamma_2 \vdash \circ : \perp \mid \Delta_1}\ \textsc{Pr}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\circ : \perp \mid [t/x]\Delta_1}\ \textsc{Cut}$$

transforms into the proof

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma \vdash t : A \mid \Delta\end{array} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, x : A, \Gamma_2 \vdash \Delta_1\end{array}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1}\ \textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \circ :\perp \mid \Delta \mid [t/x]\Delta_1}\ \textsc{Pr}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid \circ :\perp \mid [t/x]\Delta_1}\ \textsc{Series of Exchanges}
$$

Clearly, $[t/x]\circ = \circ$.

Case: secondary hypothesis: left introduction of exchange. The proof

$$
\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma \vdash t : A \mid \Delta\end{array} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, x : A, \Gamma_2, w : B, y : C, \Gamma_3 \vdash \Delta_1\end{array}}{\Gamma_1, x : A, \Gamma_2, y : C, w : B, \Gamma_3 \vdash \Delta_1}\ \textsc{ExL}}{\Gamma_1, \Gamma, \Gamma_2, y : C, w : B, \Gamma_3 \vdash \Delta \mid [t/x]\Delta_1}\ \textsc{Cut}
$$

tranforms into the proof

$$
\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma \vdash t : A \mid \Delta\end{array} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, x : A, \Gamma_2, w : B, y : C, \Gamma_3 \vdash \Delta_1\end{array}}{\Gamma_1, \Gamma, \Gamma_2, w : B, y : C, \Gamma_3 \vdash \Delta \mid [t/x]\Delta_1}\ \textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2, y : C, w : B, \Gamma_3 \vdash \Delta \mid [t/x]\Delta_1}\ \textsc{ExL}
$$

Clearly, all terms are equivalent.

Case: secondary hypothesis: right introduction of exchange. The proof

$$
\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma \vdash t : A \mid \Delta\end{array} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 : B \mid t_2 : C \mid \Delta_2\end{array}}{\Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_2 : C \mid t_1 : B \mid \Delta_2}\ \textsc{ExR}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_2 : C \mid [t/x]t_1 : B \mid [t/x]\Delta_2}\ \textsc{Cut}
$$

is transformed into

$$
\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \hline \Gamma \vdash t : A \mid \Delta\end{array} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \hline \Gamma_1, x : A, \Gamma_2 \vdash \Delta_1 \mid t_1 : B \mid t_2 : C \mid \Delta_2\end{array}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_1 : B \mid [t/x]t_2 : C \mid [t/x]\Delta_2}\ \textsc{Cut}}{\Gamma_1, \Gamma, \Gamma_2 \vdash \Delta \mid [t/x]\Delta_1 \mid [t/x]t_2 : C \mid [t/x]t_1 : B \mid [t/x]\Delta_2}\ \textsc{ExR}
$$

Clearly, all terms are equivalent.

$\square$

Corollary 7 (Cut-Elimination)**.** *Cut-elimination holds for FILL.*

4 Full LNL Models

One of the difficult questions considering the categorical models of linear logic was how to model Girard's exponential, !, which is read "of course". The ! modality can be used to translate intuitionistic logic into intuitionistic linear logic, and so the correct categorical interpretation of ! should involve a relationship between a cartesian closed category, and the model of intuitionistic linear logic.

de Paiva gave some of the first categorical models of both classical and intuitionistic linear logic in her thesis [7]. She showed that a particular dialectica category called $\mathsf{Dial}_2(\mathsf{Sets})$ is a model of FILL where ! is interpreted as a comonad which produces natural comonoids, see page 76 of [7].

Definition 8. *The category* $\mathsf{Dial}_2(\mathsf{Sets})$ *consists of*

- *objects that are triples,* $A = (U, X, \alpha)$*, where* U *and* X *are sets, and* $\alpha \subseteq U \times X$ *is a relation, and*
- *maps that are pairs* $(f, F) : (U, X, \alpha) \longrightarrow (V, Y, \beta)$ *where* $f : U \longrightarrow V$ *and* $F : Y \longrightarrow X$ *such that*
  - *For any* $u \in U$ *and* $y \in Y$*,* $\alpha(u, F(y))$ *implies* $\beta(f(u), y)$*.*

Suppose $A = (U, X, \alpha)$*,* $B = (V, Y, \beta)$*, and* $C = (W, Z, \gamma)$*. Then identities are given by* $(\mathsf{id}_U, \mathsf{id}_X) : A \longrightarrow A$*. The composition of the maps* $(f, F) : A \longrightarrow B$ *and* $(g, G) : B \longrightarrow C$ *is defined as* $(f; g, G; F) : A \longrightarrow C$*.*

In her thesis de Paiva defines a particular class of dialectica categories called GC over a base category C, see page 41 of [7]. The category $\mathsf{Dial}_2(\mathsf{Sets})$ defined above can be seen as an instantiation of GC by setting C to be the category Sets of sets and functions between them. This model is a non-trivial model, and does not model classical logic; see [7] page 58.

Seely gave a different, syntactic categorical model that confirmed that the of-course exponential should be modeled by a comonad [20]. However, Seely's model turned out to be unsound, as pointed out by Bierman [3]. This then prompted Bierman, Hyland, de Paiva, and Benton to define another categorical model called linear categories (Definition 9) that are sound, and also model ! using a monoidal comonad [3].

Definition 9. *A* ***linear category****,* $\mathcal{L}$*, consists of:*

- *A symmetric monoidal closed category* $\mathcal{L}$*,*
- *A symmetric monoidal comonad* $(!, \epsilon, \delta, m_{A,B}, m_I)$ *such that*
  - *For every free !-coalgebra* $(!A, \delta_A)$ *there are two distinguished monoidal natural transformations* $e_A : !A \longrightarrow I$ *and* $d_A : !A \longrightarrow !A \otimes !A$ *which form a commutative comonoid and are coalgebra morphisms.*
  - *If* $f : (!A, \delta_A) \longrightarrow (!B, \delta_B)$ *is a coalgebra morphism between free coalgebras, then it is also a comonoid morphism.*

This definition is the one given by Bierman in his thesis, see [3] for full definitions.

Intuitionistic logic can be interpreted in a linear category as a full subcategory of the category of !-coalgebras for the comonad, see proposition 17 of [3].

Benton gave a more balanced view of linear categories called LNL models.

Definition 10. *A* ***linear/non-linear model (LNL model)*** *consists of*

- *a cartesian closed category* $(\mathcal{C}, 1, \times, \Rightarrow)$*,*
- *a SMCC* $(\mathcal{L}, I, \otimes, \multimap)$*, and*
- *a pair of symmetric monoidal functors* $(G, n) : \mathcal{L} \longrightarrow \mathcal{C}$ *and* $(F, m) : \mathcal{C} \longrightarrow \mathcal{L}$ *between them that form a symmetric monoidal adjunction with* $F \dashv G$*.*

See Benton, [2], for the definitions of symmetric monoidal functors and adjunctions.

A non-trivial consequence of the definition of a LNL model is that the ! modality can indeed be interpreted as a monoidal comonad. Suppose $(\mathcal{L}, \mathcal{C}, F, G)$ is a LNL model. Then the comonad is given by $(!, \epsilon : ! \longrightarrow \mathsf{Id}, \delta : ! \longrightarrow !!)$ where $! = FG$, ϵ is the counit of the adjunction and δ is the natural transformation $\delta_A = F(\eta_{G(A)})$, see page 15 of [2]. We recall the following result from Benton [2]:

Theorem 11 (LNL Models and Linear Categories)**.**

i. (Section 2.2.1 of [2]) Every LNL model is a linear category.

ii. (Section 2.2.2 of [2]) Every linear category is a LNL model.

Proof. The proof of part i. is a matter of checking that each part of the definition of a linear category can be constructed using the definition of a LNL model. See lemmata 3-7 of [2].

As for the proof of part ii. Given a linear category we have a SMCC and so the difficulty of proving this result is constructing the CCC and the adjunction between both parts of the model. Suppose $\mathcal{L}$ is a linear category. Benton constructs the CCC out of the full subcategory of Eilenberg-Moore category $\mathcal{L}^!$ whose objects are exponentiable coalgebras denoted $\mathsf{Exp}(\mathcal{L}^!)$. He shows that this subcategory is cartesian closed, and contains the (co)Kleisli category, $\mathcal{L}_!$, Lemma 11 on page 23 of [2]. The required adjunction $F : \mathsf{Exp}(\mathcal{L}^!) \longrightarrow L : G$ can be defined using the adjunct functors $F(A, h_A) = A$ and $G(A) = (!A, \delta_A)$, see lemmata 13 - 16 of [2]. □

Next we show that the category $\mathsf{Dial}_2(\mathsf{Sets})$ is a full version of a linear category. First, we extend the definitions of linear categories and LNL models to be equipped with the necessary categorical structure to model par and its unit.

Definition 12. *A* ***full linear category****, $\mathcal{L}$, consists of a linear category $(\mathcal{L}, \top, \otimes, \multimap, !A, e_A, d_A)$, a symmetric monoidal structure on L, $(\perp, \invamp)$, and distribution natural transformations $\mathsf{dist}_1 : A \otimes (B \invamp C) \longrightarrow (A \otimes B) \invamp C$ and $\mathsf{dist}_2 : (A \invamp B) \otimes C \longrightarrow A \invamp (B \otimes C)$.*

Definition 13. *A* ***full linear/non-linear model (full LNL model)*** *consists of a LNL model $(\mathcal{L}, \mathcal{C}, F, G)$, and a symmetric monoidal structure on L, $(\perp, \invamp)$, as above.*

Our result is to first prove that $\mathsf{Dial}_2(\mathsf{Sets})$ is a full linear category, and then using the proof by Benton that linear categories are LNL models we obtain that $\mathsf{Dial}_2(\mathsf{Sets})$ is a full LNL model, but in order for this to work we need to know that $\mathsf{Dial}_2(\mathsf{Sets})$ has a symmetric monoidal comonad $(!, \epsilon, \delta, m_{A,B}, m_I)$. However, at the time of de Paiva's thesis it was not known that the comonad modeling the of-course exponential needed to be monoidal. We were able to show that the maps $m_{A,B}$ and m_I exist in the more general setting of dialectica categories, and thus, these maps exist in $\mathsf{Dial}_2(\mathsf{Sets})$. Intuitively, given two objects $A = (X, U, \alpha)$ and $B = (V, Y, \beta)$ of $\mathsf{Dial}_2(\mathsf{Sets})$ the map $m_{A,B}$ is defined as the pair $(\mathsf{id}_{U \times V}, F)$, where $F = (F_1, F_2)$, $F_1 : (U \times V) \Rightarrow (V \Rightarrow X)^* \longrightarrow V \Rightarrow (U \Rightarrow X^*)$ and $F_2 : (U \times V) \Rightarrow (U \Rightarrow Y)^* \longrightarrow U \Rightarrow (V \Rightarrow Y^*)$. The maps F_1 and F_2 build the sequence of all the results of applying each function in the input sequence to the input coordinate.

We can now show our main result of this section.

Lemma 14. *The category $\mathsf{Dial}_2(\mathsf{Sets})$ is a full linear category.*

Proof. This proof holds by constructing each piece of a full linear category using the structure of $\mathsf{Dial}_2(\mathsf{Sets})$[2]. We use some notation to make it easier to define and use functions over sequences. Given a function $g : A \longrightarrow X^*$ we will denote taking the ith projection of the sequence returned by $g(a)$ for some $a \in A$ by $g(a)_i$. To construct set-theoretic anonymous functions we use λ-notation. Lastly, we often use let-expressions to pattern match on sequences. For example, $\mathsf{let}\,(x_1, \ldots, x_i) = g(a)\,\mathsf{in}\,(f(x_1), \cdots, f(x_i))$.

First, we must construct a linear category. It is well known that Sets is a CCC, and in fact, locally cartesian closed, and so by using the results of de Paiva's thesis we can easily see that $\mathsf{Dial}_2(\mathsf{Sets})$ is symmetric monoidal closed:

[2]This proof was formalized in the Agda proof assistant see the file `https://github.com/heades/cut-fill-agda/blob/master/FullLinCat.agda`

- (Definition 7, page 43 of [7]). Suppose $A, B \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$. Then there are sets X, Y, V, and U, and relations $\alpha \subseteq U \times X$ and $\beta \subseteq V \times Y$, such that, $A = (U, X, \alpha)$ and $B = (V, Y, \beta)$. The tensor product of A and B can now be defined by $A \otimes B = (U \times V, (V \Rightarrow X) \times (U \Rightarrow Y), \alpha \otimes \beta)$, where $(- \Rightarrow -)$ is the internal hom of Sets. We define $((u, v), (f, g)) \in \alpha \otimes \beta$ if and only if $(u, f(v)) \in \alpha$ and $(v, g(u)) \in \beta$.

  Suppose $A = (U, X, \alpha)$, $B = (V, Y, \beta)$, $C = (W, Z, \gamma)$, and $D = (S, T, \delta)$ objects of $\mathsf{Dial_2}(\mathsf{Sets})$, and $m_1 = (f, F) : A \longrightarrow C$ and $m_2 = (g, G) : B \longrightarrow D$ are maps of $\mathsf{Dial_2}(\mathsf{Sets})$. Then the map $m_1 \otimes m_2 : A \otimes B \longrightarrow C \otimes D$ is defined by $(f \times g, F_\otimes)$ where $f \times g$ is the ordinary cartesian product functor in Sets, and we define $F \otimes G$ as follows:

$$\begin{array}{l} F_\otimes : (S \Rightarrow Z) \times (W \Rightarrow T) \longrightarrow (V \Rightarrow X) \times (U \Rightarrow Y) \\ F_\otimes(h_1, h_2) = (\lambda v.F(h_1(g(v))), \lambda u.G(h_2(f(u)))) \end{array}$$

  It is straightforward to confirm the relation condition on maps for $m_1 \otimes m_2$.

- (Definition 7, page 44 of [7]). Suppose $1 \in \mathsf{Obj}(\mathsf{Sets})$ is the final object, and $\mathsf{id}_1 \subseteq 1 \times 1$. Then we can define tensors unit by the object $\top = (1, 1, \mathsf{id}_1)$.

- Suppose $A = (U, X, \alpha)$ is an object of $\mathsf{Dial_2}(\mathsf{Sets})$. Then the map $\lambda_A : \top \otimes A \longrightarrow A$ is defined by $(\hat{\lambda}_U, F_\lambda)$ where $\hat{\lambda}_U$ is the left unitor for the cartesian product in Sets, $F_\lambda(x) = (\diamond, \lambda y.x) : X \longrightarrow (U \Rightarrow 1) \times (1 \Rightarrow X)$, and $\diamond$ is the terminal arrow in Sets. It is easy to see that both $\hat{\lambda}_U$ and F_λ have inverses, and thus, λ_A has an inverse. It is straightforward to confirm the relation condition on maps for λ_A and its inverse.

- Suppose $A = (U, X, \alpha)$ is an object of $\mathsf{Dial_2}(\mathsf{Sets})$. Then the map $\rho_A : A \otimes \top \longrightarrow A$ is defined similarly to λ_A given above.

- Suppose $A = (U, X, \alpha) \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$ and $B = (V, Y, \beta) \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$. Then we define the map $\beta_{A,B} : A \otimes B \longrightarrow B \otimes A$ by $(\hat{\beta}_{U,V}, \hat{\beta}_{V \Rightarrow X, U \Rightarrow Y})$ where $\hat{\beta}$ is the symmetry of the cartesian product in Sets. Again, it is straightforward to see that β has an inverse, and the relation condition on maps is satisfied.

- Suppose $A = (U, X, \alpha) \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$, $B = (V, Y, \beta) \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$, and $C = (W, Z, \gamma) \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$. Then we define $\alpha_{A,B,C} : (A \otimes B) \otimes C \longrightarrow A \otimes (B \otimes C)$ by $(\hat{\alpha}_{U,V,W}, F_\alpha)$ where $\hat{\alpha}_{U,V,W}$ is the associator for the cartesian product in Sets and F_α is defined as follows:

$$\begin{array}{l} F_\alpha : ((V \times W) \Rightarrow X) \times (U \Rightarrow ((W \Rightarrow Y) \times (V \Rightarrow Z))) \longrightarrow (W \Rightarrow ((V \Rightarrow X) \times (U \Rightarrow Y))) \times ((U \times V) \Rightarrow Z) \\ F_\alpha(h_1, h_2) = (\lambda w.(\lambda v.h_1(v, w), \lambda u.h_2(u)_1(w)), \lambda(u, v).h_2(u)_2(v)) \end{array}$$

  The inverse of $\alpha_{A,B,C}$ is similar, and it is straightforward to confirm the relation condition on maps.

- (Definition 9, page 44 of [7]). Suppose $A, B \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$. Then there are sets X, Y, V, and U, and relations $\alpha \subseteq U \times X$ and $\beta \subseteq V \times Y$, such that, $A = (U, X, \alpha)$ and $B = (V, Y, \beta)$. Then we define the internal hom of $\mathsf{Dial_2}(\mathsf{Sets})$ by $A \multimap B = ((U \Rightarrow V) \times (Y \Rightarrow X), U \times Y, \alpha \Rightarrow \beta)$. We define $((f, g), (u, y)) \in \alpha \Rightarrow \beta$ if and only if whenever $(u, g(y)) \in \alpha$, then $(f(u), y) \in \beta$. The locally cartesian closed structure of Sets guarantees that for any two objects $A, B \in \mathsf{Dial_2}(\mathsf{Sets})$ the internal hom $A \multimap B \in \mathsf{Dial_2}(\mathsf{Sets})$ exists.

Using the constructions above $\mathsf{Dial_2}(\mathsf{Sets})$ is a SMCC by Proposition 24 on page 46 of [7].

Next we define the symmetric monoidal comonad $(!, \epsilon, \delta, m_{A,B}, m_I)$ of the linear category:

- (Section 4.5, on page 76 of [7]). The endofunctor $! : \mathsf{Dial_2}(\mathsf{Sets}) \longrightarrow \mathsf{Dial_2}(\mathsf{Sets})$ is defined as follows:

  - Objects. Suppose $A = (U, X, \alpha) \in \mathsf{Obj}(\mathsf{Dial_2}(\mathsf{Sets}))$. Then we set $!A = (U, U \Rightarrow X^*, !\alpha)$, where $(u, f) \in !\alpha$ if and only if $(u, f(u)_1) \in \alpha$ and $\cdots$ and $(u, f(u)_i) \in \alpha$ where $f(u)$ is a sequence of length i.

– Morphisms. Suppose $A = (U, X, \alpha) \in \mathsf{Obj}(\mathsf{Dial_2(Sets)})$, $B = (V, Y, \beta) \in \mathsf{Obj}(\mathsf{Dial_2(Sets)})$, and $(f, F) : A \longrightarrow B \in \mathsf{Mor}(\mathsf{Dial_2(Sets)})$. Then we define $!(f, F) = (f, !F) :!A \longrightarrow !B$, where $!F(g) = \lambda x.F^*(g(f(x))) : V \Rightarrow Y^* \longrightarrow U \Rightarrow X^*$.

- (Section 4.5, page 77 of [7]). The endofunctor ! defined above is the functor part of the comonad $(!, \epsilon, \delta)$. Suppose $A = (U, X, \alpha) \in \mathsf{Obj}(\mathsf{Dial_2(Sets)})$. Then the co-unit $\epsilon :!A \longrightarrow A$ is defined by $\epsilon = (\mathsf{id}_U, F_0)$ where $F_0(x) = \lambda y.(x) : X \longrightarrow U \Rightarrow X^*$. Furthermore, the co-multiplication $\delta_A :!A \longrightarrow !!A$ is defined by $\delta_A = (\mathsf{id}_U, F_1)$ where $F_1(g) = \lambda u.(f_1(u) \circ \cdots \circ f_i(u)) : U \Rightarrow (U \Rightarrow X^*)^* \longrightarrow U \Rightarrow X^*$ where $g(u) = (f_1, \ldots, f_i)$.

- The following diagrams commute:

$$
\begin{tikzcd}
!A \arrow[r, "\delta_A"] \arrow[d, "\delta_A"'] & !!A \arrow[d, "!\delta_A"] \\
!!A \arrow[r, "\delta_{!A}"'] & !!!A
\end{tikzcd}
\qquad
\begin{tikzcd}
 & !A \arrow[dl, equal] \arrow[d, "\delta_A"] \arrow[dr, equal] & \\
!A & !!A \arrow[l, "\epsilon_{!A}"] \arrow[r, "!\epsilon_A"'] & !A
\end{tikzcd}
$$

We show the left most diagram commutes first. It suffices to show that $\delta_a; !\delta_A = (\mathsf{id}_U, !F_1; F_1) = (\mathsf{id}_U, F_1; F_1)$. Suppose $g \in U \Rightarrow (U \Rightarrow X^*)^*$. Then

$$
\begin{array}{l}
F_1(F_1(g)) \\
= F_1(\lambda u.g(u)_1(u) \circ \cdots \circ g(u)_i(u)) \\
= \lambda u.g(u)_1(u)_1(u) \circ \cdots \circ g(u)_1(u)_j(u) \circ \cdots \circ g(u)_i(u)_1(u) \circ \cdots \circ g(u)_i(u)_k(u)
\end{array}
$$

Consider the other direction in the diagram.

$$
\begin{array}{rcl}
F_1(!F_1(g)) & = & F_1(\lambda x.F_1^*(g(x))) \\
 & = & \lambda u.F_1(g(u)_1)(u) \circ \cdots \circ F_1(g(u)_i)(u)
\end{array}
$$

Note that we have the following:

$$
\begin{array}{rcl}
F_1(g(u)_1)(u) & = & g(u)_1(u)_1(u) \circ \cdots \circ g(u)_1(u)_k(u) \\
 & \vdots & \\
F_1(g(u)_i)(u) & = & g(u)_i(u)_1(u) \circ \cdots \circ g(u)_i(u)_k(u)
\end{array}
$$

Clearly, the above reasoning implies that $F_1; F_1 = !F_1; F_1$.

Now we prove that the second diagram commutes, but we break it into two. We define $\delta_A; !\epsilon_A = (\mathsf{id}_U, !F_0; F_1)$ where for any $g \in U \Rightarrow X^*$,

$$
\begin{array}{rcl}
(!F_0; F_1)(g) & = & F_1(!F_0(g)) \\
 & = & F_1(\lambda u'.F_0^*(g(u'))) \\
 & = & F_1(\lambda u'.(\lambda y.(g(u')_1), \ldots, (\lambda y.g(u')_i))) \\
 & = & \lambda u.(g(u)_1) \circ \cdots \circ (g(u)_i) \\
 & = & g
\end{array}
$$

and we can define $\delta_A; \epsilon_{!A} = (\mathsf{id}_U, F_0; F_1)$ where for any $g \in U \Rightarrow X^*$,

$$
\begin{array}{rcl}
(F_0; F_1)(g) & = & F_1(F_0(g)) \\
 & = & F_1(\lambda y.(g)) \\
 & = & \lambda u.g(u) \\
 & = & g
\end{array}
$$

We can see by the reasoning above that $!F_0; F_1 = F_0; F_1 = \mathsf{id}_{U \Rightarrow X^*}$.

- The monoidal natural transformation $m_\top : \top \longrightarrow !\top$ is defined by $m_\top = (\mathsf{id}_1, \lambda f.\star)$ where $\star \in \top$. It is easy to see that the relation condition on maps for $\mathsf{Dial}_2(\mathsf{Sets})$ is satisfied. The following two diagrams commute:

$$\begin{tikzcd}
\top \arrow[r, "m_\top"] \arrow[d, "m_\top"'] & !\top \arrow[d, "\delta_\top"] \\
!\top \arrow[r, "!m_\top"'] & !!\top
\end{tikzcd}
\qquad
\begin{tikzcd}
\top \arrow[d, "m_\top"'] \arrow[dr, equal] & \\
!\top \arrow[r, "\epsilon_\top"'] & \top
\end{tikzcd}$$

It is straightforward to see that the above two diagrams commute using the fact that the second coordinate of $m_\top$ is a constant function.

- The monoidal natural transformation $m_{A,B} :!A \otimes !B \longrightarrow !(A \otimes B)$ is defined by $m_{A,B} = (\mathsf{id}_{U \times V}, F_2)$. We need to define F_2, but two auxiliary functions are needed first:

$$\begin{array}{l}
h_1 : (U \times V) \Rightarrow ((V \Rightarrow X) \times (U \Rightarrow Y))^* \longrightarrow (V \Rightarrow (U \Rightarrow X^*)) \\
h_1(g, v, u) = (f_1(v), \ldots, f_i(v)) \text{ where } g(u, v) = ((f_1, g_1), \ldots, (f_i, g_i))
\end{array}$$

$$\begin{array}{l}
h_2 : (U \times V) \Rightarrow ((V \Rightarrow X) \times (U \Rightarrow Y))^* \longrightarrow (U \Rightarrow (V \Rightarrow Y^*)) \\
h_2(g, u, v) = (g_1(u), \ldots, g_i(u)) \text{ where } g(u, v) = ((f_1, g_1), \ldots, (f_i, g_i))
\end{array}$$

Then $F_2(g) = (h_1(g), h_2(g))$. In order for $m_{A,B}$ to be considered a full fledge map in $\mathsf{Dial}_2(\mathsf{Sets})$ we have to verify that the relation condition on maps is satisfied. Suppose $(u, v) \in U \times V$ and $g \in (U \times V) \Rightarrow ((V \Rightarrow X) \times (U \Rightarrow Y))^*$, where $g(u, v) = ((f_1, g_1), \ldots, (f_i, g_i))$. Then we know the following by definition:

$$\begin{array}{lll}
((u, v), F(g)) \in !\alpha \otimes !\beta & \text{iff} & ((u, v), (h_1(g), h_2(g))) \in !\alpha \otimes !\beta \\
& \text{iff} & (u, h_1(g)(v)) \in !\alpha \text{ and } (v, h_2(g)(u)) \in !\beta \\
& \text{iff} & (u, f_1(v)) \in \alpha \text{ and } \cdots \text{ and } (u, f_i(v)) \text{ and} \\
& & (v, g_1(u)) \in \beta \text{ and } \cdots \text{ and } (v, g_i(u))
\end{array}$$

and

$$\begin{array}{lll}
((u, v), g) \in !(\alpha \otimes \beta) & \text{iff} & ((u, v), (f_1, g_1)) \in \alpha \otimes \beta \text{ and } \cdots \text{ and} \\
& & ((u, v), (f_i, g_i)) \in \alpha \otimes \beta \\
& \text{iff} & (u, f_1(v)) \in \alpha \text{ and } (v, g_1(u)) \in \beta \text{ and } \cdots \text{ and} \\
& & (u, f_i(v)) \in \alpha \text{ and } (v, g_i(u)) \in \beta
\end{array}$$

The previous definitions imply that $((u, v), F(g)) \in !\alpha \otimes !\beta$ implies $((u, v), g) \in !(\alpha \otimes \beta)$. Thus, $m_{A,B}$ is a map in $\mathsf{Dial}_2(\mathsf{Sets})$.

At this point we show that the following diagrams commute:

$$\begin{tikzcd}
!\top \otimes !A \arrow[r, "m_{\top,A}"] \arrow[dr, phantom, "\mathrm{A}"] & !(\top \otimes A) \arrow[d, "!\lambda_A"] \\
\top \otimes !A \arrow[u, "m_\top \otimes \mathsf{id}_{!A}"] \arrow[r, "\lambda_{!A}"'] & !A
\end{tikzcd}
\qquad
\begin{tikzcd}
!A \otimes !\top \arrow[r, "m_{A,\top}"] \arrow[dr, phantom, "\mathrm{B}"] & !(A \otimes \top) \arrow[d, "!\rho_A"] \\
!A \otimes \top \arrow[u, "\mathsf{id}_{!A} \otimes m_\top"] \arrow[r, "\rho_{!A}"'] & !A
\end{tikzcd}$$

$$
\begin{tikzcd}
!A \otimes !B \arrow[r, "m_{A,B}"] \arrow[d, "\beta"'] \arrow[dr, phantom, "\text{C}"] & !(A \otimes B) \arrow[d, "!\beta"] \\
!B \otimes !A \arrow[r, "m_{B,A}"'] & !(B \otimes A)
\end{tikzcd}
$$

$$
\begin{tikzcd}
(!A \otimes !B) \otimes !C \arrow[r, "m_{A,B} \otimes \mathsf{id}_{!C}"] \arrow[d, "\alpha_{!A,!B,!C}"'] & !(A \otimes B) \otimes !C \arrow[r, "m_{A \otimes B, C}"] \arrow[d, phantom, "\text{D}"] & !((A \otimes B) \otimes C) \arrow[d, "!\alpha_{A,B,C}"] \\
!A \otimes (!B \otimes !C) \arrow[r, "\mathsf{id}_{!A} \otimes m_{B,C}"'] & !A \otimes !(B \otimes C) \arrow[r, "m_{A, B \otimes C}"'] & !(A \otimes (B \otimes C))
\end{tikzcd}
$$

We first prove that diagram A commutes, and then diagrams B, and C will commute using similar reasoning. Following this we show that diagram D commutes. It suffices to show that

$$(m_\top \otimes id_{!A}); m_{\top,A}; !\lambda_A = (\hat{\lambda}_U, \lambda g.F_\otimes(F_2(F_\lambda(g)))) = (\hat{\lambda}_U, \lambda g.(\diamond, \lambda t.\lambda u.g(u))).$$

Suppose $g \in U \Rightarrow X^*$, $(\star, u) \in 1 \times U$, and $g(u) = (x_1, \ldots, x_i)$. Then

$$
\begin{array}{rcl}
F_\lambda(g)(\star, u) & = & (\lambda x'.(\diamond, \lambda y.x'))^*(g(\hat{\lambda}_U(\star, u))) \\
& = & (\lambda x'.(\diamond, \lambda y.x'))^*(g(u)) \\
& = & (\lambda x'.(\diamond, \lambda y.x'))^*(x_1, \ldots, x_i) \\
& = & ((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i))
\end{array}
$$

This implies that

$$F_\lambda(g) = \lambda(\star, u).\mathsf{let}\,(x_1, \ldots, x_i) = g(u)\,\mathsf{in}\,((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i)).$$

Using this reasoning we can see the following:

$$
\begin{array}{rcl}
F_2(F_\lambda(g)) & = & (\lambda u.\lambda t.\mathsf{let}\,((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i)) = F_\lambda(g)(t, u)\,\mathsf{in} \\
& & (\diamond(u), \ldots, \diamond(u)), \\
& & \lambda t.\lambda u.\mathsf{let}\,((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i)) = F_\lambda(g)(t, u)\,\mathsf{in} \\
& & ((\lambda y.x_1)(t), \ldots, (\lambda y.x_1)(t))) \\
& = & (\lambda u.\lambda t.\mathsf{let}\,((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i)) = F_\lambda(g)(t, u)\,\mathsf{in} \\
& & (\star, \ldots, \star), \\
& & \lambda t.\lambda u.\mathsf{let}\,((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i)) = F_\lambda(g)(t, u)\,\mathsf{in} \\
& & (x_1, \ldots, x_1)) \\
& = & (\lambda u.\lambda t.(\star, \ldots, \star), \\
& & \lambda t.\lambda u.\mathsf{let}\,((\diamond, \lambda y.x_1), \ldots, (\diamond, \lambda y.x_i)) = F_\lambda(g)(t, u)\,\mathsf{in} \\
& & (x_1, \ldots, x_1)) \\
& = & (\lambda u.\lambda t.(\star, \ldots, \star), \lambda t.\lambda u.g(u))
\end{array}
$$

Finally, the previous allows us to infer the following:

$$F_\otimes(F_2(F_\lambda(g))) \quad = \quad (\diamond, \lambda t.\lambda u.g(u))$$

Thus, we obtained our desired result.

We show that diagram D commutes by observing that

$$\begin{array}{rcl}(m_{A,B}\otimes !\mathsf{id}_{!C}); m_{A\otimes B,C}; !\alpha_{A,B,C} & = & (\mathsf{id}, F_\otimes); (\mathsf{id}, F_2); (\hat{\alpha}, !F_\alpha) \\ & = & (\hat{\alpha}, !F_\alpha; F_2; F_\otimes) \\ & = & (\hat{\alpha}, F_2; F_\otimes; F_\alpha) \\ & = & (\hat{\alpha}, F_\alpha); (\mathsf{id}, F_\otimes); (\mathsf{id}, F_2)\end{array}$$

It suffices to show that $!F_\alpha; F_2; F_\otimes = F_2; F_\otimes; F_\alpha$:

$$\begin{array}{rcl}(!F_\alpha; F_2; F_\otimes)(g) & = & F_\otimes(F_2(!F_\alpha(g))) \\ & = & F_\otimes(F_2(\lambda x.F^*_\alpha(g(x))))\end{array}$$

Suppose $h_1 = \lambda v.\lambda u.\mathsf{let}\,((f_1, g_1), \ldots, (f_i, g_i)) = F^*_\alpha(g(u, v))\,\mathsf{in}\,(f_1(v), \ldots, f_i(v))$,

$h_2 = \lambda u.\lambda v.\mathsf{let}\,((f_1, g_1), \ldots, (f_i, g_i)) = F^*_\alpha(g(u, v))\,\mathsf{in}\,(g_1(u), \ldots, g_i(u))$, and

$$\begin{array}{l}F^*_\alpha(g(u, v)) = \mathsf{let}\,((f'_1, g'_1), \ldots, (f'_j, g'_j)) = g(u, v)\,\mathsf{in} \\ \quad (\lambda w.(\lambda v'.f'_1(v', w), \lambda u.g'_1(u)_1(w)), \lambda(u, v').g'_1(u)_2(v')), \ldots, \\ \quad (\lambda w.(\lambda v'.f'_j(v', w), \lambda u.g'_j(u)_1(w)), \lambda(u, v').g'_j(u)_2(v'))).\end{array}$$

Then we can simplify h_1 and h_2 as follows:

$$\begin{array}{l}h_1 = \lambda v.\lambda u.\mathsf{let}\,((f'_1, g'_1), \ldots, (f'_j, g'_j)) = g(u, v)\,\mathsf{in} \\ \quad ((\lambda v'.f'_1(v', v), \lambda u'.g'_1(u')_1(v)), \ldots, (\lambda v'.f'_j(v', v), \lambda u'.g'_j(u')_1(v))) \\ \text{and} \\ h_2 = \lambda u.\lambda v.\mathsf{let}\,(u_1, u_2) = u\,\mathsf{in} \\ \quad \mathsf{let}\,((f'_1, g'_1), \ldots, (f'_j, g'_j)) = g((u_1, u_2), v)\,\mathsf{in} \\ \quad\quad (g'_1(u_1)_2(u_2), \ldots, g'_j(u_1)_2(u_2))\end{array}$$

By the definition of F_2 the previous reasoning implies:

$$\begin{array}{rcl}F_\otimes(F_2(\lambda x.F^*_\alpha(g(x)))) & = & F_\otimes(h_1, h_2) \\ & = & (\lambda v.F_2(h_1(v)), h_2)\end{array}$$

Expanding the definition of $F_2(h_1(v))$ in the above definitions yields:

$$F_2(h_1(v)) \quad = \quad (h'_1, h'_2)$$

where

$$\begin{array}{l}h'_1 = \lambda v''.\lambda u''.(f'_1(v'', v), \ldots, f'_j(v'', v)) \\ h'_2 = \lambda u''.\lambda v''.(g'_1(u')_1(v), \ldots, g'_j(u')_1(v))\end{array}$$

At this point we can see that

$$(\lambda v.F_2(h_1(v)), h_2) \quad = \quad (\lambda v.(h'_1, h'_2), h_2)$$

We now simplify $F_2; F_\otimes; F_\alpha$. We know by definition:

$$F_2(g) \quad = \quad (h''_1, h''_2)$$

where

$$\begin{array}{rcl} h_1'' & = & \lambda v.\lambda u.\mathsf{let}\,((f_1', g_1'), \ldots, (f_j', g_j')) = g(u,v)\,\mathsf{in} \\ & & \quad \mathsf{let}\,(v', v'') = v\,\mathsf{in}\,(f_1'(v', v''), \ldots, f_k'(v', v'')) \\ \text{and} & & \\ h_2'' & = & \lambda u.\lambda v.\mathsf{let}\,((f_1', g_1'), \ldots, (f_j', g_j')) = g(u,v)\,\mathsf{in} \\ & & \quad (g_1'(u), \ldots, g_k'(u)) \end{array}$$

This implies that

$$\begin{array}{rcl} F_\alpha(F_\otimes(F_2(g))) & = & F_\alpha(F_\otimes(h_1'', h_2'')) \\ & = & F_\alpha(h_1'', \lambda u_4.F_2(h_2''(u_4))) \\ & = & (\lambda w.(\lambda v.h_1''(v,w), \lambda u.F_2(h_2''(u))_1(w)), \lambda(u,v).F_2(h_2''(u))_2(v)) \end{array}$$

Finally, by expanding the definition of F_2 in the last line of the above reasoning we can see that

$$(\lambda v.(h_1', h_2'), h_2) = (\lambda w.(\lambda v.h_1''(v,w), \lambda u.F_2(h_2''(u))_1(w)), \lambda(u,v).F_2(h_2''(u))_2(v))$$

modulo currying of set-theoretic functions.

- There are two coherence diagrams that $m_{A,B}$ and δ must ad hear to. They are listed as follows:

$$\begin{tikzcd} !A \otimes !B \arrow[r, "m_{A,B}"] \arrow[d, "\epsilon_A \otimes \epsilon_B"'] \arrow[dr, phantom, "\mathrm{E}"] & !(A \otimes B) \arrow[d, "\epsilon_{A\otimes B}"] \\ A \otimes B \arrow[r, equal] & A \otimes B \end{tikzcd}
\qquad
\begin{tikzcd} !A \otimes !B \arrow[rr, "m_{A,B}"] \arrow[d, "\delta_A \otimes \delta_B"'] & \mathrm{F} & !(A \otimes B) \arrow[d, "\delta_{A\otimes B}"] \\ !!A \otimes !!B \arrow[r, "m_{!A,!B}"'] & !(!A \otimes !B) \arrow[r, "!m_{A,B}"'] & !!(A \otimes B) \end{tikzcd}$$

Diagram E holds by simply expanding the definitions using an arbitrary input of a pair of functions. We now show diagram F commutes.

It suffices to show the following:

$$\begin{array}{rcl} m_{A,B}; \delta_{A\otimes B} & = & (\mathsf{id}_{U\times V}, F_1; F_2) \\ & = & (id_{u\times V}, !F_2; F_2; F_\otimes) \\ & = & (\delta_A \otimes \delta_B); m_{!A,!B}; !m_{A,B} \end{array}$$

Suppose $g \in (U \times V) \Rightarrow ((U \times V) \Rightarrow ((V \Rightarrow X) \times (U \Rightarrow Y))^*)^*$. Then we know by the type of g and the definition of F_1 it must be the case that $F_1(g)$ first extracts all of the functions $(f_1, \ldots, f_i)$ returned by $g(u,v)$ for arbitrary $u \in U$ and $v \in V$ – note that each f_i returns a sequence of pairs of functions, $((f_i', g_i'), \ldots, (f_j', g_j'))$ – then $F_1(g)$ returns the concatenation of all of these sequences. Finally, $F_2(F_1(g))$ returns two functions $h_1(v,u)$ and $h_2(u,v)$, where h_1 returns the sequence $(f_i'(v), \ldots, f_j'(v))$, and h_2 returns the sequence $(g_i'(u), \ldots, g_j'(u))$ from the sequence returned by $F_1(g)$. Note that each f_i' and g_i' returns a pair of functions.

Now consider applying $!F_2; F_2; F_\otimes$ to g. The function $!F_2$ will construct the function $\lambda x.F_2^*(g(x))$ by definition, and $F_2^*(g(x))$ will construct a sequence of pairs of functions $((h_1', h_1''), \ldots, (h_k', h_k''))$. The function g as we saw above returns a sequence of functions, $(f_1, \ldots, f_i)$, where each f_i returns a sequence of pairs of functions, $((f_i', g_i'), \ldots, (f_j', g_j'))$. This tells us that by definition $h_k'(v,u)$ will return the sequence $(f_i'(v), \ldots, f_j'(v))$ and $h_k''(u,v)$ will construct the sequence $(g_1'(u), \ldots, g_j'(u))$. Applying F_2 to $\lambda x.F_2^*(g(x))$ will construct two more functions $t_1(v,u)$ and $t_2(u,v)$ where the first returns the sequence of functions $(h_1'(v), \ldots, h_k'(v))$, and the second returns $(h_1''(u), \ldots, h_k''(u))$. Finally, applying the function $F_\otimes$ to the pair (t_1, t_2) will result in a pair of functions

$$
\begin{array}{rcl}
(\lambda v.F_1(t_1(v)), \lambda u.F_1(t_2(u))) & = & (\lambda v.\lambda u.h'_1(v)(u) \circ \cdots \circ h'_k(v)(u), \\
& & \lambda u.\lambda v.h''_1(u)(v) \circ \cdots \circ h''_k(u)(v)) \\
& = & (\lambda v.\lambda u.(f'_i(v), \ldots, f'_j(v)), \\
& & \lambda u.\lambda v.(g'_i(u), \ldots, g'_k(u)))
\end{array}
$$

We can now see that the pair $(\lambda v.\lambda u.(f'_i(v), \ldots, f'_j(v)), \lambda u.\lambda v.(g'_i(u), \ldots, g'_k(u)))$ is indeed equivalent to the pair (h_1, h_2) given above, and thus, the diagram commutes.

Next we must show that whenever $(!A, \delta)$ is a free comonoid, we have the distinguished natural transformations $e_A :!A \longrightarrow \top$ and $d_A :!A \longrightarrow !A \otimes !A$. Suppose $!A = (U, U \Rightarrow X^*)$ and $(!A, \delta)$ is a free comonoid. Then we have the following definitions:

- (Proposition 53, page 77 of [7]). We define $e_A :!A \longrightarrow \top$ as the pair $(\diamond, \lambda x.\lambda u.())$, where $\diamond$ is the terminal map on U and $()$ is the empty sequence.
- (Proposition 53, page 77 of [7]). We define $d_A :!A \longrightarrow !A \otimes !A$ as the pair (Δ, θ) where $\Delta : U \longrightarrow U \times U$ is the diagonal map in Sets, and

$$
\begin{array}{l}
\theta : ((U \times U) \Rightarrow X^*) \times ((U \times U) \Rightarrow X^*) \longrightarrow U \Rightarrow X^* \\
\theta(f, g) = \lambda u.f(u, u) \circ g(u, u).
\end{array}
$$

The maps e_A and d_A must satisfy several coherence diagrams.

- We must show that the map e_A is a monoidal natural transformation. This requires that the following diagrams hold (for any arbitrary map f):

$$
\begin{tikzcd}
!A \arrow[r, "e_A"] \arrow[d, "!f"'] \arrow[dr, phantom, "\mathrm{G}"] & \top \arrow[d, equal] \\
!B \arrow[r, "e_B"'] & \top
\end{tikzcd}
\qquad
\begin{tikzcd}
\top \arrow[d, "m_I"'] \arrow[dr, equal] \arrow[dr, phantom, "\mathrm{H}"', shift right=3] & \\
!\top \arrow[r, "e_\top"'] & \top
\end{tikzcd}
\qquad
\begin{tikzcd}
!A \otimes !B \arrow[r, "e_A \otimes e_B"] \arrow[d, "m_{A,B}"'] \arrow[dr, phantom, "\mathrm{I}"] & \top \otimes \top \arrow[d, "\lambda"] \\
!(A \otimes B) \arrow[r, "e_{A \otimes B}"'] & \top
\end{tikzcd}
$$

Diagrams G and H follow easily by the definition of e_A and m_I. We now show that diagram I commutes. It suffices to show the following:

$$
\begin{array}{rcl}
(e_A \otimes e_B); \lambda & = & (\diamond_U \times \diamond_V, F_\otimes); (\hat{\lambda}_\top, F_\lambda) \\
& = & ((\diamond_U \times \diamond_V); \hat{\lambda}_\top, F_\lambda; F_\otimes) \\
& = & (\diamond_{U \times V}, F_\lambda; F_\otimes) \\
& = & (\diamond_{U \times V}, F_2(\lambda u.())) \\
& = & (\diamond_{U \times V}, (\lambda x.\lambda u.()); F_2) \\
& = & (\mathsf{id}_{U \times V}; \diamond_{U \times V}, (\lambda x.\lambda u.()); F_2) \\
& = & (\mathsf{id}_{U \times V}, F_2); (\diamond_{U \times V}, \lambda x.\lambda u.()) \\
& = & m_{A,B}; e_{A \otimes B}
\end{array}
$$

It suffices to show $F_\lambda; F_\otimes = F_2(\lambda u.())$, but this easily follows by definition.

- The map d_A must be a monoidal natural transformation. This requires the following diagrams to commute:

$$
\begin{tikzcd}
!A \arrow[r, "d_A"] \arrow[d, "!f"'] \arrow[dr, phantom, "\mathrm{J}"] & !A \otimes !A \arrow[d, "!f \otimes !f"] \\
!B \arrow[r, "d_B"'] & !B \otimes !B
\end{tikzcd}
\qquad
\begin{tikzcd}
\top \arrow[r, "\lambda^{-1}"] \arrow[d, "m_\top"'] \arrow[dr, phantom, "\mathrm{K}"] & \top \otimes \top \arrow[d, "m_\top \otimes m_\top"] \\
!\top \arrow[r, "d_\top"'] & !\top \otimes !\top
\end{tikzcd}
$$

$$
\begin{tikzcd}
!A \otimes !B \arrow[r, "d_A \otimes d_B"] \arrow[d, "m_{A,B}"'] & (!A \otimes !A) \otimes (!B \otimes !B) \arrow[r, "iso"] \arrow[d, phantom, "\mathrm{L}"] & (!A \otimes !B) \otimes (!A \otimes !B) \arrow[d, "m_{A,B} \otimes m_{A,B}"] \\
!(A \otimes B) \arrow[rr, "d_{A \otimes B}"] & & !(A \otimes B) \otimes !(A \otimes B)
\end{tikzcd}
$$

Diagrams J and K follow easily from unfolding their definitions. We show that diagram L next. The morphism *iso* in $\mathsf{Dial_2(Sets)}$ is a isomorphism that can be built out of the SMCC structure. For its definition in terms of the SMCC maps see footnote 9 on page 141 of [3], but we give a direct definition instead. We will need the following definitions:

$$
\hat{iso}((u,u'),(v,v')) \quad = \quad ((u,v),(u',v'))
$$

$$
\begin{array}{rcl}
F_{iso}(f,g) & = & (\lambda(v',v'').(\lambda u'.\lambda u''.f(u',v')_1(v'',u''), \lambda u'.\lambda u''.g(u',v')_1(v'',u'')),\\
& & \lambda(u',u'').(\lambda v'.\lambda v''.f(u',v')_2(u'',v''), \lambda v'.\lambda v''.g(u',v')_1(u'',v'')))
\end{array}
$$

$$
\begin{array}{rcl}
F^{-1}_{iso}(h_1,h_2) & = & (\lambda(u,v).(\lambda v'.\lambda u'.h_1(v,v')_1(u,u'), \lambda u'.\lambda v'.h_2(u,u')_2(v,v')),\\
& & \lambda(u,v).(\lambda v'.\lambda u'.h_1(v,v')_2(u,u'), \lambda u'.\lambda v'.h_2(u,u')_2(v,v')))
\end{array}
$$

We omit the proof that F_{iso} is an isomorphism, but it is straightforward. Now $iso = (\hat{iso}, F_{iso})$.

It suffices to show the following:

$$
\begin{array}{rcl}
(d_A \otimes d_B); iso; (m_{A,B} \otimes m_{A,B}) & = & (\Delta_U \times \Delta_V, F_\otimes); (\hat{iso}, F_{iso}); (\mathsf{id}_{(U\times V)\times(U\times V)}, F_\otimes)\\
& = & ((\Delta_U \times \Delta_V); \hat{iso}; \mathsf{id}_{(U\times V)\times(U\times V)}, F_\otimes; F_{iso}; F_\otimes)\\
& = & ((\Delta_U \times \Delta_V); \hat{iso};, F_\otimes; F_{iso}; F_\otimes)\\
& = & (\Delta_{U\times V}, \Theta; F_2)\\
& = & (\mathsf{id}_{U\times V}; \Delta_{U\times V}, \Theta; F_2)\\
& = & (\mathsf{id}_{U\times V}, F_2); (\Delta_{U\times V}, \Theta)\\
& = & m_{A,B}; d_{A,B}
\end{array}
$$

At this point it suffices to show that $F_\otimes; F_{iso}; F_\otimes = \Theta; F_2$, but this follows using similar reasoning as above, because $F_{iso}; F_\otimes$ will reorganize the streams obtained by applying g_1 and g_2, and then the final $F_\otimes$ combines these sequences using Θ. However, F_2 does the same reorganization, and then the streams are combined using Θ.

- Suppose $A \in \mathsf{Obj(Dial_2(Sets))}$. Then we must show that $(!A, d_A, e_A)$ is a commutative comonoid, but this follows from the following diagrams:

$$\begin{tikzcd}
& !A \arrow[dl, "\rho^{-1}"'] \arrow[d, "d_A"] \arrow[dr, "\lambda^{-1}"] \arrow[dl, phantom, shift left=6ex, "\mathrm{M}"] \arrow[dr, phantom, shift right=6ex, "\mathrm{N}"] & \\
!A \otimes \top & !A \otimes !A \arrow[l, "\mathsf{id}_{!A} \otimes e_A"] \arrow[r, "e_A \otimes \mathsf{id}_{!A}"'] & \top \otimes !A
\end{tikzcd}
\qquad
\begin{tikzcd}
!A \arrow[r, "d_A"] \arrow[d, equal] \arrow[dr, phantom, "\mathrm{O}"] & !A \otimes !A \arrow[d, "\beta_{!A,!A}"] \\
!A \arrow[r, "d_A"'] & !A \otimes !A
\end{tikzcd}$$

$$\begin{tikzcd}
!A \arrow[rr, "d_A"] \arrow[d, "d_A"'] \arrow[drr, phantom, "\mathrm{P}"] & & !A \otimes !A \arrow[d, "\mathsf{id}_{!A} \otimes d_A"] \\
!A \otimes !A \arrow[r, "d_A \otimes \mathsf{id}_{!A}"'] & (!A \otimes !A) \otimes !A \arrow[r, "\alpha"'] & !A \otimes (!A \otimes !A)
\end{tikzcd}$$

We prove that diagram N commutes, and then diagrams M and O will commute by similar reasoning. Following this we prove that diagram P commutes.

It suffices to show the following:

$$\begin{array}{rcl}
d_A; (e_A \otimes \mathsf{id}_{!A}) & = & (\Delta, \Theta); (\diamond \times \mathsf{id}_U, F_\otimes) \\
& = & (\Delta; (\diamond \times \mathsf{id}_U), F_\otimes; \Theta) \\
& = & (\lambda u.(\diamond(u), u), F_\otimes; \Theta) \\
& = & (\hat{\lambda}_U^{-1}, F_\otimes; \Theta) \\
& = & (\hat{\lambda}^{-1}, F_{\lambda^{-1}}) \\
& = & \lambda^{-1}
\end{array}$$

We can easily see that the following holds by definition:

$$\begin{array}{rcl}
(F_\otimes; \Theta)(g_1, g_2) & = & \Theta(F_\otimes(g_1, g_2)) \\
& = & \Theta(\lambda v.\lambda u.(), \lambda u.g_2(\diamond(u))) \\
& = & \lambda u.() \circ g_2(\diamond(u, u)) \\
& = & \lambda u.g_2(\diamond(u, u)) \\
& = & \lambda^{-1}(g_1, g_2)
\end{array}$$

Now we show that diagram P commutes. However, it is straightforward to show that the following holds:

$$\begin{array}{rcl}
d_A; (\mathsf{id}_{!A} \otimes d_A) & = & (\Delta, \Theta); (\mathsf{id}_U \times \Delta, F_\otimes) \\
& = & (\Delta; (\mathsf{id}_U \times \Delta), F_\otimes; \Theta) \\
& = & (\Delta; (\mathsf{id}_U \times \Delta), F_\alpha; F_\otimes; \Theta) \\
& = & (\Delta; (\Delta \times \mathsf{id}_U); \hat{\alpha}, F_\alpha; F_\otimes; \Theta) \\
& = & (\Delta, \Theta); (\Delta \times \mathsf{id}_U, F_\otimes); (\hat{\alpha}, F_\alpha) \\
& = & d_A; (d_A \otimes \mathsf{id}_{!A}); \alpha
\end{array}$$

We can see that $F_\otimes; \Theta = F_\alpha; F_\otimes; \Theta$, because the right-hand side does the same as the left-hand side, but first reorganizes and does it on the opposite association.

- The map e_A must be a coalgebra morphism which amounts to requiring that the following diagram commute:

$$\begin{tikzcd}
!A \arrow[r, "e_A"] \arrow[d, "\delta_A"'] \arrow[dr, phantom, "\mathrm{Q}"] & \top \arrow[d, "m_\top"] \\
!!A \arrow[r, "!e_A"'] & !\top
\end{tikzcd}$$

Diagram Q commutes by unfolding definitions, and the fact that the second coordinate of e_A is a constant function.

- The map d_A must also be a coalgebra morphism, and hence, the following diagram must commute:

$$\begin{tikzcd}
!A \arrow[rr, "\delta_A"] \arrow[d, "d_A"'] \arrow[drr, phantom, "\mathrm{R}"] & & !!A \arrow[d, "!d_A"] \\
!A \otimes !A \arrow[r, "\delta_A \otimes \delta_A"'] & !!A \otimes !!A \arrow[r, "m_{!A,!A}"'] & !(!A \otimes !A)
\end{tikzcd}$$

It suffices to show that the following holds:

$$\begin{array}{lll}
\delta_A; !d_A & = & (\mathsf{id}_U, F_1); (\Delta, !\Theta) \\
 & = & (\Delta, !\Theta; F_1) \\
 & = & (\Delta; \mathsf{id}_{U\times U}; \mathsf{id}_{U\times U}, !\Theta; F_1) \\
 & = & (\Delta; (\mathsf{id}_U \times \mathsf{id}_U); \mathsf{id}_{U\times U}, !\Theta; F_1) \\
 & = & (\Delta; (\mathsf{id}_U \times \mathsf{id}_U); \mathsf{id}_{U\times U}, F_2; F_\otimes; \Theta) \\
 & = & (\Delta, \Theta); (\mathsf{id}_U \times \mathsf{id}_U, F_\otimes); (\mathsf{id}_{U\times U}, F_2) \\
 & = & d_A; (\delta_A \otimes \delta_A); m_{!A,!A}
\end{array}$$

Now consider the following:

$$\begin{array}{lll}
!(\Theta; F_1)(g) & = & F_1(!\Theta(g)) \\
 & = & F_!(\lambda x.\Theta^*(g(x))) \\
 & = & \lambda p.\mathsf{let}\,(f_1, \cdots, f_i) = \Theta^*(g(p))\,\mathsf{in}\,f_1(p) \circ \cdots \circ f_i(p)
\end{array}$$

Furthermore, consider the following:

$$\begin{array}{lll}
(F_2; F_\otimes; \Theta)(g) & = & \Theta(F_\otimes(h_1, h_2)) \\
 & = & \Theta(\lambda x.F_1(h_1(x)), \lambda y.F_1(h_2(y))) \\
 & = & \lambda u.F_1(h_1(u,u)) \circ F_1(h_2(u,u))
\end{array}$$

where

$$\begin{array}{lll}
h_1(u,u') & = & \mathsf{let}\,(f_1, \ldots, f_i) = g(u,u')\,\mathsf{in}\,(f_1(u), \ldots, f_i(u)) \\
h_2(u,u') & = & \mathsf{let}\,(f_1, \ldots, f_i) = g(u,u')\,\mathsf{in}\,(f_1(u'), \ldots, f_i(u'))
\end{array}$$

At this point we can see that $\Theta; F_1 = F_2; F_\otimes; \Theta$ by the previous reasoning and the definition of F_1.

- Finally, we show that when given a coalgebra morphism, f, between free coalgebras $(!A, \delta_A)$ and $(!B, \delta_B)$, i.e. making the following diagram commute:

$$\begin{tikzcd}
!A \arrow[r, "f"] \arrow[d, "\delta_A"'] \arrow[dr, phantom, "\mathrm{S}"] & !B \arrow[d, "\delta_B"] \\
!!A \arrow[r, "!f"'] & !!B
\end{tikzcd}$$

Then it must be the case that the following commutes:

$$\begin{tikzcd}
\top \arrow[d, equal] & !A \arrow[l, "e_A"'] \arrow[r, "d_A"] \arrow[d, "f"] & !A\otimes !A \arrow[d, "f\otimes f"] \\
\top & !B \arrow[l, "e_B"] \arrow[r, "d_B"'] & !B\otimes !B
\end{tikzcd}$$

It is straightforward to show that the previous diagram commutes using the definitions of the respective morphisms and the assumption that f is a coalgebras morphism.

□

Corollary 15. *The category* $\mathsf{Dial}_2(\mathsf{Sets})$ *is a full LNL model.*

Proof. This follows directly from the previous lemma and Theorem 11 which shows that linear categories are LNL models. □

5 Tensorial Logic

Melliès and Tabareau introduced tensorial logic as a means of generalizing linear logic to a theory of tensor and a non-involutive negation called tensorial negation. That is, instead of an isomorphism $A = \neg\neg A$ we have only a natural transformation $A \longrightarrow \neg\neg A$ [15]. Tensorial logic makes the claim that tensor and tensorial negation are more fundamental than tensor and negation defined via implication. This is at odds with FILL where implication is considered to be fundamental. In this section we show that multiplicative tensorial logic can be modeled by any symmetric monoidal closed category containing an object $\perp$ that is dual to the unit of tensor (Lemma 20). Thus, any model of FILL, e.g. $\mathsf{Dial}_2(\mathsf{Sets})$, can be seen as a model of multiplicative tensorial logic. In fact, we will show that tensorial negation arises as a simple property of implication (Lemma 19). In addition, we show that $\mathsf{Dial}_2(\mathsf{Sets})$ is not only a model of multiplicative tensorial logic, but a model of full tensorial logic.

A categorical model of tensorial logic is a symmetric monoidal category with a tensorial negation.

Definition 16. *A* ***tensorial negation*** *on a symmetric monoidal category* $(\mathcal{C}, \otimes, I)$ *is defined as a functor* $\neg : \mathcal{C} \longrightarrow \mathcal{C}^{\mathsf{op}}$ *together with a family of bijections* $\phi_{A,B,C} : \mathsf{Hom}_{\mathcal{C}}(A \otimes B, \neg C) \cong \mathsf{Hom}_{\mathcal{C}}(A, \neg(B \otimes C))$ *natural in* A*,* B*, and* C*. Furthermore, the following diagram must commute:*

$$\begin{tikzcd}
\mathsf{Hom}(A\otimes(B\otimes C),\neg D) \arrow[rr, "\mathsf{Hom}(\alpha_{A,B,C},\mathsf{id}_{\neg D})"] \arrow[dd, "\phi_{A,B\otimes C,D}"'] & & \mathsf{Hom}((A\otimes B)\otimes C,\neg D) \arrow[d, "\phi_{A\otimes B,C,D}"] \\
 & & \mathsf{Hom}(A\otimes B,\neg(C\otimes D)) \arrow[d, "\phi_{A,B,C\otimes D}"] \\
\mathsf{Hom}(A,\neg((B\otimes C)\otimes D)) \arrow[rr, "\mathsf{Hom}(\mathsf{id}_A,\neg\alpha_{B,C,D})"'] & & \mathsf{Hom}(A,\neg(B\otimes(C\otimes D)))
\end{tikzcd}$$

The most basic form of tensorial logic is called multiplicative tensorial logic and only consists of tensor and a tensorial negation. The model of multiplicative tensorial logic is called a dialogue category.

Definition 17. *A* ***dialogue category*** *is a symmetric monoidal category equipped with a tensorial negation.*

At this point we show that any symmetric monoidal closed category with an object, $\perp$, dual to the unit of tensor is a dialogue category.

Definition 18. *A* ***dual-unit monoidal category (DU-MC)****,* $(\mathcal{C}, \otimes, \top)$ *is a monoidal category with an object* $\perp$ *dual to* $\top$*.*

Having the dual to $\top$ in a monoidal closed category allows one to define a negation functor using the internal hom as $\neg A = A \multimap \perp$. We can show that this negation functor is actually a tensorial negation using the following result.

Lemma 19. *In any monoidal closed category, $\mathcal{C}$, there is a natural bijection $\phi_{A,B,C,D} : \mathsf{Hom}_{\mathcal{C}}(A \otimes B, C \multimap D) \cong \mathsf{Hom}_{\mathcal{C}}(A, (B \otimes C) \multimap D)$. Furthermore, the following diagram commutes:*

$$\begin{tikzcd}
\mathsf{Hom}(A \otimes (B \otimes C), D \multimap E) \arrow[r, "\mathsf{Hom}(\alpha_{A,B,C}, \mathsf{id}_{D \multimap E})"] \arrow[dd, "\phi_{A,B\otimes C,D,E}"'] & \mathsf{Hom}((A \otimes B) \otimes C, D \multimap E) \arrow[d, "\phi_{A\otimes B,C,D,E}"] \\
& \mathsf{Hom}(A \otimes B, (C \otimes D) \multimap E) \arrow[d, "\phi_{A,B,C\otimes D,E}"] \\
\mathsf{Hom}(A, ((B \otimes C) \otimes D) \multimap E) \arrow[r, "\mathsf{Hom}(\mathsf{id}_A, \alpha_{B,C,D} \multimap E)"'] & \mathsf{Hom}(A, (B \otimes (C \otimes D)) \multimap E)
\end{tikzcd}$$

Proof. Suppose $\mathcal{C}$ is a monoidal closed category. Then we can define $\phi(f : A \otimes B \longrightarrow C \multimap D) = \mathsf{cur}(\alpha^{-1}; \mathsf{cur}^{-1}(f))$ and $\phi^{-1}(g : A \longrightarrow (B \otimes C) \multimap D) = \mathsf{cur}(\alpha; \mathsf{cur}^{-1}(g))$. Clearly, these are mutual inverses, and hence, ϕ is a bijection. Naturality of ϕ easily follows.

Suppose $f : A \otimes (B \otimes C) \longrightarrow D \multimap E$. The required diagram commutes by the following equational reasoning:

$$\begin{array}{rcll}
\phi(\phi(\alpha; f)) & = & \phi(\mathsf{cur}(\alpha^{-1}; \mathsf{cur}^{-1}(\alpha; f))) & \text{(Definition)} \\
& = & \mathsf{cur}(\alpha^{-1}; \mathsf{cur}^{-1}(\mathsf{cur}(\alpha^{-1}; \mathsf{cur}^{-1}(\alpha; f)))) & \text{(Definition)} \\
& = & \mathsf{cur}(\alpha^{-1}; (\alpha^{-1}; \mathsf{cur}^{-1}(\alpha; f))) & \text{(Inverses)} \\
& = & \mathsf{cur}((\alpha^{-1}; \alpha^{-1}); \mathsf{cur}^{-1}(\alpha; f)) & \text{(Associativity)} \\
& = & \mathsf{cur}((\alpha^{-1}; \alpha^{-1}); (\alpha \otimes \mathsf{id}); \mathsf{cur}^{-1}(f)) & \text{(Naturality of } \mathsf{cur}) \\
& = & \mathsf{cur}((\mathsf{id} \otimes \alpha^{-1}); \alpha^{-1}; (\alpha^{-1} \otimes \mathsf{id}); (\alpha \otimes \mathsf{id}); \mathsf{cur}^{-1}(f)) & \text{(Monoidal Pentagon)} \\
& = & \mathsf{cur}((\mathsf{id} \otimes \alpha^{-1}); \alpha^{-1}; (\alpha^{-1}; \alpha \otimes \mathsf{id}); \mathsf{cur}^{-1}(f)) & \text{(Functorality)} \\
& = & \mathsf{cur}((\mathsf{id} \otimes \alpha^{-1}); \alpha^{-1}; (\mathsf{id} \otimes \mathsf{id}); \mathsf{cur}^{-1}(f)) & \text{(Inverses)} \\
& = & \mathsf{cur}((\mathsf{id} \otimes \alpha^{-1}); \alpha^{-1}; \mathsf{id}; \mathsf{cur}^{-1}(f)) & \text{(Functorality)} \\
& = & \mathsf{cur}((\mathsf{id} \otimes \alpha^{-1}); \alpha^{-1}; \mathsf{cur}^{-1}(f)) & \text{(Identity)} \\
& = & \mathsf{cur}(\alpha^{-1}; \mathsf{cur}^{-1}(f)); (\alpha^{-1} \multimap \mathsf{id}) & \text{(Naturality of } \mathsf{cur}) \\
& = & \phi(f); (\alpha^{-1} \multimap \mathsf{id}) & \text{(Definition)}
\end{array}$$

□

Replacing D and E in the previous result with $\perp$ yields the definition of tensorial negation using the negation functor $\neg A = A \multimap \perp$.

Lemma 20. *Any DU-SMCC, $(\mathcal{C}, \otimes, \top, \multimap, \perp)$, is a model of multiplicative tensorial logic.*

Proof. Suppose $(\mathcal{C}, \otimes, \top, \multimap, \perp)$ is a DU-SMCC. It suffices to show that that we can define a tensorial negation. Define the negation functor by $\neg A = A \multimap \perp$. Finally, by Lemma 19 $\neg A$ is a tensorial negation. Therefore, $\mathcal{C}$ is a dialogue category. □

Corollary 21. $\mathsf{Dial}_2(\mathsf{Sets})$ *is a model of multiplicative tensorial logic.*

Proof. We have already shown $\mathsf{Dial}_2(\mathsf{Sets})$ to be a model of FILL, and thus, has a SMCC structure as well as the dual to the unit of tensor which is the unit of par[3]. □

Extending a model of multiplicative tensorial logic with an exponential resource modality yields a model of full tensorial logic.

[3]We give a full proof using the previous result in the formalization see the file `https://github.com/heades/cut-fill-agda/blob/master/Tensorial.agda`.

Definition 22. *A **resource modality** on a symmetric monoidal category $(\mathcal{C}, \otimes, I)$ is an adjunction with a symmetric monoidal category $(\mathcal{M}, \otimes', I')$:*

$$
\begin{tikzcd}
\mathcal{M} \arrow[r, bend left, "F"] & \mathcal{C} \arrow[l, bend left, "G"]
\end{tikzcd}
$$

*A resource modality is called an **exponential resource modality** if $\mathcal{M}$ is cartesian where $\otimes'$ is the product and I' is the terminal object.*

A model of full tensorial logic is defined to be a model of multiplicative tensorial logic with an exponential resource modality. We now know that $\mathsf{Dial}_2(\mathsf{Sets})$ is a model of multiplicative tensorial logic. By constructing the co-Kleisli category which consists of the !-coalgebras as objects, and happens to be cartesian, we can show that $\mathsf{Dial}_2(\mathsf{Sets})$ is a model of full tensorial logic. The adjunction with the co-Kleisli category naturally arises from the proof that $\mathsf{Dial}_2(\mathsf{Sets})$ is a full LNL model (Corollary 15).

Lemma 23. *The category* $\mathsf{Dial}_2(\mathsf{Sets})$ *is a model of full tensorial logic.*

Proof. It suffices to show that there is an adjunction between $\mathsf{Dial}_2(\mathsf{Sets})$ and a cartesian category. Define the category $\mathsf{Dial}_2(\mathsf{Sets})_!$ as follows:

- Take as objects $(U, (U \Rightarrow X^*), \alpha_!)$ where U and X are sets, and $\alpha \subseteq U \times (U \Rightarrow X^*)$.
- Take as morphisms $(f, F) : (U, (U \Rightarrow X^*), \alpha_!) \longrightarrow (V, (V \Rightarrow Y^*), \beta_!)$ where $f : U \longrightarrow V$ and $F : (V \Rightarrow Y^*) \longrightarrow (U \Rightarrow X^*)$ subject to the same condition on morphisms as $\mathsf{Dial}_2(\mathsf{Sets})$. Composition and identities are defined similarly to $\mathsf{Dial}_2(\mathsf{Sets})$.

Now we show that $\mathsf{Dial}_2(\mathsf{Sets})_!$ is cartesian. Notice that $\mathsf{Dial}_2(\mathsf{Sets})_!$ is a subcategory of $\mathsf{Dial}_2(\mathsf{Sets})$, and there is a functor $J : \mathsf{Dial}_2(\mathsf{Sets}) \longrightarrow \mathsf{Dial}_2(\mathsf{Sets})_!$ which is defined equivalently to the endofunctor ! from the proof of Lemma 14. In fact, $\mathsf{Dial}_2(\mathsf{Sets})_!$ is the co-Kleisli category with objects free !-coalgebras and is cartesian closed [8]. However, we only need the fact that it is cartesian.

To show that $\mathsf{Dial}_2(\mathsf{Sets})_!$ is cartesian it suffices to show that J preserves the cartesian structure of $\mathsf{Dial}_2(\mathsf{Sets})$ – the proof that $\mathsf{Dial}_2(\mathsf{Sets})$ is cartesian can be found on page 48 of [7].

- Suppose $A = (U, X, \alpha)$ and $B = (V, Y, \beta)$ are objects of $\mathsf{Dial}_2(\mathsf{Sets})$. Then we define $A \times B = (U \times V, (X + Y), \alpha \times \beta)$, where $((u, v), i) \in \alpha \times \beta$ iff when $i \in X$, then $(u, i) \in \alpha$, otherwise when $i \in Y$, then $(v, i) \in \beta$. Now the cartesian product in $\mathsf{Dial}_2(\mathsf{Sets})_!$ is defined as $J(A \times B)$.
- The terminal object in $\mathsf{Dial}_2(\mathsf{Sets})$ is defined by $(\top, \bot, \alpha_\top)$ where $\top$ and $\bot$ are the terminal and initial objects in Sets respectively, and $(x, y) \in \alpha_\top$ iff true. The proof that this is terminal, and is the unit to the cartesian product can be found in the formal development.
- Suppose $A = (U, X, \alpha)$, $B = (V, Y, \beta)$, and $C = (W, Z, \gamma)$ are objects of $\mathsf{Dial}_2(\mathsf{Sets})$. Then the we define the following morphisms – the proofs of the morphism conditions have been omitted, but the details can be found in the formal development:
  - Define the first projection by $(\pi_1, F_{\pi_1}) : J(A \times B) \longrightarrow J(A)$, where π_1 is the first projection in Sets, and F_{π_1} takes a function $f : U \longrightarrow X^*$ and a pair $(u, v) \in U \times V$, and returns the sequence $(X + Y)^*$ by mapping the first coproduct injection over $f(u, v)$.
  - The second projection is defined similar to the first, $(\pi_2, F_{\pi_2}) : J(A \times B) \longrightarrow J(B)$, but F_{π_2} maps the second projection instead of the first.
  - Suppose $j_1 = (f, F) : J(C) \longrightarrow J(A)$ and $j_2 = (g, G) : J(C) \longrightarrow J(B)$ are morphisms in $\mathsf{Dial}_2(\mathsf{Sets})_!$. Then we define the morphism $(j_1, j_2) = ((f, g), \lambda j.\lambda w.F(h_1(j), w) \circ F(h_2(j), w)) : J(C) \longrightarrow J(A \times B)$, where $(f, g) = \lambda w.(f(w), g(w))$ is the unique morphism from the cartesian structure of Sets, and $h_1(j) = \lambda u.\iota_1(j(u, g(w)))$ and $h_2(j) = \lambda u.\iota_2(j(u, f(w)))$. Notice that F is defined in terms of unique morphisms, and thus, (j_1, j_2) is unique.

$$\frac{}{A \vdash A}\ \text{LL\_Ax} \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ \text{LL\_Cut} \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\ \text{LL\_Tl} \qquad \frac{}{\cdot \vdash \top}\ \text{LL\_Tr}$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}\ \text{LL\_Tenl} \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'}\ \text{LL\_Tenr} \qquad \frac{}{\perp \vdash \cdot}\ \text{LL\_Pl} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \perp, \Delta}\ \text{LL\_Pr}$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \parr B \vdash \Delta, \Delta'}\ \text{LL\_Parl} \qquad \frac{\Gamma \vdash \Delta, A, B, \Delta'}{\Gamma \vdash \Delta, A \parr B, \Delta'}\ \text{LL\_Parr} \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, A \multimap B, \Gamma' \vdash \Delta, \Delta'}\ \text{LL\_Impl}$$

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta}\ \text{LL\_Impr} \qquad \frac{\Gamma, A, B \vdash \Delta}{\Gamma, B, A \vdash \Delta}\ \text{LL\_Exl} \qquad \frac{\Gamma \vdash \Delta_1, A, B, \Delta_2}{\Gamma \vdash \Delta_1, B, A, \Delta_2}\ \text{LL\_Exr}$$

Figure 3: Multi-Conclusion Linear Logic

All of the previous morphisms satisfy the usual diagram for cartesian products.

By Lemma 15 we know $\mathsf{Dial}_2(\mathsf{Sets})$ is a full LNL model, and thus, there is an adjunction between $\mathsf{Dial}_2(\mathsf{Sets})$ and $\mathsf{Dial}_2(\mathsf{Sets})_!$ where the left-adjoint is the forgetful functor $G : \mathsf{Dial}_2(\mathsf{Sets})_! \longrightarrow \mathsf{Dial}_2(\mathsf{Sets})$, and the right adjoint is the free functor $J : \mathsf{Dial}_2(\mathsf{Sets}) \longrightarrow \mathsf{Dial}_2(\mathsf{Sets})_!$. Therefore, $\mathsf{Dial}_2(\mathsf{Sets})$ is a model of full tensorial logic. $\square$

5.1 Double Negation Translation

In this section we show that we can use intuitionistic negation – which we showed was tensorial in the previous section – to define a negative translation of multi-conclusion linear logic (Figure 3) into FILL where implication plays a central role. Melliès and Tabareau give a negative translation of the multiplicative fragment of linear logic into tensorial logic [16] using tensor as the main connective. For example, they define $(A \otimes B)^N = \neg(\neg(A)^N \otimes \neg(B)^N)$, and thus, they simulate par using tensor and negation. This definition would cause some syntactic issues with FILL, because the left-rule to par requires the let-pattern term defined in Definition 3, thus, encoding par in terms of tensor would require the let-pattern term to be used in the left-rule for tensor. It is understandable why this definition is defined the way it is given that they are translating into tensorial logic. However, in FILL we have par, and so we modify their translation into one that better fits FILL.

The following definition defines a translation of linear logic formulas into FILL formulas.

Definition 24. *The following is the double-negation translation of linear logic into FILL:*

$$\begin{array}{lcl}
(\top)^N & = & \top \\
(\perp)^N & = & \perp \\
(A^{\perp})^N & = & \neg((A)^N) \\
(A \parr B)^N & = & \neg\neg((A)^N) \parr \neg\neg((B)^N) \\
(A \otimes B)^N & = & \neg\neg((A)^N) \otimes \neg\neg((B)^N)
\end{array}$$

The main point of the previous definition is that because FILL has intuitionistic versions of all of the operators of linear logic we can give a very natural translation that preserves these operators by double negating their arguments.

At this point we need to extend the translation of linear logic formulas to sequents. However, we must be careful, because in FILL implication has the FILL restriction, and thus, if we choose the wrong translation then we will run into problems trying to satisfy the FILL condition. The method we employ here is to first translate a linear logic sequent into a single-sided sequent, and then translate that to FILL using the well-known translation. That is, it is easy to see that any linear logic sequent $A_1, \ldots, A_i \vdash B_1, \ldots, B_j$ is logically equivalent to the sequent $\cdot \vdash {A_1}^{\perp}, \dots, {A_i}^{\perp}, B_1, \dots, B_j$. Then we translate the latter into FILL as $x_1 : \neg(({A_1}^{\perp})^N), \dots, x_i : \neg(({A_i}^{\perp})^N), y_1 : \neg((B_1)^N), \dots, y_j : \neg((B_j)^N) \vdash \cdot$ for any free variables $x_1, \dots, x_i$ and $y_1, \dots, y_j$, but this is equivalent to $x_1 : \neg\neg((A_1)^N), \dots, x_i : \neg\neg((A_i)^N), y_1 : \neg((B_1)^N), \dots, y_j : \neg((B_j)^N) \vdash \cdot$. The astute reader will realize that this is indeed the translation of single-sided linear logic into single-conclusion intuitionistic linear logic. This translation also has the benefit that we do not have to worry about mentioning terms in the statement of the result.

Lemma 25 (Negative Translation)**.** *If $A_1, \dots, A_i \vdash B_1, \dots, B_j$ is derivable, then for any unique fresh variables $x_1, \dots, x_i$, and $y_1, \dots, y_j$, the sequent $x_1 : \neg\neg((A_1)^N), \dots, x_i : \neg\neg((A_i)^N), y_1 : \neg((B_1)^N), \dots, y_j : \neg((B_j)^N) \vdash \cdot$ is derivable.*

Proof. This is a proof by induction on the form of the assumed sequent $A_1, \dots, A_i \vdash B_1, \dots, B_j$. Throughout the proof if $\Gamma = A_1, \dots, A_i$, then $\neg((\Gamma)^N) = x_1 : \neg((A_1)^N), \dots, x_i : \neg((A_i)^N)$ for some unique fresh variables $x_1, \dots, x_i$. We will use this notation when applying the inductive hypothesis to make the presentation easier to read.

Case.

$$\dfrac{}{A \vdash A}\ \textsc{Ax}$$

It suffices to derive the sequent $x : \neg\neg((A)^N), y : \neg((A)^N) \vdash \cdot$ which is equivalent to the sequent $x : \neg((A)^N) \multimap \perp, y : \neg((A)^N) \vdash \cdot$. The latter is derivable as follows:

$$\dfrac{\dfrac{}{y : \neg((A)^N) \vdash y : \neg((A)^N)}\ \textsc{Ax} \qquad \dfrac{}{w : \perp \vdash \cdot}\ \textsc{Pl}}{x : \neg((A)^N) \multimap \perp, y : \neg((A)^N) \vdash \cdot}\ \textsc{ImpL}$$

Case.

$$\dfrac{\Gamma_1 \vdash A, \Delta_1 \quad \Gamma_2, A \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ \textsc{Cut}$$

By the induction hypothesis we know the following:

$$\neg(\neg((\Gamma_1)^N)), x : \neg((A)^N), \neg((\Delta_1)^N) \vdash \cdot$$
$$\neg(\neg((\Gamma_2)^N)), y : \neg(\neg((A)^N)), \neg((\Delta_2)^N) \vdash \cdot$$

It suffices to derive the sequent $\neg(\neg((\Gamma_1)^N)), \neg(\neg((\Gamma_2)^N)), \neg((\Delta_1)^N), \neg((\Delta_2)^N) \vdash \cdot$ which we do as follows:

$$\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma_1)^N)), x : \neg((A)^N), \neg((\Delta_1)^N) \vdash \cdot}{\neg(\neg((\Gamma_1)^N)), x : \neg((A)^N), \neg((\Delta_1)^N) \vdash \circ : \perp}\ \textsc{Pr}}{\neg(\neg((\Gamma_1)^N)), \neg((\Delta_1)^N) \vdash \lambda x.\circ : \neg(\neg((A)^N))}\ \textsc{ImpR} \qquad \neg(\neg((\Gamma_2)^N)), y : \neg(\neg((A)^N)), \neg((\Delta_2)^N) \vdash \cdot}{\neg(\neg((\Gamma_1)^N)), \neg(\neg((\Gamma_2)^N)), \neg((\Delta_1)^N), \neg((\Delta_2)^N) \vdash \cdot}\ \textsc{Cut}$$

Case.

$$\dfrac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\ \textsc{Tl}$$

First, note that $(\top)^N = \top$. By the induction hypothesis we know $\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \cdot$. We obtain our result by the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), z : \top \vdash \cdot}\ \textsc{Tl}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), z : \top \vdash t :\perp}\ \textsc{Pr}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash t : \neg\top}\ \textsc{ImpR} \quad \dfrac{}{y :\perp\vdash \cdot}\ \textsc{Pl}}{\neg(\neg((\Gamma)^N)), x : \neg(\neg\top), \neg((\Delta)^N) \vdash \cdot}\ \textsc{ImpL}$$

Note that $\mathsf{let}\, x\, \mathsf{be}\, *\, \mathsf{in} \cdot = \cdot$ for any variable x.

Case.

$$\dfrac{}{\cdot \vdash \top}\ \textsc{Tr}$$

If suffices to derive the sequent $x : \neg((\top)^N) \vdash \cdot$, but this is equivalent to the sequent $x : \neg\top \vdash \cdot$ which is derivable as follows:

$$\dfrac{\dfrac{}{\cdot \vdash * : \top}\ \textsc{Tr} \qquad \dfrac{}{y :\perp\vdash \cdot}\ \textsc{Pl}}{x : \neg\top \vdash \cdot}\ \textsc{ImpL}$$

Case.

$$\dfrac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}\ \textsc{TenL}$$

By the induction hypothesis we have the following sequent:

$$\neg(\neg((\Gamma)^N)), x : \neg\neg((A)^N), y : \neg\neg((B)^N), \neg((\Delta)^N) \vdash \cdot$$

Our result follows from the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), w : \neg\neg((A)^N), v : \neg\neg((B)^N), \neg((\Delta)^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), w : \neg\neg((A)^N), v : \neg\neg((B)^N), \neg((\Delta)^N) \vdash \circ :\perp}\ \textsc{Pr}}{\neg(\neg((\Gamma)^N)), x : \neg\neg((A)^N) \otimes \neg\neg((B)^N), \neg((\Delta)^N) \vdash \mathsf{let}\, x\, \mathsf{be}\, (w \otimes v)\, \mathsf{in} \circ :\perp}\ \textsc{TenL}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \lambda x.\mathsf{let}\, x\, \mathsf{be}\, (w \otimes v)\, \mathsf{in} \circ : \neg(\neg\neg((A)^N) \otimes \neg\neg((B)^N))}\ \textsc{ImpR} \quad \dfrac{}{y :\perp\vdash \cdot}\ \textsc{Pl}}{\neg(\neg((\Gamma)^N)), z : \neg\neg(\neg\neg((A)^N) \otimes \neg\neg((B)^N)), \neg((\Delta)^N) \vdash \cdot}\ \textsc{ImpL}$$

Case.

$$\dfrac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'}\ \textsc{TenR}$$

By the induction hypothesis we know the following:

$$\begin{array}{l} \neg(\neg((\Gamma)^N)), x : \neg((A)^N), \neg((\Delta)^N) \vdash \cdot \\ \neg(\neg((\Gamma')^N)), y : \neg((B)^N), \neg((\Delta')^N) \vdash \cdot \end{array}$$

We obtain our result by the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), x : \neg((A)^N), \neg((\Delta)^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), x : \neg((A)^N), \neg((\Delta)^N) \vdash \circ : \perp}\ \textsc{Pr}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \lambda x.\circ : \neg\neg((A)^N)}\ \textsc{ImpR} \qquad \dfrac{\dfrac{\neg(\neg((\Gamma')^N)), y : \neg((B)^N), \neg((\Delta')^N) \vdash \cdot}{\neg(\neg((\Gamma')^N)), y : \neg((B)^N), \neg((\Delta')^N) \vdash \circ : \perp}\ \textsc{Pr}}{\neg(\neg((\Gamma')^N)), \neg((\Delta')^N) \vdash \lambda y.\circ : \neg\neg((B)^N)}\ \textsc{ImpR}}{\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash (\lambda x.\circ) \otimes (\lambda y.\circ) : \neg\neg((A)^N) \otimes \neg\neg((B)^N)}\ \textsc{TenR} \qquad \dfrac{}{w : \perp \vdash \cdot}\ \textsc{Pl}}{\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), z : \neg(\neg\neg((A)^N) \otimes \neg\neg((B)^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \cdot}\ \textsc{ImpL}$$

Case.

$$\dfrac{}{\perp \vdash \cdot}\ \textsc{Pl}$$

It suffices to show that $z : \neg((\perp)^N) \vdash \cdot$ is derivable, which is equivalent to showing that the sequent $z : \neg\neg(\perp) \vdash \cdot$ is derivable. We obtain our result by the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x : \perp \vdash \cdot}\ \textsc{Pl}}{x : \perp \vdash \circ : \perp}\ \textsc{Pr}}{\cdot \vdash \lambda x.\circ : \neg(\perp)}\ \textsc{ImpR} \qquad \dfrac{}{w : \perp \vdash \cdot}\ \textsc{Pl}}{z : \neg\neg(\perp) \vdash \cdot}\ \textsc{ImpL}$$

Case.

$$\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \perp, \Delta}\ \textsc{Pr}$$

By the induction hypothesis we know the following sequent is derivable:

$$\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \cdot$$

It suffices to show that $\neg(\neg((\Gamma)^N)), z : \neg((\perp)^N), \neg((\Delta)^N) \vdash \cdot$ which is equivalent to the sequent $\neg(\neg((\Gamma)^N)), z : \neg(\perp), \neg((\Delta)^N) \vdash \cdot$. The latter is derivable as follows:

$$\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \circ : \perp}\ \textsc{Pr} \qquad \dfrac{}{w : \perp \vdash \cdot}\ \textsc{Pl}}{\neg(\neg((\Gamma)^N)), z : \neg(\perp), \neg((\Delta)^N) \vdash \cdot}\ \textsc{ImpL}$$

Case.

$$\dfrac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \parr B \vdash \Delta, \Delta'}\ \textsc{ParL}$$

By the induction hypothesis we know the following:

$$\neg(\neg((\Gamma)^N)), x : \neg\neg((A)^N), \neg((\Delta)^N) \vdash \cdot$$
$$\neg(\neg((\Gamma')^N)), v : \neg\neg((B)^N), \neg((\Delta')^N) \vdash \cdot$$

It suffices to show that the sequent

$$\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), z : \neg\neg((A \parr B)^N), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \cdot$$

is derivable which is equivalent to the sequent

$$\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), z : \neg\neg(\neg\neg((A)^N) \parr \neg\neg((B)^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \cdot.$$

The latter is derivable as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), x : \neg\neg((A)^N), \neg((\Delta)^N) \vdash \cdot \qquad \neg(\neg((\Gamma')^N)), v : \neg\neg((B)^N), \neg((\Delta')^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), y : \neg\neg((A)^N) \parr \neg\neg((B)^N), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \cdot}\,\textsc{ParL}}{\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), y : \neg\neg((A)^N) \parr \neg\neg((B)^N), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \circ : \bot}\,\textsc{Pr}}{\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \lambda y.\circ : \neg(\neg\neg((A)^N) \parr \neg\neg((B)^N))}\,\textsc{ImpR} \qquad \dfrac{}{w : \bot \vdash \cdot}\,\textsc{Pl}}{\neg(\neg((\Gamma)^N)), \neg(\neg((\Gamma')^N)), z : \neg\neg(\neg\neg((A)^N) \parr \neg\neg((B)^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash \cdot}\,\textsc{ImpL}$$

Case.

$$\dfrac{\Gamma \vdash \Delta, A, B, \Delta'}{\Gamma \vdash \Delta, A \parr B, \Delta'}\,\textsc{ParR}$$

By the induction hypothesis we know the following is derivable:

$$\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), x : \neg((A)^N), y : \neg((B)^N), \neg((\Delta')^N) \vdash \cdot$$

It suffices to show that the sequent $\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), z : \neg((A \parr B)^N), \neg((\Delta')^N)$ is derivable, but this sequent is equivalent to the sequent $\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), z : \neg(\neg\neg((A)^N) \parr \neg\neg((B)^N)), \neg((\Delta')^N)$. We derive the latter as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), x : \neg((A)^N), y : \neg((B)^N), \neg((\Delta')^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), x : \neg((A)^N), y : \neg((B)^N), \neg((\Delta')^N) \vdash \circ : \bot}\,\textsc{Pr}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), x : \neg((A)^N), \neg((\Delta')^N) \vdash (\lambda y.\circ) : \neg\neg((B)^N)}\,\textsc{ImpR}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), x : \neg((A)^N), \neg((\Delta')^N) \vdash \circ : \bot, (\lambda y.\circ) : \neg\neg((B)^N)}\,\textsc{Pr}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash (\lambda x.\circ) : \neg\neg((A)^N), (\lambda y.\circ) : \neg\neg((B)^N)}\,\textsc{ImpR}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), \neg((\Delta')^N) \vdash (\lambda x.\circ) \parr (\lambda y.\circ) : \neg\neg((A)^N) \parr \neg\neg((B)^N)}\,\textsc{ParR} \qquad \dfrac{}{w : \bot \vdash \cdot}\,\textsc{Pl}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N), z : \neg(\neg\neg((A)^N) \parr \neg\neg((B)^N)), \neg((\Delta')^N)}\,\textsc{ImpL}$$

Case.

$$\dfrac{\Gamma, A \vdash \Delta}{\Gamma \vdash A^{\perp}, \Delta}\,\textsc{NegL}$$

By the induction hypothesis we know the following:

$$\neg(\neg((\Gamma)^N)), x : \neg\neg((A)^N), \neg((\Delta)^N) \vdash \cdot$$

It suffices to show that the sequent $\neg(\neg((\Gamma)^N)), x : \neg((A^{\perp})^N), \neg((\Delta)^N) \vdash \cdot$, but this is equivalent to the sequent $\neg(\neg((\Gamma)^N)), x : \neg\neg((A)^N), \neg((\Delta)^N) \vdash \cdot$, but this is equivalent to the induction hypothesis.

Case.

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, A^{\perp} \vdash \Delta}\ \textsc{Negl}$$

By the induction hypothesis we know the following:

$$\neg(\neg((\Gamma)^N)), x : \neg((A)^N), \neg((\Delta)^N) \vdash \cdot$$

It suffices to show that the sequent $\neg(\neg((\Gamma)^N)), z : \neg\neg((A^{\perp})^N), \neg((\Delta)^N) \vdash \cdot$, but this is equivalent to the sequent $\neg(\neg((\Gamma)^N)), z : \neg\neg(\neg((A)^N)), \neg((\Delta)^N) \vdash \cdot$. The latter is derived as follows:

$$\dfrac{\dfrac{\dfrac{\neg(\neg((\Gamma)^N)), x : \neg((A)^N), \neg((\Delta)^N) \vdash \cdot}{\neg(\neg((\Gamma)^N)), x : \neg((A)^N), \neg((\Delta)^N) \vdash \circ : \perp}\ \textsc{Pr}}{\neg(\neg((\Gamma)^N)), \neg((\Delta)^N) \vdash \lambda x.\circ : \neg(\neg((A)^N))}\ \textsc{Impr} \qquad w : \perp \vdash \cdot}{\neg(\neg((\Gamma)^N)), z : \neg\neg(\neg((A)^N)), \neg((\Delta)^N) \vdash \cdot}\ \textsc{Impl}$$

Case.

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, B, A \vdash \Delta}\ \textsc{Exl}$$

This case follows by first applying the induction hypothesis and then applying the rule Exl.

Case.

$$\frac{\Gamma \vdash \Delta_1, A, B, \Delta_2}{\Gamma \vdash \Delta_1, B, A, \Delta_2}\ \textsc{Exr}$$

This case follows by first applying the induction hypothesis and then applying the rule Exl.

□

6 Conclusion and Future Work

We first gave the definition of full intuitionistic linear logic using the left rule for par proposed by Bellin in Section 2. We then directly proved cut-elimination of FILL in Section 3 by adapting the well-known cut-elimination procedure for classical linear logic to FILL. In Section 4 we showed that the category $\mathsf{Dial}_2(\mathsf{Sets})$, a model of FILL, is a full LNL model by showing that it is a full linear category, and then replaying the proof that linear categories are LNL models by Benton. In Section 5 we showed that $\mathsf{Dial}_2(\mathsf{Sets})$ is a model of full tensorial logic. Finally, we gave a double-negation translation of multi-conclusion linear logic into FILL that explicitly uses both tensor and par.

Future work. Lorenzen games are a particular type of game semantics for various logics developed by Lorenz, Felscher, and Rahman et al. [12, 18]. Lorenzen games consist of a first-order language consisting of the logical connectives of the logic one wishes to study. Then the structure of the games are defined by two types of rules: particle rules and structural rules. The particle rules describe how formulas can be attacked or defended based on the formulas main connective. Then the structural rules orchestrate the particle rules as the game progresses. They describe the overall organization of the game.

Rahman showed that Lorenzen games could be defined for classical linear logic [17]. He was able to define a sound and complete semantics in Lorenzen games for classical linear logic, but he does mention that one could adopt a particular structural rule that enforces intuitionism. We plan to show that by adopting this rule we actually obtain a sound and complete semantics in Lorenzen games for FILL.

References

[1] Gianluigi Bellin. Subnets of proof-nets in multiplicative linear logic with mix. *Mathematical Structures in Computer Science*, 7:663–669, 12 1997.

[2] P. N. Benton. A mixed linear and non-linear logic: Proofs, terms and models (preliminary report). Technical Report UCAM-CL-TR-352, University of Cambridge Computer Laboratory, 1994.

[3] G. M. Bierman. *On Intuitionistic Linear Logic.* PhD thesis, Wolfson College, Cambridge, December 1993.

[4] Gavin Bierman. A note on full intuitionistic linear logic. *Annals of Pure and Applied Logic*, 79(3):281 – 287, 1996.

[5] Torben Braüner and Valeria Paiva. A formulation of linear logic based on dependency-relations. In Mogens Nielsen and Wolfgang Thomas, editors, *Computer Science Logic*, volume 1414 of *Lecture Notes in Computer Science*, pages 129–148. Springer Berlin Heidelberg, 1998.

[6] R. Clouston, J. Dawson, R. Gore, and A. Tiu. Annotation-Free Sequent Calculi for Full Intuitionistic Linear Logic – Extended Version. *ArXiv e-prints*, July 2013.

[7] Valeria de Paiva. *The Dialectica Categories.* PhD thesis, University of Cambridge, 1988.

[8] Valeria de Paiva. Dialectica categories. In J. Gray and A. Scedrov, editors, *Categories in Computer Science and Logic*, volume 92, pages 47–62. American Mathematical Society, 1989.

[9] Valeria de Paiva and Luiz Carlos Pereira. A short note on intuitionistic propositional logic with multiple conclusions. *MANUSCRITO - Rev. Int. Fil.*, 28(2):317 – 329, jul-dez 2005.

[10] A. G. Dragalin. *Mathematical Intuitionism. Introduction to Proof Theory*, volume 67 of *Translations of Mathematical Monographs*. Amerian Mathematical Society, 1988.

[11] Martin Hyland and Valeria de Paiva. Full intuitionistic linear logic (extended abstract). *Annals of Pure and Applied Logic*, 64(3):273 – 291, 1993.

[12] Laurent Keiff. Dialogical logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy.* Summer 2011 edition, 2011.

[13] S. Maehara. Eine darstellung der intuitionistischen logik in der klassischen. *Nagoya Mathematical Journal*, 7:45–64, 1954.

[14] Paul-Andre Mellies. *Categorical Semantics of Linear Logic.* 2009.

[15] Paul-André Melliés and Nicolas Tabareau. Linear continuations and duality. Technical report, Université Paris VII, 2008.

[16] Paul-André Melliès and Nicolas Tabareau. Resource modalities in tensor logic. *Annals of Pure and Applied Logic*, 161(5):632 – 653, 2010.

[17] S. Rahman. Un desafio para las teorias cognitivas de la competencia log- ica: los fundamentos pragmaticos de la semantica de la logica linear. *Manuscrito*, XXV(2):381–432, October 2002.

[18] Shahid Rahman and Laurent Keiff. On how to be a dialogician. In Daniel Vanderveken, editor, *Logic, Thought and Action*, volume 2 of *Logic, Epistemology, and the Unity of Science*, pages 359–408. Springer Netherlands, 2005.

[19] Harold Schellinx. Some syntactical observations on linear logic. *Journal of Logic and Computation*, 1(4):537–559, 1991.

[20] R. Seely. Linear logic, *-autonomous categories and cofree coalgebras. In *Computer Science Logic*, 1989.

[21] G. Takeuti. *Proof Theory.* Amsterdam: North-Holland, 1975.