Agda and Dialectica Categories for the Lambek Calculus

Valeria de Paiva and Harley Eades III

August 2016

Introduction

Lambek introduced his homonymous calculus (originally called the 'Syntactic Calculus') for proposed applications in Linguistics. However the calculus got much of its cult following and reputation by being a convenient, well-behaved prototype of a Gentzen sequent calculus without any structural rules.

This note recalls a Dialectica model of the Lambek Calculus presented by the first author in the Amsterdam Colloquium in 1991 [5]. Here, like then, we approach the Lambek Calculus from the perspective of Linear Logic, so we are interested in the basic sequent calculus with no structural rules, except associativity of tensors. In that earlier work we took for granted the syntax of the calculus and only discussed the exciting possibilities of categorical models of linear-logic-like systems. Many years later we find that the work on models is still interesting and novel and that it might inform some of the most recent work on the relationship between categorial grammars and notions of distributional semantics [4].

Moreover, the type theoretical notions that were left undiscussed are now more amenable to verification, using some new automated tools. Since the Amsterdam Colloquium proceedings were never published, we have decided to revisit some of the mathematical work, using the new tools that have been developed for type theory and sequent proof systems in the time that elapsed. Thus, we implement the calculus in Agda [3] and we use `Ott` [13] to check that we do not have trivial mistakes in our term systems.

The goal is to show that our implementation can shed new light on some of the issues that remained open. Mostly we wanted to check the correctness of the semantic proposals put forward since Szabo's seminal book [14] and, for future work, on the applicability and fit of the original systems to their intended uses.

Overview

The Syntactic Calculus was first introduced by Joachim Lambek in 1958 [11]. Since then the rechristened Lambek Calculus has had as its motivation provide

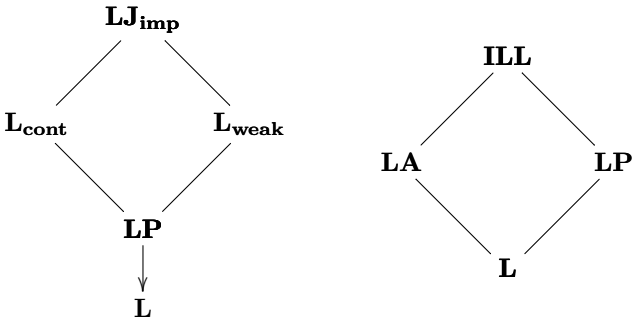
an explanation of the mathematics of sentence structure, starting from the author's algebraic intuitions. The Lambek calculus is the core of logical Categorial Grammar. The first use of the term "categorial grammar" seems to be in the title of Bar-Hillel, Gaifman and Shamir (1960), but categorial grammar began with Ajdukiewicz (1935) quite a few years earlier. After a period of ostracism, around 1980 the Lambek Calculus was taken up by logicians interested in Computational Linguistics, especially the ones interested in Categorial Grammars.

The work on Categorial Grammar was given a serious impulse by the advent of Girard's Linear Logic at the end of the 1980s. Girard [7] showed that there is a full embedding, preserving proofs, of Intuitionistic Logic into Linear Logic with a modality "!". This meant that Linear Logic, while paying attention to resources, could always code up faithfully Classical Logic and hence one could, as Girard put it, 'have one's cake and eat it', paying attention to resources, if desired, but always forgetting this accounting, if preferred. This meant also that several new systems of resource logics were conceived and developed and these refined resource logics were applied to several areas of Computer Science.

In Computational Linguistics, the Lambek calculus has seen a significant number of works written about it, apart from a number of monographs that deal with logical and linguistic aspects of the generalized type-logical approach. For a shorter introduction, see Moortgat's entry on the Stanford Encyclopedia of Philosophy on Type Logical Grammar [12].

Type Logical Grammar situates the type-logical approach within the framework of Montague's Universal Grammar and presents detailed linguistic analyses for a substantive fragment of syntactic and semantic phenomena in the grammar of English. Type Logical Semantics offers a general introduction to natural language semantics studied from a type-logical perspective.

This meant that a series of systems, implemented or not, were devised that used the Lambek Calculus or variants of Linear Logic as their basis. These systems can be as expressive as Intuitionistic Logic and the claim is that they are more precise i.e. they make finer distinctions. From the beginning it was clear that the Lambek Calculus is the multiplicative fragment of non-commutative Intuitionistic Linear Logic (there was a dispute as the original system did not introduce the constant corresponding to the nullary case of the tensor product, here written as I). Hence several interesting questions, considered for Linear Logic, could also be asked of the Lambek Calculus. One of them, posed by Morrill et al is whether we can extend the Lambek calculus with a modality that does for the structural rule of *(exchange)* what the modality *of course* '!' does for the rules of *(weakening)* and *(contraction)*. A preliminary proposal, which answers this question affirmatively, is set forward in this paper. The answer was provided in semantical terms in the first version of this work. Here we provide also the more syntactic description of these modalities. Building up from work of Galatos, Ciabattoni and Terui in [**?**] and others that describe how to transform systems of axioms into cut-free sequent rules, we aim to refine the algebraization of proof theory.



We first recall Linear Logic and provide the transformations to show that the Lambek Calculus L really is the multiplicative fragment of non-commutative Intuitionistic Linear Logic. Then we describe the usual String Semantics for the Lambek Calculus L and generalize it, using a categorical perspective in the second section. The third section recalls our Dialectica model for the Lambek Calculus. Finally, in the fourth section we discuss modalities and some untidiness of the Curry-Howard correspondence for the fragments of Linear Logic in question.

1 The Lambek Calculus

The Lambek Calculus, formerly the Syntactic Calculus L, due to J. Lambek [11], was created to capture the logical structure of sentences. Lambek introduced what we think of as a substructural logic with an operator denoting concatenation, $A \otimes B$, and two implications relating the order of phrases, $A \leftharpoonup B$ and $A \rightharpoonup B$. The first implication corresponds to a phrase of type A when followed by a phrase of type B, and the second is a phrase of type B when proceeded by a phrase of type A.

The Lambek Calculus can be presented as a non-commutative intuitionistic multiplicative linear logic. The syntax of formulas and contexts of the logic are as follows:

$$\begin{array}{ll} \text{(formulas)} & A, B, C ::= I \mid A \otimes B \mid A \leftharpoonup B \mid A \rightharpoonup B \\ \text{(contexts)} & \Gamma ::= A \mid \Gamma_1, \Gamma_2 \end{array}$$

We denote mapping the modalities over an arbitrary context by $!\Gamma$ and $\kappa\Gamma$. The inference rules are defined in Figure 1.

Because the operator $A \otimes B$ denotes the type of concatenations the types $A \otimes B$ and $B \otimes A$ are not equivalent, and hence, L is non-commutative which explains why implication must be broken up into two operators $A \leftharpoonup B$ and $A \rightharpoonup B$. In the following subsections we give two extensions of L: one with the well-known modality of-course of linear logic which adds weakening and contraction, and a second with a new modality adding exchange.

$$\frac{}{A \vdash A}\ \textsc{ax} \qquad \frac{}{\cdot \vdash I}\ \textsc{UR} \qquad \frac{\Gamma_2 \vdash A \qquad \Gamma_1, A, \Gamma_3 \vdash B}{\Gamma_1, \Gamma_2, \Gamma_3 \vdash B}\ \textsc{cut}$$

$$\frac{\Gamma_1, \Gamma_2 \vdash A}{\Gamma_1, I, \Gamma_2 \vdash A}\ \textsc{UL} \qquad \frac{\Gamma, A, B, \Gamma' \vdash C}{\Gamma, A \otimes B, \Gamma' \vdash C}\ \textsc{TL}$$

$$\frac{\Gamma_1 \vdash A \qquad \Gamma_2 \vdash B}{\Gamma_1, \Gamma_2 \vdash A \otimes B}\ \textsc{TR} \qquad \frac{\Gamma_2 \vdash A \qquad \Gamma_1, B, \Gamma_3 \vdash C}{\Gamma_1, A \rightharpoonup B, \Gamma_2, \Gamma_3 \vdash C}\ \textsc{IRL}$$

$$\frac{\Gamma_2 \vdash A \qquad \Gamma_1, B, \Gamma_3 \vdash C}{\Gamma_1, \Gamma_2, B \leftharpoonup A, \Gamma_3 \vdash C}\ \textsc{ILL} \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightharpoonup B}\ \textsc{IRR}$$

$$\frac{A, \Gamma \vdash B}{\Gamma \vdash B \leftharpoonup A}\ \textsc{ILR}$$

Figure 1: The Lambek Calculus: L

2 Extensions to the Lambek Calculus

The linear modality, $!A$, read "of-course A" was first proposed by Girard [7] as a means of encoding non-linear logic in both classical and intuitionistic forms in linear logic. For example, non-linear implication $A \rightharpoonup B$ is usually encoded into linear logic by $!A \multimap B$. Since we have based L on non-commutative intuitionistic linear logic it is straightforward to add the of-course modality to L. The rules for the of-course modality are defined by the following rules:

$$\frac{\Gamma_1, !A, \Gamma_2, !A, \Gamma_3 \vdash B}{\Gamma_1, !A, \Gamma_2, \Gamma_3 \vdash B}\ \textsc{C} \qquad \frac{\Gamma_1, \Gamma_2 \vdash B}{\Gamma_1, !A, \Gamma_2 \vdash B}\ \textsc{W}$$

$$\frac{!\Gamma \vdash B}{!\Gamma \vdash !B}\ \textsc{BR} \qquad \frac{\Gamma_1, A, \Gamma_2 \vdash B}{\Gamma, !A, \Gamma_2 \vdash B}\ \textsc{BL}$$

The rules C and W add contraction and weakening to L in a controlled way. Then the other two rules allow for linear formulas to be injected into the modality; and essentially correspond to the rules for necessitation found in S4[**?**]. Thus, under the of-course modality the logic becomes non-linear. We will see in Section 4.1 that these rules define a comonad. We call the extension of L with the of-course modality $\mathrm{L}_!$.

2.1 Adding Exchange to the Lambek Calculus

As we remarked above, one leading question of the Lambek Calculus is can exchange be added in a similar way to weakening and contraction? That is, can we add a new modality that adds the exchange rule to L in a controlled way?

The answer to this question is positive, and the rules for this new modality are as follows:

$$\frac{\kappa\Gamma \vdash B}{\kappa\Gamma \vdash \kappa B}\ \textsc{Er} \qquad \frac{\Gamma_1, A, \Gamma_2 \vdash B}{\Gamma_1, \kappa A, \Gamma_2 \vdash B}\ \textsc{El}$$

$$\frac{\Gamma_1, \kappa A, B, \Gamma_2 \vdash C}{\Gamma_1, B, \kappa A, \Gamma_2 \vdash C}\ \text{E1} \qquad \frac{\Gamma_1, A, \kappa B, \Gamma_2 \vdash C}{\Gamma_1, \kappa B, A, \Gamma_2 \vdash C}\ \text{E2}$$

The first two rules are similar to of-course, but the last two add exchange to formulas under the κ-modality. We call L with the exchange modality L_κ. Thus, unlike intuitionistic linear logic where any two formulas can be exchanged L_κ restricts exchange to only formulas under the exchange modality. Just like of-course the exchange modality is modeled categorically as a comonad; see Section 4.1.

3 Algebraic Semantics

In Lambek's original paper [11] introducing his calculus L, albeit without modalities, he introduced an algebraic semantics that is now called the String Semantics for L. The semantics begins with a non-empty set of expressions denoted V^+, and then by modeling formulas of L by subsets of V^+ it proceeds by defining operations on these subsets, which correspond to the logical connectives of L. Each operation is defined as follows (using our notation for the logical connectives):

$$\begin{aligned} A \otimes B &= \{xy \in \mathrm{V}^+ \mid x \in A \text{ and } y \in B\} \\ A \leftharpoonup B &= \{x \in \mathrm{V}^+ \mid \text{for all } y \in B, xy \in A\} \\ A \rightharpoonup B &= \{y \in \mathrm{V}^+ \mid \text{for all } x \in A, xy \in B\} \end{aligned}$$

Let $\mathcal{N} = \mathcal{P}(\mathrm{V}^+)$ be the powerset of V^+. Then we can view each of the above definitions as binary operations with type $\mathcal{N}\times\mathcal{N}\longrightarrow\mathcal{N}$. In fact, $\mathcal{N}$ has a natural order induced by set containment, and concatenation, $A \otimes B$, gives $\mathcal{N}$ a non-commutative monoidal structure, where the unit $I = \{\epsilon\}$ is the set containing the empty sequence:

(associativity)	$A \otimes (B \otimes C) = (A \otimes B) \otimes C$
(unit)	$A \otimes I = A = I \otimes A$
(non-commutativity)	$A \otimes B \neq B \otimes A$, in general

There happens to be a more general structure underlying the previous semantics. We now make this structure explicit using the tools developed by Hyland and de Paiva [8]. The ordering on $\mathcal{N}$ induces a poset $(\mathcal{N}, \subseteq)$, but even more so, $\mathcal{N}$ is also a monoid $(\mathcal{N}, \otimes, I)$, but that is not all, these two structures are compatible, that is, given $A \subseteq B$ the following hold:

$$\begin{aligned} A \otimes C \subseteq B \otimes C, \text{ for all } C \in \mathcal{N} \\ C \otimes A \subseteq C \otimes B, \text{ for all } C \in \mathcal{N} \end{aligned}$$

Abstracting this structure out yields what is call an ordered non-commutative monoid.

Definition 1. *An **ordered non-commutative monoid**, $(M, \leq, \circ, e)$, is a poset $(M, \leq)$ with a given compatible monoidal structure $(M, \circ, e)$. That is, a set M equipped with a binary relation, $\leq\colon M \times M \longrightarrow \mathsf{2}$, satisfying:*

$$\begin{array}{cc} \textit{(reflexivity)} & a \leq a \textit{ for all } a \in M \\ \textit{(transitivity)} & a \leq b \textit{ and } b \leq c, \textit{ implies that } a \leq c \textit{ for all } a, b, c \in M \\ \textit{(antisymmetry)} & a \leq b \textit{ and } b \leq a, \textit{ implies that } a = b \end{array}$$

together with a monoidal multiplication $\circ : M \times M \longrightarrow M$ and a distinguished object $e \in M$ satisfying the following:

$$\begin{array}{cc} \textit{(associativity)} & a \circ (b \circ c) = (a \circ b) \circ c \\ \textit{(identity)} & e \circ a = a = a \circ e \end{array}$$

The structures are compatible in the sense that, if $a \leq b$, then the following hold:

$$\begin{array}{c} a \circ c \leq b \circ c \textit{ for any } c \in M \\ c \circ a \leq c \circ b \textit{ for any } c \in M \end{array}$$

It is easy to see that the previous definition accounts for all of the structure we have described so far, and thus, we may conclude that $(\mathcal{N}, \subseteq, \otimes, I)$ is an ordered non-commutative monoid, however, this definition is not able to model the implication operations $A \mathrel{\reflectbox{$\rightarrow$}} B$ and $A \multimap B$. To do this we need to understand how implication relates to the ordered non-commutative monoid structure. Notice that the following hold:

$$\begin{array}{c} A \otimes (A \multimap B) \subseteq B \\ (A \mathrel{\reflectbox{$\rightarrow$}} B) \otimes A \subseteq B \end{array}$$

Furthermore, there are no larger objects of $\mathcal{N}$ with these properties. Abstracting this results in the notion of a biclosed poset.

Definition 2. *Suppose $(M, \leq, \circ, e)$ is an ordered non-commutative monoid. If there exists a largest $x \in M$ such that $a \circ x \leq b$ for any $a, b \in M$, then we denote x by $a \multimap b$ and called it the **left-pseudocomplement** of a w.r.t b. Additionally, if there exists a largest $x \in M$ such that $x \circ a \leq b$ for any $a, b \in M$, then we denote x by $a \mathrel{\reflectbox{$\rightarrow$}} b$ and called it the **right-pseudocomplement** of a w.r.t b.*

*A **biclosed poset**, $(M, \leq, \circ, e, \multimap, \mathrel{\reflectbox{$\rightarrow$}})$, is an ordered non-commutative monoid, $(M, \leq, \circ, e)$, such that $a \multimap b$ and $a \mathrel{\reflectbox{$\rightarrow$}} b$ exist for any $a, b \in M$.*

At this point we have everything we need to model the Lambek Calculus L without modalities.

Lemma 3. *Any biclosed poset $(M, \leq, \circ, e, \multimap, \mathrel{\reflectbox{$\rightarrow$}})$ is a model for the Lambek Calculus L without modalities.*

Proof. First suppose we have an assignment $(-)^0$ which assigns to each formula of L an element of M. Then if $\Gamma \vdash A$ holds we show that $(\Gamma)^0 \leq (A)^0$. This proof can easily be completed by induction on the form $\Gamma \vdash A$. □

4 Categorical Models

Historically, the Lambek calculus arose from its intended categorical model, biclosed symmetric monoidal categories, as discussed by Lambek in [10]. Clearly all this happened almost three decades before Girard introduced Linear Logic, hence there were no modalities or exponentials in this setting.

Categorically, one models the of-course modality as a functor endowed with the structure of a comonad $(!, \delta_A, \varepsilon_A)$ with some additional structure. That is, there are maps $\delta_A : !A \longrightarrow !!A$ and $\varepsilon_A : !A \longrightarrow A$ subject to a few coherence diagrams, and maps $c_A : !A \longrightarrow !A \otimes !A$ and $w_A : !A \longrightarrow I$. Using this structure we can interpret the rules of the of-course modality.

Consider the rule C from Section 2, and suppose we have a map $\Gamma \otimes (!A \otimes !A) \xrightarrow{f} B$, then we can obtain a new map $\Gamma \otimes !A \xrightarrow{\mathsf{id}_\Gamma \otimes c_A} \Gamma \otimes (!A \otimes !A) \xrightarrow{f} B$. The rule W is similar, but we start with a map $\Gamma \xrightarrow{f} B$ and then we can define the map $\Gamma \otimes !A \xrightarrow{\mathsf{id}_\Gamma \otimes w_A} \Gamma \otimes I \xrightarrow{\cong} \Gamma \xrightarrow{f} B$. Notice that the previous map exploits the fact that I is the unit for tensor. Now consider the rule BL, and suppose we have a map $!\Gamma \xrightarrow{f} B$, then we may obtain a second map using the fact that of-course is a functor $!\Gamma \xrightarrow{\delta} !!\Gamma \xrightarrow{!f} !B$. Finally, consider the rule BR and suppose we have a map $\Gamma \otimes A \xrightarrow{f} B$, then we can construct the map $\Gamma \otimes !A \xrightarrow{\mathsf{id}_\Gamma \otimes \varepsilon_A} \Gamma \otimes A \xrightarrow{f} B$.

This analysis tells us a few things about interpreting logics into categorical models. Sequents, $\Gamma \vdash B$, are interpreted as morphisms, $\Gamma \xrightarrow{f} B$, where Γ is I if it is empty, or it is the tensor product of the interpretations of its formulas. Then interpreting inference rules amounts to starting with the morphisms corresponding to the premises, and then building a map corresponding to the conclusion. The cut-elimination procedure is defined by a set of equations between derivations, and hence, in the model corresponds to equations between morphisms. The various coherence diagrams relating the structure of the model enforce that these equations hold.

We can similarly interpret the rules for the exchange modality. That is, as a functor endowed with the structure of a second comonad, but also with the maps $e_1 : A \otimes \kappa B \longrightarrow \kappa B \otimes A$ and $e_2 : \kappa A \otimes B \longrightarrow B \otimes \kappa A$. Then, each of the inference rules for the exchange modality can easily be interpreted into the model.

4.1 Dialectica Lambek Spaces

A sound and complete categorical model of each of the Lambek calculi we have been exploring can be given in a modification to de Paiva's dialectica categories [6]. Originally, dialectica categories arose from de Paiva's thesis work on the categorical model of Gödel's Dialectica interpretation. In fact, dialectica categories are one of the first sound categorical models of intuitionistic linear logic

with the linear modalities. We show in this section that they can be adapted to a sound a complete model, called dialectica Lambek spaces, for the Lambek calculus with both the exchange and of-course modalities. Due to the complexities of working with dialectica categories we have formally verified this section in the proof assistant Agda[1] [3].

Dialectica categories arise as constructions over a given monoidal category. Suppose $\mathcal{C}$ is such a category. Then in complete generality the objects of the dialectica category over $\mathcal{C}$ are triples (U, X, α) where U and X are objects of $\mathcal{C}$, and $\alpha : A \longrightarrow U \otimes X$ is a subobject of the tensor product in $\mathcal{C}$ of U and X. Thus, we can think of α as a relation over $U \otimes X$. If we specialize the category $\mathcal{C}$ to the category of sets and functions, Set, then we obtain what is called a dialectica space. Dialectica spaces are a useful model of full intuitionistic linear logic [9].

Now morphisms between objects (U, X, α) and (V, Y, β) are pairs (f, F) where $f : U \longrightarrow V$ and $F : Y \longrightarrow X$ are morphisms of $\mathcal{C}$ such that the following pullback condition holds:

$$(U \otimes F)^{-1}(\alpha) \leq (f \otimes Y)^{-1}(\beta)$$

In dialectica spaces this condition becomes the following:

$$\forall u \in U. \forall y \in Y. \alpha(u, F(y)) \leq \beta(f(u), y)$$

The latter reveals that we can think of the condition on morphisms as a weak adjoint condition. Finally, through some nontrivial reasoning on this structure we can show that this is indeed a category; for the details see the formal development. Dialectica categories are related to the Chu construction [**?**] and to GAME_{κ_1} [**?**].

To some extent the underlying category $\mathcal{C}$ controls the kind of structure we can expect in the dialectica category over $\mathcal{C}$. de Paiva later showed [**?**] that by changing the relation used in objects and the relation used in the weak adjoint condition also controls the type of structure we obtain in the dialectica category, but the structure of the underlying category and the structure of the underlying relations must be compatible. She shows that one can abstract this notion of relation out as a parameter in the dialectica construction, and we denote this by $\mathsf{Dial}_L(\mathcal{C})$ where L is the structure controlling the relations used in the construction. For example, $\mathsf{Dial}_2(\mathsf{Set})$ is the category of dialectica spaces. Thus, we can see dialectica categories as really a framework of categorical models of various logics. Depending on which category we start with and which structure we use for the relations in the construction we will obtain different models for different logics.

The underlying category we will choose here is the category Set, but the structure we will define our relations over will be biclosed posets (Definition 2).

Definition 4. *Suppose $(M, \leq, \circ, e, \rightharpoonup, \leftharpoonup)$ is a biclosed poset. Then we define the category of* ***dialectica Lambek spaces****, $\mathsf{Dial}_M(\mathsf{Set})$, as follows:*

[1]The complete formalization can be found online at `https://github.com/heades/dialectica-spaces/blob/Lambek/NCDialSets.agda`.

- *objects, or dialectica Lambek spaces, are triples* (U, X, α) *where* U *and* X *are sets, and* $\alpha : U \times X \longrightarrow M$ *is a generalized relation over* M*, and*

- *maps that are pairs* $(f, F) : (U, X, \alpha) \longrightarrow (V, Y, \beta)$ *where* $f : U \longrightarrow V$*, and* $F : Y \longrightarrow X$ *are functions such that the following weak adjointness condition holds:*

$$\forall u \in U. \forall y \in Y. \alpha(u, F(y)) \leq \beta(f(u), y)$$

Notice here that the biclosed poset is used as the target of the relations in objects, but also as the relation in the weak adjoint condition on morphisms. This will allow the structure of the biclosed poset to lift up into $\mathsf{Dial}_M(\mathsf{Set})$.

We claim that $\mathsf{Dial}_M(\mathsf{Set})$ is a model of the Lambek calculus with modalities. First, we show that it is a model of the Lambek calculus without modalities. Thus, we must show that $\mathsf{Dial}_M(\mathsf{Set})$ is monoidal biclosed.

Definition 5. *Suppose* (U, X, α) *and* (V, Y, β) *are two objects of* $\mathsf{Dial}_M(\mathsf{Set})$*. Then their tensor product is defined as follows:*

$$(U, X, \alpha) \otimes (V, Y, \beta) = (U \times V, (V \to X) \times (U \to Y), \alpha \otimes \beta)$$

where $- \to -$ *is the function space from* Set*, and* $(\alpha \otimes \beta)((u, v), (f, g)) = \alpha(u, f(v)) \circ \beta(g(u), v)$*.*

The identity of the tensor product just defined is $I = (\mathbb{1}, \mathbb{1}, e)$, where $\mathbb{1}$ is the terminal object in Set, and e is the unit of the biclosed poset. It is straightforward to show that the tensor product is functorial, one can define the left and right unitors, and the associator for tensor; see the formalization for the definitions. In addition, all of the usual monoidal diagrams hold [6]. Take note of the fact that this tensor product is indeed non-commutative, because the non-commutative multiplication of the biclosed poset is used to define the relation of the tensor product.

The tensor product has two right adjoints making $\mathsf{Dial}_M(\mathsf{Set})$ biclosed.

Definition 6. *Suppose* (U, X, α) *and* (V, Y, β) *are two objects of* $\mathsf{Dial}_M(\mathsf{Set})$*. Then two internal-homs can be defined as follows:*

$$(U, X, \alpha) \multimap (V, Y, \beta) = ((U \to V) \times (Y \to X), U \times Y, \alpha \multimap \beta)$$
$$(V, Y, \beta) \multimap (U, X, \alpha) = ((U \to V) \times (Y \to X), U \times Y, \alpha \multimap \beta)$$

These two definitions are functorial, where the first is contravariant in the first argument and covariant in the second, but the second internal-hom is covariant in the first argument and contravariant in the second. The relations in the previous two definitions prevent these two from collapsing into the same object, because of the use of the left and right pseudo-compliments. It is straightforward to show that the following bijections hold:

$$\mathsf{Hom}(A \otimes B, C) \cong \mathsf{Hom}(B, A \multimap C)$$
$$\mathsf{Hom}(A \otimes B, C) \cong \mathsf{Hom}(A, C \multimap B)$$

Therefore, $\mathsf{Dial}_M(\mathsf{Set})$ is biclosed, and we obtain the following result.

Theorem 7. $\mathsf{Dial}_M(\mathsf{Set})$ *is a sound and complete model for the Lambek calculus* L *without modalities.*

4.1.1 Modalities

We now extend $\mathsf{Dial}_M(\mathsf{Set})$ with two modalities: the usual modality, of-course, denoted $!A$, and the exchange modality denoted κA. However, we must first extended biclosed posets to include an exchange operation.

Definition 8. *A* ***biclosed poset with exchange*** *is a poset* $(M, \leq, \circ, e, \rightharpoonup, \leftharpoondown)$ *equipped with an unary operation* $\kappa : M \to M$ *satisfying the following:*

$$\begin{array}{cc} \textit{(Compatibility)} & a \leq b \textit{ implies } \kappa a \leq \kappa b \textit{ for all } a, b, c \in M \\ \textit{(Minimality)} & \kappa a \leq a \textit{ for all } a \in M \\ \textit{(Duplication)} & \kappa a \leq \kappa\kappa a \textit{ for all } a \in M \\ \textit{(Left Exchange)} & \kappa a \circ b \leq b \circ \kappa a \textit{ for all } a, b \in M \\ \textit{(Right Exchange)} & a \circ \kappa b \leq \kappa b \circ a \textit{ for all } a, b \in M \end{array}$$

Compatibility results in $\kappa : M \to M$ being a functor in the biclosed poset, and the remainder of the axioms imply that κ is a comonad extending the biclosed poset with left and right exchange.

We can now define the two modalities in $\mathsf{Dial}_M(\mathsf{Set})$ where M is a biclosed poset with exchange; clearly we know $\mathsf{Dial}_M(\mathsf{Set})$ is also a model of the Lambek calculus without modalities by Theorem 7 because M is a biclosed poset.

Definition 9. *Suppose* (U, X, α) *is an object of* $\mathsf{Dial}_M(\mathsf{Set})$ *where* M *is a biclosed poset with exchange. Then the* ***of-course*** *and* ***exchange*** *modalities can be defined as follows:*

$$\begin{aligned} !(U, X, \alpha) &= (U, U \to X^*, !\alpha) \\ \kappa(U, X, \alpha) &= (U, X, \kappa\alpha) \end{aligned}$$

where X^* *is the free commutative monoid on* X*,* $(!\alpha)(u, f) = \alpha(u, x_1) \circ \cdots \circ \alpha(u, x_i)$ *for* $f(u) = (x_1, \ldots, x_i)$*, and* $(\kappa\alpha)(u, x) = \kappa(\alpha(u, x))$*.*

This definition highlights a fundamental difference between the two modalities. The definition of the exchange modality relies on an extension of biclosed posets with essentially the exchange modality in the category of posets. However, the of-course modality is defined by the structure already present in $\mathsf{Dial}_M(\mathsf{Set})$, specifically, the structure of Set.

Both of the modalities have the structure of a comonad. That is, there are monoidal natural transformations $\varepsilon_! : !A \longrightarrow A$, $\varepsilon_\kappa : \kappa A \longrightarrow A$, $\delta_! : !A \longrightarrow !!A$, and $\delta_\kappa : \kappa A \longrightarrow \kappa\kappa A$ which satisfy the appropriate diagrams; see the formalization for the full proofs. Furthermore, these comonads come equipped with arrows $e : A \longrightarrow I$, $d : A \longrightarrow A \otimes A$, $\beta L : \kappa A \otimes B \longrightarrow B \otimes \kappa A$, and $\beta R : A \otimes \kappa B \longrightarrow \kappa B \otimes A$. Thus, we arrive at the following result.

Theorem 10. *Suppose* M *is a biclosed poset with exchange. Then* $\mathsf{Dial}_M(\mathsf{Set})$ *is a sound and complete model for the Lambek calculi* $L_!$*,* L_κ*, and* $L_{!\kappa}$*.*

5 Typed Lambek Calculi

In this section we introduce typed calculi for each of the logics discussed so far. Each type system is based on the term assignment for Full Intuitionistic Linear Logic introduced by Hyland and de Paiva [9]. We show that they are all strongly normalizing and confluent, but we do not give full detailed proofs of each of these properties, because they are straightforward consequences of the proofs of strong normalization and confluence for intuitionistic linear logic. In fact, we will reference Bierman's thesis often within this section. The reader may wish to review Section 3.5 on page 88 of [2].

5.1 The Typed Lambek Calculus: λL

The first system we cover is the Lambek calculus without modalities. This system can be seen as the initial core of each of the other systems we introduce below, and thus, we will simply extend the results here to those other systems.

The syntax for patterns, terms, and contexts are described by the following grammar:

$$\begin{array}{rl} \text{(patterns)} & p := - \mid x \mid \mathsf{unit} \mid p_1 \otimes p_2 \\ \text{(terms)} & t := x \mid \mathsf{unit} \mid t_1 \otimes t_2 \mid \lambda_l x : A.t \mid \lambda_r x : A.t \mid \mathsf{app}_l\ t_1\ t_2 \mid \\ & \mathsf{app}_r\ t_1\ t_2 \mid \mathsf{let}\ t_1\ \mathsf{be}\ p\ \mathsf{in}\ t_2 \\ \text{(contexts)} & \Gamma := \cdot \mid x : A \mid \Gamma_1, \Gamma_2 \end{array}$$

Contexts are sequences of pairs of free variables and types. Patterns are only used in the let-expression which is itself used to eliminate logical connectives within the left rules of L. All variables in the pattern of a let-expression are bound. The remainder of the terms are straightforward.

The typing rules can be found in Figure 2 and the reduction rules in Figure 3. The typing rules are as one might expect. Finally, the reduction rules were extracted from the cut-elimination procedure for L. We denote the reflexive and transitive closure of the $\rightsquigarrow$ by $\rightsquigarrow^*$. We call a term with no β-redexes a normal form, and we denote normal forms by n. In the interest of space we omit the congruence rules from the definition of the reduction relation; we will do this for each calculi introduced throughout this section. The other typed calculi we introduce below will be extensions of λL, thus, we do not reintroduce these rules each time for readability.

Strong normalization. It is well known that intuitionistic linear logic (ILL) is strongly normalizing, for example, see Bierman's thesis [2] or Benton's beautiful embedding of ILL into system F [1]. It is fairly straightforward to define a reduction preserving embedding of λL – and as we will see the other calculi can be as well – into ILL. Intuitionistic linear logic can be obtained from λL by replacing the rules T_IRL, T_ILL, T_IRR, and T_ILR with the following two rules:

$$\frac{\Gamma_2 \vdash t_1 : A \qquad \Gamma_1, x : B, \Gamma_3 \vdash t_2 : C}{\Gamma_1, z : A \multimap B, \Gamma_2, \Gamma_3 \vdash [z\ t_1/x]t_2 : C}\ \text{T\_IL} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x : A.t : A \multimap B}\ \text{T\_IR}$$

$$\frac{}{x : A \vdash x : A}\ \text{T\_VAR} \qquad \frac{}{\cdot \vdash \mathsf{unit} : I}\ \text{T\_UR}$$

$$\frac{\Gamma_2 \vdash t_1 : A \qquad \Gamma_1, x : A, \Gamma_3 \vdash t_2 : B}{\Gamma_1, \Gamma_2, \Gamma_3 \vdash [t_1/x]t_2 : B}\ \text{T\_CUT}$$

$$\frac{\Gamma_1, \Gamma_2 \vdash t : A}{\Gamma_1, x : I, \Gamma_2 \vdash \mathsf{let}\ x\ \mathsf{be\ unit\ in}\ t : A}\ \text{T\_UL}$$

$$\frac{\Gamma, x : A, y : B, \Gamma' \vdash t : C}{\Gamma, z : A \otimes B, \Gamma' \vdash \mathsf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathsf{in}\ t : C}\ \text{T\_TL}$$

$$\frac{\Gamma_1 \vdash t_1 : A \qquad \Gamma_2 \vdash t_2 : B}{\Gamma_1, \Gamma_2 \vdash t_1 \otimes t_2 : A \otimes B}\ \text{T\_TR}$$

$$\frac{\Gamma_2 \vdash t_1 : A \qquad \Gamma_1, x : B, \Gamma_3 \vdash t_2 : C}{\Gamma_1, z : A \multimap B, \Gamma_2, \Gamma_3 \vdash [\mathsf{app}_r\ z\ t_1/x]t_2 : C}\ \text{T\_IRL}$$

$$\frac{\Gamma_2 \vdash t_1 : A \qquad \Gamma_1, x : B, \Gamma_3 \vdash t_2 : C}{\Gamma_1, \Gamma_2, z : B \multimap A, \Gamma_3 \vdash [\mathsf{app}_l\ z\ t_1/x]t_2 : C}\ \text{T\_ILL}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda_r x : A.t : A \multimap B}\ \text{T\_IRR} \qquad \frac{x : A, \Gamma \vdash t : B}{\Gamma \vdash \lambda_l x : A.t : B \multimap A}\ \text{T\_ILR}$$

Figure 2: Typing Rules for The Typed Lambek Calculus: λL

In addition, contexts are considered multisets, and hence, exchange is handled implicitly.

At this point we define the following embeddings.

Definition 11. *We embed types and terms of λL into ILL as follows:*

$$\frac{}{\mathsf{app}_l\,(\lambda_l x : A.t_2)\;t_1 \rightsquigarrow [t_1/x]t_2}\;\text{R\_BetaL}$$

$$\frac{}{\mathsf{app}_r\,(\lambda_r x : A.t_2)\;t_1 \rightsquigarrow [t_1/x]t_2}\;\text{R\_BetaR}$$

$$\frac{}{\mathsf{let}\;t_1\;\mathsf{be}\;\mathsf{unit}\;\mathsf{in}\;[\mathsf{unit}/z]t_2 \rightsquigarrow [t_1/z]t_2}\;\text{R\_BetaU}$$

$$\frac{}{\mathsf{let}\;t_1 \otimes t_2\;\mathsf{be}\;x \otimes y\;\mathsf{in}\;t \rightsquigarrow [t_1/x][t_2/y]t}\;\text{R\_BetaT1}$$

$$\frac{}{\mathsf{let}\;t_1\;\mathsf{be}\;x \otimes y\;\mathsf{in}\;[x \otimes y/z]t_2 \rightsquigarrow [t_1/x]t_2}\;\text{R\_BetaT2}$$

$$\frac{}{[\mathsf{let}\;t_1\;\mathsf{be}\;\mathsf{unit}\;\mathsf{in}\;t_2/z]t_3 \rightsquigarrow \mathsf{let}\;t_1\;\mathsf{be}\;\mathsf{unit}\;\mathsf{in}\;[t_2/z]t_3}\;\text{R\_NatU}$$

$$\frac{}{[\mathsf{let}\;t_1\;\mathsf{be}\;x \otimes y\;\mathsf{in}\;t_2/z]t_3 \rightsquigarrow \mathsf{let}\;t_1\;\mathsf{be}\;x \otimes y\;\mathsf{in}\;[t_2/z]t_3}\;\text{R\_NatT}$$

$$\frac{}{\mathsf{let}\;\mathsf{unit}\;\mathsf{be}\;\mathsf{unit}\;\mathsf{in}\;t \rightsquigarrow t}\;\text{R\_LetU}$$

Figure 3: Rewriting Rules for The Typed Lambek Calculus: λL

Types:

$$\begin{aligned} I^{\mathsf{e}} &= I \\ (A \otimes B)^{\mathsf{e}} &= A^{\mathsf{e}} \otimes B^{\mathsf{e}} \\ (A \rightharpoonup B)^{\mathsf{e}} &= A^{\mathsf{e}} \multimap B^{\mathsf{e}} \\ (A \leftharpoonup B)^{\mathsf{e}} &= A^{\mathsf{e}} \multimap B^{\mathsf{e}} \end{aligned}$$

Terms:

$$\begin{aligned} x^{\mathsf{e}} &= x \\ \mathsf{unit}^{\mathsf{e}} &= \mathsf{unit} \\ (t_1 \otimes t_2)^{\mathsf{e}} &= {t_1}^{\mathsf{e}} \otimes {t_2}^{\mathsf{e}} \\ (\mathsf{let}\;t_1\;\mathsf{be}\;p\;\mathsf{in}\;t_2)^{\mathsf{e}} &= \mathsf{let}\;{t_1}^{\mathsf{e}}\;\mathsf{be}\;p\;\mathsf{in}\;{t_2}^{\mathsf{e}} \\ (\lambda_l x : A.t)^{\mathsf{e}} &= \lambda x : A.t^{\mathsf{e}} \\ (\lambda_r x : A.t)^{\mathsf{e}} &= \lambda x : A.t^{\mathsf{e}} \\ (\mathsf{app}_l\;t_1\;t_2)^{\mathsf{e}} &= {t_1}^{\mathsf{e}}\;{t_2}^{\mathsf{e}} \\ (\mathsf{app}_r\;t_1\;t_2)^{\mathsf{e}} &= {t_1}^{\mathsf{e}}\;{t_2}^{\mathsf{e}} \end{aligned}$$

The previous embeddings can be extended to contexts in the straightforward way, and to sequents as follows:

$$(\Gamma \vdash t : A)^{\mathsf{e}} = \Gamma^{\mathsf{e}} \vdash t^{\mathsf{e}} : A^{\mathsf{e}}$$

This embedding preserves typing and reduction.

Lemma 12 (Type Preserving Embedding)**.** *If* $\Gamma \vdash t : A$ *in* λL*, then* $\Gamma^{\mathsf{e}} \vdash t^{\mathsf{e}} : A^{\mathsf{e}}$ *in ILL.*

Proof. This holds by straightforward induction on the form of the assumed typing derivation. □

Lemma 13 (Reduction Preserving Embedding)**.** *If* $t_1 \rightsquigarrow t_2$ *in* λL*, then* ${t_1}^{\mathsf{e}} \rightsquigarrow {t_2}^{\mathsf{e}}$ *in ILL.*

Proof. This holds by straightforward induction on the form of the assumed reduction derivation. □

The previous two results imply that λL is strongly normalizing, because ILL is.

Corollary 14 (Strong Normalization)**.** *If* $\Gamma \vdash t : A$*, then* t *is strongly normalizing.*

Proof. Suppose t is not strongly normalizing. Then there exists at least one infinite descending chain starting with t of the form $t \rightsquigarrow t' \rightsquigarrow t'' \rightsquigarrow \cdots$. Lemma 13 implies that there must also be an infinite descending chain of the form $t^{\mathsf{e}} \rightsquigarrow t'^{\mathsf{e}} \rightsquigarrow t''^{\mathsf{e}} \rightsquigarrow \cdots$, but we know this is impossible, because ILL is strongly normalizing. □

Confluence. The Church-Rosser property is well known to hold for ILL modulo commuting conversions, for example, see Theorem 19 of [2] on page 96. Since λL is essentially a subsystem of ILL, it is straightforward, albeit lengthly, to simply redefine Bierman's candidates (Definitions 17, 19, 20, and 21 of Section 5.1 of ibid.), and then carry out a similar proof as Bierman's (Theorem 19 on page 96 of ibid.). However, giving the details here would not be fruitful nor enlightening, and hence, we omit the proof. Thus, we have the following:

Theorem 15 (Confluence)**.** *The reduction relation,* $\rightsquigarrow$*, modulo the commuting conversions is confluent.*

5.2 The Typed Lambek Calculus: $\lambda\mathbf{L}_!$

The calculus we introduce in this section is an extension of λL with the of-course modality $!A$. This extension follows from ILL exactly. The syntax of types and terms of λL are extended as follows:

$$\begin{array}{llll} \text{(types)} & A & := & \cdots \mid !A \\ \text{(terms)} & t & := & \cdots \mid \mathsf{copy}\ t'\ \mathsf{as}\ t_1, t_2\ \mathsf{in}\ t \mid \mathsf{discard}\ t'\ \mathsf{in}\ t \mid \mathsf{promote}_!\ t'\ \mathsf{for}\ t''\ \mathsf{in}\ t \mid \\ & & & \mathsf{derelict}_!\ t \end{array}$$

The new type and terms are what one might expect, and are the traditional syntax used for the of-course modality. We add the following typing rules to

$$\frac{}{\mathsf{derelict}_!\ (\mathsf{promote}_!\ \vec{t}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t_1) \rightsquigarrow [\vec{t}/\vec{x}]t_1}\ \text{R\_BetaDR}$$

$$\frac{}{\mathsf{discard}\ (\mathsf{promote}_!\ \vec{t}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t_1)\ \mathsf{in}\ t_2 \rightsquigarrow \mathsf{discard}\ \vec{t}\ \mathsf{in}\ t_2}\ \text{R\_BetaDI}$$

$$\frac{t_1' = \mathsf{promote}_!\ \vec{w}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t_1 \qquad t_1'' = \mathsf{promote}_!\ \vec{z}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t_1}{\mathsf{copy}\ (\mathsf{promote}_!\ \vec{t}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t_1)\ \mathsf{as}\ w, z\ \mathsf{in}\ t_2 \rightsquigarrow \mathsf{copy}\ \vec{t}\ \mathsf{as}\ \vec{w}, \vec{z}\ \mathsf{in}\ [t_1'/w][t_1''/z]t_2}\ \text{R\_BetaC}$$

$$\frac{}{[\mathsf{discard}\ t\ \mathsf{in}\ t_1/x]t_2 \rightsquigarrow \mathsf{discard}\ t\ \mathsf{in}\ [t_1/x]t_2}\ \text{R\_NatD}$$

$$\frac{}{[\mathsf{copy}\ t\ \mathsf{as}\ x, y\ \mathsf{in}\ t_1/x]t_2 \rightsquigarrow \mathsf{copy}\ t\ \mathsf{as}\ x, y\ \mathsf{in}\ [t_1/x]t_2}\ \text{R\_NatC}$$

Figure 4: Rewriting Rules for The Typed Lambek Calculus: $\lambda\text{L}_!$

λL:

$$\frac{\Gamma_1, x :!A, \Gamma_2, y :!A, \Gamma_3 \vdash t : B}{\Gamma_1, z :!A, \Gamma_2, \Gamma_3 \vdash \mathsf{copy}\ z\ \mathsf{as}\ x, y\ \mathsf{in}\ t : B}\ \text{T\_C}$$

$$\frac{\Gamma_1, \Gamma_2 \vdash t : B}{\Gamma_1, x :!A, \Gamma_2 \vdash \mathsf{discard}\ x\ \mathsf{in}\ t : B}\ \text{T\_W}$$

$$\frac{\vec{x} :!\Gamma \vdash t : B}{\vec{y} :!\Gamma \vdash \mathsf{promote}_!\ \vec{y}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t :!B}\ \text{T\_BR}$$

$$\frac{\Gamma_1, x : A, \Gamma_2 \vdash t : B}{\Gamma_1, y :!A, \Gamma_2 \vdash [\mathsf{derelict}_!\ y/x]t : B}\ \text{T\_BL}$$

Finally, the reduction rules can be found in Figure 4. The equality used in the R_BetaC rule is definitional, meaning, that the rule simply gives the terms on the right side of the equation the name on the left side, and nothing more. This makes the rule easier to read.

Strong normalization. Showing strong normalization for $\lambda\text{L}_!$ easily follows by a straightforward extension of the embedding we gave for λL.

Definition 16. *The following is an extension of the embedding of λL into ILL resulting in an embedding of types and terms of $\lambda L_!$ into ILL:*

Types:

$$(!A)^{\mathsf{e}} \;=\; !A^{\mathsf{e}}$$

Terms:

$$\begin{array}{rcl}
(\mathsf{copy}\ t'\ \mathsf{as}\ t_1, t_2\ \mathsf{in}\ t)^{\mathsf{e}} & = & \mathsf{copy}\ t'^{\mathsf{e}}\ \mathsf{as}\ t_1{}^{\mathsf{e}}, t_2{}^{\mathsf{e}}\ \mathsf{in}\ t^{\mathsf{e}} \\
(\mathsf{discard}\ t'\ \mathsf{in}\ t)^{\mathsf{e}} & = & \mathsf{discard}\ t'^{\mathsf{e}}\ \mathsf{in}\ t^{\mathsf{e}} \\
(\mathsf{promote}_!\ t'\ \mathsf{for}\ t''\ \mathsf{in}\ t)^{\mathsf{e}} & = & \mathsf{promote}_!\ t'^{\mathsf{e}}\ \mathsf{for}\ t''^{\mathsf{e}}\ \mathsf{in}\ t^{\mathsf{e}} \\
(\mathsf{derelict}_!\ t)^{\mathsf{e}} & = & \mathsf{derelict}_!\ t^{\mathsf{e}}
\end{array}$$

Just as before this embedding is type preserving and reduction preserving.

Lemma 17 (Type Preserving Embedding)**.** *If* $\Gamma \vdash t : A$ *in* $\lambda L_!$*, then* $\Gamma^{\mathsf{e}} \vdash t^{\mathsf{e}} : A^{\mathsf{e}}$ *in ILL.*

Proof. This holds by straightforward induction on the form of the assumed typing derivation. □

Lemma 18 (Reduction Preserving Embedding)**.** *If* $t_1 \rightsquigarrow t_2$ *in* $\lambda L_!$*, then* $t_1{}^{\mathsf{e}} \rightsquigarrow t_2{}^{\mathsf{e}}$ *in ILL.*

Proof. This holds by straightforward induction on the form of the assumed reduction derivation. □

Finally, we can use the previous two results to conclude strong normalization, because ILL is.

Corollary 19 (Strong Normalization)**.** *If* $\Gamma \vdash t : A$*, then* t *is strongly normalizing.*

Confluence. The Church-Rosser property also holds for $\lambda\mathrm{L}_!$, and can be shown by straightforwardly applying a slightly modified version of Bierman's proof [2] just as we did for λL. Thus, we have the following:

Theorem 20 (Confluence)**.** *The reduction relation,* $\rightsquigarrow$*, modulo the commuting conversions is confluent.*

5.3 The Typed Lambek Calculus: $\lambda\mathbf{L}_\kappa$

The next calculus we introduce is also an extension of λL with a modality that adds exchange to $\lambda\mathrm{L}_\kappa$ denoted κA. It is perhaps the most novel of the calculi we have introduced so far. The syntax of types and terms of λL are extended as follows:

$$\begin{array}{llll}
\text{(types)} & A & := & \cdots \mid \kappa A \\
\text{(terms)} & t & := & \cdots \mid \mathsf{exchange}_{\mathsf{l}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3 \mid \mathsf{exchange}_{\mathsf{r}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3 \mid \\
 & & & \quad \mathsf{promote}_\kappa\ t'\ \mathsf{for}\ t''\ \mathsf{in}\ t \mid \mathsf{derelict}_\kappa\ t
\end{array}$$

The syntax for types has been extended to include the exchange modality, and the syntax of terms follow suit. The terms $\mathsf{exchange}_{\mathsf{l}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3$

and $\mathsf{exchange}_{\mathsf{r}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3$ are used to explicitly track uses of exchange throughout proofs. We add the following typing rules to λL:

$$\frac{\Gamma_1, x_1 : \kappa A, y_1 : B, \Gamma_2 \vdash t : C}{\Gamma_1, y_2 : B, x_2 : \kappa A, \Gamma_2 \vdash \mathsf{exchange}_{\mathsf{l}}\ y_2, x_2\ \mathsf{with}\ x_1, y_1\ \mathsf{in}\ t : C}\ \text{T\_E1}$$

$$\frac{\Gamma_1, x_1 : A, y_1 : \kappa B, \Gamma_2 \vdash t : C}{\Gamma_1, y_2 : \kappa B, x_2 : A, \Gamma_2 \vdash \mathsf{exchange}_{\mathsf{r}}\ y_2, x_2\ \mathsf{with}\ x_1, y_1\ \mathsf{in}\ t : C}\ \text{T\_E2}$$

$$\frac{\vec{x} : \kappa\Gamma \vdash t : B}{\vec{y} : \kappa\Gamma \vdash \mathsf{promote}_{\kappa}\ \vec{y}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t : \kappa B}\ \text{T\_ER}$$

$$\frac{\Gamma_1, x : A, \Gamma_2 \vdash t : B}{\Gamma_1, y : \kappa A, \Gamma_2 \vdash [\mathsf{derelict}_{\kappa}\ y/x]t : B}\ \text{T\_EL}$$

$$\frac{}{\mathsf{derelict}_{\kappa}\ (\mathsf{promote}_{\kappa}\ \vec{t}\ \mathsf{for}\ \vec{x}\ \mathsf{in}\ t_1) \rightsquigarrow [\vec{t}/\vec{x}]t_1}\ \text{R\_BetaEDR}$$

$$\frac{}{[\mathsf{exchange}_{\mathsf{l}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3/z]t_4 \rightsquigarrow \mathsf{exchange}_{\mathsf{l}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ [t_3/z]t_4}\ \text{R\_NatEL}$$

$$\frac{}{[\mathsf{exchange}_{\mathsf{r}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3/z]t_4 \rightsquigarrow \mathsf{exchange}_{\mathsf{r}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ [t_3/z]t_4}\ \text{R\_NatER}$$

Figure 5: Rewriting Rules for The Typed Lambek Calculus: $\lambda\mathrm{L}_{\kappa}$

The reduction rules are in Figure 5, and are vary similar to the rules from $\lambda\mathrm{L}_!$.

Strong normalization. Similarly, we show that we can embed $\lambda\mathrm{L}_{\kappa}$ into ILL, but the embedding is a bit more interesting.

Definition 21. *The following is an extension of the embedding of λL into ILL resulting in an embedding of types and terms of $\lambda\mathrm{L}_{\kappa}$ into ILL:*

Types:

$$(\kappa A)^{\mathsf{e}} = !A^{\mathsf{e}}$$

Terms:

$$\begin{aligned}
(\mathsf{exchange}_{\mathsf{l}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3)^{\mathsf{e}} &= [t_2{}^{\mathsf{e}}/x][t_1{}^{\mathsf{e}}/y]t_3{}^{\mathsf{e}} \\
(\mathsf{exchange}_{\mathsf{r}}\ t_1, t_2\ \mathsf{with}\ x, y\ \mathsf{in}\ t_3)^{\mathsf{e}} &= [t_2{}^{\mathsf{e}}/x][t_1{}^{\mathsf{e}}/y]t_3{}^{\mathsf{e}} \\
(\mathsf{promote}_{\kappa}\ t'\ \mathsf{for}\ t''\ \mathsf{in}\ t)^{\mathsf{e}} &= \mathsf{promote}_{!}\ t'^{\mathsf{e}}\ \mathsf{for}\ t''^{\mathsf{e}}\ \mathsf{in}\ t^{\mathsf{e}} \\
(\mathsf{derelict}_{\kappa}\ t)^{\mathsf{e}} &= \mathsf{derelict}_{!}\ t^{\mathsf{e}}
\end{aligned}$$

The embedding just defined translates the exchange modality into the of-course modality of ILL. We do this so as to preserve the comonadic structure of the exchange modality. One might think that we could simply translate the exchange modality to the identity, but as Benton showed [1], this would result

in an embedding that does not preserve reductions. Furthermore, the left and right exchange terms are translated away completely, but this works because ILL contains exchange in general, and hence, does not need to be tracked explicitly. It is now straightforward to prove the following results.

Lemma 22 (Type Preserving Embedding)**.** *If* $\Gamma \vdash t : A$ *in* $\lambda L_!$*, then* $\Gamma^{\mathsf{e}} \vdash t^{\mathsf{e}} : A^{\mathsf{e}}$ *in ILL.*

Proof. This holds by straightforward induction on the form of the assumed typing derivation. □

Lemma 23 (Reduction Preserving Embedding)**.** *If* $t_1 \rightsquigarrow t_2$ *in* $\lambda L_!$*, then* ${t_1}^{\mathsf{e}} \rightsquigarrow {t_2}^{\mathsf{e}}$ *in ILL.*

Proof. This holds by straightforward induction on the form of the assumed reduction derivation. □

Finally, we can use the previous two results to conclude strong normalization, because ILL is.

Corollary 24 (Strong Normalization)**.** *If* $\Gamma \vdash t : A$*, then* t *is strongly normalizing.*

Confluence. The Church-Rosser property also holds for $\lambda \mathrm{L}_\kappa$, and the proof is nearly identical to the proof that $\lambda \mathrm{L}_!$ is confluent. Thus, we have the following:

Theorem 25 (Confluence)**.** *The reduction relation,* $\rightsquigarrow$*, modulo the commuting conversions is confluent.*

5.4 The Typed Lambek Calculus: $\lambda \mathbf{L}_{!\kappa}$

If we combine all three of the previous typed Lambek calculi, then we obtain the typed Lambek calculus $\lambda \mathrm{L}_{!\kappa}$. We do not explicitly define the syntax, typing rules, and reduction rules here, because we would be repeating what we have already defined in the previous sections. The benefits of this system is that it provides the structure of the Lambek calculus with the benefits of having exchange, and the of-course modality, but both are carefully tracked within the proofs.

Strong normalization for this calculus can be proved similarly to the previous calculi by simply merging the embeddings together. Thus, both modalities of $\lambda \mathrm{L}_{!\kappa}$ would merge into the of-course modality of ILL. The Church-Rosser property also holds for $\lambda \mathrm{L}_{!\kappa}$ by extending the proof of confluence for ILL by Bierman [2] just as we did for the other systems. Thus, we have the following results.

Theorem 26 (Strong Normalization)**.** *If* $\Gamma \vdash t : A$*, then* t *is strongly normalizing.*

Theorem 27 (Confluence)**.** *The reduction relation,* $\rightsquigarrow$*, modulo the commuting conversions is confluent.*

6 Conclusion

References

[1] P. N. Benton. Strong normalisation for the linear term calculus. *Journal of Functional Programming*, 5:65–80, 1 1995.

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

[3] Ana Bove, Peter Dybjer, and Ulf Norell. A brief overview of agda-a functional language with dependent types. *TPHOLs*, 5674:73–78, 2009.

[4] Bob Coecke, Edward Grefenstette, and Mehrnoosh Sadrzadeh. Lambek vs. lambek: Functorial vector space semantics and string diagrams for Lambek calculus. *Annals of Pure and Applied Logic*, 164(11):1079–1100, 2013.

[5] Valeria de Paiva. A Dialectica model of the Lambek calculus. In *8th Amsterdam Logic Colloquium*, 1991.

[6] Valeria C V de Paiva. *The dialectica categories*. PhD thesis, 1990. Computer Laboratory, University of Cambridge, PhD Thesis.

[7] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1 – 101, 1987.

[8] Martin Hyland and Valeria de Paiva. Lineales. *"O que nos faz pensar", Special number in Logic, Cadernos do Dept. de Filosofia da PUC", Pontificial Catholic University of Rio de Janeiro*, 1991.

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

[10] J. Lambek. *Categorial Grammars and Natural Language Structures*, chapter Categorial and Categorical Grammars, pages 297–317. Springer Netherlands, Dordrecht, 1988.

[11] Joachim Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, pages 154–170, 1958.

[12] Michael Moortgat. Typelogical grammar. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2014 edition, 2014.

[13] P. Sewell, F. Nardelli, S. Owens, G. Peskine, T. Ridge, S. Sarkar, and R. Strnisa. Ott: Effective tool support for the working semanticist. In *Journal of Functional Programming*, volume 20, pages 71–122, 2010.

[14] M. E. Szabo. *Algebra of Proofs*. Studies in Logic and the Foundations of Mathematics, Vol. 88, North-Holland, 1978, 1979.