13 The Lambek Calculus and the Lambda Calculus

1 Introduction

In this chapter, we shall learn about two calculi: the *Lambek calculus* and the *Lambda calculus*. Before turning to the exposition of these two calculi, readers might find it useful to know what is meant by the word *calculus*[1] here. A calculus is a system for calculation, which means that it comprises a set of expressions and rules for transforming expressions in the set into other expressions in the same set. For this reason, the area of mathematics often called *analysis* by mathematicians is also called differential and integral *calculus*, for it comprises a set of expressions and the rules of differentiation and integration by which one differentiates or integrates expressions in the set. The deductive part of CPL is also called the *propositional calculus*, for it too comprises a set of expressions, the formulae of CPL, and rules, the deduction rules, by which formulae are transformed into formulae. Indeed, we learned about one version of CPL known as the Gentzen sequent calculus, where the expressions transformed by the rules were not formulae, but sequents, which nonetheless themselves contain formulae.

The Lambek calculus, called a syntactic calculus by its formulator, the German born, Canadian mathematician Joachim (Jim) Lambek (1922–2014), has its origins in the work of Polish logician Kazimierz Ajdukiewicz (1935), who, inspired by ideas of Edmund Husserl (1900 bk. 5) and of Stanislaw Leśniewski, developed a calculus, a set of expressions and rules, for determining which strings of symbols in a logical notation are formulae and which are not. Though Ajdukiewicz alludes to the application of these ideas to the study of natural language, it is only with the work of the Austrian born, Israeli philosopher Yehoshua Bar-Hillel (1953) that its application to natural language is first seriously explored. Jim Lambek (1958) expanded the mathematical ideas of his predecessors into his syntactic calculus, pointing out the expanded calculus's application to the determination of which expressions in a natural language are correct expressions and which are not.

1. The word *calculus* in Latin means pebble. Pebbles were used by the Romans and other ancient people to do arithmetic calculation.

We shall also learn about the Lambda calculus. In the 1930s, starting with work in logic by the Austrian logician Kurt Gödel (1906–1978), a number of logicians turned their minds to trying to give a precise characterization of a computable function. Inspired by Gödel's work, the English logician, Alan Turing (1912–1954) arrived at another characterization, in which the functions defined were metaphorically called *machines* and now are called *Turing machines*. Still another characterization, due to Alonzo Church (1903–1995), an American logician, is given in terms of what he called the *Lambda calculus*.[2] The Lambda calculus comprises a set of expressions for functions, one key symbol of which is the lowercase of the Greek letter lambda (λ), and rules for transforming these expressions in expressions of the same set.

In this chapter, we shall first present the rudiments of the Lambek calculus, then the rudiments of the Lambda calculus. Finally, in preparation for the application of the Lambek and Lambda calculi to the study of natural language, we shall extend both and establish an isomorphism between them.

2 The Lambek Calculus

The Lambek calculus comprises a set of expressions, which we shall call *formulae,* and a set of rules whereby formulae are transformed into formulae, which we shall call *deduction rules*. The formulae of the Lambek calculus are obtained by forming complex expressions from a set of basic expressions and the three connectives of $\backslash$, $/$, and $\cdot$. We take our basic expressions to be the atomic formulae AF, the atomic formulae we used for CPL, though other sets could have served the same purpose. Here, then, is the formal definition of the set of all Lambek formulae, or LF.

Definition 1 Formulae of the Lambek calculus
Let AF be the atomic formula.

(1) $\text{AF} \subseteq \text{LF}$;

(2) If $\alpha, \beta \in \text{LF}$, then $\alpha\backslash\beta, \alpha/\beta, \alpha \cdot \beta \in \text{LF}$;

(3) Nothing else is in LF.

For example, if p, q, and r are atomic Lambek formulae, then not only are p, q, and r Lambek formulae, but so are $(p\backslash q)$, (q/r), $(r \cdot p)$, $\big((p \cdot q)/r\big)$, $\big((q/(q/r)\big)$, $\big((p\backslash q)/(r \cdot p)\big)$, and so on. We shall also adopt the usual convention of omitting the final pair of parentheses of a Lambek formula. Thus, instead of $(p\backslash q)$, $\big((p \cdot q)/r\big)$, and $\big((p\backslash q)/(r \cdot p)\big)$, we shall often write $p\backslash q$, $(p \cdot q)/r$, and $(p\backslash q)/(r \cdot p)$.

2. Other characterizations of computable functions are one due to the Polish born, American mathematician, Emil Post (1897–1954), called *Post machines,* arrived at independently from the work of Alan Turing, as well as others due to Joachim Lambek (1922–2014), called metaphorically *abaci,* and Marvin Minsky (1927–2016), an American computer scientist, to mention but a few.

We now turn to deduction in the Lambek calculus. We shall consider three presentations: the first is the presentation of formula deduction in the tree format, which is analogous to the presentation of formula natural deduction in the tree format; the second is the presentation of sequent deduction, also, in the tree format, which is analogous to the presentation of sequent natural deduction in the tree format; and the third is the presentation of Gentzen sequent deduction, again, in the tree format, which is analogous to the presentation of the Gentzen sequent calculus. Since each of the presentations of Lambek calculus given here uses the tree format, we shall usually omit this qualification hereafter.

An important question is whether these three presentations of the Lambek calculus are equivalent. We shall establish that they are. It should be noted that, in chapter 7, we could have asked the question of whether the five presentations, four natural deduction presentations and one the Gentzen sequent calculus presentation, of CPL are equivalent. To avoid burdening readers with too many questions in chapter 7, we assumed that readers would take it for granted that the five presentations are equivalent.

2.1 Formula Deduction

We begin with the definition of formula deduction for the Lambek calculus.

Definition 2 Formula deduction (tree format)
A deduction of a formula from a list of formulae, the premises of the deduction, is a tree of formulae each one of which is either a formula at the top of the tree, in which case it is taken from among the set of premises, or is obtained from formulae immediately above it in the formula tree by one of the rules specified below.

In analogy with natural deduction, we have six rules, an elimination and an introduction rule for each of the three basic connectives. Let us consider first the elimination and introduction rules for $\backslash$.

$\backslash$	Elimination	Introduction
	$\dfrac{\alpha \qquad \alpha\backslash\beta}{\beta}$	$\dfrac{\dfrac{[\,\alpha\,] \quad \vdots}{\beta}}{\alpha\backslash\beta}$

This pair of rules is similar to the elimination and introduction rules for $\rightarrow$ given for natural deduction in the formula tree format, repeated here for ease of reference.

→	Elimination	Introduction
	$\alpha, \alpha \to \beta$ / β	[α] ⋮ β / $\alpha \to \beta$

However, there is an important difference. We begin with the difference between the elimination rules. Formulae in the premises of the → elimination rule for formula natural deduction (tree format) are unordered with respect to one another. Indeed, the comma in the notation for the premise signals this fact. What this lack of order means is that the rule applies whether the formula tree terminating in α is to the left or the right of the formula tree terminating in $\alpha \to \beta$. In case of the \ elimination rule, no comma occurs in the notation for the rule's premise. This signals that the order of the formula trees terminating in $\alpha \backslash \beta$ and in α makes a difference: if the formula tree terminating in $\alpha \backslash \beta$ occurs to the immediate right of the formula tree terminating in α, the rule applies; if it occurs to the immediate left, it does not.

Though the rules for → introduction and \ introduction look similar, they differ in one crucial respect. In any application of the rule of → introduction in a tree formula deduction of CPL, when an instance of $\alpha \to \beta$ occurs under an instance of the formula β, each instance of the formula α above the instance of the formula β is enclosed in square brackets. However, in any application of the rule of \ introduction in a tree formula deduction of Lambek Calculus, when an instance of $\alpha \backslash \beta$ occurs under an instance of the formula β, only the leftmost instance of the formula α above the instance of the formula β is enclosed in square brackets.

We turn next to the pair of rules for /. This pair of rules is just like the pair for \, except that the order of the formulae relevant to the application of the rules is reversed. The / elimination rule applies only if the formula tree terminating in β/α is to the left of the formula tree terminating in α. The / introduction rule applies only if the formula α is the top, rightmost formula of the deduction terminating in β.

/	Elimination	Introduction
	$\beta/\alpha \quad \alpha$ / β	⋮ [α] ⋮ ⋮ β / β/α

Finally, we come to the pair of rules for ·.

·	Elimination	Introduction
	$\begin{array}{c} \vdots \quad \alpha \cdot \beta \quad \vdots \\ \vdots \quad \overline{\qquad\qquad} \quad \vdots \\ \vdots \quad \alpha \ \beta \quad \vdots \\ \vdots \quad \vdots \ \vdots \quad \vdots \\ \hline \gamma \end{array}$	$\begin{array}{c} \alpha \qquad \beta \\ \hline \alpha \cdot \beta \end{array}$

The · elimination rule is reminiscent of the $\vee$ elimination rule in natural deduction. Just as the rule of $\vee$ elimination in CPL is used to derive a formula of the form γ from a formula of the form $\alpha \vee \beta$, provided that there are formulae of the form $\alpha \rightarrow \gamma$ and $\beta \rightarrow \gamma$ available in the deduction, so the rule of · elimination in CPL is used to derive a formula of the form γ from a formula of the form $\alpha \cdot \beta$, provided that the formula γ is deducible both from the formula α and the formula β. Moreover, here too order is pertinent: if α is to the left of β in the formula $\alpha \cdot \beta$, then the subtree terminating in γ must have two top nodes, one labeled with α and another labelled with β with α occurring to the immediate left of β.

We now define formula deducibility in the Lambek calculus.

Definition 3 Formula deducibility
Let α be a formula in LF. Let Γ be a list of formulae taken from LF. Then, α is deducible in the formula tree format from Γ, or $\Gamma \vdash_{FD} \alpha$, iff there is a deduction (in the formula tree format) terminating with α such that (1) each formula at the top of the tree that is not enclosed in square brackets appears in Γ and (2) the order of appearance of the formulae unenclosed in square brackets at the top of the tree is the same as the order of the formulae listed in Γ.

It is important to stress that Γ here denotes a finite list of formulae, not a set of formulae. In a list, the left to right order of the occurrences of formulae matters: transpose any two occurrences of distinct formulae in the list and a different list results. Similarly, in a list, redundancies matter: lists which are the same, except that one has two instances of the same formula one next to the other and the other has only one instance of the formula are different lists. The deducibility relation, then, is a relation between a finite list of formulae and a single formula.

Here is a number of theorems for the formula deduction presentation of the Lambek calculus.

THEOREMS

1.1 $\alpha \cdot (\beta \cdot \gamma) \vdash_{FD} (\alpha \cdot \beta) \cdot \gamma$ 1.2 $(\alpha \cdot \beta) \cdot \gamma \vdash_{FD} \alpha \cdot (\beta \cdot \gamma)$

2.1 $\alpha \vdash_{FD} (\alpha \cdot \beta)/\beta$ 2.2 $\beta \vdash_{FD} \alpha \backslash (\alpha \cdot \beta)$

3.1 $(\alpha/\beta) \cdot \beta \vdash_{FD} \alpha$ 3.2 $\alpha \cdot (\alpha \backslash \beta) \vdash_{FD} \beta$

4.1	$\alpha \vdash_{FD} (\beta/\alpha)\backslash\beta$	4.2	$\alpha \vdash_{FD} \beta/(\alpha\backslash\beta)$
5.1	$(\gamma/\beta)\cdot(\beta/\alpha) \vdash_{FD} \gamma/\alpha$	5.2	$(\alpha\backslash\beta)\cdot(\beta\backslash\gamma) \vdash_{FD} \alpha\backslash\gamma$
6.1	$\gamma/\beta \vdash_{FD} (\gamma/\alpha)/(\beta/\alpha)$	6.2	$\beta\backslash\gamma \vdash_{FD} (\alpha\backslash\gamma)\backslash(\alpha\backslash\beta)$
7.1	$(\alpha\backslash\beta)/\gamma \vdash_{FD} \alpha\backslash(\beta/\gamma)$	7.2	$\alpha\backslash(\beta/\gamma) \vdash_{FD} (\alpha\backslash\beta)/\gamma$
8.1	$(\alpha/\beta)/\gamma \vdash_{FD} \alpha/(\gamma\cdot\beta)$	8.2	$\alpha/(\gamma\cdot\beta) \vdash_{FD} (\alpha/\beta)/\gamma$

We now prove some of these theorems, leaving the proof of the others to readers as an exercise.

PROOF: Theorem 1.1

$$\dfrac{\dfrac{\dfrac{\alpha \quad \dfrac{\alpha\cdot(\beta\cdot\gamma)}{\beta\cdot\gamma}\cdot E}{\beta}\cdot E \quad \gamma}{\alpha\cdot\beta}\cdot I}{(\alpha\cdot\beta)\cdot\gamma}\cdot I$$

PROOF: Theorem 4.1

$$\dfrac{\dfrac{[\beta/\alpha] \quad \alpha}{\beta}/E}{(\beta/\alpha)\backslash\beta}\backslash I$$

PROOF: Theorem 7.1

$$\dfrac{\dfrac{[\alpha] \quad \dfrac{(\alpha\backslash\beta)/\gamma \quad [\gamma]}{\alpha\backslash\beta}/E}{\beta}\backslash E}{\dfrac{\beta/\gamma}{\alpha\backslash(\beta/\gamma)}/I}\backslash I$$

Exercises: Formula deductions

1. Using the formula deduction rules of this section, establish the theorems which have not already been established.
2. Use the same formula deduction rules to establish the following.

1.1	$\alpha\ \beta\ \gamma \vdash_{FD} (\alpha\cdot\beta)\cdot\gamma$	1.2	$\alpha\ \beta\ \gamma \vdash_{FD} \alpha\cdot(\beta\cdot\gamma)$
2.1	$\alpha/\beta\ \beta \vdash_{FD} \alpha$	2.2	$\alpha\ \alpha\backslash\beta \vdash_{FD} \beta$
3.1	$\gamma/\beta\ \beta/\alpha \vdash_{FD} \gamma/\alpha$	3.2	$\alpha\backslash\beta\ \beta\backslash\gamma \vdash_{FD} \alpha\backslash\gamma$

2.2 Sequent Deduction

Let us turn to sequent deductions in the Lambek calculus. We should emphasize, to begin with, the point made just below the definition of formula deducibility for the Lambek calculus (Df. 3). A sequent in the sequent deduction presentation of the Lambek calculus comprises a finite list of formulae, the turnstile and a single formula. This notion of a

sequent shares with Gentzen's notion of a sequent, discussed at the end of chapter 7, the idea that the antecedent of a sequent is a finite list of formulae, though Gentzen additionally allowed the list to be empty.

Sequent deduction in the Lambek calculus is defined as follows.

Definition 4 Sequent deduction (tree format)
A deduction of a sequent is a tree of sequents each one of which either is an axiom or is obtained from sequents immediately above it in the sequent tree by one of the rules specified below.

Like the rules for sequent deduction (tree format) of CPL, which comprise an axiom and pairs of rules for each propositional connective, the rules for sequent deduction in the Lambek calculus comprise an axiom and pairs of rules for each connective.

Axiom

Axiom
$\dfrac{}{\alpha \vdash_{SD} \alpha}$

Next come the pair of rules for the elimination and introduction of \.

\	Elimination	Introduction
	$\dfrac{\Gamma \vdash_{SD} \alpha,\ \Delta \vdash_{SD} \alpha \backslash \beta}{\Gamma\ \Delta \vdash_{SD} \beta}$	$\dfrac{\alpha\ \Gamma \vdash_{SD} \beta}{\Gamma \vdash_{SD} \alpha \backslash \beta}$

This pair of rules for the Lambek calculus is very similar to the pair of rules for the elimination and introduction or $\rightarrow$ for deductions in CPL in the sequent tree format.

$\rightarrow$	Elimination	Introduction
	$\dfrac{\Gamma \vdash_{SD} \alpha\ ,\ \Delta \vdash_{SD} \alpha \rightarrow \beta}{\Gamma \cup \Delta \vdash_{SD} \beta}$	$\dfrac{\Gamma \cup \{\alpha\} \vdash_{SD} \beta}{\Gamma \vdash_{SD} \alpha \rightarrow \beta}$

We observe that the difference between the two pairs is the ordering. First, what is to the left of the turnstile in the Lambek calculus is a list of formulae, whose order is that of the list, whereas what is to the left of the turnstile in CPL is a set of formulae, for which no order is defined. The order of the list is pertinent to the formulation of both rules for the \.

In the case of the \ elimination rule, the left-hand side of the sequent in the conclusion of the rule comprises the list of formulae from which α is deducible followed by the list of formulae from which $\alpha\backslash\beta$ is deducible. In the case of the \ introduction rule, the left-hand side of the sequent in the conclusion of the rule is the same as the left-hand side of the sequent in the premise of the rule, except that the initial formula in the list has been removed.

It should be no surprise that the pair of rules for / are just like the pair for \, except that the points pertaining to order just mentioned are reversed. Readers should check that the comments about order do indeed apply, mutatis mutandis, to the pair of rules for \.

/	Elimination	Introduction
	$\dfrac{\Gamma \vdash_{SD} \alpha \ , \ \Delta \vdash_{SD} \beta/\alpha}{\Delta \ \Gamma \vdash_{SD} \beta}$	$\dfrac{\Gamma \ \alpha \vdash_{SD} \beta}{\Gamma \vdash_{SD} \beta/\alpha}$

Next we come to the pair of rules for · elimination and introduction.

·	Elimination	Introduction
	$\dfrac{\Delta \vdash_{SD} \alpha \cdot \beta, \Gamma \ \alpha \ \beta \ \Theta \vdash_{SD} \gamma}{\Gamma \ \Delta \ \Theta \vdash_{SD} \gamma}$	$\dfrac{\Gamma \vdash_{SD} \alpha, \ \Delta \vdash_{SD} \beta}{\Gamma \ \Delta \vdash_{SD} \alpha \cdot \beta}$

We add one further rule, for which there is no counterpart among the rules for the deductions in the formula tree format, namely, the *cut* rule. It is so called, since it permits one to cut out, as it were, a formula that occurs both as the formula on the right side of a sequent's turnstile and in the list of formulae on the left side of a sequent's turnstile. This turns out to be a handy rule, which we shall use in some of the following proofs. As we shall see, this rule is dispensable: any sequent that can be derived with its aid can also be derived without it.

Cut
$\dfrac{\Delta \vdash_{SD} \alpha, \Gamma\alpha\Theta \vdash_{SD} \beta}{\Gamma\Delta\Theta \vdash_{SD} \beta}$

In the preceding statement of the rules, we allow the variables ranging over a list of formulae from LF to be empty only if that assumption does not result in the list to the left

of the turnstile being empty. Thus, the rule for · elimination permits either Δ or Θ or both to be empty; in all other cases, the lists are nonempty.

The analogs of the theorems for the formula deduction presentation of the Lambek calculus also hold for the sequent deduction presentation. Readers are asked to prove these theorems in the exercises.

THEOREMS

1.1	$\alpha\ \beta\ \gamma \vdash_{SD} (\alpha \cdot \beta) \cdot \gamma$	1.2	$\alpha\ \beta\ \gamma \vdash_{SD} \alpha \cdot (\beta \cdot \gamma)$
2.1	$\alpha \vdash_{SD} (\alpha \cdot \beta)/\beta$	2.2	$\beta \vdash_{SD} \alpha\backslash(\alpha \cdot \beta)$
3.1	$\alpha/\beta\ \beta \vdash_{SD} \alpha$	3.2	$\alpha\ \alpha\backslash\beta \vdash_{SD} \beta$
4.1	$\alpha \vdash_{SD} (\beta/\alpha)\backslash\alpha$	4.2	$\alpha \vdash_{SD} \beta/(\alpha\backslash\beta)$
5.1	$\gamma/\beta\ \beta/\alpha \vdash_{SD} \gamma/\alpha$	5.2	$\alpha\backslash\beta\ \beta\backslash\gamma \vdash_{SD} \alpha\backslash\gamma$
6.1	$\gamma/\beta \vdash_{SD} (\gamma/\alpha)/(\beta/\alpha)$	6.2	$\beta\backslash\gamma \vdash_{SD} (\alpha\backslash\gamma)\backslash(\alpha\backslash\beta)$
7.1	$(\alpha\backslash\beta)/\gamma \vdash_{SD} \alpha\backslash(\beta/\gamma)$	7.2	$\alpha\backslash(\beta/\gamma) \vdash_{SD} (\alpha\backslash\beta)/\gamma$
8.1	$(\alpha/\beta)/\gamma \vdash_{SD} \alpha/(\gamma \cdot \beta)$	8.2	$\alpha/(\gamma \cdot \beta) \vdash_{SD} (\alpha/\beta)/\gamma$

PROOF: Theorem 2.1

$$\dfrac{\dfrac{\alpha \vdash_{SD} \alpha \qquad \beta \vdash_{SD} \beta}{\alpha\ \beta \vdash_{SD} \alpha \cdot \beta}\cdot I}{\alpha \vdash_{SD} (\alpha \cdot \beta)/\beta}/I$$

To be complete, the next proof requires that the proofs for theorems 3.1 and 4.1 be placed above the starting points of what is displayed.

PROOF: Theorem 5.1

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\beta/\alpha\ \alpha \vdash_{SD} \beta}/E \qquad \dfrac{\vdots}{\beta \vdash_{SD} (\gamma/\beta)\backslash\gamma}\backslash I}{(\beta/\alpha)\ \alpha \vdash_{SD} (\gamma/\beta)\backslash\gamma}cut}{(\gamma/\beta)\ \beta/\alpha\ \alpha \vdash_{SD} \gamma}\backslash E}{\gamma/\beta\ \beta/\alpha \vdash_{SD} \gamma/\alpha}/I$$

PROOF: Theorem 8.1

$$\dfrac{\dfrac{\dfrac{}{\gamma \cdot \beta \vdash_{SD} \gamma \cdot \beta}axiom \qquad \dfrac{\dfrac{\dfrac{}{(\alpha/\beta)/\gamma \vdash_{SD} (\alpha/\beta)/\gamma}axiom}{(\alpha/\beta)/\gamma\ \ \gamma \vdash_{SD} \alpha/\beta}/E}{(\alpha/\beta)/\gamma\ \ \gamma\ \ \beta \vdash_{SD} \alpha}/E}{(\alpha/\beta)/\gamma\ \ \gamma \cdot \beta \vdash_{SD} \alpha}\cdot E}{(\alpha/\beta)/\gamma \vdash_{SD} \alpha/(\gamma \cdot \beta)}/I$$

Exercises: Sequent deductions

1. Using the sequent deduction rules, establish the theorems not established in this section.
2. Use the sequent deduction rules to establish the following.

1.1	$\alpha \cdot (\beta \cdot \gamma) \vdash_{SD} (\alpha \cdot \beta) \cdot \gamma$	1.2	$(\alpha \cdot \beta) \cdot \gamma \vdash_{SD} \alpha \cdot (\beta \cdot \gamma)$
2.1	$(\alpha / \beta) \cdot \beta \vdash_{SD} \alpha$	2.2	$\alpha \cdot (\alpha \backslash \beta) \vdash_{SD} \beta$
3.1	$(\gamma / \beta) \cdot (\beta / \alpha) \vdash_{SD} \gamma / \alpha$	3.2	$(\alpha \backslash \beta) \cdot (\beta \backslash \gamma) \vdash_{SD} \alpha \backslash \gamma$

2.2.1 Equivalence of formula deduction and sequent deduction

We now wish to show that the formula deduction presentation of the Lambek calculus is equivalent to the sequent deduction presentation. What we mean by equivalence is that the very same pairs comprising a finite list of formulae and a single formula are paired by $\vdash_{FD}$ as are paired by $\vdash_{SD}$: in other words, that, for each Θ, a finite list of formulae, and for each θ, a formula, $\Theta \vdash_{FD} \theta$ iff $\Theta \vdash_{SD} \theta$. We shall prove this in two long steps. The first long step is to show that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{FD} \theta$, then $\Theta \vdash_{SD} \theta$. The second is to show the converse: for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{SD} \theta$, then $\Theta \vdash_{FD} \theta$.

FIRST LONG STEP

To take the first long step, we must first introduce the concept of the depth of a tree. Each point where an expression, be it a formula or a sequent, occurs in a tree is a *node*. A *branch* in a tree is a sequence of all the nodes from a node in the tree, to one node immediately above it, and so forth, up to a top node. Thus, a tree has, from its bottom node, precisely as many branches as top nodes. The number of nodes in a branch is the branch's length. The *depth* of a tree is the length of its longest branch.[3]

The concept of the depth of a tree permits us to put into one group all trees of depth 1, into another all trees of depth 2, and in general, in its own group, all trees of depth n, a positive integer.

Next, we recall that $\Theta \vdash_{FD} \theta$ holds iff there is a formula deduction tree whose top formulae, from left to right, are the formulae listed in Θ and whose bottom formula is θ. Now, should we show that, for each instance of $\Theta \vdash_{FD} \theta$ which is underpinned by a formula deduction tree of depth 1, there is a sequent deduction whose bottom sequent is $\Theta \vdash_{SD} \theta$, and that, for each instance of $\Theta \vdash_{FD} \theta$ which is underpinned by a formula deduction tree of depth 2, there is a sequent deduction whose bottom sequent is $\Theta \vdash_{SD} \theta$, and that, for each instance of $\Theta \vdash_{FD} \theta$ which is underpinned by a formula deduction tree of depth 3, there is a sequent deduction whose bottom sequent is $\Theta \vdash_{SD} \theta$, and indeed that, for each

3. On this definition, a tree may have more than one longest branch. Of course, if there are two longest branches, they have the same length.

instance of $\Theta \vdash_{FD} \theta$ which is underpinned by a formula deduction tree of depth n, there is a sequent deduction whose bottom sequent is $\Theta \vdash_{SD} \theta$, then it follows that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{FD} \theta$, then $\Theta \vdash_{SD} \theta$.

It is not possible to write out a proof of what we want to prove by writing out a proof starting from the case of formula deduction trees of depth 1, then the case of formula deduction trees of depth 2, and so on, since we shall never come to the end. What we can, and must, do is to apply a kind of proof which uses the principle known as the *principle of mathematical induction*. To apply this principle here means that we first show that, for each instance of $\Theta \vdash_{FD} \theta$ which is underpinned by a formula deduction of depth 1, we have a sequent deduction of $\Theta \vdash_{SD} \theta$. This part of the proof is known as this proof's *base case*. Next, we show that, if for each instance of $\Theta \vdash_{FD} \theta$ which is underpinned by formula deduction of depth n or less, we have a sequent deduction of $\Theta \vdash_{SD} \theta$, then for each instance of $\Delta \vdash_{FD} \beta$ which is underpinned by formula deduction of depth $n+1$ or less, we have a sequent deduction of $\Delta \vdash_{SD} \beta$. This part of the proof is known as this proof's *induction case*. Once the base case and the induction case have been completed, we shall invoke the principle of mathematical induction to conclude that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{FD} \theta$, then $\Theta \vdash_{SD} \theta$.

BASE CASE

$\Theta \vdash_{FD} \theta$ results from a formula deduction of depth 1. Such a formula deduction comprises a single formula, call it α. Thus, $\Theta \vdash_{FD} \theta$ is just $\alpha \vdash_{FD} \alpha$. However, $\alpha \vdash_{SD} \alpha$ is an axiom of sequent deduction. We have thereby proved the base case.

INDUCTION CASE

INDUCTION HYPOTHESIS

For each $\Theta \vdash_{FD} \theta$ which is underpinned by a formula deduction of depth n or less (where $n > 1$), there is a sequent deduction of $\Theta \vdash_{SD} \theta$.

Now consider any sequent $\Theta \vdash_{FD} \theta$ whose underlying formula deduction is of depth $n+1$. This formula deduction must result from the application of one of the six rules of formula deduction to one formula deduction, or possibly two, one of depth n, the other (if there is one) of depth no greater than n. By the induction hypothesis, the one or two formula deductions have corresponding sequent deductions which can be extended to a sequent deduction whose bottom sequent is $\Theta \vdash_{SD} \theta$. Since we must show this to be so no matter which of the six formula deduction rules is used in the bottom step of the formula deduction of depth $n+1$, we have six cases to consider. We now consider each of these cases in turn.

CASE: $\backslash$ elimination

Suppose that $\Theta \vdash_{FD} \theta$ is underpinned by a formula deduction the last applied rule of which is $\backslash$ elimination. Then the underlying formula deduction has the following form:

$$\dfrac{\begin{array}{cc}\Gamma & \Delta \\ \vdots & \vdots \\ \alpha & \alpha\backslash\beta\end{array}}{\beta}\ \backslash E$$

where Θ is $\Gamma\ \Delta$ and θ is β. By the induction hypothesis, we know that there are sequent deductions whose bottom sequents are $\Gamma \vdash_{SD} \alpha$ and $\Delta \vdash_{SD} \alpha\backslash\beta$ respectively. We can use these two sequent deductions to obtain a sequent deduction whose bottom sequent is $\Gamma\ \Delta \vdash_{SD} \beta$, as follows.

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \vdash_{SD} \alpha\ IH & \Delta \vdash_{SD} \alpha\backslash\beta\ IH\end{array}}{\Gamma\ \Delta \vdash_{SD} \beta}\ \backslash E$$

CASE: $\backslash$ introduction

Suppose that $\Theta \vdash_{FD} \theta$ is underpinned by a formula deduction the last applied rule of which is $\backslash$ introduction. Then the underlying formula deduction has this form:

$$\dfrac{\begin{array}{cc}[\alpha] & \Delta \\ \vdots & \vdots \\ \beta & \end{array}}{\alpha\backslash\beta}\ \backslash I$$

where Θ is Δ and θ is $\alpha\backslash\beta$. By the induction hypothesis, we know that there is a sequent deduction whose bottom sequent is $\alpha\ \Delta \vdash_{SD} \beta$. Using the rule of $\backslash$ introduction, we extend this sequent deduction to one whose bottom sequent is $\Delta \vdash_{SD} \alpha\backslash\beta$.

$$\dfrac{\begin{array}{c}\vdots \\ \alpha\ \Delta \vdash_{SD} \beta\ IH\end{array}}{\Delta \vdash_{SD} \alpha\backslash\beta}\ \backslash I$$

CASE: $/$ elimination

Exercise.

CASE: $/$ introduction

Exercise.

CASE: $\cdot$ elimination

This proof is omitted.[4]

4. The form of this rule, which is different from the other rules, gives rise to a complication the accommodation of which occasions a prolixity of detail best avoided.

CASE: · introduction

Suppose that $\Theta \vdash_{FD} \theta$ is underpinned by a formula deduction the last applied rule of which is · introduction. Then the form of the underlying formula deduction is this:

$$\dfrac{\begin{array}{cc}\Gamma & \Delta \\ \vdots & \vdots \\ \alpha & \beta\end{array}}{\alpha \cdot \beta}\ {\cdot}I$$

where Θ is $\Gamma\ \Delta$ and θ is $\alpha \cdot \beta$. By the induction hypothesis, we know that there are two sequent deductions, one of $\Gamma \vdash_{SD} \alpha$ and another of $\Delta \vdash_{SD} \beta$. We can use these sequent deductions to obtain a sequent deduction of $\Gamma\ \Delta \vdash_{SD} \alpha \cdot \beta$, as follows.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{SD} \alpha}\ IH \qquad \dfrac{\vdots}{\Delta \vdash_{SD} \beta}\ IH}{\Gamma\ \Delta \vdash_{SD} \alpha \cdot \beta}\ {\cdot}I$$

This completes the proof of the induction case. We now invoke the principle of mathematical induction to conclude that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{FD} \theta$, then $\Theta \vdash_{SD} \theta$.

SECOND LONG STEP

The second step is to show that, for each Γ, a finite list of formulae, and for each α, a formula, if $\Gamma \vdash_{SD} \alpha$, then $\Gamma \vdash_{FD} \alpha$. Again, the proof is done using the principle of mathematical induction. We first show that any sequent deduction of depth 1 has a corresponding formula deduction. This is the *base case*. Next, we show that, if for each instance of $\Theta \vdash_{SD} \theta$ which is underpinned by sequent deduction of depth n or less, we have a formula deduction of $\Theta \vdash_{FD} \theta$, then for each instance of $\Delta \vdash_{SD} \beta$ which is underpinned by formula deduction of depth $n + 1$ or less, we have a formula deduction of $\Delta \vdash_{FD} \beta$. This is the proof's *induction case*. From the base case and the induction case, we conclude, invoking the principle of mathematical induction that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{SD} \theta$, then $\Theta \vdash_{FD} \theta$.

BASE CASE

$\Theta \vdash_{SD} \theta$ results from a sequent deduction of depth 1. Such a sequent deduction can only be an axiom. It therefore must have the form $\alpha \vdash_{SD} \alpha$, for some formula α. However, $\alpha \vdash_{FD} \alpha$ follows from the fact that a single formula comprises a formula deduction. Thus is established the base case.

INDUCTION CASE

INDUCTION HYPOTHESIS

For each sequent deduction of $\Theta \vdash_{SD} \theta$ of depth n or less (where $n > 1$), there is a formula deduction which underpins $\Theta \vdash_{FD} \theta$.

Now consider any sequent deduction of depth $n+1$ whose bottom sequent is $\Theta \vdash_{SD} \theta$. This sequent deduction must result from the application of one of the six rules of sequent deduction to one sequent deduction, or possibly two, one of depth n, the other, if there is one, of depth no greater than n. By the induction hypothesis, the one or two sequent deductions have corresponding formula deductions which can be extended to a formula deduction whose bottom formula is θ and whose assumptions comprise the formula list Θ. Since we must show this to be so no matter which of the six sequent deduction rules is used in the last step of the sequent deduction of depth $n+1$, we have six cases to consider. We now consider each of these cases in turn.

CASE: $\backslash$ elimination

Suppose that $\Theta \vdash_{SD} \theta$ results from a sequent deduction the last applied rule of which is $\backslash$ elimination. Then the underlying sequent deduction has the following form:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{SD} \alpha} \qquad \dfrac{\vdots}{\Delta \vdash_{SD} \alpha\backslash\beta}}{\Gamma\ \Delta \vdash_{SD} \beta}\ \backslash E$$

where Θ is $\Gamma\ \Delta$ and θ is β. By the induction hypothesis, we know that there are formula deductions whereby $\Gamma \vdash_{FD} \alpha$ and $\Delta \vdash_{FD} \alpha\backslash\beta$ both hold. We can use the corresponding two formula deductions to obtain a formula deduction of $\Gamma\ \Delta \vdash_{FD} \beta$, as follows.

$$\dfrac{\begin{array}{cc}\Gamma & \Delta \\ \vdots & \vdots \\ \dfrac{\alpha}{}\ IH & \dfrac{\alpha\backslash\beta}{}\ IH\end{array}}{\beta}\ \backslash E$$

CASE: $\backslash$ introduction

Exercise.

CASE: / elimination

Exercise.

CASE: / introduction

Suppose that $\Theta \vdash_{SD} \theta$ results from a sequent deduction the last applied rule of which is / introduction. Then the underlying sequent deduction has the following form:

$$\dfrac{\dfrac{\vdots}{\Gamma\ \alpha \vdash_{SD} \beta}}{\Gamma \vdash_{SD} \beta/\alpha}\ /I$$

where Θ is Γ and θ is β/α. By the induction hypothesis, we know that there is a formula deduction whereby $\Gamma\ \alpha \vdash_{FD} \beta$ holds. We can use the corresponding formula deduction to obtain a formula deduction whereby $\Gamma \vdash_{FD} \beta/\alpha$ holds, as follows.

$$\dfrac{\dfrac{\Gamma \qquad [\alpha]}{\begin{array}{cc}\vdots & \vdots \\ \beta & \end{array}}\,IH}{\beta/\alpha}\,/I$$

CASE: $\cdot$ elimination

Exercise.

CASE: $\cdot$ introduction

Exercise.

Once readers supply a proof for the formula deduction corresponding to the sequent deduction whose last step uses $\cdot I$, they will have shown that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{SD} \theta$, then $\Theta \vdash_{FD} \theta$.

Exercises: Equivalence of formula deduction and sequent deduction

1. Show that for each positive integer n, there is a formula deduction tree of depth n.
2. Supply the proofs signalled as exercises in this section.

2.3 Gentzen Deduction

At last we come to Gentzen deduction for the Lambek calculus.

Definition 5 Gentzen Deduction (tree format)
A deduction of a sequent is a tree of sequents each one of which either is an axiom or is obtained from sequents immediately above it in the sequent tree by one of the rules specified below.

As we recall from chapter 7, the Gentzen sequent calculus uses the same axiom and introduction rules as the deduction rules for the sequent tree format, but it replaces the elimination rules with another set of introduction rules. What are called introduction rules in the sequent tree format are relabelled as right introduction rules in the Gentzen sequent calculus and the new set of introduction rules are called left introduction rules. An analogous change is made here with respect to the Lambek calculus.

Here, then, is the axiom for the Gentzen deduction presentation of the Lambek calculus.

Axiom
$\alpha \vdash_{GD} \alpha$

Here are the introduction rules for the Gentzen deduction of the Lambek calculus.

$\backslash$	Left	Right
	$\dfrac{\Gamma \vdash_{GD} \alpha \ , \ \Delta \ \beta \ \Theta \vdash_{GD} \gamma}{\Delta \ \Gamma \ \alpha\backslash\beta \ \Theta \vdash_{GD} \gamma}$	$\dfrac{\alpha \ \Gamma \vdash_{GD} \beta}{\Gamma \vdash_{GD} \alpha\backslash\beta}$

$/$	Left	Right
	$\dfrac{\Gamma \vdash_{GD} \alpha \ , \ \Delta \ \beta \ \Theta \vdash_{GD} \gamma}{\Delta \ \beta/\alpha \ \Gamma \ \Theta \vdash_{GD} \gamma}$	$\dfrac{\Gamma \ \alpha \vdash_{GD} \beta}{\Gamma \vdash_{GD} \beta/\alpha}$

$\cdot$	Left	Right
	$\dfrac{\Gamma \ \alpha \ \beta \ \Delta \vdash_{GD} \gamma}{\Gamma \ \alpha \cdot \beta \ \Delta \vdash_{GD} \gamma}$	$\dfrac{\Gamma \vdash_{GD} \alpha \ , \ \Delta \vdash_{GD} \beta}{\Gamma \ \Delta \vdash_{GD} \alpha \cdot \beta}$

Cut
$\dfrac{\Delta \vdash_{GD} \alpha \ , \ \Gamma \ \alpha \ \Theta \vdash_{GD} \beta}{\Gamma \ \Delta \ \Theta \vdash_{GD} \beta}$

The analogs of the theorems for the formula deduction of the Lambek calculus also hold for the Gentzen deduction. Readers are asked to prove these theorems in the exercises.

THEOREMS

1.1	$\alpha \cdot (\beta \cdot \gamma) \vdash_{GD} (\alpha \cdot \beta) \cdot \gamma$	1.2	$(\alpha \cdot \beta) \cdot \gamma \vdash_{GD} \alpha \cdot (\beta \cdot \gamma)$
2.1	$\alpha \vdash_{GD} (\alpha \cdot \beta)/\beta$	2.2	$\beta \vdash_{GD} \alpha\backslash(\alpha \cdot \beta)$
3.1	$\alpha/\beta \ \beta \vdash_{GD} \alpha$	3.2	$\alpha \ \alpha\backslash\beta \vdash_{GD} \beta$
4.1	$\alpha \vdash_{GD} (\beta/\alpha)\backslash\beta$	4.2	$\alpha \vdash_{GD} \beta/(\alpha\backslash\beta)$
5.1	$\gamma/\beta \ \beta/\alpha \vdash_{GD} \gamma/\alpha$	5.2	$\alpha\backslash\beta \ \beta\backslash\gamma \vdash_{GD} \alpha\backslash\gamma$
6.1	$\gamma/\beta \vdash_{GD} (\gamma/\alpha)/(\beta/\alpha)$	6.2	$\beta\backslash\gamma \vdash_{GD} (\alpha\backslash\gamma)\backslash(\alpha\backslash\beta)$
7.1	$(\alpha\backslash\beta)/\gamma \vdash_{GD} \alpha\backslash(\beta/\gamma)$	7.2	$\alpha\backslash(\beta/\gamma) \vdash_{GD} (\alpha\backslash\beta)/\gamma$
8.1	$(\alpha/\beta)/\gamma \vdash_{GD} \alpha/(\gamma \cdot \beta)$	8.2	$\alpha/(\gamma \cdot \beta) \vdash_{GD} (\alpha/\beta)/\gamma$

PROOF: Theorem 3.1

$$\dfrac{\dfrac{}{\alpha \vdash_{GD} \alpha}\,axiom \qquad \dfrac{}{\beta \vdash_{GD} \beta}\,axiom}{(\alpha/\beta)\ \beta \vdash_{GD} \alpha}\,/L$$

PROOF: Theorem 6.1

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\gamma \vdash_{GD} \gamma}\,axiom \qquad \dfrac{}{\beta \vdash_{GD} \beta}\,axiom}{\gamma/\beta\ \beta \vdash_{GD} \gamma}\,/L \qquad \dfrac{}{\alpha \vdash_{GD} \alpha}\,axiom}{\gamma/\beta\ \beta/\alpha\ \alpha \vdash_{GD} \gamma}\,/L}{\gamma/\beta\ \beta/\alpha \vdash_{GD} \gamma/\alpha}\,/R}{\gamma/\beta \vdash_{GD} (\gamma/\alpha)/\beta/\alpha}\,/R$$

Exercises: Gentzen deduction

1. Using the Gentzen deduction rules, establish the theorems not established in this section.
2. Use the Gentzen deduction rules to establish the following.

1.1	$\alpha \cdot (\beta \cdot \gamma) \vdash_{GD} (\alpha \cdot \beta) \cdot \gamma$	1.2	$(\alpha \cdot \beta) \cdot \gamma \vdash_{GD} \alpha \cdot (\beta \cdot \gamma)$
2.1	$(\alpha/\beta) \cdot \beta \vdash_{GD} \alpha$	2.2	$\alpha \cdot (\alpha\backslash\beta) \vdash_{GD} \beta$
3.1	$(\gamma/\beta) \cdot (\beta/\alpha) \vdash_{GD} \gamma/\alpha$	3.2	$(\alpha\backslash\beta) \cdot (\beta\backslash\gamma) \vdash_{GD} \alpha\backslash\gamma$

2.3.1 Equivalence of sequent deduction and Gentzen deduction

As we are about to see, the sequent deduction presentation of the Lambek calculus is equivalent to the Gentzen deduction presentation: that is, for each Θ, a finite list of formulae, and for each θ, a formula, $\Theta \vdash_{SD} \theta$ iff $\Theta \vdash_{GD} \theta$.

As with the proof of the equivalence of the formula deduction presentation and the sequent deduction presentation of the Lambek calculus, we proceed in two steps. The first step is to show that, for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{SD} \theta$, then $\Theta \vdash_{GD} \theta$. The second is to show the converse: for each Θ, a finite list of formulae, and for each θ, a formula, if $\Theta \vdash_{GD} \theta$, then $\Theta \vdash_{SD} \theta$. Since both presentations share half of their rules, these two steps will be shorter than their earlier counterparts. Each step is a proof using mathematical induction.

FIRST STEP

As with each of the two steps for the proof of the equivalence of the formula deduction presentation of the Lambek Calculus and the sequent deduction presentation, we shall use the principle of mathematical induction.

BASE CASE

$\Theta \vdash_{SD} \theta$ results from a sequent deduction of depth 1. Such a sequent deduction can only be an axiom. It therefore must have the form $\alpha \vdash_{SD} \alpha$, for some formula α. However, $\alpha \vdash_{GD} \alpha$ is an axiom of Gentzen deduction. The base case is thus established.

INDUCTION CASE

INDUCTION HYPOTHESIS

For each $\Theta \vdash_{SD} \theta$ which results from a formula deduction of depth n or less (where $n > 1$), there is a Gentzen deduction of $\Theta \vdash_{GD} \theta$.

Now consider any sequent deduction of depth $n+1$ whose bottom sequent is $\Theta \vdash_{SD} \theta$. This sequent deduction must result from the application of one of the six rules of sequent deduction to one sequent deduction, or possibly two, one of depth n, the other (if there is one) of depth no greater than n. By the induction hypothesis, the one or two sequent deductions have corresponding Gentzen deductions which can be extended to a Gentzen deduction whose bottom sequent is $\Theta \vdash_{GD} \theta$. Since we must show this to be so no matter which of the six sequent deduction rules is used in the last step of the sequent deduction of depth $n+1$, we have six cases to consider. However, since the introduction rules of sequent deduction are the same as the right introduction rules of Gentzen deduction, we need only consider three cases, namely those sequent deductions where the last rule used is one of the three elimination rules.

CASE: $\backslash$ elimination

Suppose that $\Theta \vdash_{SD} \theta$ results from a sequent deduction the last applied rule of which is $\backslash$ elimination. Then the underlying sequent deduction has the following form:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{SD} \alpha} \qquad \dfrac{\vdots}{\Delta \vdash_{SD} \alpha\backslash\beta}}{\Gamma\ \Delta \vdash_{SD} \beta}\ \backslash E$$

where Θ is $\Gamma\ \Delta$ and θ is β. By the induction hypothesis, we know that there are Gentzen deductions whose bottom sequents are $\Gamma \vdash_{GD} \alpha$ and $\Delta \vdash_{GD} \alpha\backslash\beta$, respectively. We can use these corresponding two Gentzen deductions to obtain a Gentzen deduction whose bottom sequent is $\Gamma\ \Delta \vdash_{GD} \beta$, as follows.

$$\dfrac{\dfrac{\vdots}{\Delta \vdash_{GD} \alpha\backslash\beta}\ IH \qquad \dfrac{\dfrac{\vdots}{\Gamma \vdash_{GD} \alpha}\ IH \qquad \dfrac{\dfrac{}{\alpha \vdash_{GD} \alpha}\ axiom \qquad \dfrac{}{\beta \vdash_{GD} \beta}\ axiom}{\alpha\ \alpha\backslash\beta \vdash_{GD} \beta}\ \backslash L}{\Gamma\ \alpha\backslash\beta \vdash_{GD} \beta}\ cut}{\Gamma\ \Delta \vdash_{GD} \beta}\ cut$$

CASE: $/$ elimination

Exercise.

CASE: $\cdot$ elimination

Suppose that $\Theta \vdash_{SD} \theta$ results from a sequent deduction the last applied rule of which is $\cdot$ elimination. Then the underlying sequent deduction has the following form:

$$\dfrac{\dfrac{\vdots}{\Delta \vdash_{SD} \alpha \cdot \beta} \qquad \dfrac{\vdots}{\Gamma\ \alpha\ \beta\ \Upsilon \vdash_{SD} \gamma}}{\Gamma\ \Delta\ \Upsilon \vdash_{SD} \gamma}\ {\cdot}E$$

where Θ is $\Gamma\ \Delta\ \Upsilon$ and θ is γ. By the induction hypothesis, we know that there are Gentzen deductions whose bottom sequents are $\Delta \vdash_{GD} \alpha \cdot \beta$ and $\Gamma\ \alpha\ \beta\ \Upsilon \vdash_{GD} \gamma$ respectively. We can use the corresponding two Gentzen deductions to obtain a Gentzen deduction whose bottom sequent is $\Gamma\ \Delta\ \Upsilon \vdash_{GD} \gamma$, shown here.

$$\dfrac{\dfrac{\vdots}{\Delta \vdash_{GD} \alpha \cdot \beta}\ IH \qquad \dfrac{\dfrac{\vdots}{\Gamma\ \alpha\ \beta\ \Upsilon \vdash_{GD} \gamma}\ IH}{\Gamma\ \alpha \cdot \beta\ \Upsilon \vdash_{GD} \gamma}\ {\cdot}L}{\Gamma\ \Delta\ \Upsilon \vdash_{GD} \gamma}\ cut$$

We have now established the first step. Before starting the second step, we establish two so-called *derived rules* for sequent deduction presentation of the Lambek Calculus. These derived rules permit one to shorten sequent deductions. They are, in principle, eliminable. Thus, they do not alter in any way the set of sequents provable in the sequent deduction presentation. They do, however, shorten the length of sequent deductions. We shall use these derived rules in the second step.

The first derived rule, labelled $DR \cdot LE$, permits one to replace any formula of the form $(\ldots(\alpha_1 \cdot \alpha_2)\ldots) \cdot \alpha_n$ in a list Γ with the list $\alpha_1\ \alpha_2 \ldots \alpha_n$. We restate the rule in the usual format for sequent deduction rules as follows:

$$\dfrac{\Gamma\ (\ldots(\alpha_1 \cdot \alpha_2)\ldots) \cdot \alpha_n\ \Delta \vdash_{SD} \beta}{\Gamma\ \alpha_1\ \alpha_2 \ldots\ \alpha_n\ \Delta \vdash_{SD} \beta}\ DR\cdot LE$$

(Both Γ and Δ may be empty lists.)

Note that, if $n = 2$, then $(\ldots(\alpha_1 \cdot \alpha_2)\ldots) \cdot \alpha_n$ is just $(\alpha_1 \cdot \alpha_2)$

The second derived rule, labelled $DR \cdot LI$, permits one to replace any sublist of formulae $\alpha_1\ \alpha_2 \ldots \alpha_n$ in the list Γ with the single formula $(\ldots(\alpha_1 \cdot \alpha_2)\ldots) \cdot \alpha_n$.

$$\dfrac{\Gamma\ \alpha_1\ \alpha_2 \ldots\ \alpha_n\ \Delta \vdash_{SD} \beta}{\Gamma\ (\ldots(\alpha_1 \cdot \alpha_2)\ldots) \cdot \alpha_n\ \Delta \vdash_{SD} \beta}\ DR\cdot LI$$

(Both Γ and Δ may be empty lists.)

We prove that these derived rules hold in the sequent deduction presentation of the Lambek Calculus using the principle of mathematical induction. The base case considers an

instance with just two formulae, α_1 and α_2.[5] The induction case hypothesizes that the rule holds for n formulae, α_1 through α_n, and shows that the rule also holds for $n+1$ formulae, α_1 through α_{n+1}.

PROOF: $DR \cdot LI$

BASE CASE

We wish to show that, if there is a sequent deduction of $\Gamma\ \alpha_1 \cdot \alpha_2\ \Delta \vdash_{SD} \beta$, then there is a sequent deduction of $\Gamma\ \alpha_1\ \alpha_2\ \Delta \vdash_{SD} \beta$. The proof consists in showing that a sequent deduction of $\Gamma\ \alpha_1 \cdot \alpha_2\ \Delta \vdash_{SD} \beta$ can be extended to a sequent deduction of $\Gamma\ \alpha_1\ \alpha_2\ \Delta \vdash_{SD} \beta$. The last proof line of the supposed sequent deduction of $\Gamma\ \alpha_1 \cdot \alpha_2\ \Delta \vdash_{SD} \beta$ is labelled with S.

$$\dfrac{\dfrac{\dfrac{}{\alpha_1 \vdash_{SD} \alpha_1}\ axiom \quad \dfrac{}{\alpha_2 \vdash_{SD} \alpha_2}\ axiom}{\alpha_1\ \alpha_2 \vdash_{SD} \alpha_1 \cdot \alpha}\ \cdot R \qquad \dfrac{\vdots}{\Gamma\ \alpha_1 \cdot \alpha_2\ \Delta \vdash_{SD} \beta}\ S}{\Gamma\ \alpha_1\ \alpha_2\ \Delta \vdash_{SD} \beta}\ cut$$

INDUCTION CASE

We now turn to the induction case in the proof of this derived rule.

INDUCTION HYPOTHESIS

$$\dfrac{\Gamma\ (\ldots(\alpha_1 \cdot \alpha_2)\ldots) \cdot \alpha_n\ \Delta \vdash_{SD} \beta}{\Gamma\ \alpha_1\ \alpha_2 \ldots\ \alpha_n\ \Delta \vdash_{SD} \beta}\ IH$$

Suppose that there is a sequent deduction whose bottom sequent is $\Theta\ (\ldots(\gamma_1 \cdot \gamma_2)\ldots) \cdot \gamma_{n+1}\ \Upsilon \vdash_{SD} \delta$. This deduction can be extended to a sequent deduction whose bottom sequent is $\Theta\ \gamma_1\ \gamma_2\ \ldots\ \gamma_{n+1}\ \Upsilon \vdash_{SD} \delta$. We display the extended deduction below, abbreviating $(\ldots(\gamma_1 \cdot \gamma_2)\ldots) \cdot \gamma_n$ as $\bar{\gamma}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\bar{\gamma} \vdash_{SD} \bar{\gamma}}\ axiom \quad \dfrac{}{\gamma_{n+1} \vdash_{SD} \gamma_{n+1}}\ axiom}{\bar{\gamma}\ \gamma_{n+1} \vdash_{SD} \bar{\gamma} \cdot \gamma_{n+1}}\ \cdot I \qquad \dfrac{\vdots}{\Theta\ \bar{\gamma} \cdot \gamma_{n+1}\ \Upsilon \vdash_{SD} \delta}\ S}{\Theta\ \bar{\gamma}\ \gamma_{n+1}\ \Upsilon \vdash_{SD} \delta}\ cut}{\Theta\ \gamma_1\ \ldots\ \gamma_{n+1}\ \Upsilon \vdash_{SD} \delta}\ IH$$

PROOF: $DR \cdot LE$

Exercise.

5. Those familiar with proofs using the principle of mathematical induction know that one may take as the base case an instance of just one formula, α_1.

SECOND STEP

We now turn to the second step. Once again we proceed with a proof using the principle of mathematical induction.

BASE CASE

$\Theta \vdash_{GD} \theta$ results from a Gentzen deduction of depth 1. Such a Gentzen deduction can only be an axiom. It therefore must have the form $\alpha \vdash_{GD} \alpha$, for some formula α. However, $\alpha \vdash_{SD} \alpha$ is an axiom of sequent deduction. This completes the base case.

INDUCTION CASE

We now turn to the induction case of the second step.

INDUCTION HYPOTHESIS

For each $\Theta \vdash_{GD} \theta$ which results from a Gentzen deduction of depth n or less (where $n > 1$), there is a sequent deduction of $\Theta \vdash_{SD} \theta$.

Now consider any Gentzen deduction of depth $n + 1$ whose bottom sequent is $\Theta \vdash_{GD} \theta$. This Gentzen deduction must result from the application of one of the six rules of Gentzen deduction to one Gentzen deduction, or possibly two, one of depth n, the other, if there is one, of depth no greater than n. By the induction hypothesis, the one or two Gentzen deductions have corresponding sequent deductions which can be extended to a sequent deduction whose bottom sequent is $\Theta \vdash_{SD} \theta$. Since we must show this to be so no matter which of the six Gentzen deduction rules is used in the last step of the sequent deduction of depth $n + 1$, we have six cases to consider. However, again, since the introduction rules of sequent deduction are the same as the right introduction rules of Gentzen deduction, we in fact have only three cases to consider, namely those Gentzen deductions where the last rule used is one of the three left introduction rules.

CASE: $\backslash$ left introduction

Suppose that $\Theta \vdash_{GD} \theta$ results from a Gentzen deduction the last applied rule of which is $\backslash$ left introduction. Then the underlying Gentzen deduction has the following form:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{GD} \alpha} \qquad \dfrac{\vdots}{\Delta\ \beta\ \Upsilon \vdash_{GD} \gamma}}{\Delta\ \Gamma\ \alpha\backslash\beta\ \Upsilon \vdash_{GD} \gamma}\ \backslash L$$

where Θ is $\Delta\ \Gamma\ \alpha\backslash\beta\ \Upsilon$ and θ is γ. By the induction hypothesis, we know that there are sequent deductions whose bottom sequents are $\Gamma \vdash_{SD} \alpha$ and $\Delta\ \beta\ \Upsilon \vdash_{SD} \gamma$ respectively. We can use these two sequent deductions to obtain a sequent deduction whose bottom sequent is $\Delta\ \Gamma\ \alpha\backslash\beta\ \Upsilon \vdash_{SD} \gamma$.

We display the extended sequent deduction below. Though the deduction is not especially subtle, its display does not fit easily on one page. We therefore divide the deduction

into three subdeductions. The first subdeduction uses one part of the induction hypothesis to arrive at the sequent $\Gamma\ \alpha\backslash\beta \vdash_{SD} \beta$. The second subdeduction uses the second part of the induction hypothesis to arrive at the sequent $\beta \vdash_{SD} \bar{\delta}\backslash(\gamma\,/\,\bar{\upsilon})$, where we abbreviate the formula corresponding to the formula list Δ as $\bar{\delta}$ and the one corresponding to the formula list Υ as $\bar{\upsilon}$. The third subdeduction uses the two sequents arrived at by the first and second subdeductions to arrive at the desired sequent, $\Delta\ \Gamma\ \alpha\backslash\beta\ \Upsilon \vdash_{SD} \gamma$. Finally, in the first subdeduction we omit the sequent deduction proof of Theorem 4.2

FIRST SUBDEDUCTION:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash_{SD} \alpha}\ IH \qquad \dfrac{\vdots}{\alpha \vdash_{SD} \beta/(\alpha\backslash\beta)}\ Theorem\ 4.2}{\Gamma \vdash_{SD} \beta/(\alpha\backslash\beta)}\ cut \qquad \dfrac{}{\alpha\backslash\beta \vdash_{SD} \alpha\backslash\beta}\ axiom}{\Gamma\ \alpha\backslash\beta \vdash_{SD} \beta}\ /E$$

SECOND SUBDEDUCTION:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Delta\ \beta\ \Upsilon \vdash_{SD} \gamma}\ IH}{\bar{\delta}\ \beta\ \Upsilon \vdash_{SD} \gamma}\ DR{\cdot}LI}{\bar{\delta}\ \beta\ \bar{\upsilon} \vdash_{SD} \gamma}\ DR{\cdot}LI}{\bar{\delta}\ \beta \vdash_{SD} \gamma\,/\,\bar{\upsilon}}\ /I}{\beta \vdash_{SD} \bar{\delta}\backslash(\gamma\,/\,\bar{\upsilon})}\ \backslash I$$

THIRD SUBDEDUCTION:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Gamma\ \alpha\backslash\beta \vdash_{SD} \beta}\ first \qquad \dfrac{\vdots}{\beta \vdash_{SD} \bar{\delta}\backslash(\gamma\,/\,\bar{\upsilon})}\ second}{\Gamma\ \alpha\backslash\beta \vdash_{SD} \bar{\delta}\backslash(\gamma\,/\,\bar{\upsilon})}\ cut \qquad \dfrac{}{\bar{\delta} \vdash_{SD} \bar{\delta}}\ axiom}{\bar{\delta}\ \Gamma\ \alpha\backslash\beta \vdash_{SD} \gamma\,/\,\bar{\upsilon}}\ \backslash E \qquad \dfrac{}{\bar{\upsilon} \vdash_{SD} \bar{\upsilon}}\ axiom}{\bar{\delta}\ \Gamma\ \alpha\backslash\beta\ \bar{\upsilon} \vdash_{SD} \gamma}\ /E$$
$$\dfrac{\dfrac{\bar{\delta}\ \Gamma\ \alpha\backslash\beta\ \bar{\upsilon} \vdash_{SD} \gamma}{\Delta\ \Gamma\ \alpha\backslash\beta\ \bar{\upsilon} \vdash_{SD} \gamma}\ DR{\cdot}LE}{\Delta\ \Gamma\ \alpha\backslash\beta\ \Upsilon \vdash_{SD} \gamma}\ DR{\cdot}LE$$

PROOF: / left introduction

Exercise.

PROOF: $\cdot$ left introduction

Suppose that $\Theta \vdash_{GD} \theta$ results from a Gentzen deduction the last applied rule of which is $\cdot$ left introduction. Then the underlying Gentzen deduction has the following form:

$$\dfrac{\dfrac{\vdots}{\Gamma\ \alpha\ \beta\ \Delta \vdash_{GD} \gamma}}{\Gamma\ \alpha \cdot \beta\ \Delta \vdash_{GD} \gamma}\ {\cdot}L$$

where Θ is $\Gamma\ \alpha \cdot \beta\ \Delta$ and θ is γ. By the induction hypothesis, we know that there is a sequent deduction whose bottom sequent is $\Gamma\ \alpha \cdot \beta\ \Delta \vdash_{SD} \gamma$ corresponding to the Gentzen deduction whose bottom sequent is $\Gamma\ \alpha\ \beta\ \Delta \vdash_{GD} \gamma$. We extend this sequent deduction to obtain a sequent deduction whose bottom sequent is $\Gamma\ \alpha \cdot \beta\ \Delta \vdash_{GD} \gamma$.

$$\dfrac{\dfrac{}{\alpha \cdot \beta \vdash_{SD} \alpha \cdot \beta}\ axiom \qquad \dfrac{\vdots}{\Gamma\ \alpha\ \beta\ \Theta \vdash_{SD} \gamma}\ IH}{\Gamma\ \alpha \cdot \beta\ \Theta \vdash_{SD} \gamma}\ {\cdot}E$$

Exercises: Equivalence of sequent deduction and Gentzen deduction

1. Prove, without availing yourself of the equivalences discussed, that the formula deduction presentation of the Lambek calculus is equivalent to the Gentzen deduction presentation: that is, for each Θ, a finite list of formulae, and for each θ, a formula, $\Theta \vdash_{FD} \theta$ iff $\Theta \vdash_{GD} \theta$.

2.4 Cut Elimination

We now turn to the cut elimination theorem for the Lambek calculus. The cut elimination theorem, as its name suggests, asserts that the cut rule can be eliminated. In other words, for any deduction from axioms alone of a sequent in the Lambek calculus in which it is applied, there is a corresponding deduction from axioms of the very same sequent in which the cut rule is not applied at all. Or, to put it another way, every sequent theorem of the Lambek calculus can be proved without any application of the cut rule.

The proof proceeds by induction on the degree of a cut. The degree of a cut is the sum of the degree of the rule's constituents. Recall the form of the cut rule.

$$\dfrac{\Gamma \vdash_{GD} \alpha \qquad \Delta\ \alpha\ \Theta \vdash_{GD} \beta}{\Delta\ \Gamma\ \Theta \vdash_{GD} \beta}\ cut$$

The constituents of an application of the cut rule are: Γ, Δ, Θ, α, and β. The degree of a formula is the number of appearances of a connective in the formula and the degree of a list is the sum of the degrees of the formulae in the list. The Lambek calculus has, besides the cut rule, six rules and one axiom. The six rules comprise three R rules, one for each connective, and three L rules, again one for each connective. The proof is to show that, for any application of the cut rule in a deduction, either the cut rule can be eliminated

altogether or it can be replaced by other applications of the cut rule of lesser degree. Though the resulting deduction's depth may increase or decrease by one, its number of nodes never increases. This means that, by repeated elimination of applications of the cut rule or by repeated reductions of the degree of any application of the cut rule, any proof applying the cut rule can be turned into one that does not apply it at all; this can be done in a finite number of eliminations or reductions. In total, there are fourteen cases to consider. There are seven cases to consider for the left-hand premise of the cut rule and seven cases for the right-hand premise. The seven cases comprise the case when the premise in question is an axiom, the three cases where the premise is the last step in a subdeduction resulting from an application of an L rule, and the three cases where the premise is the last step in a subdeduction resulting from an application of an R rule.

We shall start with the two cases in which one of the premises is an axiom.

CASE 1.1

We begin by instantiating the left premise as the axiom $\gamma \vdash_{GD} \gamma$.[6] This means that γ is substituting for α as well as for Γ. The result is the following:

$$\frac{\gamma \vdash_{GD} \gamma \qquad \Delta\ \gamma\ \Theta \vdash_{GD} \beta}{\Delta\ \gamma\ \Theta \vdash_{GD} \beta}\ cut$$

The conclusion of this application of the cut rule is identical with its right-hand premise. Thus, this application of the cut rule is entirely superfluous: any deduction in which such a configuration is found can be shortened by excising both the left-and right-hand sides.

CASE 1.2

Next, we instantiate the right premise as the axiom $\gamma \vdash_{GD} \gamma$. This means that Δ and Θ are the empty list and that α and β are γ. One thereby obtains the following instantiation.

$$\frac{\Gamma \vdash_{GD} \gamma \qquad \gamma \vdash_{GD} \gamma}{\Gamma \vdash_{GD} \gamma}\ cut$$

This time the conclusion of the application of the cut rule is identical with its left-hand premise. Again, the application of the cut rule is entirely superfluous.

We now turn to the cases where one of the premises results from one of the rules. We shall consider initially the cases where the L rules apply, considering first the three cases where their application results in the left-hand premise and then the three cases where their application results in the right-hand premise.

6. The two premises are unordered with respect to one another so that, technically speaking, there are no left- and right-hand premises. However, we shall persist in distinguishing the two premises as left and right, thinking of the particular expression of the cut rule stated earlier.

CASE 2.1: Application of the L rules

CASE 2.1.1: Left-hand premise

We have three subcases to consider, corresponding to applications of the / L rule, the \ L rule, and the · L rule, which yield the left-hand premise.

SUBCASE: / L rule (left-hand premise)

Let the left-hand premise, which is of the form $\Gamma \vdash_{GD} \alpha$, result from an application of the / L rule. Then its list has a formula of the form δ/γ. Furthermore, because the left-hand premise results from an application of / L, it itself is immediately dominated by two premises, one of the form $\Psi \vdash_{GD} \gamma$ and the other of the form $\Upsilon\ \delta\ \Lambda \vdash_{GD} \alpha$. These premises, in turn, require Γ in the left-hand premise to have the form $\Upsilon\ \delta/\gamma\ \ \Psi\ \Lambda$. Thus, the left-hand premise has the form $\Upsilon\ \delta/\gamma\ \ \Psi\ \Lambda \vdash_{GD} \alpha$. All this implies that the application of the cut rule that we are considering appears in the following configuration.

$$\dfrac{\dfrac{\Psi \vdash_{GD} \gamma \qquad \Upsilon\ \delta\ \Lambda \vdash_{GD} \alpha}{\Upsilon\ \delta/\gamma\ \ \Psi\ \Lambda \vdash_{GD} \alpha}\ /L \qquad \Delta\ \alpha\ \Theta \vdash_{GD} \beta}{\Delta\ \Upsilon\ \delta/\gamma\ \ \Psi\ \Lambda\ \Theta \vdash_{GD} \beta}\ cut$$

We can now rearrange this portion of any longer deduction in which the previous configuration appears into the same deduction, but replacing the previous portion by the following.

$$\dfrac{\Psi \vdash_{GD} \gamma \qquad \dfrac{\Upsilon\ \delta\ \Lambda \vdash_{GD} \alpha \qquad \Delta\ \alpha\ \Theta \vdash_{GD} \beta}{\Delta\ \Upsilon\ \delta\ \Lambda\ \Theta \vdash_{GD} \beta}\ cut}{\Delta\ \Upsilon\ \delta/\gamma\ \ \Psi\ \Lambda\ \Theta \vdash_{GD} \beta}\ /L$$

REMARKS

1. The depth of the second deduction tree could increase or decrease by 1.
2. Each deduction tree has the same number of nodes.
3. Each deduction tree has exactly the same number of applications of the cut rule.
4. The application of the cut rule in the portion of the second deduction is higher in its deduction tree than the application of the cut rule in the portion of the first deduction tree.
5. Moreover, the degree of the application of the cut rule in the portion of the first deduction tree is greater than the degree of the application of the cut rule in the portion of the second deduction tree. Since $d(\delta/\gamma) = d(\delta) + d(\gamma) + 1$, $d(\Delta\ \Theta\ \Psi\ \Upsilon\ \Lambda\ \alpha\ \beta\ \delta/\gamma) > d(\Delta\ \Theta\ \Upsilon\ \Lambda\ \alpha\ \beta\ \delta)$.

SUBCASE: \ L rule (left-hand premise)

Let $\Gamma \vdash_{GD} \alpha$ result from an application of the \ L rule to yield the left-hand premise. (This is left as an exercise.)

SUBCASE: · L rule (left-hand premise)

Let the left-hand premise, which is of the form $\Gamma \vdash_{GD} \alpha$, result from an application of the · L rule. Then its list has a formula of the form $\delta \cdot \gamma$. Γ, then, has the form $\Upsilon\ \delta \cdot \gamma\ \Lambda$. So, the left-hand premise is $\Upsilon\ \delta \cdot \gamma\ \Lambda \vdash_{GD} \alpha$. Now this sequent results from an application of · L. For this rule to apply, the left-hand premise must be immediately dominated by a premise of the form $\Upsilon\ \delta\ \gamma\ \Lambda \vdash_{GD} \alpha$. Putting all this together, one obtains the following instantiation of the cut rule whose application we are considering.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Upsilon\ \delta\ \gamma\ \Lambda \vdash_{GD} \alpha}}{\Upsilon\ \delta \cdot \gamma\ \Lambda \vdash_{GD} \alpha}\cdot L \qquad \dfrac{\vdots}{\Delta\ \alpha\ \Theta \vdash_{GD} \beta}}{\Delta\ \Upsilon\ \delta \cdot \gamma\ \Lambda\ \Theta \vdash_{GD} \beta}\ cut$$

We can now rearrange this portion of any longer deduction in which the preceding portion appears into the same deduction, but replacing the preceding portion by the following:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Upsilon\ \delta\ \gamma\ \Lambda \vdash_{GD} \alpha} \qquad \dfrac{\vdots}{\Delta\ \alpha\ \Theta \vdash_{GD} \beta}}{\Delta\ \Upsilon\ \delta\ \gamma\ \Lambda\ \Theta \vdash_{GD} \beta}\ cut}{\Delta\ \Upsilon\ \delta \cdot \gamma\ \Lambda\ \Theta \vdash_{GD} \alpha}\cdot L$$

REMARKS

1. The depth of the second deduction tree could increase by 1.
2. Each deduction tree has the same number of nodes.
3. Each deduction tree has exactly the same number of applications of the cut rule.
4. The application of the cut rule in the portion of the second deduction is higher in its deduction tree than the application of the cut rule in the portion of the first deduction tree.
5. Moreover, the degree of the application of the cut rule in the portion of the first deduction tree is greater than the degree of the application of the cut rule in the portion of the second deduction tree. Since $d(\delta \cdot \gamma) = d(\delta) + d(\gamma) + 1$, $d(\Delta\ \Theta\ \Psi\ \Upsilon\ \Lambda\ \alpha\ \beta\ \delta \cdot \gamma) > d(\Delta\ \Theta\ \Upsilon\ \Lambda\ \alpha\ \beta\ \delta\ \gamma)$.

CASE 2.1.2: Right-hand premise

Again, we have three subcases to consider, corresponding to applications of / L rule, \ L rule, and the · L rule, this time to the right-hand premise.

SUBCASE: / L rule (right-hand premise)

Let the right-hand premise, which is of the form $\Delta\ \alpha\ \Theta \vdash_{GD} \beta$, result from an application of the / L rule. So α is a formula of the form δ/γ . Since δ/γ results from an application of

/ L rule, Θ must have the form of $\Xi\ \Lambda$, where Λ may be empty, but Ξ is not. The right-hand premise, then, has the form $\Upsilon\ \delta/\gamma\ \ \Xi\ \Lambda \vdash_{GD} \beta$ and it is immediately dominated by two sequents, one of the form $\Xi \vdash_{GD} \gamma$ and the other of the form $\Upsilon\ \delta\ \Lambda \vdash_{GD} \beta$. In addition, the left-hand premise has the form $\Psi \vdash_{GD} \delta/\gamma$. This, in turn, requires that it result from an application of the / R rule and that it be immediately dominated by a sequent of the form $\Psi\ \gamma \vdash_{GD} \delta$. Putting this all together, one obtains the following instantiation of the application of the cut rule.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Psi\ \gamma \vdash_{GD} \delta}}{\Psi \vdash_{GD} \delta/\gamma}\,/R \qquad \dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \gamma} \qquad \dfrac{\vdots}{\Upsilon\ \delta\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \delta/\gamma\ \ \Xi\ \Lambda \vdash_{GD} \beta}\,/L}{\Upsilon\ \Psi\ \Xi\ \Lambda \vdash_{GD} \beta}\,cut$$

Any deduction that includes such a configuration can be rearranged into a deduction where the relevant portion is reconfigured as follows.

$$\dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \gamma} \qquad \dfrac{\dfrac{\vdots}{\Psi\ \gamma \vdash_{GD} \delta} \qquad \dfrac{\vdots}{\Upsilon\ \delta\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \Psi\ \gamma\ \ \Lambda \vdash_{GD} \beta}\,cut}{\Upsilon\ \Psi\ \Xi\ \Lambda \vdash_{GD} \beta}\,cut$$

REMARKS

1. The depth of the second deduction tree could decrease by 1.
2. The second deduction tree has one less node than the first.
3. Although the portion of the second deduction tree shown has two applications of the cut rule while the portion of the first has only one, each of the applications in the portion of the second deduction tree has a lesser degree than does the single application of the cut rule in the portion shown of the first deduction tree. Since $d(\delta/\gamma) = d(\delta) + d(\gamma) + 1$, $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta/\gamma) > d(\Psi\ \Upsilon\ \Lambda\ \beta\ \delta\ \gamma)$ and $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta/\gamma) > d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \gamma)$.

SUBCASE: \ L rule (right-hand premise)

Let $\Delta\ \alpha\ \Theta \vdash_{GD} \beta$ result from an application of the \ L rule to the right-hand premise. (This is left as an exercise.)

SUBCASE: $\cdot$ L rule (right-hand premise)

Let $\Delta\ \alpha\ \Theta \vdash_{GD} \beta$ result from an application of the $\cdot$ L rule. So α is a formula of the form $\delta \cdot \gamma$. Since the right-hand premise results from an application of the $\cdot$ L rule, it has the form $\Upsilon\ \delta \cdot \gamma\ \ \Lambda \vdash_{GD} \beta$ and it is immediately dominated by a sequent of the form $\Upsilon\ \delta\ \gamma\ \ \Lambda \vdash_{GD} \beta$. At the same time, the left-hand premise has the form $\Gamma \vdash_{GD} \delta \cdot \gamma$. So, it has resulted from an application of the $\cdot$ R rule. Thus, Γ has two sub lists, Ξ and T, and

the left-hand premise is immediately dominated by two sequents of the form $\Xi \vdash_{GD} \delta$ and $\Psi \vdash_{GD} \gamma$. The following is, then, the instantiation of the application of the cut rule we are considering.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \delta} \quad \dfrac{\vdots}{\Psi \vdash_{GD} \gamma}}{\Xi\ \Psi \vdash_{GD} \delta \cdot \gamma}\cdot R \quad \dfrac{\dfrac{\vdots}{\Upsilon\ \delta\ \gamma\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \delta \cdot \gamma\ \Lambda \vdash_{GD} \beta}\cdot L}{\Upsilon\ \Xi\ \Psi\ \Lambda \vdash_{GD} \beta}\ cut$$

The preceding configuration within any deduction can be rearranged in the following way to yield a fresh deduction:

$$\dfrac{\dfrac{\vdots}{\Psi \vdash_{GD} \gamma} \quad \dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \delta} \quad \dfrac{\vdots}{\Upsilon\ \delta\ \gamma\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \Xi\ \gamma\ \Lambda \vdash_{GD} \beta}\ cut}{\Upsilon\ \Xi\ \Psi\ \Lambda \vdash_{GD} \beta}\ cut$$

REMARKS

1. The depth of the second deduction tree could decrease by 1.
2. The second deduction tree has one less node than the first.
3. Although the portion of the second deduction tree shown has two applications of the cut rule, while the portion of the first has only one, each of the applications in the portion of the second deduction tree shown has a lesser degree than does the single application of the cut rule in the portion shown of the first deduction tree. Since $d(\delta \cdot \gamma) = d(\delta) + d(\gamma) + 1$, $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta \cdot \gamma) > d(\Xi\ \Upsilon\ \Lambda\ \beta\ \delta\ \gamma)$ and $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta \cdot \gamma) > d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \gamma)$.

CASE 2.2: Application of the R rules

We now turn to the deductions with applications of the R rules. Again, we start with those cases where the left-hand premise of the cut rule results from an application of an R rule and then we turn to those cases where the right-hand premise results from such applications.

CASE 2.2.1: Left-hand premise

We have three subcases to consider, corresponding to applications of the / R rule, the \ R rule, and the · R rule.

SUBCASE: / R rule (left-hand premise)

Let the left-hand premise, which is of the form $\Gamma \vdash_{GD} \alpha$, result from an application of the / R rule. So α is a formula of the form δ/γ. The list of the right-hand premise, then, contains the formula δ/γ. So, the premise itself has the form $\Delta\ \delta/\gamma\ \Theta \vdash_{GD} \beta$. But then this sequent will have resulted from an application of the / L rule. This implies that it is

immediately dominated by two premises, one of the form $\Xi \vdash_{GD} \gamma$ and the other of the form $\Upsilon\ \delta\ \Lambda \vdash_{GD} \beta$. The application of the cut rule under consideration, then, is instantiated as follows.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Psi\ \gamma \vdash_{GD} \delta}}{\Psi \vdash_{GD} \delta/\gamma}\ /R \qquad \dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \gamma} \qquad \dfrac{\vdots}{\Upsilon\ \delta\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \delta/\gamma\ \ \Xi\ \Lambda \vdash_{GD} \beta}\ /L}{\Upsilon\ \Psi\ \Xi\ \Lambda \vdash_{GD} \beta}\ cut$$

This arrangement can be reconfigured as follows.

$$\dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \gamma} \qquad \dfrac{\dfrac{\vdots}{\Psi\ \gamma \vdash_{GD} \delta} \qquad \dfrac{\vdots}{\Upsilon\ \delta\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \Psi\ \gamma\ \ \Lambda \vdash_{GD} \beta}\ cut}{\Upsilon\ \Psi\ \Xi\ \Lambda \vdash_{GD} \beta}\ cut$$

REMARKS

1. The depth of the second deduction tree could decrease by 1.
2. The second deduction tree has one less node than the first.
3. Although the portion of the second deduction tree shown has two applications of the cut rule, while the portion of the first has only one, each of the applications in the portion of the second deduction tree shown has a lesser degree than does the single application of the cut rule in the portion shown of the first deduction tree. Since $d(\delta/\gamma) = d(\delta) + d(\gamma) + 1$, $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta/\gamma) > d(\Xi\ \Psi\ \delta\ \gamma)$ and $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta/\gamma) > d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta)$.

SUBCASE: $\backslash$ R rule (left-hand premise)

Let the left-hand premise result from an application of the $\backslash$ R rule. (This is left as an exercise.)

SUBCASE: $\cdot$ R rule (left-hand premise)

Let the left-hand premise result from an application of the rule $\cdot$ R. Thus, the left-hand premise has the form $\Xi\ \Psi \vdash_{GD} \delta \cdot \gamma$ and it is immediately dominated by two premises, $\Xi \vdash_{GD} \delta$ and $\Psi \vdash_{GD} \gamma$. At the same time, the right-hand premise has the form $\Upsilon\ \delta \cdot \gamma\ \ \Lambda \vdash_{GD} \beta$. It results from an application of the $\cdot$ L rule and so it is immediately dominated by the sequent $\Upsilon\ \delta\ \gamma\ \ \Lambda \vdash_{GD} \beta$. The application of the cut rule under consideration instantiates to the following.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \delta\ \Psi \vdash_{GD} \gamma}}{\Xi\ \Psi \vdash_{GD} \delta \cdot \gamma}\ \cdot R \qquad \dfrac{\dfrac{\vdots}{\Upsilon\ \delta\ \gamma\ \ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \delta \cdot \gamma\ \ \Lambda \vdash_{GD} \beta}\ \cdot L}{\Upsilon\ \Xi\ \Psi\ \Lambda \vdash_{GD} \beta}\ cut$$

The preceding configuration within any deduction can be rearranged in the following way to yield a fresh deduction:

$$\dfrac{\dfrac{\vdots}{\Psi \vdash_{GD} \gamma} \qquad \dfrac{\dfrac{\vdots}{\Xi \vdash_{GD} \delta} \qquad \dfrac{\vdots}{\Upsilon\ \delta\ \gamma\ \Lambda \vdash_{GD} \beta}\ cut}{\Upsilon\ \Xi\ \gamma\ \Lambda \vdash_{GD} \beta}}{\Upsilon\ \Xi\ \Psi\ \Lambda \vdash_{GD} \beta}\ cut$$

REMARKS

1. The depth of the second deduction tree could decrease by 1.
2. The second deduction tree has one less node.
3. Although the portion of the second deduction tree shown has two applications of the cut rule while the portion of the first has only one, each of the applications in the portion of the second deduction tree shown has a lesser degree than does the single application of the cut rule in the portion shown of the first deduction tree. Since $d(\delta \cdot \gamma) = d(\delta) + d(\gamma) + 1$, $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta \cdot \gamma) > d(\Xi\ \Upsilon\ \Lambda\ \beta\ \delta\ \gamma)$ and $d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \delta \cdot \gamma) > d(\Xi\ \Psi\ \Upsilon\ \Lambda\ \beta\ \gamma)$.

CASE 2.2.2: Right-hand premise
Again, we have three subcases to consider corresponding to the applications of the / R rule, the \ R rule, and the · R rule.

SUBCASE: / R rule (right-hand premise)

Let the right-hand premise result from an application of the / R rule. So β, the final formula of the right-hand premise is of the form δ/γ. Since this sequent results from an application of the / R rule, it must be immediately dominated by a sequent of the form $\Delta\ \alpha\ \Theta\ \gamma \vdash_{GD} \delta$. The deduction whose final sequent results from an application of the cut rule has, then, the following configuration:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{GD} \alpha} \qquad \dfrac{\dfrac{\vdots}{\Delta\ \alpha\ \Theta\ \gamma \vdash_{GD} \delta}\ /R}{\Delta\ \alpha\ \Theta \vdash_{GD} \delta/\gamma}}{\Delta\ \Gamma\ \Theta \vdash_{GD} \delta/\gamma}\ cut$$

This arrangement can be reconfigured as follows.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{GD} \alpha} \qquad \dfrac{\vdots}{\Delta\ \alpha\ \Theta\ \gamma \vdash_{GD} \delta}\ cut}{\dfrac{\Delta\ \Gamma\ \Theta\ \gamma \vdash_{GD} \delta}{\Delta\ \Gamma\ \Theta \vdash_{GD} \delta/\gamma}\ /R}$$

REMARKS

1. The depth of the second deduction tree could increase by 1.
2. Each deduction tree has the same number of nodes.
3. Each deduction tree has exactly the same number of applications of the cut rule.
4. The application of the cut rule in the portion of the second deduction is higher in its deduction tree than the application of the cut rule in the portion of the first deduction tree.
5. Moreover, the degree of the application of the cut rule in the portion of the first deduction tree is greater than the degree of the application of the cut rule in the portion of the second deduction. Since $d(\delta/\gamma) = d(\gamma) + d(\delta) + 1$, $d(\Gamma\ \Delta\ \Theta\ \alpha\ \delta/\gamma) > d(\Gamma\ \Delta\ \Theta\ \alpha\ \delta\ \gamma)$.

SUBCASE: \ R rule (right-hand premise)

Let the right-hand premise result from the rule \ R. (This is left as an exercise.)

SUBCASE: · R rule (right-hand premise)

Let the right-hand premise result from the rule · R. It follows that the final formula of the right-hand premise, β, is of the form $\delta \cdot \gamma$. Since this sequent results from an application of the rule · R, it has the form $\Delta\ \alpha\ \Lambda\ \Upsilon \vdash_{GD} \delta \cdot \gamma$. This sequent is immediately dominated by a pair of sequents, one of the form $\Delta\ \alpha\ \Lambda \vdash_{GD} \delta$ and another of the form $\Upsilon \vdash_{GD} \gamma$. The deduction whose bottom sequent results from an application of the cut rule, then, has the following configuration:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash_{GD} \alpha} \qquad \dfrac{\dfrac{\vdots}{\Delta\ \alpha\ \Lambda \vdash_{GD} \delta} \qquad \dfrac{\vdots}{\Upsilon \vdash_{GD} \gamma}}{\Delta\ \alpha\ \Lambda\ \Upsilon \vdash_{GD} \delta \cdot \gamma}\ {\cdot R}}{\Delta\ \Gamma\ \Lambda\ \Upsilon \vdash_{GD} \delta \cdot \gamma}\ cut$$

This arrangement can be reconfigured as follows.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash_{GD} \alpha} \qquad \dfrac{\vdots}{\Delta\ \alpha\ \Lambda \vdash_{GD} \delta}}{\Delta\ \Gamma\ \Lambda \vdash_{GD} \delta}\ cut \qquad \dfrac{\vdots}{\Upsilon \vdash_{GD} \gamma}}{\Delta\ \Gamma\ \Lambda\ \Upsilon \vdash_{GD} \delta \cdot \gamma}\ {\cdot R}$$

REMARKS

1. The depth of the second deduction tree could increase by 1.
2. Each deduction tree has the same number of nodes.
3. Each deduction tree has exactly the same number of applications of the cut rule.
4. The application of the cut rule in the portion of the second deduction is higher in its deduction tree than the application of the cut rule in the portion of the first deduction tree.

5. Moreover, the degree of the application of the cut rule in the portion of the first deduction tree is greater than the degree of the application of the cut rule in the portion of the second deduction tree. Since $d(\delta \cdot \gamma) = d(\gamma) + d(\delta) + 1$, $d(\Gamma\ \Delta\ \Lambda\ \Upsilon\ \alpha\ \delta \cdot \gamma) > d(\Gamma\ \Delta\ \Lambda\ \Upsilon\ \alpha\ \delta)$.

We have now completed the proof of the cut elimination theorem for the Lambek calculus. We have shown that, for any proof in the Lambek calculus that uses the cut rule, there is a proof of the same sequent without any use of the cut rule. We established this result using mathematical induction by showing that any proof in which the cut rule is used can be modified so that its use is either eliminated or is replaced by one or two uses each of a lesser degree. Since each proof has a finite number of nodes and each modification does not increase the number of nodes, any use of the cut rule can be ultimately eliminated.

Exercises: Cut elimination

1. Complete the proofs omitted in this section.

3 The Lambda Calculus

We now come to the Lambda calculus. Since it is a calculus, it comprises a set of expressions and rules for transforming expressions in the set into other expressions in the same set. The expressions are expressions for functions and the rules transform expressions for functions into expressions for functions. In section 3.1, we shall see how the expressions are formed; in section 3.2, how the expressions are expressions for functions; and in section 3.3, how to transform these expressions into other expressions.

3.1 Notation of the Lambda Calculus

There are various versions of the notation for the Lambda calculus. We shall be concerned just with two, one known as the *single typed,* or *monotyped,* Lambda calculus[7] and the other as the *simply typed* Lambda calculus. Though we shall be mainly concerned with the latter, we begin with the former, simpler version, as a propaedeutic to the latter.

3.1.1 Notation: Monotyped

The two basic categories of symbols in the Lambda calculus are constants, or CN, and variables, or VR. We are already familiar with both these categories of symbols from CQL, where variables were distinguished from individual and relational symbols, which are CQL's counterpart to the constants of the Lambda calculus. And just as VR is disjoint from IS and RS in CQL, so VR is disjoint from CN in the Lambda calculus.

We begin our discussion of the notation of the Lambda calculus with the notation of the monotyped version.

7. Another customary, but potentially misleading, name is the *untyped* Lambda calculus.

Definition 6 Monotyped terms of the Lambda Calculus (syncategorematic version)

(1.1) $\mathrm{CN} \subseteq \mathrm{TM}$;

(1.2) $\mathrm{VR} \subseteq \mathrm{TM}$;

(2.1) If $\sigma \in \mathrm{TM}$ and $\tau \in \mathrm{TM}$, then $(\sigma\,\tau) \in \mathrm{TM}$;

(2.2) If $\upsilon \in \mathrm{VR}$ and $\tau \in \mathrm{TM}$, then $(\lambda\upsilon.\tau) \in \mathrm{TM}$;

(3) Nothing else is in TM.

The two clauses in (1) of definition 6 require that the constants and the variables be terms. These terms are *atomic terms*. The two clauses in (2) enlarge the set of terms to include composite terms. The composite terms are formed in two ways: the first way (2.1) concatenates any two terms and encloses the result with a pair of parentheses, and the second way (2.2) introduces the lambda operator (λ) followed by a variable, a period, and a term and encloses the result with a pair of parentheses. By analogy with the terminology in chapter 11, it is useful to divide a term of the form $(\lambda\upsilon.\tau)$ into its first part $\lambda\upsilon$, which comprises the lambda operator and the variable and which we shall call the (*lambda*) *prefix,* and its second part τ, which comprises the remainder of the term and which we call the (*lambda*) *matrix*.

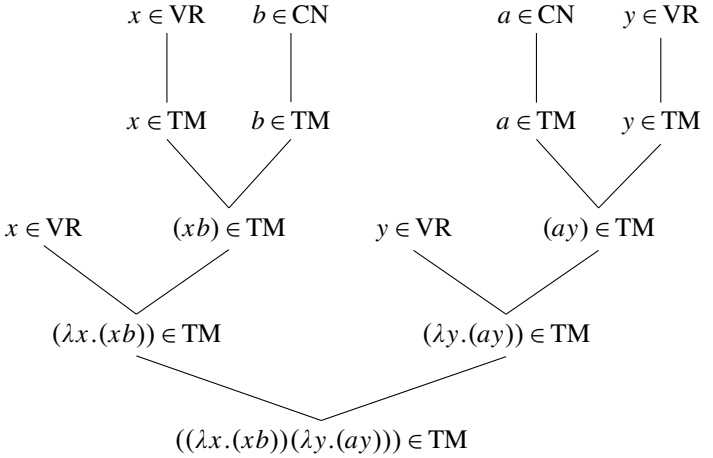
As we shall see in section 3.2, each clause in (2) introduces a way to express an operation: the latter does so by introducing a complex symbol comprising the symbol λ, the Greek letter called lambda, and a variable; the former does so through simple juxtaposition. This practice is familiar from secondary school algebra, where multiplication of two variables or of a parameter and a variable or of a numeral and a variable is expressed by simple juxtaposition. For example, one may write xy for $x \cdot y$, or ay for $a \cdot y$ or $2y$ for $2 \cdot y$.

We shall adopt here conventions for constants and variables of the Lambda calculus analogous to those we adopted in chapter 11 for CQL. We shall use for constants lowercase letters from the beginning of the Roman alphabet, such as *a, b,* and *c;* and if a large number of constants is required, we shall use the letter *c,* subscripted with positive integers. Similarly, we shall use for variables lowercase letters from the end of the Roman alphabet, such as *w, x, y,* and *z;* and again, if a large number of variables are required, we shall use the letter υ, subscripted with positive integers. In addition, for terms, atomic or composite, we shall use the letters *t, s,* and *p*. Should a larger number of terms be needed, we shall subscript *t* with positive integers.

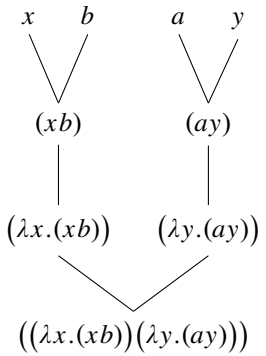
Before turning to the abbreviatory conventions governing parentheses to be adopted hereafter, let us pause to see how we display in a tree diagram, of the kind we first met in chapter 1, the reasoning whereby we establish whether an expression is indeed a term in the monotyped Lambda calculus.

Suppose that *a* and *b* are constants and that *x* and *y* are variables. By clause (1.1) *a* and *b* are terms and by clause (1.2) *x* and *y* are also terms. Since *a* and *y* are both terms, by clause (2.1), (ay) is a term; similarly, since *b* and *x* are terms, (xb) is a term, also by clause (2.1). Since *x* and *y* are variables and (ay) and (xb) are terms, then by clause (2.2) both

$\big(\lambda x.(xb)\big)$ and $\big(\lambda y.(ay)\big)$ are terms. And since these last two expressions are terms, then by clause (2.1) $\big(\big(\lambda x.(xb)\big)\big(\lambda y.(ay)\big)\big)$ is a term. This reasoning is displayed in the following diagram.



Just as we simplified such diagrams in the case of CPL, CPDL, and CQL, we simplify the term formation diagrams, first by restricting our attention to terms, thereby omitting the nodes labeled with CN and VR, and second, as all the remaining expressions are terms, we omit the indication that they are members of TM. The result is that the preceding diagram simplifies to the following.



Now we turn to abbreviatory conventions governing parentheses. There are three. The first, already familiar to us from the conventions governing the notations for CPL, CPDL, and CQL, is to omit the outer most parentheses. Thus, for example, the formula in CPL, $\big((p \wedge q) \to r\big)$, is abbreviated as $(p \wedge q) \to r$. Similarly, the terms $(\sigma\,\tau)$ and $(\lambda\upsilon.\tau)$ are

abbreviated as $\sigma\,\tau$ and $\lambda v.\tau$, respectively. This convention applies only to a term, not to any subterm of a term.

We now turn to the second convention. Without any specific guidance, the arithmetic expression $4+2\cdot 3$ is ambiguous: either it could denote 10, which results from carrying out the operation of multiplication before carrying out the operation of addition, or it could denote 18, which results from carrying out the operation of addition before carrying out the operation of multiplication. The use of parentheses makes unambiguous which operation is performed first. Thus, the expression $4+(2\cdot 3)$ unambiguously denotes 10, while the expression $(4+2)\cdot 3$ unambiguously denotes 18. A convention one can adopt is to give the multiplication sign priority over the addition sign. Under this convention, the expression without parentheses, $4+2\cdot 3$, unambiguously denotes 10. To denote 18, one then must resort to parentheses and write the expression $(4+2)\cdot 3$.

The second convention is of this kind. The two clauses in (2) of the definition of the set of terms (definition 6) introduce notation for two operations, the application operation and the abstraction operation. The convention is that the application operation has priority over the abstraction operation. More exactly, if the term to the right of a lambda prefix itself results from clause (2.1), then the parentheses introduced by the rule are omitted. Thus, $\lambda v.\sigma\,\tau$ abbreviates, by the second convention, $\lambda v.(\sigma\,\tau)$, which itself abbreviates, by the first convention, $\big(\lambda v.(\sigma\,\tau)\big)$. $\lambda v.\sigma\,\tau$ is to be distinguished from $(\lambda v.\sigma)\tau$ in the same way as $4+2\cdot 3$ is from $(4+2)\cdot 3$.

The third convention pertains to expressions of application. We shall assume that application associates to the left. Thus, the term $(\rho\sigma)\tau$, which, by the first convention, abbreviates $\big((\rho\sigma)\tau\big)$, is abbreviated further by the second convention as $\rho\sigma\,\tau$. This convention does not compromise the distinction between the terms $(\rho\sigma)\tau$ and $\rho(\sigma\,\tau)$. Only the first is subject to this convention. The third convention also applies to subterms of a term.

To shore up our understanding of these conventions, let us consider the term $\big(\lambda x.\big((yx)\big(\lambda x.(zx)\big)\big)\big)$. By the first convention, one eliminates the outer parentheses, obtaining $\lambda x.\big((yx)\big(\lambda x.(zx)\big)\big)$. By the second convention, one eliminates the outer parentheses enclosing the subterm $\big((yx)\big(\lambda x.(zx)\big)\big)$, obtaining $\lambda x.(yx)\big(\lambda x.(zx)\big)$. The same convention permits the elimination of the parentheses surrounding the subterm (zx), yielding $\lambda x.(yx)(\lambda x.zx)$. Finally, by the third convention, one eliminates the parentheses surrounding the subterm (yx), yielding $\lambda x.yx(\lambda x.zx)$.

Just as in CQL, the scope of an occurrence of a quantifier and the scope of the prefix in which it occurs is the same, so in the Lambda calculus, the scope of an occurrence of the lambda operator symbol and the scope of the prefix in which it occurs is the same. Moreover, the definition of the scope of an occurrence of a lambda prefix is completely analogous to the definition of the scope of an occurrence of a quantifier prefix.

Definition 7 Scope of a lambda prefix
The scope of an occurrence of a lambda prefix in a term τ is the subterm of τ, which contains the lambda prefix's occurrence but whose immediate subterms do not.

We now define what it is for an occurrence of a lambda prefix to bind a particular occurrence of a variable. For example, the scope of the first lambda operator in the term $(\lambda x.((yx)(\lambda x.(zx))))$ is the entire formula, whereas that of the second lambda operator is $(\lambda x.(zx))$.

Definition 8 Binding
Let τ be in TM. Let λv be a lambda prefix in τ and let u be a variable in τ. (An occurrence of) the lambda prefix λv binds (an occurrence of) the variable u iff

(1) $v = u$ (i.e., v and u are the same variable);
(2) (The occurrence of) u is within the scope of (the occurrence of) the lambda prefix λv; and
(3) No other occurrence of the lambda prefix λv has the relevant occcurrence of u within its scope and is itself within the scope of the relevant occurrence of the lambda prefix λv.

Evidently, then, the lambda prefix λx binds both occurrences of the variable x in the term $\lambda x.x$, which, by the first convention, abbreviates $(\lambda x.x)$. The lambda prefix λy binds both occurrences of the variable y in the term $\lambda y.xy$, which abbreviates $(\lambda y.(yx))$. But, the lambda prefix λy does not bind both occurrences of the variable y in the term $(\lambda y.x)y$, a conventional abbreviation of $((\lambda y.x)y)$. Finally, let us consider again the term $\lambda x.yx(\lambda x.zx)$, an abbreviation of $(\lambda x.((yx)(\lambda x.(zx))))$, as we explained earlier. In this term, the second occurrence of the lambda prefix λx binds the last two occurrences of the variable x in the term, while the first occurrence of the lambda prefix binds only the first two occurrences of the variable x.

We now distinguish, as we did in CQL, between variables bound by a lambda prefix and free variables, that is, variables unbound by any lambda prefix.

Definition 9 Bound variable
(An occurrence of) a variable v is bound in a term τ iff (an occurrence of) a lambda prefix in the term τ binds it.

Definition 10 Free variable
(An occurrence of) a variable v is free in a term τ iff no (occurrence of a) lambda prefix in the term τ binds it.

Thus, in the term $\lambda x.yx(\lambda x.zx)$, each occurrence of the variable x is bound and neither the occurrence of y nor the occurrence of z is bound. No single occurrence of a variable may be both bound and free, though the variable itself may be. For example, the variable y is both free and bound in the term in $\lambda x.xy(\lambda y.yz)$, while each of its occurrences is one or the other but not both.

The foregoing distinction between variables that are bound and variables that are free is used to distinguish two kinds of terms: those that are closed and those that are open. An open term is a term with at least one free variable occurring in it.

Definition 11 Open term
A term is open iff it has at least one (occurrence of a) free variable.

A closed term is one in which no free variables occur.

Definition 12 Closed term
A term is closed iff it has no (occurrences of a) free variable.

It is convenient to define a function that applies to a term and retrieves any variable with at least one free occurrence within the term.

Definition 13 Free variables in a term

(1) $\mathrm{Fvr}(\upsilon) = \{\upsilon\}$, for each $\upsilon \in \mathrm{VR}$;
(2) $\mathrm{Fvr}(c) = \emptyset$, for each $c \in \mathrm{CN}$;
(3) $\mathrm{Fvr}(\sigma\,\tau) = \mathrm{Fvr}(\sigma) \cup \mathrm{Fvr}(\tau)$;
(4) $\mathrm{Fvr}(\lambda\upsilon.\tau) = \mathrm{Fvr}(\tau) - \{\upsilon\}$.

Open and closed terms could also be defined in terms of the function that determines, for any terms, what its free variables are. A term τ is closed just in case $\mathrm{Fvr}(\tau)$ is empty and it is open otherwise.

In chapter 11, section 1, we were introduced to the syntactic operation of substitution. It was defined as an operation that mapped terms to terms and formulae to formulae by replacing each occurrence of a free variable with an occurrence of an individual symbol. Here it will be defined as an operation that maps terms to terms by replacing each occurrence of a free variable with an occurrence of a term. The term that substitutes for the free variable may be any term, a constant or a variable, or indeed a composite term.

Definition 14 Substitution of a term for a free variable
Let ρ, σ, and τ be terms and let u, υ, and w be variables.

(1.1) $[\tau/\upsilon]\sigma = \sigma$, if $\sigma \in \mathrm{CN}$;
(1.2.1) $[\tau/\upsilon]\sigma = \sigma$, if $\sigma \in \mathrm{VR}$ and $\sigma \neq \upsilon$;
(1.2.2) $[\tau/\upsilon]\sigma = \tau$, if $\sigma \in \mathrm{VR}$ and $\sigma = \upsilon$;

(2.1) $[\tau/\upsilon](\rho\sigma) = [\tau/\upsilon]\rho[\tau/\upsilon]\sigma$;
(2.2.1) $[\tau/\upsilon](\lambda w.\rho) = \lambda w.\rho$, if $w = \upsilon$;
(2.2.2) $[\tau/\upsilon](\lambda w.\rho) = \lambda w.[\tau/\upsilon]\rho$, if $w \neq \upsilon$.

The substitution operation substitutes a term for a free variable in a term. Should the term in which the substitution is to take place have no free variables, then the term to which the substitution operation applies is unaltered. Thus, as clause (1.1) makes clear, if the term in which the substitution is to take place is a constant, then the substitution operation leaves the constant unaltered. Should the term contain a free occurrence of a variable, but the

variable is not the same as the substituens, then again the term to which the substitution operation applies is left unaltered. As clause (1.2.1) makes clear, a term that is just a variable, and therefore free, is unaltered by a substitution operation if the substituens is a distinct variable. Should the term in which the substituion is to take place be just a variable and should it be identical with the substituens, then the substitution takes place. Clause (1.2.2) states this.

Next, if a composite term comprises two terms, the substitution operation applied to the composite term is just the result of the operation applied to its two immediate subterms. If the composite term comprises a lambda prefix and a term, whether the substitution operation applied to the composite term alters the term depends on whether the term in question has at least one occurrence of the substituens and its occurrence is free. And, it is free just in case the substituens is distinct from the variable in the lambda prefix.

We illustrate these various cases next. The expression above indicates the substitution to take place and the expression into which the substitution takes place, while the expression below is the result of the substitution.

CLAUSE			
(1.1)	$[a/x]b$	$[y/x]b$	$[\lambda z.az/x]b$
	b	b	b
(1.2.1)	$[a/x]y$	$[z/x]y$	$[\lambda z.az/x]y$
	y	y	y
(1.2.2)	$[a/x]x$	$[z/x]y$	$[\lambda z.az/x]x$
	a	y	$\lambda z.az$
(2.1)	$[a/x](xy)$	$[z/x](xy)$	$[\lambda z.az/x](xy)$
	ay	zy	$(\lambda z.az)y$
(2.2.1)	$[a/x](\lambda z.xz)$	$[y/x](\lambda z.xz)$	$[\lambda w.yw/x](\lambda z.xz)$
	$\lambda z.az$	$\lambda z.yz$	$\lambda z.(\lambda w.yw)z$
(2.2.2)	$[a/x](\lambda x.xz)$	$[y/x](\lambda x.xz)$	$[\lambda w.yw/x](\lambda x.xz)$
	$\lambda x.xz$	$\lambda x.xz$	$\lambda x.xz$

Before turning to exploring some of the details of substitution, we remind readers of the two neologisms introduced in chapter 11 in connection with substitution: namely, the word *substituens,* or τ in the notation of $[\tau/\upsilon]$, the term replacing the various occurrences of a free variable, υ, and the word *substituendum*, or υ in the notation of $[\tau/\upsilon]$, the free variable whose various free occurrences τ replaces.

When substitution was defined for expressions of CQL, a substituendum is a variable and the substituens is an individual symbol. However, whereas in CQL only individual symbols served as substituens, in the Lambda calculus any term, atomic or composite,

closed or open, may serve as a substituens. Since substitution in CQL replaces a free variable with only an individual symbol, such a replacement never disturbs the binding relations between an occurrence of a quantifier prefix and the occurrences of the variable it binds. Because substitution in a term of the Lambda calculus permits open terms to substitute for the free occurrences of a variable, the relationship between the occurrence of a quantifier prefix and the free occurrences of a variable may be disturbed.

Let us see a few examples of how this relationship can be disturbed. Consider the substitution of the variable x for the free variable y in the term $\lambda x.xy$. The result is $\lambda x.xx$: that is, $[x/y]\lambda x.xy = \lambda x.xx$. The term into which the substitution is to take place has one lambda prefix λx, one bound occurrence of x, and one free occurrence of y. The term that results has one lambda prefix λx, two bound occurrences of x, and no free occurrence of any variable. This substitution contrasts with the substitution of the variable z for the free variable y in the term $\lambda x.xy$. The result is $\lambda x.xz$: that is, $[z/y]\lambda x.xy = \lambda x.xz$. The one lambda prefix λx binds the same variables in the same positions as it did before the substitution, and the position of the only free occurrence of a variable in $\lambda x.xy$ is the position of the only free occurrence of a variable in $\lambda x.xz$, though the variable has changed. Here is another example of a disturbance of the binding relationship. The substituens y in the term $\lambda z.yz$ is free before substitution and occurs as bound after substitution for v in the term $\lambda y.vy$, for $[\lambda z.yz/v]\lambda y.vr = \lambda y.(\lambda z.yz)r$.

Such a disturbance does not arise if the substituens is a closed term. If it is an open term, the substituendum position is not within the scope of a lambda prefix whose variable is among the free variables in the substituens. For example, no harm arises in substituting the closed term b for y in $\lambda x.xy$. Moreover, no harm arises in substituting the closed term $\lambda z.zc$ for x in $\lambda y.xy$. Nor does any harm arise in substituting an open term, say $\lambda z.zv$ for x in $\lambda x.xy$, since the free variable v in $\lambda z.zv$ is distinct from the variable x, which occurs in the lambda prefix of the term containing the substituendum.

We now define a condition which, if observed, ensures that when a substitution takes place, the relationship between the occurrences of lambda prefixes and the variable occurrences they bind will not be disturbed.

Definition 15 Free for a term to substitute for a variable
Let σ, τ be terms and let u and v be variables. Then, σ is free to substitute for v in τ iff one of the following holds:

(1.1) $\tau \in \mathrm{CN}$;
(1.2) $\tau \in \mathrm{VR}$;
(2.1) τ has the form $(\pi\rho)$ and σ is free to substitute for v in both π and ρ;
(2.2) τ has the form $\lambda u.\pi$ and σ is free to substitute for v in π and if $v \in \mathrm{Fvr}(\pi)$, then $u \notin \mathrm{Fvr}(\sigma)$.

Thus, x is not free to substitute for y in $\lambda x.xy$, though a, a constant, clearly is. In fact, any closed term is free to substitute for y in $\lambda x.xy$. Moreover, any variable distinct from x, say z, is free to substitute for y in $\lambda x.xy$, as is any term, provided the substituens does not have x among its free variables.

Exercises: Monotyped Lambda calculus

1. For each sequence of symbols, determine whether it is a term of the Lambda calculus, a term by convention only or not a term at all. If it is a term by convention, state the relevant convention. If it is not a term, explain why it is not.

(a) ab
(b) (xy)
(c) $(z.\lambda z)$
(d) $\lambda xy.xyz$
(e) $\big(\lambda x.\big(\lambda y.(\lambda z.a)\big)\big)$
(f) $abcd$
(g) $\lambda z.(\lambda x.x)$
(h) $\lambda z.y\lambda x.x$
(i) $\big(\lambda a.(ay)\big)$
(j) $\big(\lambda z.\big((\lambda x.x)\big(\lambda y.\big((xy)z\big)\big)\big)\big)$
(k) $\big(z(\lambda z.x)\big)$
(l) $\lambda z.(\lambda x.x)(\lambda y.xyz)$
(m) $\lambda x.\lambda x.x$
(n) $\lambda x.(ax)$
(o) $\lambda x.x(\lambda y.y)$
(p) $\lambda x.x(\lambda y.yxx)x$

2. For each variable occurrence in each term, state which lambda prefix occurrence binds which variable occurrence and state which variable occurrences are free.

(a) $y\lambda x.z$
(b) $\lambda x.\lambda y.xyz$
(c) $\big(\lambda y.xz(\lambda z.ya)\big)\big(\lambda x.wy(\lambda z.z)\big)$
(d) $\lambda x.(\lambda x.y)zx$
(e) $\lambda x.(\lambda y.yz)(xy)$
(f) $\big(z(\lambda z.x)\big)$

3. In each case, determine which term results from the application of the substitution function.

(a) $[x/y]y$
(b) $[\lambda z.wz/v]\lambda y.vx$
(c) $[\lambda x.yx/x]\lambda z.zx$
(d) $[\lambda x.xy/y]y$
(e) $[\lambda x.xy/x]\lambda z.(\lambda x.x)(\lambda y.xyz)$
(f) $[\lambda y.ayx/z]\big(z(\lambda z.x)\big)$

4. State which substituenda, listed in the first column, are free to substitute for the substituens, specified in the second column, in the term, specified in the same row as the substituens but in the third column.

SUBSTITUENDA	SUBSTITUENS	TERM
a	z	az
x	x	az
y	y	$\lambda y.ay$
z	z	$\lambda x.az$
$\lambda x.bx$	z	$\lambda y.ayz$
$\lambda y.ay$	x	$\lambda x.\lambda y.yx$
$\lambda z.xy$	w	$\lambda x.yw(\lambda y.xy)$
$\lambda x.\lambda y.xy$	y	$\lambda x.yw(\lambda y.xy)$
$\lambda w.wy$		
$\lambda x.xy$		
$\lambda x.xw(\lambda y.by)$		

5. Using the definition for Fvr (definition 13) as a guide, provide an explicit definition of the set of variables occurring in a term, using the notation Ovr. Also write an explicit definition for a function Bvr that identifies variables with at least one bound occurrence in a term.

3.1.2 Notation: Simply typed

We now turn to the simply typed Lambda calculus. In the monotyped Lambda calculus, there is but one type, TM; in the simply typed Lambda calculus, there is a (countably) infinite number of types. Our first task is to define this infinite set of types. To define this set of types, we shall avail ourselves of an index set. Readers may recall that we were able to partition the set of relational symbols, introduced in chapter 9, into a countably infinite number of subsets, where the one-place relational symbols are in one set, the two-place relational symbols in a second set, and in general, the n-place relational symbols are in an nth set. Indeed, what we did was to index the family of sets using the positive integers: an n-place relational symbol is a member of the set RS_n. The set of relational symbols, RS, then is just $\bigcup_{i \in \mathbb{Z}^+} \mathrm{RS}_i$. In a similar fashion, we can define the set of all nonlogical symbols for CPDL as follows: let NS_0 be IS and, for each $i \in \mathbb{Z}^+$, let NS_i be RS_i. NS, the set of nonlogical symbols, is then defined as $\bigcup_{n \in \mathbb{N}} \mathrm{NS}_n$. In this way, the natural numbers are used as indices whereby a countably infinite set of categories, or types, is defined and into which categories, or types, the nonlogical constants (the individual symbols and the relational symbols) of CQL, can be put.

In the same way, we shall define a countably infinite set of indices and use them to define a countably infinite set of categories, or types, by which atomic terms of the simply typed Lambda calculus can be categorized, or typed. Not surprisingly, the set of indices, called types, is defined recursively. The set of types, named Typ, includes two atomic types, e and t, as well as the composite types obtained from them, stated in the following definition.

Definition 16 Set of simple types
Let e and t be distinct entities.

(1) $e, t \in \text{Typ}$;
(2) If $x, y \in \text{Typ}$, then so is (x/y);
(3) Nothing else is in Typ.

As with the notation of CQL, we assume that the set of constants and the set of variables are disjoint from one another. In addition, as we did with the relational symbols of CPDL, we shall segregate both the constants and the variables into disjoint subsets, labeling distinct subsets with distinct types, just as we labeled the disjoint subsets of the set of relational symbols of CPDL with distinct positive integers. Thus, if x and y are distinct types, then CN_x, the *constants of type x,* and CN_y, the *constants of type y,* are disjoint from one another, as are the *variables of type x,* or VR_x, and the *variables of type y,* or VR_y. The set of all constants, or CN, then is just $\bigcup_{x \in \text{Typ}} \text{CN}_x$ and the set of all variables, or VR, is $\bigcup_{x \in \text{Typ}} \text{VR}_x$. We shall refer to the members of either VR or CN as *atomic terms*.

We now wish to extend the atomic terms to include *composite terms,* just as we extended the atomic formulae, or propositional variables of CPL, to include composite formulae. We shall do this by extending the atomic terms of each type.

Definition 17 Formation of simply typed terms (syncategorematic version)
Let x and y be in Typ.

(1) ATOMIC TERMS

(1.1) $\text{CN}_x \subseteq \text{TM}_x$;

(1.2) $\text{VR}_x \subseteq \text{TM}_x$;

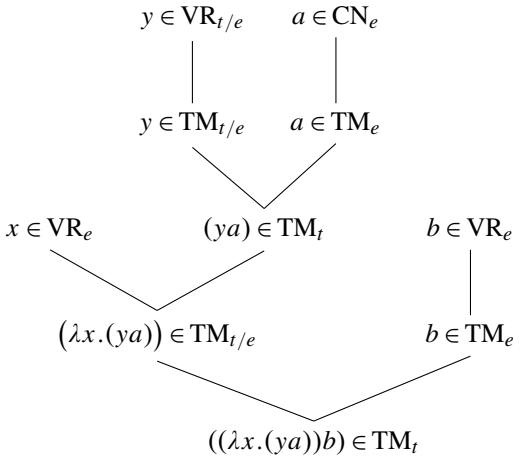
(2) COMPOSITE TERMS

(2.1) If $\sigma \in \text{TM}_{x/y}$ and $\tau \in \text{TM}_y$, then $(\sigma\,\tau) \in \text{TM}_x$;

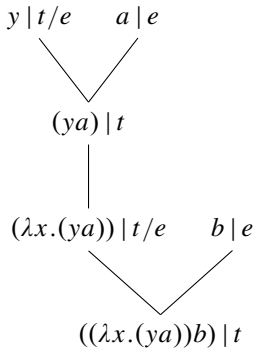
(2.2) If $v \in \text{VR}_y$ and $\tau \in \text{TM}_x$, then $(\lambda v.\tau) \in \text{TM}_{x/y}$.

Just as we defined CN to be $\bigcup_{x \in \text{Typ}} \text{CN}_x$ and VR to be $\bigcup_{x \in \text{Typ}} \text{VR}_x$, we define TM to be $\bigcup_{x \in \text{Typ}} \text{TM}_x$.

Let us see how these formation rules work. Suppose that $a,\ b \in \text{CN}_e,\ x \in \text{VR}_e$, and $y \in \text{VR}_{t/e}$. By clause (1.1) $a, b \in \text{TM}_e$ and by clause (1.2) $y \in \text{TM}_{t/e}$. Since $y \in \text{TM}_{t/e}$ and $a \in \text{TM}_e$, it follows by clause (2.1) that $ya \in \text{TM}_t$. Since $x \in \text{VR}_e$ and $ya \in \text{TM}_t$, clause (2.2) shows us that $\lambda x.ya \in \text{TM}_{t/e}$. Finally, clause (2.1) establishes that $(\lambda x.ya)b \in \text{TM}_t$, since $\lambda x.ya \in \text{TM}_{t/e}$ and $b \in \text{TM}_e$. As we have come to expect, such reasoning can be recapitulated in a tree diagram:



Here too we can simplify the diagram, though not as much as we were able to with the diagrams for the monotyped terms. The reason is that whether two terms can form a term depends on which category each belongs to. For example, there is no term formed from two terms, one taken from $\mathrm{TM}_{t/e}$ and one taken from $\mathrm{TM}_{e/t}$. Indeed, no two terms taken from the very same TM_x, regardless of what x is, can be put together to form a term. Moreover, even if one term is taken from $\mathrm{TM}_{t/e}$ and another from TM_e, they form a term only if the second one is put after the first one. It follows, then, that in simplifying the diagrams, we cannot omit information about the type of the category. We can, however, confine our attention to terms and we can compress the labels so that labels mention only the term and its type.



Attentive reader will have noticed that the assignment of types in the term tree conform to the rules of / Elimination and / Introduction of the Lambek calculus. We shall discuss this in detail in section 4.

We adopt the same abbreviatory conventions for the simply typed Lambda calculus as we did for the monotyped Lambda calculus. The various notions of the monotyped

Lambda calculus—namely, of scope, of binding, of bound and free variables, of open and closed terms, of substitution, and of being free to substitute for a variable in a term—all apply mutatis mutandis to the simply typed Lambda calculus. Two definitions, nonetheless, deserve comment. Binding is a relation between the occurrence of a lambda prefix in a term and the occurrence of a free variable in the same term. In a Lambda Calculus with more than one type, the variable in the lambda prefix must be of the same type as the variable occurrence that the lambda prefix binds. However, it turns out that the definition as stated for the monotyped Lambda calculus applies here without any change, for the first clause of its definiens, which requires the variable occurrence in the lambda prefix occurrence and the variable occurrence that it binds be occurrences of the same variable, and so, ipso facto, the variable occurrences of variables of the same type.

In contrast, the definition of substitution requires the explicit mention of types. If substitution is to yield a term, then the substituens must have the same type as the free variable for which it substitutes. Here is the revised definition of substitution containing the emendation.

Definition 18 Substitution of a term for a free variable (simply typed)
Let ρ, σ and τ be terms and let u, v, and w be variables. Let term τ and variable v be of the same type.

(1.1) $[\tau/v]\sigma = \sigma$, if $\sigma \in \mathrm{CN}$;

(1.2) $[\tau/v]\sigma = \sigma$, if $\sigma \in \mathrm{VR}$ and $\sigma \neq v$;

(1.3) $[\tau/v]\sigma = \tau$, if $\sigma \in \mathrm{VR}$ and $\sigma = v$;

(2.1) $[\tau/v](\rho\sigma) = [\tau/v]\rho[\tau/v]\sigma$;

(2.2.1) $[\tau/v](\lambda w.\rho) = \lambda w.\rho$, if $w = v$;

(2.2.2) $[\tau/v](\lambda w.\rho) = \lambda w.[\tau/v]\rho$, if $w \neq v$.

Since the definition of being free to substitute for a variable explicitly uses the notation for substitution, its definition for the simply typed Lambda calculus also requires the explicit mention of types.

Definition 19 Free for a term to substitute for a variable
Let σ, τ be terms and let u and v be variables. Let term σ and variable v be of the same type. Then, σ is free to substitute for v in τ iff one of the following holds:

(1.1) $\tau \in \mathrm{CN}$;

(1.2) $\tau \in \mathrm{VR}$;

(2.1) τ has the form $(\pi\rho)$ and σ is free to substitute for v in both π and ρ;

(2.2) τ has the form $\lambda u.\pi$ and σ is free to substitute for v in π and if $v \in \mathrm{Fvr}(\pi)$, then $u \notin \mathrm{Fvr}(\sigma)$.

Exercises: Simply typed Lambda calculus

1. Let a, b, c, and d be terms and let x, y, and z be variables, where c and x are of type e, d, and y are of type t/e, b and z are of type $(t/e)/(t/e)$, and a of type $t/(t/e)$. Determine which of the following are well-formed terms, taking into account the abbreviations introduced in this section. If a term is well formed, restore its unabbreviated form, and for each well-formed term, determine its type.

(a)	cd	(e)	$\lambda x.dx$	(i)	$a(zd)$	(m)	$(\lambda x.xy)a$
(b)	dc	(f)	zyc	(j)	$\lambda y.ad$	(n)	$(\lambda z.dx)z$
(c)	zy	(g)	$z(yc)$	(k)	$(\lambda x.yc)c$	(o)	$\lambda x.adx$
(d)	da	(h)	ay	(l)	$a(zd)$		

2. Using the same type assignment as used in the previous example, determine the type of p and the type of q as they occur within the following terms. Assume that each term is of type t. In the event that either p or q admits of more than one type, indicate what they are.

(a)	pdx	(c)	pad	(e)	$p(qd)$
(b)	$p(dx)$	(d)	$q(byp)$	(f)	$q(pbd)$

3.2 Semantics: Functional Structures

In this section, we shall show how the terms of the simply typed Lambda calculus denote functions. We do this by defining a structure for the constants of the simply typed Lambda calculus. However, before presenting the definition, we shall illustrate the formulation of the definition by restating the definition for a structure for a CPDL signature.

Recall that the point of a structure is to assign values to each of the nonlogical symbols used in the notation of CPDL. To do this, one requires a way to characterize the nonlogical symbols. To this end, we defined a CPDL signature, which is a trio, comprising the set of individual symbols, the set of relational symbols, and the adicity function that assigns to each relational symbol its adicity. Just now, we saw an alternative characterization of the set of nonlogical symbols for CPDL. Using this new characterization of the notation of CPDL, let us recharacterize what a structure for CPDL is. Recall that a structure for CPDL is the ordered pair $\langle U, i\rangle$, where i is the interpretation function and U is a nonempty set. i's domain is the set NS, which was defined to be $\mathrm{IS} \cup \mathrm{RS}$, and its codomain comprises various sets built from U. However, not any function from the domain of symbols to the codomain is an interpretation function. Certain constraints must be respected. In particular, each member of IS, that is, of NS_0, is assigned a member of U and each member of RS_n, that is, of NS_n, is assigned a set of n-tuples drawn from U.

By exhaustively segregating the members of the codomain of an interpretation function into a family of sets, one can index the members of the family in such a way that the constraint on the function becomes easy to state. Here is how: let D_0 be U, let D_1 be

Pow(U), and in general, for each $n > 1$, let D_n be Pow(U^n). Finally, let D be $\bigcup_{i \in \mathbb{N}} D_i$. The codomain of an interpretation function becomes simply D.

We then define a structure for CPDL as follows.

Let NS, or $\bigcup_{n \in \mathbb{N}} \mathrm{NS}_n$, be the set of nonlogical symbols for CPDL.
$\langle U, i \rangle$ is a structure for NS iff

(1) U is a nonempty set;
(2) i is a function from NS into D such that, each $n \in \mathbb{N}$, if $\sigma \in \mathrm{RS}_n$, then $i(\sigma) \in D_n$.

As with CPDL, we start with a nonempty set, called U. We let U be D_e and we let D_t be the set with just the truth values T and F. We use the other types to extend our names for various subsets of the codomain of the interpretation function as follows: $D_{x/y} = D_x^{D_y}$, that is, Fnc(D_y, D_x), or the set of all functions from D_y into D_x. We let $D = \bigcup_{x \in \mathrm{Typ}} D_x$. An interpretation function for the simply typed Lambda calculus, then, is any function from CN into D respecting the types.

Definition 20 Structure for CN for the simply typed Lambda calculus
Let CN be the set of constants of the simply typed Lambda Calculus.
$\langle U, i \rangle$ is a structure for CN iff

(1) U is a nonempty set;
(2) i is a function from CN into D such that, for each $x \in \mathrm{Typ}$, if $\tau \in \mathrm{CN}_x$ then $i(\tau) \in D_x$.

To assign a value to terms that have free variables, one requires a variable assignment. In the simply typed Lambda calculus, variables have types and they must be assigned values from the codomain of the corresponding type.

Definition 21 Variable assignment for the simply typed Lambda calculus
A variable assignment is any function g from VR into U such that if $\upsilon \in \mathrm{VR}_x$ then $g(\upsilon) \in D_x$.

Equipped with a structure and a variable assignment, we can now proceed to assign values to composite terms, based on the values assigned to the atomic terms, in exactly a way analogous to the way used for CQL.

Definition 22 Extension for the simply typed Lambda calculus
Let M, or $\langle U, i \rangle$, be a structure for CN. Let g be a variable assignment.

(1.1) If $\tau \in \mathrm{CN}$, then $[\tau]_g^M = i(\tau)$;
(1.2) If $\tau \in \mathrm{VR}$, then $[\tau]_g^M = g(\tau)$;
(2.1) If $\sigma \in \mathrm{TM}_{x/y}$ and $\tau \in \mathrm{TM}_y$, then $[\sigma\tau]_g^M = [\sigma]_g^M([\tau]_g^M)$;
(2.2) If $\upsilon \in \mathrm{VR}_y$ and $\tau \in \mathrm{TM}_x$, then $[\lambda\upsilon.\tau]_g^M = f$, where f is a function from D_y into D_x such that $f(a) = [\tau]_{g_{\upsilon \mapsto a}}^M$, for each a in D_y.

We note that the expression in (2.1) expresses the application of a function to a value and its interpretation is the application of the function to the value: $[\sigma]^M_g$ is the function, $[\tau]^M_g$ is a value to which the function applies, and $[\sigma\,\tau]^M_g$ is the result of application of the function to the value. The expression in (2.2) expresses the abstraction of a function from the function denoted by its matrix: the function is the one that arises by permitting the values assigned to the variable υ in $[\tau]^M_{g_{\upsilon\mapsto a}}$, assigning any other variables free in τ the values assigned to them by the variable assignment g.

By the definition of a structure for CN (definition 20) and by the definition of a variable assignment (definition 21), we know that each atomic term is assigned a value suited to its type. But what about composite terms? Could it be that $\tau \in \mathrm{TM}_x$ but $[\tau]^M_g \notin D_x$; that is, could it be, for example, that $a \in \mathrm{TM}_{(t/e)/e}$ but $[a]^M_g \notin D_{(t/e)/e}$? The answer is that such a situation does not arise. And it is established by a proof using the principle of mathematical induction.

Fact 1 Type soundness
For each $\tau \in \mathrm{TM}_x$, for each structure M and for each variable assignment g, $[\tau]^M_g \in D_x$.

PROOF
Let τ be a term. Let M, or $\langle U,i\rangle$, be a structure and g a variable assignment.

BASE CASE

Suppose that τ is an atomic term. Then, either it is in CN_x, for some $x \in \mathrm{Typ}$, or it is in VR_x, for some $x \in \mathrm{Typ}$. Suppose, on the one hand, that $\tau \in \mathrm{CN}_x$. Then, by clause (2) of the definition of a structure (definition 20), $i(\tau) \in D_x$. And by clause (1.1) of the definition of an extension (definition 22), $[\tau]^M_g \in D_x$. Suppose, on the other hand, that $\tau \in \mathrm{VR}_x$. Then, by the definition of a variable assignment (definition 21), $g(\tau) \in D_x$. And by clause (1.2) of the definition of an extension (definition 22), $[\tau]^M_g \in D_x$. Thus, whether the atomic term τ is a constant or a variable, $[\tau]^M_g \in D_x$.

INDUCTION CASE

Suppose that τ, a member of TM_x, is a composite term. Then, it arises either from clause (2.1) or from clause (2.2) of the definition of the formation of terms (definition 17). In the former case, there is $y \in \mathrm{Typ}$ such that $\sigma \in \mathrm{TM}_{x/y}$, $\pi \in \mathrm{TM}_y$, and $\sigma\,\pi \in \mathrm{TM}_x$; in the latter case, there are $y,z \in \mathrm{Typ}$ such that $x = z/y$, $\upsilon \in \mathrm{VR}_y$, $\rho \in \mathrm{TM}_z$ and $\lambda\upsilon.\rho \in \mathrm{TM}_{z/y}$.

INDUCTION HYPOTHESIS
Let $[\sigma]^M_g$ be in $D_{x/y}$, $[\pi]^M_g$ in D_y, $[\upsilon]^M_g$ in D_y, and $[\rho]^M_g$ in D_x.

CASE 1

Suppose that τ, a member of TM_x, has the form $\sigma\,\pi$. Then, by the induction hypothesis, $[\sigma]^M_g \in D_{x/y}$ and $[\pi]^M_g \in D_y$. Now $D_{x/y} = D_x^{D_y}$, which just is $\mathrm{Fnc}(D_y, D_x)$, or the set of all functions from D_y into D_x. Thus, $[\sigma]^M_g([\pi]^M_g) \in D_x$. But, by clause (2.1) of the definition of extension (definition 22), $[\sigma\,\pi]^M_g = [\sigma]^M_g([\pi]^M_g)$. Therefore, $[\sigma\,\pi]^M_g \in D_x$.

CASE 2

Suppose that τ, a member of TM_x, has the form $\lambda v.\rho$. Then, by the induction hypothesis, $[v]_g^M \in D_y$ and $[\rho]_g^M \in D_z$. By clause (2.2) of definition 22, $[\lambda v.\rho]_g^M = f$, where f is a function from D_y into D_z, that is, $f \in \mathrm{Fnc}(D_y, D_z)$, or $D_z^{D_y}$, which is just $D_{z/y}$. So $[\lambda v.\rho]_g^M \in D_{z/y}$. But $x = z/y$. Therefore, $[\lambda v.\rho]_g^M \in D_x$.

Thus, in either case, if $\tau \in \mathrm{TM}_x$, then $[\tau]_g^M \in D_x$. Therefore, by the principle of mathematical induction, we conclude that for each $\tau \in \mathrm{TM}_x$, for each structure M and for each variable assignment g, $[\tau]_g^M \in D_x$.

Definition 23 Equivalence of terms

Let σ and τ be terms of the simply typed Lambda calculus. Then, $\sigma \equiv \tau$, or σ and τ are equivalent, iff, for each structure M and for each variable assignment g, $[\sigma]_g^M = [\tau]_g^M$.

3.3 Deduction

We have introduced the notation of the Lambda calculus, and we have shown that it is used to express functions. The latter is not surprising, since the point of devising the notation is to have a systematic way to express functions. Two questions arise: are there different terms that are terms for the same function? If so, how can one establish that two terms for the same function are indeed terms denoting the same function?

To understand better what the import of these questions is, let us consider two familiar, analogous problems, one that we learned to tackle in primary school, another that we learned to tackle in secondary school. Let us begin with the one from primary school.

Consider the set of what one might call the elementary arithmetic expressions. It comprises the set of Indic numerals (IN) as well as all the terms that can be formed from them using the symbols for addition and multiplication. We call the set of all such expressions EA and we define it as follows.

Definition 24 Elementary arithmetic expressions

(1) $\mathrm{IN} \subseteq \mathrm{EA}$;

(2.1) if $\sigma, \tau \in \mathrm{EA}$, then $(\sigma + \tau) \in \mathrm{EA}$;

(2.2) if $\sigma, \tau \in \mathrm{EA}$, then $(\sigma \cdot \tau) \in \mathrm{EA}$;

(3) nothing else is in EA.

Expressions of EA include: 0, 1, $1298 + 3422$, $155 \cdot (18 + 34)$ and $(542 + 174) \cdot 352$. Clearly, each natural number has many different expressions for it. Indeed, there are infinitely many expressions in EA for each natural number. One might wonder how one determines whether two members of EA are expressions for the same natural number? The answer is to use the rules we have learned in primary school for addition and multiplication to convert one expression to another. The usual strategy is to convert a more complex expression into a simpler one, with a view to converting the two expressions into the simplest expression, an Indic numeral. Two complex expressions

are expressions for the same natural number just in case the two complex expressions can be converted, in a finite number of steps, into the same Indic numeral, using the rules for addition and multiplication. By way of illustration, consider the following three expressions of EA: $\big((3\cdot 5)+(7\cdot 11)\big)\cdot(3+7)\big)$, $(1+3)\cdot\big((6+4)\cdot(15+5)\big)$, and $\big((13\cdot 17)\cdot 2\big)+\big(\big(5\cdot(5\cdot 3)\big)\cdot(3\cdot 2)\big)$.

$$\begin{array}{c}\big((3\cdot 5)+(7\cdot 11)\big)\cdot(3+7)\big)\\ \big(\big(15+(7\cdot 11)\big)\cdot(3+7)\big)\\ \big((15+77)\cdot(3+7)\big)\\ \big(92\cdot(3+7)\big)\\ (92\cdot 10)\\ 920\end{array}$$

$$\begin{array}{c}(1+3)\cdot\big((6+4)\cdot(15+5)\big)\\ 4\cdot\big((6+4)\cdot(15+5)\big)\\ 4\cdot\big(10\cdot(15+8)\big)\\ 4\cdot(10\cdot 23)\\ 4\cdot 230\\ 920\end{array}$$

$$\begin{array}{c}\big((13\cdot 17)\cdot 2\big)+\big(\big(5\cdot(5\cdot 3)\big)\cdot(3\cdot 2)\big)\\ (221\cdot 2)+\big(\big(5\cdot(5\cdot 3)\big)\cdot(3\cdot 2)\big)\\ 442+\big(\big(5\cdot(5\cdot 3)\big)\cdot(3\cdot 2)\big)\\ 442+\big((5\cdot 15)\cdot(3\cdot 2)\big)\\ 442+\big(75\cdot(3\cdot 2)\big)\\ 442+(75\cdot 6)\\ 442+450\\ 892\end{array}$$

We see from these examples that the first two expressions from EA, $\big((3\cdot 5)+(7\cdot 11)\big)\cdot(3+7)\big)$ and $(1+3)\cdot\big((6+4)\cdot(15+5)\big)$, denote the same natural number, whereas the third expression, $\big((13\cdot 17)\cdot 2\big)+\big(\big(5\cdot(5\cdot 3)\big)\cdot(3\cdot 2)\big)$, denotes a different one.

In a similar fashion, one can define the set of all polynomials in, say, the variable x and ask which of these polynomial expressions are expressions of the same function. Two polynomial expressions are expressions of the same function just in case each can be converted into the other, using the various rules of elementary algebra. A typical strategy is to convert the two polynomials into the same polynomial. For example, the polynomials $x^2-5+x+3$ and $(x+2)\cdot(x-1)$ can be both be converted into the polynomial x^2+x-2, as a result of which we know that they are polynomials denoting the same function.

The Lambda calculus, as we saw in section 3.2, is a general notation for expressing functions. When, one might wonder, are two such expressions expressions of the same function? In particular, are there rules whereby each term can be converted into the other, as a result of which we know that the two terms denote the same function? The answer is that there are such rules. The rules, as presented here, are framed in terms of the conversion relation, denoted by $\Rightarrow$.[8] In particular, we are interested in defining a conversion

8. This symbol is used here to denote a different relation from the relation it was used to denote in chapter 3.

relation between pairs of terms. We do so by defining a fundamental conversion relation and then enlarging it recursively. The fundamental conversion relation comprises three kinds of conversions: *alpha* conversion, *beta* conversion, and *eta* conversion.

Recall that the Lambda calculus has the lambda prefix that binds variables. CQL also has quantifier prefixes that bind variables. In chapter 11, we learned that pairs of formulae of CQL, such as $\forall x\, Px$ and $\forall y\, Py$ as well as such pairs as $\forall z\, Rzx$ and $\forall y\, Ryx$, are logically equivalent. Indeed, we said that they are mere alphabetical variants one of the other. The Lambda calculus also has terms that are alphabetical variants. We could say, after the fashion of the characterization of alphabetically variant formulae of CQL, that terms are alphabetical variants of one another just in case, first, they have been formed in the same way, second, their corresponding occurrences of constants are occurrences of the same constant, and, third, their corresponding free variable occurrences are occurrences of the same variable. In other words, two terms are alphabetical variants if they differ only with respect to the variables bound by their corresponding lambda prefixes: where one has a lambda prefix λu binding the occurrences of the variable u, the other has a lambda prefix λv binding the occurrences of the variable v in positions corresponding to the occurrences of u bound by the occurrence of λu. The terms $\lambda x.ax$ and $\lambda y.ay$, for example, are alphabetical variants, whereas the terms $\lambda x.ax$ and $\lambda y.by$ are not, nor are the terms $\lambda x.xa$ and $\lambda y.ay$.

Alpha conversion relates any term to a distinct term that is an alphabetical variant of it. Alpha conversion stipulates that, for each term τ and for variables u and v, if $u \notin \mathrm{Fvr}(\tau)$ and u is free to substitute for v in τ, then $\lambda v.\tau \Rightarrow \lambda u.[u/v]\tau$. We now reformulate in a form suited for deductions as follows.

ALPHA (α) CONVERSION

Alpha (α) conversion
$\lambda v.\tau \Rightarrow \lambda u.[u/v]\tau$ u is free to substitute for v in τ. $u \notin \mathrm{Fvr}(\tau)$.

Notice that nothing is written above the line. This means that alpha conversion is an axiom. There is a condition on the axiom, which is stated below it.

Let us see whether a pair of terms, $\lambda v.\tau$ and $\lambda u.[u/v]\tau$, related by alpha conversion, are indeed alphabetical variants, as described previously. Clearly, the tree diagrams for the two terms are exactly alike, since the latter term arises from the former by substituting for the free occurrences of v in τ occurrences of u. Moreover, the two conditions on alpha conversion guarantee that the number of free occurrences of v in τ is the same as the number of free occurrences of u in $[u/v]\tau$. Let us see how. Suppose, on the one hand, that $[u/v]\tau$ has more free occurrences of u than τ has of free occurrences of v. Then τ must

have free occurrences of u. But the second condition of alpha conversion rules this out. For example, $\lambda y.yy$ and $\lambda x.xy$ are not alphabetical variants, as the number of free occurrences of y in yy is greater than the number of free occurrences of x in xy. At the same time, neither of $\lambda y.yy$ and $\lambda x.xy$ can arise from the other by alpha conversion. $\lambda x.xy$ cannot arise from $\lambda y.yy$ by alpha conversion, since $\lambda x.[x/y]yy$ yields $\lambda x.xx$, which is distinct from the term $\lambda x.xy$.[9] And $\lambda y.yy$ cannot arise from $\lambda x.xy$ by alpha conversion, for if it did, y would have to substitute for x in xy, contrary to the second condition, which requires that y not be free in xy. In contrast, $\lambda x.xy$ and $\lambda z.zy$ are alphabetical variants and each can arise from the other by alpha conversion.

Suppose, on the other hand, that τ has more free occurrences of v than $[u/v]\tau$ has free occurrences of u. Then τ must have free occurrences of v whose corresponding occurrences of u in $[u/v]\tau$ are not free. This means that, in the substitution of u for v in τ, an occurrence of u becomes bound. But the first condition of alpha conversion rules this out. Consider, for example, $\lambda y.\lambda y.yy$ and $\lambda x.\lambda y.xy$. The former is not an alphabetical variant of the latter, as $\lambda y.yy$ has fewer free occurrences of x than $\lambda y.xy$. However, $\lambda y.\lambda y.yy$ does not arise from $\lambda x.\lambda y.xy$ by alpha conversion, for it would require y to substitute for x in $\lambda y.xy$, contrary to the first condition, which requires that y be free to substitute for x in $\lambda y.xy$. In contrast, $\lambda z\lambda y.zy$ is an alphabetical variant of $\lambda x\lambda y.xy$ and $\lambda z\lambda y.zy$ can be obtained from $\lambda x\lambda y.xy$ by alpha conversion, since z is not free in xy and z is free to substitute for x in xy.

The next rule is that of beta conversion. Readers are no doubt familiar with the practice learned in secondary school of substituting a numeral for a variable in a polynomial: For example, one substitutes the numeral 5 into the polynomial $2x^2 + 3x - 4$. The result is: $2 \cdot 5^2 + 3 \cdot 5 - 4$, which equals 51. Now, customary mathematical practice permits such polynomials to serve both to denote a function and to denote a value. The latter is possible, if in the context of discussion the variable has been stipulated to have a value. Thus, should somewhere in the context it be stipulated that x takes on the value of 5, then $2x^2 + 3x - 4$ denotes, not the function, but the number 51. By prefixing the lambda prefix λx to a polynomial, any free occurrence of the variable x in the polynomial becomes bound, indeed bound by the prefixed λx, which thereby shields it from assuming any value from the context. Thus, while $2x^2 + 3x - 4$ may denote either a function or a number, $\big(\lambda x.(2x^2 + 3x - 4)\big)$ denotes only the function. To show that one wishes to apply this function to a number, say 5, one writes $\big(\big(\lambda x.(2x^2 + 3x - 4)\big)5\big)$. Beta conversion says that the term $\big(\big(\lambda x.(2x^2 + 3x - 4)\big)5\big)$ converts to the term $(2 \cdot 5^2 + 3 \cdot 5 - 4)$.

Let us look at the details of the rule of beta conversion. So long as either the substituens has no variables or the term into which the substitution takes place has no lambda prefix, no unwanted consequences arise. However, in the Lambda calculus, any term may be a substituens and the term into which the substitution is to take place may have lambda prefixes. This opens up the possibility that an open term, once it has substituted for a free

9. Of course, $\lambda x.xx$ is an alphabetical variant of $\lambda y.yy$.

variable in a second term, becomes a closed term. Consider $(\lambda x.\lambda y.xy)(\lambda z.zy)$. Notice that the third occurrence of the variable y is free. Hence, the composite term is open. However, beta conversion yields the term $\lambda y.(\lambda z.zy)y$, which has no free variables and so is a closed term. It is therefore necessary to restrict the rule of beta conversion to cases where the substituendum is free to substitute for the variable in the term into which the substitution is to take place. Beta conversion states that, for all terms σ and τ and for each variable υ, if σ is free to substitute for υ in τ, then $(\lambda\upsilon.\tau)\sigma \Rightarrow [\sigma/\upsilon]\tau$. This too can be reformulated in a form suited to deductions.

BETA (β) CONVERSION

Beta (β) conversion
$\dfrac{}{(\lambda\upsilon.\tau)\sigma \Rightarrow [\sigma/\upsilon]\tau}$ σ is free to substitute for υ in τ.

Again, we have an axiom, where a condition on the form of the axiom is written below it.

Finally, we come to eta conversion. The equivalence it asserts for terms of the Lambda calculus is the counterpart of the equivalence of two formulae of CQL where one formula, say α, is the quantifier matrix of the other, $\exists\upsilon\alpha$, and the quantifier prefix of the latter, $\exists\upsilon$, binds no variable in its matrix, α: for example, $\exists x(Pa \wedge Ryc)$ and $Pa \wedge Ryc$.

Eta conversion says that, for each term τ and for each variable υ, if $\upsilon \notin \mathrm{Fvr}(\tau)$, then $\lambda\upsilon.\tau\upsilon \Rightarrow \tau$. Here it is stated as an axiom for a deduction.

ETA (η) CONVERSION

Eta (η) conversion
$\dfrac{}{\lambda\upsilon.\tau\upsilon \Rightarrow \tau}$ $\upsilon \notin \mathrm{Fvr}(\tau)$

As before, we have an axiom, where a condition on the form of the axiom is written below it.

Suppose that q denotes the squaring function on the natural numbers. If we take the universe to be natural numbers, then q has type e/e: apply it to a natural number, say 3, and one obtains the natural number 9. Let x be a variable of type e. qx is therefore a term of type e, though it denotes nothing until x, the free variable, is assigned a value. From the

term qx, we can form the term $\lambda x.qx$. This term is of type e/e. Since the Indic numerals are constants of type e, let us compare the terms q and $\lambda x.qx$. $q0$ is 0, $q1$ is 1, $q2$ is 4, and so on $(\lambda x.qx)0$, by beta conversion, converts to $q0$, which is 0; $(\lambda x.qx)1$ to $q1$, which is 1; $(\lambda x.qx)2$ converts to $q2$, which is 4. In other words, the two lambda expressions are expressions for the same function, the squaring function on the set of natural numbers. We therefore wish the term $\lambda x.qx$ to convert to the term q. However, $\lambda x.(px)x$ should not eta convert to px. For example, let p be a function of type $(e/e)/e$ that when applied to a natural number, say 5, returns a function, which when applied to a number adds 5 to it. For example, $(p5)2$ is 7 and $(p5)6$ is 11, while $(p3)2$ is 5, $(p3)4$ is 7. Now, $\big(\lambda x.(px)x\big)2$ is 4, $\big(\lambda x.(px)x\big)3$ is 6, and in general, $\big(\lambda x.(px)x\big)n$ is $2n$. However, $p2$ is not a natural number; rather, it is a function that adds 2 to a natural number.

The foregoing conversions are also spoken of as reductions. The idea is that the term to the left of $\Rightarrow$ reduces, or simplifies, to the term to the right. This, obviously, is not the case for alpha conversion. However, alpha conversion preserves the structure of the term and therefore does not render the right-hand term any more complex. Now eta conversion certainly does amount to a simplication, or reduction, for the term to the right of $\Rightarrow$ is a proper subterm of the term to the left. Beta conversion also relates a more complex term to a simpler one in the simply typed Lambda calculus, though it may not in the monotyped Lambda calculus. As readers can easily check, beta conversion applied to the term $(\lambda x.xx)(\lambda y.yyy)$ of the monotyped Lambda calculus yields a more complex term. Thus, in the simply typed Lambda calculus, the second term of the conversion relation is either as simple as or simpler than the first term.

We do not wish to confine conversion relation only to the terms directly related by alpha, beta, and eta conversion. In fact, nothing in the conversion rules permit us to assert $\lambda x.x \Rightarrow \lambda x.x$; in particular, this expression is not an instance of alpha conversion. However, just as in classical logic, where each formula is deducible from itself, we wish that each term in the Lambda calculus be a conversion of itself. In other words, we wish for the relation denoted by $\Rightarrow$ to be reflexive.

REFLEXIVITY OF CONVERSION

Reflexivity of conversion
$\dfrac{}{\tau \Rightarrow \tau}$

This too is an axiom. There are no conditions on its form.

At the same time, we wish for the relation to be transitive. In other words, if one term is related by conversion to a second and the second to a third, we wish that the first be related by conversion to the third. Thus, for example, $\lambda x.\big((\lambda y.y)x\big)a \Rightarrow (\lambda y.y)a$ and $(\lambda y.y)a \Rightarrow a$. But again, nothing in the rules permit us to assert $\lambda x.\big((\lambda y.y)x\big)a \Rightarrow a$.

TRANSITIVITY OF CONVERSION

Transitivity of conversion
$\dfrac{\sigma \Rightarrow \tau,\ \tau \Rightarrow \rho}{\sigma \Rightarrow \rho}$

Finally, we wish that the conversion relation be *congruent,* or consistent, with application and abstraction. Thus, as the terms $(\lambda y.xy)a$ and $(\lambda z.zy)b$ convert to xa and by, respectively, so we wish that the composite term $\big((\lambda y.xy)\big)a\big)\big((\lambda z.zy)b\big)$ convert to the composite term $(xa)(by)$: in other words, $\big(\big((\lambda y.(xy)\big)a\big)\big((\lambda z.zy)b\big) \Rightarrow (xa)(by)$. This is the point of the next rule: if $\sigma \Rightarrow \tau$ and $\rho \Rightarrow \pi$, then $\sigma\rho \Rightarrow \tau\pi$.

APPLICATION CONGRUENCE OF CONVERSION

Application congruence of conversion
$\dfrac{\sigma \Rightarrow \tau,\ \rho \Rightarrow \pi}{\sigma\rho \Rightarrow \tau\pi}$

Similarly, we expect conversion to be congruent, or consistent, with abstraction. Thus, if $(\lambda y.zxy)a$ converts to $zxa,$ that is, $(\lambda y.zxy)a \Rightarrow zxa,$ then we wish that $\big(\lambda x.(\lambda y.zxy)a\big)$ convert to $\lambda x.zxa,$ that is, $\big(\lambda x.(\lambda y.zxy)a\big) \Rightarrow \lambda x.zxa$. More generally, if $\sigma \Rightarrow \tau$, then $\lambda v.\sigma \Rightarrow \lambda v.\tau$.

ABSTRACTION CONGRUENCE OF CONVERSION

Abstraction congruence of conversion
$\dfrac{\sigma \Rightarrow \tau}{\lambda v.\sigma \Rightarrow \lambda v.\tau}$

We now define the conversion relation as the transitive, reflexive closure of the relations of alpha, beta, and eta conversions, which is congruent with respect to application and abstraction. To restate the definition formally, we first define the notion of a *conversion formula,* which is any expression of the form $\sigma \Rightarrow \tau$, where σ and τ are lambda terms.

Definition 25 Conversion relation (tree format)
Let σ and τ be lambda terms. $\sigma \Rightarrow \tau$ holds iff there is a tree of conversion formulae each one of which either is a formula at the top of the tree, in which case it is an instance of

one of the axioms specified above, or is a conversion formula obtained from conversion formulae immediately above it in the formula tree by one of the rules specified above.

Exercises: Deduction

1. Define the relation of alphabetical variant for the formula of CQL.

2. Assume that each of the following expressions are all terms. For each term in the first column, which of the other two terms in the same row are its alphabetical variants?

ax	ay	ax
$\lambda x.x$	$\lambda x.x$	$\lambda y.y$
$\lambda x.b$	$\lambda y.b$	$\lambda x.a$
$\lambda x.ax$	$\lambda v.av$	$\lambda v.va$
$\lambda x.xy$	$\lambda z.zv$	$\lambda z.zy$
$\lambda x.\lambda y.xy$	$\lambda v.\lambda w.vw$	$\lambda y.\lambda x.yx$
$\lambda x.xc(\lambda y.ay)$	$\lambda y.yc(\lambda y.ay)$	$\lambda v.vc(\lambda z.az)$

3. Assume that each of the following expressions are all terms. Simplify them as much as possible using alpha, beta, and eta conversions. It may be useful to restore parentheses.

(a)	$\lambda x.xa$	(m)	$(\lambda z.\lambda x.zxx)(ca)b$
(b)	$(\lambda x.bx)a$	(n)	$\lambda z.yz$
(c)	$\lambda z.lzy$	(o)	$(\lambda x.\lambda y.cyx)ab$
(d)	$(\lambda x.cxy)a$	(p)	$\lambda x.ax$
(e)	$(\lambda x.\lambda y.cxy)(ab)$	(q)	$(\lambda x.\lambda y.ayx)y$
(f)	$(\lambda x.\lambda x.cx)ab$	(r)	$\big(\lambda z.\lambda y.\lambda x.a(zx)(zy)\big)(cab)$
(g)	$(\lambda x.cxy)y$	(s)	$(\lambda w.wx)(\lambda y.\lambda x.cxy)$
(h)	$(\lambda z.\lambda z.ca)(ab)$	(t)	$(\lambda z.za)\big((\lambda v.\lambda x.vxx)(\lambda y.\lambda w.gway)\big)$
(i)	$(\lambda x.cxx)z$	(u)	$(\lambda x.x)(\lambda z.za)$
(j)	$\big(\lambda x.c(dx)\big)x$	(v)	$(\lambda x.\lambda y.xcy)ab$
(k)	$(\lambda x.xz)(\lambda v.va)$	(w)	$(\lambda x.xa)(\lambda y.yb)$
(l)	$(\lambda x.\lambda y.axy)y$	(x)	$ab\big(\lambda z.\big(\lambda v.\lambda x.v(\lambda y.dyx)\big)\big)(\lambda w.wz)\big)$

4 The Lambek Typed Lambda Calculus

In section 3, we observed that the assignment of types in the term tree for terms of the simply typed Lambda calculus conform to the rules of / elimination and / introduction of the Lambek calculus. In fact, the types required to categorize all the terms of the simply typed Lambda calculus are precisely the formulae of Lambek calculus obtained by taking two expressions, here *e* and *t,* as atomic formulae and generating from them the composite

formulae whose only connective is /. In addition, as readers can easily see by inspecting the type subscripts used in clause (2) of the definition of simply typed terms (definition 17), the assignment of types to composite terms conforms to the rules of / elimination and / introduction.

As we shall now see, one can expand the terms of the Lambda calculus in such a way that the terms in the expanded set comprise all and only terms categorized by not just one, but all three connectives of the Lambda calculus and that the corresponding formation rules reflect the rules of elimination and introduction for all three connectives. We shall call this expanded version of the Lambda calculus the *Lambek typed Lambda calculus*.

In the two versions of the Lambda calculus presented, we have expressions for functions and expressions for arguments, the latter expressions being possibly expressions also for functions. Both versions follow the predominant mathematical wont of writing an expression showing the application of a function to an argument by putting the function expression to the left and the argument expression on the right. Suppose that s denotes the successor function on the natural numbers. If we take the universe to be natural numbers, then s has type e/e: apply it to a natural number, say 3, and one obtains the natural number 4: that is, $s3$ equals 4. However, it sometimes happens that one writes the function expression on the right and the argument expression of the left. This is the custom, for example, in the notation for exponentiation. In the expression x^2, the argument expression is the variable x and the function expression is ^{2}. If we wish to reflect this custom in the Lambda calculus, then we assign the symbol introduced for the squaring function, not the type e/e, but the type $e\backslash e$, and we write for example, not $q3$ as we did in section 3.3, but $3q$ instead. The Lambek typed Lambda calculus allows for function expressions whose argument expressions are written to their right and for function expressions whose argument expressions are written to their left. A function expression whose argument expression is written to its right has the familiar type α/β, where β is the type of the argument expression and α is the type of the composite expression obtained from the function expression and the argument expression taken in that order. A function expression whose argument expression is written to its left has the new type $\beta\backslash\alpha$, where β is the type of the argument expression and α is the type of the composite expression obtained from the argument expression and the function expression taken in that order.

The Lambek typed Lambda calculus also has expressions for pairs. Given two expressions σ and τ, we can form an expression for the pair of the entities denoted by σ and τ, namely, the expression $\mathrm{pr}(\sigma, \tau)$. Inversely, given an expression for a pair of entities, we have expressions for its first and second coordinates. Should σ be an expression for a pair, then $\mathrm{pj}_1(\sigma)$ is an expression for the first coordinate of the pair and $\mathrm{pj}_2(\sigma)$ for the second. The type of the complex expression is $\alpha \cdot \beta$, where α is the type of the first coordinate and β is the type of the second coordinate. Thus, $\mathrm{pr}(3, 7)$ is an expression that denotes the ordered pair $\langle 3, 7\rangle$ and $\mathrm{pr}(q,s)$ is the ordered pair comprising the squaring function and the successor function.

The types used for the Lambek typed Lambda calculus, which we shall call *Lambek types*, are defined as follows.

Definition 26 Set of Lambek types
Let e and t be distinct entities.

(1) $e,t \in \text{Typ}$;

(2) If $x,y \in \text{Typ}$, then so is (x/y), $(x\backslash y)$, and $(x \cdot y)$;

(3) Nothing else is in Typ.

As with the simple types of the Lambda calculus, the symbols in CN_x are *constants of type* x and the symbols in VR_x are *variables of type* x. The set of all constants, or CN, is $\bigcup_{x\in\text{Typ}} \text{CN}_x$ and the set of all variables, or VR, is $\bigcup_{x\in\text{Typ}} \text{VR}_x$. As we did with the simply typed Lambda calculus, we refer to VR and CN as *atomic terms*. We extend the atomic terms to include composite terms, just as we extended the atomic terms of the simply typed Lambda calculus to its composite terms.

Definition 27 Formation of Lambek typed lambda terms (syncategorematic version)
Let x and y be in Typ.

(1) ATOMIC TERMS

(1.1) $\text{CN}_x \subseteq \text{TM}_x$; (1.2) $\text{VR}_x \subseteq \text{TM}_x$;

(2) COMPOSITE TERMS

(2.1.1) If $\sigma \in \text{TM}_{x/y}$ and $\tau \in \text{TM}_y$, then $(\sigma\,\tau) \in \text{TM}_x$;

(2.1.2) If $\upsilon \in \text{VR}_y$ and $\tau \in \text{TM}_x$, then $(\lambda\upsilon.\tau) \in \text{TM}_{x/y}$.

(2.2.1) If $\sigma \in \text{TM}_{y\backslash x}$ and $\tau \in \text{TM}_y$, then $(\tau\,\sigma) \in \text{TM}_x$;

(2.2.2) If $\upsilon \in \text{VR}_y$ and $\tau \in \text{TM}_x$, then $(\lambda\upsilon.\tau) \in \text{TM}_{y\backslash x}$.

(2.3.1) If $\sigma \in \text{TM}_{x\cdot y}$, then $\text{pj}_1(\sigma) \in \text{TM}_x$ and $\text{pj}_2(\sigma) \in \text{TM}_y$;

(2.3.2) If $\sigma \in \text{TM}_x$ and $\tau \in \text{TM}_y$, then $\text{pr}(\sigma, \tau) \in \text{TM}_{x\cdot y}$.

The set of lambda terms for this expanded calculus is TM, or $\bigcup_{x\in\text{Typ}} \text{TM}_x$.

Punctilious readers may be wondering why, having justified the adoption of types of the form $\alpha\backslash\beta$ by the occasional mathematical practice of writing an expression for a function to the right of the expression for its argument, we have not reflected this practice in the formation of Lambek typed lambda terms by formulating clause (2.2.1) as follows—if $\sigma \in \text{TM}_{y\backslash x}$ and $\tau \in \text{TM}_y$, then $(\tau\,\sigma) \in \text{TM}_x$—thereby having expressions where the function expression occurs to the right of the expression for its argument. The reason is that having lambda terms of both types renders reading the Lambda notation more difficult, and this without any compensating gain in insight. One might then wonder why types of the form $\alpha\backslash\beta$ are introduced at all? The answer is that such types are required for the syntactic analysis of natural language expressions in a type logical grammar, a topic of chapter 14.

We now define a structure for the Lambek typed Lambda calculus. To do so, we need to pair the various types of terms with various types of values. As with the simply typed Lambda calculus, we start with a nonempty set, called U, and we let U be D_e and let D_t be the set with just the truth values, T and F. As before, we use the other types to extend our names for various subsets of the codomain of the interpretation function as follows: $D_{x/y} = \text{Fnc}(D_y, D_x)$ (or, $D_x^{D_y}$), $D_{y\backslash x} = \text{Fnc}(D_y, D_x)$, and $D_{x\cdot y} = D_x \times D_y$. We let $D = \bigcup_{x \in \text{Typ}} D_x$. An interpretation function for the Lambek typed Lambda calculus, then, is any function from CN into D respecting the types.

Definition 28 Structure for CN for the Lambek typed Lambda calculus

Let CN be the set of constants of the Lambek typed Lambda Calculus.

$\langle U, i \rangle$ is a structure for CN iff

(1) U is a nonempty set;

(2) i is a function from CN into D such that, for each $x \in Typ$, if $\tau \in \text{CN}_x$ then $i(\tau) \in D_x$.

To assign a value to terms that have free variables, one requires a variable assignment.

Definition 29 Variable assignment

A variable assignment is any function g from VR into U such that, if $\upsilon \in \text{VR}_x$, then $g(\upsilon) \in D_x$.

Equipped with a structure and a variable assignment, we can now proceed to assign values to composite terms, based on the values assigned to the atomic terms, in an exactly analogous way to the way used for the simply typed Lambda calculus.

Definition 30 Extension for the Lambek typed Lambda calculus

Let M, or $\langle U, i \rangle$, be a structure for CN. Let g be a variable assignment.

(1) ATOMIC TERMS

(1.1) If $\tau \in \text{CN}$, then $[\tau]_g^M = i(\tau)$;

(1.2) If $\tau \in \text{VR}$, then $[\tau]_g^M = g(\tau)$;

(2) COMPOSITE TERMS

(2.1.1) If $\sigma \in \text{TM}_{x/y}$ and $\tau \in \text{TM}_y$, then $[\sigma\tau]_g^M = [\sigma]_g^M([\tau]_g^M)$;

(2.1.2) If $\upsilon \in \text{VR}_y$ and $\tau \in \text{TM}_x$, then $[\lambda\upsilon.\tau]_g^M = f$, where f is a function from D_y into D_x such that $f(a) = [\tau]_{g_{\upsilon \mapsto a}}^M$, for each a in D_y;

(2.2.1) If $\sigma \in \text{TM}_{y\backslash x}$ and $\tau \in \text{TM}_y$, then $[\tau\sigma]_g^M = [\sigma]_g^M([\tau]_g^M)$;

(2.2.2) If $\upsilon \in \text{VR}_y$ and $\tau \in \text{TM}_x$, then $[\lambda\upsilon.\tau]_g^M = f$, where f is a function from D_y into D_x such that $f(a) = [\tau]_{g_{\upsilon \mapsto a}}^M$, for each a in D_y;

(2.3.1) If $\sigma \in \mathrm{TM}_{x \cdot y}$, then $[\mathrm{pj}_1(\sigma)]_g^M = \pi_2^1[\sigma]_g^M$ and $[\mathrm{pj}_2(\sigma)]_g^M = \pi_2^2[\sigma]_g^M$;

(2.3.2) If $\sigma \in \mathrm{TM}_x$ and $\tau \in \mathrm{TM}_y$, then $[\mathrm{pr}(\sigma, \tau)]_g^M = \langle [\sigma]_g^M, [\tau]_g^M \rangle$.

(π_2^1 and π_2^2 are functions that apply to ordered pairs and yield the first coordinate and the second coordinate, respectively.)

Next, we show how deductions are done when Lambek types are paired with terms from the expanded Lambda calculus. The following are the six rules of the Lambek calculus where each schematic formula is paired with a lambda term. We start with the rules for / and \ elimination. Corresponding to the rule of / elimination is the formation of a lambda term expressing function application, while corresponding to the rule of / introduction is the formation of a lambda term expression function abstraction.

/	Elimination	Introduction
	$\dfrac{x/y \mapsto f \quad y \mapsto a}{x \mapsto fa}$	$\dfrac{\begin{array}{c}\vdots \quad [\, x \mapsto v \,] \\ \vdots \qquad \vdots \\ y \mapsto f\end{array}}{\begin{array}{c}y/x \mapsto \lambda v.f \\ v \text{ fresh}\end{array}}$

(To say that a variable is *fresh* is to say that it does not occur in any term above the term in question in the deduction.)

The rules for \ elimination and introduction are similar. The rule of \ elimination corresponds to the formation of a lambda term expressing function application and the rule of \ introduction to the formation of a lambda term expressing function abstraction. (Again, we bring to readers' attention that the lambda terms resulting from the / and \ elimination rules are schematically the same, as are the lambda terms resulting from the introduction rules.)

\	Elimination	Introduction
	$\dfrac{y \mapsto a \quad y\backslash x \mapsto f}{x \mapsto fa}$	$\dfrac{\begin{array}{c}[\, x \mapsto v \,] \quad \vdots \\ \vdots \qquad \vdots \\ y \mapsto f\end{array}}{\begin{array}{c}x\backslash y \mapsto \lambda v.f \\ v \text{ fresh}\end{array}}$

We come to the last pair of rules, those for · elimination and introduction. Here appear lambda terms that are proper to the expanded Lambda calculus. The rule of · elimination corresponds to the appearance of two terms, each an immediate subterm of the term

associated with the schematic type with the ·. The rule of · introduction corresponds to the appearance of a single term whose immediate subterms of the terms associated with the types to the left and the right of the ·.

·	Elimination	Introduction
	$\vdots \quad x \cdot y \mapsto a \quad \vdots$ $\vdots \quad \vdots \quad \vdots$ $\vdots \quad x \mapsto \mathrm{pj}_1(a) \quad y \mapsto \mathrm{pj}_2(a) \quad \vdots$ $\vdots \quad \vdots \quad \vdots \quad \vdots$ $z \mapsto b$ $z \mapsto b$	$x \mapsto a \quad y \mapsto b$ $x \cdot y \mapsto \mathrm{pr}(a,b)$

We shall illustrate Lambek calculus deductions with these terms presently. But first we set out the axioms and rules whereby the lambda terms proper to the Lambek typed Lambda calculus are converted one into another. Terms proper to the expanded calculus are formed using the symbols pr, pj_1, and pj_2. The first is used to denote the pairing function, the function that forms a pair from one or two things. For example, should a and b be symbols in CN_e, denoting say 3 and 7, respectively, then $\mathrm{pr}(a, b)$ is an expression in $\mathrm{TM}_{e \cdot e}$ and denotes $\langle 3, 7 \rangle$. pj_1 and pj_2 denote two *projection* functions, the first function applies to an ordered pair yielding its first coordinate and the second, applied to the same ordered pair, yields its second coordinate. The first projection function, then, applied to $\langle 3, 7 \rangle$ yields 3, and the second projection function applied to the same ordered pair yields 7. In general, if one forms a pair and then takes the first projection of the pair formed, one gets the first element of the pair formed, and if one takes the second project of the pair, one gets the second element of the pair. These facts about the functions mean that composite terms either of the form $\mathrm{pj}_1(\mathrm{pr}(\sigma, \tau))$ or of the form $\mathrm{pj}_2(\mathrm{pr}(\sigma, \tau))$ may be converted, respectively, to the simpler terms σ and τ.

FIRST PROJECTION OF CONVERSION

First projection of conversion
$\mathrm{pj}_1(\mathrm{pr}(\sigma, \tau)) \Rightarrow \sigma$

SECOND PROJECTION OF CONVERSION

Second projection of conversion
$\mathrm{pj}_2(\mathrm{pr}(\sigma, \tau)) \Rightarrow \tau$

In addition, it is evident that there is no difference between a pair and what results from forming a pair from the first and second projections of the initial pair.

PAIRING OF CONVERSION

Pairing of conversion
$\dfrac{}{\mathrm{pr}(\mathrm{pj}_1(\sigma), \mathrm{pj}_2(\sigma)) \Rightarrow \sigma}$

FIRST PROJECTION CONGRUENCE OF CONVERSION

First projection congruence of conversion
$\dfrac{\sigma \Rightarrow \tau}{\mathrm{pj}_1(\sigma) \Rightarrow \mathrm{pj}_1(\tau)}$

SECOND PROJECTION CONGRUENCE OF CONVERSION

Second projection congruence of conversion
$\dfrac{\sigma \Rightarrow \tau}{\mathrm{pj}_2(\sigma) \Rightarrow \mathrm{pj}_2(\tau)}$

PAIRING CONGRUENCE OF CONVERSION

Pairing congruence of conversion
$\dfrac{\sigma \Rightarrow \tau \ , \rho \Rightarrow \pi}{\mathrm{pr}(\sigma, \rho) \Rightarrow \mathrm{pr}(\tau, \pi)}$

Let us now see the examples promised earlier of deductions in the Lambek typed Lambda calculus. We shall use the tree format to display the deductions. Each node of the deduction tree comprises a pair of expressions flanking the symbol $\mapsto$. The expression to the left of the symbol is a formula of the Lambek calculus and the expression to its right is a term from the Lambda calculus. Each rule applied is, in fact, a pair of rules, one from the Lambek calculus applying to the formula or formulae of a pair or of pairs of expressions, the other from the Lambda calculus, applying to the term or terms of a pair or of pairs of expressions.

Next we will present a deduction corresponding to each of the following three theorems of the Lambek calculus.

Theorem 1.1 $\alpha \cdot (\beta \cdot \gamma) \vdash_{FD} (\alpha \cdot \beta) \cdot \gamma$

Theorem 3.1 $(\alpha/\beta) \cdot \beta \vdash_{FD} \alpha$

Theorem 5.1 $(\gamma/\beta) \cdot (\beta/\alpha) \vdash_{FD} \gamma/\alpha$

However, each formula is paired with a lambda term. This means that the deductions are carried out with the deduction rules of this section, and not the deduction rules for formula deduction in the Lambek calculus. This also means that the deducibility relation is not relative to the Lambek calculus formula deduction rules, but relative to the deduction rules of the Lambek typed Lambda calculus. We therefore symbolize the deducibility relation here as $\vdash_{LL}$, rather than as $\vdash_{FD}$.

In our first deduction, which establishes the counterpart of Theorem 1.1, the Lambek calculus formula $\alpha \cdot (\beta \cdot \gamma)$ is paired with the lambda term a.

TO SHOW

$\alpha \cdot (\beta \cdot \gamma) \mapsto a \vdash_{LL} (\alpha \cdot \beta) \cdot \gamma \mapsto \mathrm{pr}\big(\mathrm{pr}\big(\mathrm{pj}_1(a), \mathrm{pj}_1\big(\mathrm{pj}_2(a)\big)\big), \mathrm{pj}_2\big(\mathrm{pj}_2(a)\big)\big)$

PROOF

$$\dfrac{\dfrac{\dfrac{\alpha \cdot (\beta \cdot \gamma) \mapsto a}{\alpha \mapsto \mathrm{pj}_1(a)} \cdot E \qquad \dfrac{\dfrac{\alpha \cdot (\beta \cdot \gamma) \mapsto a}{\beta \cdot \gamma \mapsto \mathrm{pj}_2(a)} \cdot E}{\beta \mapsto \mathrm{pj}_1\big(\mathrm{pj}_2(a)\big)} \cdot I \qquad \dfrac{\dfrac{\alpha \cdot (\beta \cdot \gamma) \mapsto a}{\beta \cdot \gamma \mapsto \mathrm{pj}_2(a)} \cdot E}{\gamma \mapsto \mathrm{pj}_2\big(\mathrm{pj}_2(a)\big)} \cdot E}{\alpha \cdot \beta \mapsto \mathrm{pr}\big(\mathrm{pj}_1(a), \mathrm{pj}_1\big(\mathrm{pj}_2(a)\big)\big)} \cdot I}{(\alpha \cdot \beta) \cdot \gamma \mapsto \mathrm{pr}\big(\mathrm{pr}\big(\mathrm{pj}_1(a), \mathrm{pj}_1\big(\mathrm{pj}_2(a)\big)\big), \mathrm{pj}_2\big(\mathrm{pj}_2(a)\big)\big)} \cdot I$$

It is worth noting that we could have equally paired the Lambek formula $\alpha \cdot (\beta \cdot \gamma)$ with the lambda term $\mathrm{pr}\big(a, \mathrm{pr}(b,c)\big)$, by assuming that the Lambek formula α is paired with the lambda term a, the Lambek formula β with the lambda term b, and the Lambek formula γ with the lambda term c. Doing so, would yield the following proof, where the tree for the proof is just the tree for the previous proof, except that the lambda terms have been changed.

TO SHOW

$\alpha \cdot (\beta \cdot \gamma) \mapsto \mathrm{pr}\big(a,\mathrm{pr}(b,c)\big) \vdash_{LL} (\alpha \cdot \beta) \cdot \gamma \mapsto \mathrm{pr}\big(\mathrm{pr}(a, b), c\big)$

PROOF

$$\dfrac{\dfrac{\dfrac{\alpha \cdot (\beta \cdot \gamma) \mapsto \mathrm{pr}\big(a, \mathrm{pr}(b, c)\big)}{\alpha \mapsto a} \cdot E \qquad \dfrac{\dfrac{\alpha \cdot (\beta \cdot \gamma) \mapsto \mathrm{pr}\big(a, \mathrm{pr}(b, c)\big)}{\beta \cdot \gamma \mapsto \mathrm{pr}(b, c)} \cdot E}{\beta \mapsto b} \cdot I \qquad \dfrac{\dfrac{\alpha \cdot (\beta \cdot \gamma) \mapsto \mathrm{pr}\big(a, \mathrm{pr}(b, c)\big)}{\beta \cdot \gamma \mapsto \mathrm{pr}(b, c)} \cdot E}{\gamma \mapsto c} \cdot E}{\alpha \cdot \beta \mapsto \mathrm{pr}(a, b)} \cdot I}{(\alpha \cdot \beta) \cdot \gamma \mapsto \mathrm{pr}\big(\mathrm{pr}(a, b), c\big)} \cdot I$$

We now turn to a pair of proofs in the Lambek typed Lambda calculus corresponding to a deduction of theorem 3.1.

TO SHOW

$(\alpha/\beta) \cdot \beta \mapsto a \vdash_{LL} \alpha \mapsto \mathrm{pj}_1(a)(\mathrm{pj}_2(a))$

PROOF

$$\dfrac{\dfrac{\dfrac{(\alpha/\beta) \cdot \beta \mapsto a}{\alpha/\beta \mapsto \mathrm{pj}_1(a)} \cdot E \qquad \dfrac{(\alpha/\beta) \cdot \beta \mapsto a}{\beta \mapsto \mathrm{pj}_2(a)} \cdot E}{\alpha \mapsto \mathrm{pj}_1(a)(\mathrm{pj}_2(a))} /E$$

TO SHOW

$(\alpha/\beta)\cdot\beta \mapsto \mathrm{pr}(a,b) \vdash_{LL} \alpha \mapsto ab$

PROOF

$$\dfrac{\dfrac{(\alpha/\beta)\cdot\beta \mapsto \mathrm{pr}(a,b)}{\alpha/\beta \mapsto a \qquad \beta \mapsto b}\cdot E}{\alpha \mapsto ab}/E$$

We conclude our illustrative deductions with two corresponding to a deduction of theorem 5.1.

TO SHOW

$(\gamma/\beta)\cdot(\beta/\alpha) \mapsto a \vdash_{LL} \gamma/\alpha \mapsto \lambda v.\mathrm{pj}_1(a)\big(\mathrm{pj}_2(a)v\big)$

PROOF

$$\dfrac{\dfrac{\dfrac{(\gamma/\beta)\cdot(\beta/\alpha) \mapsto a}{\gamma/\beta \mapsto \mathrm{pj}_1(a) \qquad \dfrac{\beta/\alpha \mapsto \mathrm{pj}_2(a) \qquad [\alpha \mapsto v]_1}{\beta \mapsto \mathrm{pj}_2(a)v}/E}\cdot E}{\gamma \mapsto \mathrm{pj}_1(a)\big(\mathrm{pj}_2(a)\big)v\big)}/E}{\gamma/\alpha \mapsto \lambda v.\mathrm{pj}_1(a)\big(\mathrm{pj}_2(a)\big)v\big)}/I_1$$

TO SHOW

$(\alpha\backslash\beta)/\gamma \mapsto a \vdash_{LL} \alpha\backslash(\beta/\gamma) \mapsto \lambda w.\lambda v.avw$

PROOF

$$\dfrac{\dfrac{[\alpha \mapsto w]_2 \qquad \dfrac{(\alpha\backslash\beta)/\gamma \mapsto a \qquad [\gamma \mapsto v]_1}{\alpha\backslash\beta \mapsto av}/E}{\beta \mapsto avw}\backslash E}{\dfrac{\beta/\gamma \mapsto \lambda v.avw}{\alpha\backslash(\beta/\gamma) \mapsto \lambda w.\lambda v.avw}/I_1}\backslash I_2$$

Exercises: Lambek typed Lambda calculus

1. Assign lambda terms to each formula and carry out the relevant deduction.
   (a) $\alpha \vdash_{FD} (\alpha\cdot\beta)/\beta$
   (b) $\alpha \vdash_{FD} (\beta/\alpha)\backslash\beta$
   (c) $\gamma/\beta \vdash_{FD} (\gamma/\alpha)/(\beta/\alpha)$
   (d) $(\alpha/\beta)/\gamma \vdash_{FD} \alpha/(\gamma\cdot\beta)$

SOLUTIONS TO SOME OF THE EXERCISES

3.1.1 Monotyped Lambda calculus

1. (a) Abbreviation (i) Neither
   (c) Neither (k) A term
   (e) A term (m) Neither
   (g) Abbreviation (o) Abbreviation

2. (a) $y\lambda x.z$ (d) $\lambda x.(\lambda x.y)zx$
   (c) $\big(\lambda y.xz(\lambda z.ya)\big)\big(\lambda x.wy(\lambda z.z)\big)$ (f) $(z(\lambda z.x))$

3. (a) $[x/y]y$ (d) $[\lambda x.xy/y]y$
   (e) $[\lambda x.xy/x]\lambda z.(\lambda x.x)(\lambda y.xyz)$
   (f) $[\lambda y.ayx/z](z(\lambda z.x))$

4. All terms are free to substitute for z in az;
   All terms are free to substitute for y in $\lambda y.ay$;
   All terms but y, $\lambda w.wy$, and $\lambda x.xy$ are free to substitute for z in $\lambda y.ayz$;
   All terms but $\lambda x.xw(\lambda y.by)$ are free to substitute for w in $\lambda x.yw(\lambda y.xy)$.

5. Definition of Ovr

(1) $\mathrm{Ovr}(v) = \{v\}$, for each $v \in \mathrm{VR}$;

(2) $\mathrm{Ovr}(c) = \emptyset$, for each $c \in \mathrm{CN}$;

(3) $\mathrm{Ovr}(\sigma\tau) = \mathrm{Ovr}(s) \cup Ovr(t)$;

(4) $\mathrm{Ovr}(\lambda v.t) = \mathrm{Ovr}(t) \cup \{v\}$.

3.1.2 Simply typed Lambda calculus

1. (a) cd (f) zyc (k) $(\lambda x.yc)c$
   (c) zy (h) ay (m) $(\lambda x.xy)a$
   (e) $\lambda x.dx$ (j) $\lambda y.ad$ (o) $\lambda x.adx$

2. (a) $p \mid (t/e)/(t/e)$ (d) $q \mid t \to t;\ p \mid e$
   (c) $p \mid \big(t/(t/e)\big)/\big(t/(t/e)\big)$ (f) $p \mid \big(x/(t/e)\big)/\big((t/e)/(t/e)\big);$
   $q \mid t/x$