PROBLEM SET EIGHT: LAMBEK CALCULUS

Background

Traditionally, logic was concerned with establishing principles of valid argumentation, or, to put it another way, with determining under what conditions a proposition (the conclusion) can be said to follow from other propositions (the premisses). More specifically, classical propositional logic sought to establish such principles solely on the basis of how the expressions which express propositions—declarative sentences—are combined, leaving aside how simple sentences themselves are constructed from smaller expressions (such as nouns, verbs, adjectives, determiners, etc.).

In this course, so far we have taken a semantic approach, focusing not on sentences but rather on the propositions that they express. We modelled propositions as elements of a pre-boolean algebra, entailment as the preorder on that algebra, and the meanings of the various ways of combining sentences as the algebra operations. But in the traditional, proof-theoretic (i.e. symbol-manipulating) approaches usually taken in introductory logic courses, one never talks about the propositions themselves but rather the sentences that express them, or more precisely, formulas in some logical language (usually classical propositional logic or classical first-order predicate logic) that serve as (extremely) idealized stand-ins for the sentences. Rather than directly analyzing the entailment relation between propositions, instead one works with a **provability** relation denoted by the 'turnstile' symbol $\vdash$) between formulas, or between finite sets of formulas and formulas, and asks which statements of the form $\Gamma \vdash A$ can we prove (where Γ is a finite set of formulas and A is a formula).

In one standard approach of this kind, called **sequent-style natural deduction**, one starts with certain sequents (statements of the form $\Gamma \vdash A$) called **axioms** and then **proves** more sequents (called **theorems**) from the axioms, using **inference rules**. The inference rules in turn are general reasoning principles of the form 'if such-and-such sequents have been proved, then one can take such-and-such sequent as proven.' To be more specific, for classical propositional logic, the most usual axioms are ones of the forms $A \vdash A$ (which corresponds to the reflexivity of entailment in the semantics), and $\mathrm{F} \vdash A$ (which corresponds to necessary falsehoods being bottoms in the semantics). And with one exception, the inference rules are introduction and elimination rules for the connectives, corresponding in the semantics to the algebraic properties of the operations which these connectives have as their meanings. For example, the introduction and elimination rules for conjunction are the proof-theoretic counterparts of the semantic fact that the propositional operation corresponding to conjunction is a glb operation, and those for implication are the proof-theoretic counterparts of the semantic fact that the propositional operation corresponding to implication is the residual of the glb operation. (The exceptional inference rule, called **weakening**, says that if $\Gamma \vdash A$, then also $\Gamma\ B \vdash A$, i.e. if one can prove something from certain hypotheses, then one can still prove it from *more* hypotheses. Many systems of logic (such as relevant logic, linear logic, and Lambek calculus) do *not* allow Weakening!)

But more generally, we can have different *kinds* of propositional logics, each of which is suited to semantic interpretation in a certain class of preordered algebras, e.g. *intuitionistic* logic to pre-heyting algebras, *bilinear* logic (aka Lambek calculus) to residuated pre-semigroups, etc. Once we generalize in this way, we are no longer committed to thinking of the elements of the preordered algebra as sentence meanings and the preorder as entailment, but different interpretations might be available (depending on the kind of logic).

Thus, in classical propositional logic, the meaning of a sequent $\Gamma \vdash A$ is that the meet of the propositions denoted by the formulas in Γ entails the proposition denoted by A. But in Lambek calculus, $\Gamma \vdash A$ means this: linguistic expressions belonging to the syntactic types listed in Γ can be strung together in the given order to form an expression of syntactic type A. (Note also that, whereas in classical logic, Γ is a *set* of formulas, in Lambek calculus, it is a *string* of formulas, because order and repetitions make a difference.)

1 The (Product-Free) Lambek Calculus

We begin by fixing a set of **atomic formulas** (say, $\{\mathrm{NP}, \mathrm{N}, \mathrm{S}\}$), and then recursively defining the set of **formulas** as follows:

a. An atomic formula is a formula.

b. If A and B are formulas, then so are $A\backslash B$ and A/B.

We then define a **sequent** to be an ordered pair $\langle\Gamma, A\rangle$ where Γ is a nonnull string of formulas and A is a formula. Γ is called the **context** of the sequent, the formula occurences in the context are called the **hypotheses**, and A is called the **statement**.

Next, we recursively define a set of sequents called the **provable** sequents. That is, we recursively define a relation, called **provability** (written '$\vdash$'), between nonempty strings of formulas and formulas. Usually we read $\Gamma \vdash A$ as 'Γ proves A' rather than '$\langle \Gamma, A\rangle$ is a provable sequent'. An assertion of this form is called a (provability) **judgment**.

Here's the definition of the provability relation (throughout, italic capital roman letters are metavariables that range over formulas, and capital Greek letters are metavariables that range over non-null strings of formulas):

Base clause: Axioms

$$A \vdash A$$

Recursion Clauses (Inference Rules)

$\backslash$ Elimination:

$$\frac{\Gamma \vdash A \qquad \Delta \vdash A\backslash B}{\Gamma\Delta \vdash B}$$

$\backslash$ Introduction:

$$\frac{A\Gamma \vdash B}{\Gamma \vdash A\backslash B}$$

/ Elimination:

$$\frac{\Gamma \vdash B/A \qquad \Delta \vdash A}{\Gamma\Delta \vdash B}$$

/ Introduction:

$$\frac{\Gamma A \vdash B}{\Gamma \vdash B/A}$$

In an inference rule, the judgments above the horizontal line are called the **premisses** and the judgment below the line is called the **conclusion**. The whole rule is to be understood as a conditional, with the conjunction of the premisses as antecdent and the conclusion as the consequent: if all premisses are true, then so is the conclusion. (An axiom can be understood as an inference rule with no premisses, in other words just a direct assertion, in this case that (no matter what A is), A proves A.) Also, in an Introduction rule, the hypothesis A which appears in the context of the premiss but not in the context of the conclusion is called the **withdrawn** hypothesis.

Note that the Elimnation rules play a role in Lambek calculus analogous to that played by Implication Elimination (aka Modus Ponens) in ordinary intuitionisitc or classical logic, while the Introduction rules are analogous to Implication Elimination (aka Hypothetical Proof or Conditionalization).

And finally, we define a **(formal) proof** of a judgment Σ to be a tree with each node labelled by a judgment, such that the following three conditions hold:

1. The root is labelled by Σ.

2. Each leaf is labelled by an axiom.

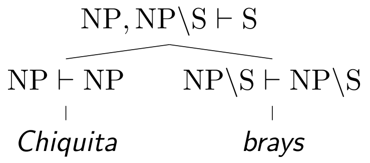
3. For each nonleaf x, the label of x is the conclusion of a rule whose premisses are the labels of x's daughters.

It should be intuitively obvious (though we omit the informal metalanguage proof) that a judgment $\Gamma \vdash A$ is true (i.e. Γ does indeed prove A) iff there is a formal proof of it. In that case we call the judgment a **theorem** (of the Lambek calculus).

2 Linguistic Application: Lambek Grammars

A **Lambek grammar** uses a Lambek calculus to recursively assign word strings to syntactic types as follows. First, we have a lexicon that assigns words (more precisely, word strings of length one) to one or more syntactic types, e.g. Chiquita : NP, brays : NP\S, etc. This is the base of the recursion.

Next, the recursion clause is this: if the strings $s_0, \ldots, s_n$ are assigned to types $A_0, \ldots, A_n$ respectively (for some $n \in \omega$), and $A_0 \ldots A_n \vdash A$, then the string $s_0 \ldots s_n$ is assigned to type A. For example, *Chiqita brays* is a sentence because NP NP\S ⊢ S is a Lambek theorem (the formal proof uses two axioms and one instance of \E):


$\mathrm{NP}, \mathrm{NP}\backslash\mathrm{S} \vdash \mathrm{S}$
$\mathrm{NP} \vdash \mathrm{NP}$
$\mathrm{NP}\backslash\mathrm{S} \vdash \mathrm{NP}\backslash\mathrm{S}$
Chiquita
brays

(Notes: (1) in proof trees, hypotheses are separated by commas, since otherwise the tree-drawing program jams them together; and (2) as a mnemonic, we write words below the corresponding axiom instances, even though technically they are not part of the formal proof.)

Problem 1

Give a formal proof of the Lambek theorem NP TV NP ⊢ S that uses only Elimination rules. (Note: 'TV' (transitive verb) abbreviates (NP\S)/NP.)

The presence of the Introduction rules changes the nature of the game. For one thing, proofs become available which don't comport well with the traditional notion of syntactic constituency (as discussed in introductory syntax courses).

Problem 2

Give a different formal proof of NP TV NP ⊢ S in which the topmost rule (the one that licenses the root S of the proof tree) is /E. (Hint: the proof tree will have FOUR leaves, not three, with the third one corresponding to a 'hypothetical' NP—analogous to what syntacticans call a 'trace'.)

In the application to linguistics, the way that the Introduction rules are used is as follows. Instead of every leaf in the proof tree corresponding to a word, as before, we now allow some leaves to correspond to 'hypothetical' expressions which are inaudible (that is, they are associated with the null string). And when an Introduction rule is used higher in the proof tree, a hypothesis can be withdrawn only if it originated from a leaf associated with a hypothetical (inaudible) expression. Additionally, a hypothesis can be withdrawn using \E (/E) only if the corresponding leaf is the leftmost (rightmost) leaf of the proof tree.

Problem 3

Prove the Lambek theorem $A/B\ B/C \vdash A/C$. (This is known as Right Composition. There's also a left-handed version.)

Problem 4

Prove the Lambek theorem $A \vdash B/(A\backslash B)$. (This is known as Type Raising on the Left. There's also a right-handed version.)

Categorial grammarians often assume that conjunctions (such as *and*, *or*, and *but*) each have an infinite number of syntactic types, each of the form $(A\backslash A)/A$, which is abbreviated $\mathrm{K}(A)$ (here A is a metavariable rsanging over syntactic types). This allows a conjunction to combine first with an A on the right, and then with an A on the left, to produce an A.

Problem 5

Use the preceding idea to give an analysis of the sentence *Juan liked, but Maria hated, Chiquita.* (Examples of this kind exemplify the phenomenon known as 'right node raising') [Hints: (1) The proof tree will have two 'traces' in it. (2) The traces in the proof tree should look like this:

$$\begin{array}{c} \mathrm{NP} \vdash \mathrm{NP} \\ | \\ \mathsf{e} \end{array}$$

where e is the null string.]