Logical Grammar: Proof-Theoretic Background

Carl Pollard

Department of Linguistics
Ohio State University

July 4, 2011

Our Technical Tools

To do linear grammar, we will make use of several different logical systems:

- linear logic
- positive intuitionistic intuitionistic logic
- typed lambda calculus
- type theory/higher order logic

To explain these, we start by introducing a kind of **proof theory** called **sequent-style natural deduction**, or just **ND** for short.

What is Proof Theory?

- **Proof theory** is the part of logic concerned with purely **syntactic** methods for determining whether a **formula** is deducible from a **collection** of formulas.
- what counts as a 'formula' varies from one proof theory to the next. Usually (as in the proof theory we will use) they are certain strings of symbols.
- Here 'syntactic' means that we are only concerned with the **form** of the formulas and not their semantic interpretation. (The part of logic that worries about that is **model theory**).
- What counts as a 'collection' also varies from one proof theory to the next
- In some proof theories, collections are taken to be sets; in others, strings. In the proof theory we will be concerned with, they will be taken to be **finite multisets**.

Finite Multisets

- Roughly speaking, finite multisets are a sort of compromise between strings and finite sets:
  - They are stringlike in the sense that **repetitions matter**.
  - But they are setlike in the sense that **order does not matter.**
- Technically, for any set S, a finite S-multiset is an equivalence class of S-strings, where two strings count as equivalent if they are permutations of each other.
- Alternatively, we can think of a finite S-multiset as a function from a finite subset of S to the positive natural numbers.
- So if we indicate multisets between square brackets, then $[A]$ is a different multiset from $[A, A]$, but $[A, B]$ and $[B, A]$ are the same multiset.

Formulas

- To define a proof theory, we first need a recursively defined set of **formulas**.
- The base of the recursion specifies some **basic** formulas.
- Then the recursion clauses tell how to get additional formulas using **connectives**.

Example: Formulas in Linear Logic (LL)

- In a simple LG of English, we might take the basic formulas to be S, NP, and N.
- The only recursion clause is: if A and B are formulas, so is $A \multimap B$.
- The connective $\multimap$ is called **linear implication** (informally called ‘lollipop’).
- Soon we will see that $\multimap$ works much like the familiar implication $\rightarrow$ of ordinary propositional logic, but with more limited options.

Note: Actually, there are many linear logics. The one we describe here, whose only connective is $\multimap$, is implicative intuitionistic linear propositional logic.

Two Notational Conventions for Formulas

- We use upper-case italic roman letters as metavariables ranging over formulas.
- $\multimap$ associates to the right, e.g. $A \multimap B \multimap C$ abbreviates $A \multimap (B \multimap C)$.

Contexts

- A finite multiset of formulas is called a **context**.
- Careful: this is a distinct usage from the notion of context as linguistically relevant features of the environment in which an expression is uttered.
- We use capital Greek letters (usually Γ or Δ) as metavariables ranging over contexts.

Sequents

- An ordered pair $\langle \Gamma, A \rangle$ of a context and a formula is called a **sequent**.
- Γ is called the **context** of the sequent and A is called the **statement** of the sequent.
- The formula occurences in the context of a sequent are called its **hypotheses** or **assumptions**.

What the Proof Theory Does

- The proof theory recursively defines a set of sequents.
- That is, it recursively defines a relation between contexts and formulas.
- The relation defined by the proof theory is called **deducibility**, **derivability**, or **provability**, and is denoted by $\vdash$ (read 'deduces', 'derives', or 'proves').

Sequent Terminology

- The metalanguage assertion that $\langle \Gamma, A \rangle \in \vdash$ is usually written $\Gamma \vdash A$.
- Such an assertion is called a **judgment**.
- The symbol '$\vdash$' that occurs between the context and the statment of a judgment is called 'turnstile'.
- If Γ is the null multiset, we usually just write $\vdash A$.
- If Γ is the singleton multiset with one occurrence of B, we write $B \vdash A$.
- Commas in the contexts of judgments represent multiset union.
- For example, if $\Gamma = A, B$ and $\Delta = B$, then $\Gamma, \Delta = A, B, B$.

Proof Theory Terminology

- The proof theory itself is a recursive definition.
- The base clauses of the proof theory are called **axioms**, and the recursion clauses are called **(inference) rules**.
- Axioms are just certain judgments.
- Rules are metalanguage conditional statements, whose antecedents are conjunctions of judgments and whose consequent is a judgment.
- The judgments in the antecedent of a rule are called the **premisses** of the rule, and the consequent is called the **conclusion** of the rule.
- Rules are notated by a horizontal line with the premisses above and the conclusion below.

Axioms of (Pure) Linear Logic

- The proof theory for (pure) linear logic has one schema of axioms, and two schemas of rules.
- The axiom schema, variously called Refl (Reflexive Axioms), Hyp (Hypotheses), or simply Ax (Axioms) looks like this:

$$A \vdash A$$

- To call this an axiom **schema** is just to say that upon replacing the metavariable A by any (not necessarily basic) formula, we get an axiom, such as

$$\mathrm{NP} \vdash \mathrm{NP}$$

Note: In LG, hypothesis axioms will be used as (the tecto part of) traces.

Rules of Linear Logic

- Modus Ponens, also called $\multimap$-Elimination:

$$\frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B} \multimap\mathrm{E}$$

- Hypothetical Proof, also called $\multimap$-Introduction:

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B} \multimap\mathrm{I}$$

- Notice that Modus Ponens **eliminates** the connective $\multimap$, in the sense that there is an occurrence of $\multimap$ in one of the premisses (called the **major** premiss; the other premiss is called the **minor** premiss).
- Whereas, Hypothetical Proof **introduces** $\multimap$, in the sense that there is an occurrence of $\multimap$ in the conclusion but not in the (single) premiss.
- Pairs of rules that eliminate and introduce connectives are characteristic of the natural-deduction style of proof theory.

Theorems of a Proof Theory

- If in fact $\Gamma \vdash A$, then we call the sequent $\langle \Gamma, A \rangle$ a **theorem** (in the present case, of linear logic).
- It is not hard to see that $\Gamma \vdash A$ if and only if there is a **proof tree** whose root is labelled with the sequent $\langle \Gamma, A \rangle$.
- Here, by a proof tree we mean an ordered tree whose nodes are labelled by sequents, such that
  - the label of each leaf node is the sequent of an axiom; and
  - the label of each nonleaf node is the sequent of the conclusion of a rule such that the sequents of the premisses of the rule are the labels of the node's daughters.

Proof Tree Notation

- In displaying a proof tree, the root appears at the bottom and the leaves at the top (so from a logician's point of view, linguist's trees are upside down).
- Even though technically the labels are sequents, we conventionally write the corresponding judgments (metalanguage assertions that the sequents are deducible).
- Instead of edges connecting mothers to daughters as in linguist's trees, we write horizontal lines with the label of the mother below and the labels of the daughters above (just as in inference rules).
- Sometimes as a mnemonic we label the horizontal line with the name of the rule schema that was instantiated there.

The Simplest Proof Tree

- The simplest possible proof tree in linear logic has just one leaf, which is also the root.
- In this case the only option is for the node to be labelled by a Hypothesis, e.g.:

$$\text{NP} \vdash \text{NP}$$

- Intuitively, this can be thought of as: suppose you had an NP; then, sure enough, you'd have an NP.
- Don't worry if this doesn't seem to make any sense yet; it will become clear what this means when we use an elaborated form of Hypothesis (namely, trace) in a linguistic analysis.

A (Very) Slightly Less Trivial Proof Tree

$$\dfrac{\mathrm{NP} \vdash \mathrm{NP}}{\vdash \mathrm{NP} \multimap \mathrm{NP}} \multimap \mathrm{I}$$

Another Proof Tree (Type Raising

$$\dfrac{\dfrac{\dfrac{\mathrm{NP} \vdash \mathrm{NP} \qquad \mathrm{NP} \multimap \mathrm{S} \vdash \mathrm{NP} \multimap \mathrm{S}}{\mathrm{NP}, \mathrm{NP} \multimap \mathrm{S} \vdash \mathrm{S}} \multimap\mathrm{E}}{\mathrm{NP} \vdash (\mathrm{NP} \multimap \mathrm{S}) \multimap \mathrm{S}} \multimap\mathrm{I}}{\vdash \mathrm{NP} \multimap (\mathrm{NP} \multimap \mathrm{S}) \multimap \mathrm{S}} \multimap\mathrm{I}$$

Note: Type Raising plays an important role in the analysis of quantificational noun phrases, topicalization, focus constructions, etc.

Positive Intuitionistic Propositional Logic (PIPL)

- PIPL is like LL but with more connectives and rules.
- The only connectives of PIPL are
  - The 0-ary connective T (read ‘true’), and
  - the two binary connectives $\rightarrow$ (intuitionistic implication) and $\wedge$ (conjunction).
- By adding still more connectives and rules to PIPL (F (false), $\vee$ (disjunction), and $\neg$ (negation)) we can get full intutionistic propositional logic (IPL) and classical propositional logic (CPL).
- PIPL underlies the typed lambda calculus (TLC) and higher order logic (HOL), which are used for notating both pheno and semantics in LG.

Axioms of (Pure) PIPL

- Like LL, PIPL has the Hypothesis schema

$$A \vdash A$$

- In addition, it has as the Truth axiom

$$\vdash \mathrm{T}$$

Rules of PIPL

- Introduction and elimination rules for implication
- Introduction and elimination rules for conjunction
- Two **structural** rules, Weakening and Contraction, which affect only the contexts of sequents

PIPL Rules for Implication

These are the same as for LL, but with $\multimap$ replaced by $\rightarrow$:

Modus Ponens, also called $\rightarrow$-Elimination:

$$\frac{\Gamma \vdash A \rightarrow B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B} \rightarrow\mathrm{E}$$

Hypothetical Proof, also called $\rightarrow$-Introduction:

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \rightarrow\mathrm{I}$$

PIPL Rules for Conjunction

The rules for conjunction include *two* elimination rules (for eliminating the left and right conjunct respectively):

$\wedge$-Elimination 1:

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \wedge\text{E1}$$

$\wedge$-Elimination 2:

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \wedge\text{E2}$$

$\wedge$-Introduction:

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \wedge B}\ \wedge\text{I}$$

PIPL Structural Rules

Weakening:

$$\frac{\Gamma \vdash A}{\Gamma, B \vdash A}\,\mathrm{W}$$

Intuitively: if we can prove something from certain assumptions, we can also prove it with more assumptions.

Contraction:

$$\frac{\Gamma, B, B \vdash A}{\Gamma, B \vdash A}\,\mathrm{C}$$

Intuitively: repeated assumptions can be eliminated.

These may seem too obvious to be worth stating, but in fact they *must* be stated, because in some logics (such as LL) they are not available!