Proof Theory for Linguists

Carl Pollard

Department of Linguistics
Ohio State University

August 25, 2016

Logics for Linguistics

Many different kinds of logic are directly applicable to formalizing theories in syntax, phonology, semantics, pragmatics, and computational linguistics. Examples:

- Lambek calculus (intuitionistic bilinear logic)
- linear logic
- intuitionistic propositional/predicate logic
- (simply) typed lambda calculus
- higher order logic
- Martin-Löf type theory
- calculus of inductive constructions

To explain these, we first introduce a kind of **proof theory** called **(Gentzen-sequent-style) natural deduction**, **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.
- Here 'syntactic' means that we are only concerned with the **form** of the formulas, not their semantic interpretation. (The part of logic concerned with that is **model theory**).
- What counts as a 'formula' varies from one proof theory to the next. Usually they are certain strings of symbols.
- Intuitively, to say that A is 'deducible' from Γ is to say that if the formulas in Γ have been 'established', then A can also be established.
- 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.
- To start with, we will take them to be **finite multisets**.

Finite Multisets

- Roughly speaking, finite multisets are a sort of compromise between strings and finite sets:
  - They are stringlike because **repetitions matter**.
  - But they are setlike because **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 recursively define set of **formulas**.
- The base of the recursion specifies some **basic** formulas.
- The recursion clauses tell how to get additional formulas using **connectives**.

Example: Formulas in Linear Logic (LL)

- The set of LL formulas is defined as follows:
  1. Any basic formula is a formula. (N.B.: we have to specify somehow what the basic formulas are.)
  2. If A and B are formulas, then so is $A \multimap B$.
  3. Nothing else is a formula.
- The connective $\multimap$ is called **linear implication** (informally called 'lollipop').
- We adopt the convention that $\multimap$ 'associates to the right', e.g. $A \multimap B \multimap C$ abbbreviates $A \multimap (B \multimap C)$, not $(A \multimap B) \multimap C$.
- As we'll see, $\multimap$ works much like the implication $\rightarrow$ of familiar propositional logic, but with fewer options.

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

Linguistic Application: Tectogrammar (1/4)

- LL is used in *categorial grammar (CG)* frameworks, such as λ-grammar, abstract categorial grammar (ACG), linear categorial grammar (LCG), and hybrid type-logical categorial grammar (HTLCG), which distinguish between **tectogrammatical structure** (also called **abstract syntax** or **syntactic combinatorics**) and **phenogrammatical structure** (also called **concrete syntax**).
- Such frameworks are sometimes called **curryesque**, after Haskell Curry, who first made this distinction (1961).
- Tectogrammatical structure drives the semantic composition.
- Phenogrammatical structure ('phenogrammar' or simply 'pheno') is concerned with surface realization, including word order and intonation.

Tectogrammar (2/4)

- In curryesque frameworks, LL formulas, called **tectotypes** (or just **tectos**) play a role analogous to that played by *nonterminals* in context-free grammar (CFG): they can be thought of as names of syntactic categories of linguistic expressions.
- A currysesque grammar has far fewer rules than a CFG, because the 'combinatory potential' of a linguistic expression is encoded in its tecto.

Tectogrammar (3/4)

In a simple LCG of English (ignoring details such as case, agreement, and verb inflectional form), we might take the basic tectos to be:

S: (ordinary) sentences

$\bar{\text{S}}$: *that*-sentences

NP: noun phrases, such as names

It: 'dummy pronoun' *it*

N: common nouns

Tectogrammar (4/4)

Some nonbasic tectos:

$N \multimap N$: attributive adjectives

$S \multimap \bar{S}$: ‘complementizer’ *that*

$NP \multimap S$: intransitive verbs

$NP \multimap NP \multimap S$: transitive verbs

$NP \multimap NP \multimap NP \multimap S$: ditransitive verbs

$NP \multimap \bar{S} \multimap S$: sentential-complement verbs

$(NP \multimap S) \multimap S$: quantificational NPs, abbreviated QP

$N \multimap QP$: determiners

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 situation in which an expression is uttered. (But some modern type-theoretic semanticists make a connection between the two.)
- 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**. (In modern type theories, this is only one of several different kinds of judgments.)
- The symbol '$\vdash$' that occurs between the context and the statment of a judgment is called 'turnstile'.
- If Γ is empty, we usually just write $\vdash A$.
- If Γ is the singleton multiset with one occurrence of B, we write $B \vdash A$.
- Commas in contexts represent multiset union, e.g. if $\Gamma = A, B$ and $\Delta = B$, then $\Gamma, \Delta = A, B, B$.

Proof Theory Terminology

- The proof theory itself is a recursive definition of the deducibility relation.
- The base clauses of the proof theory identify certain sequents, called **axioms**, as deducible,
- the recursion clauses of the proof theory, called **(inference) 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 its **premisses**, and the consequent is called its **conclusion**.
- 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) LL has one schema of axioms, and two schemas of rules.
- The axiom schema, called Refl (Reflexivity), Hyp (Hypotheses), or just 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 (a judgment that specifies) an axiom, e.g.

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

- In most forms of categorial grammar, hypotheses play a role analogous to that of *traces* in frameworks such as the Minimalist Program (MP) and head-driven phrase structure grammar (HPSG).

Rules of Linear Logic

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

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

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

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

- Modus Ponens **eliminates** the connective $\multimap$, i.e. there is an occurrence of $\multimap$ in one of the premisses (called the **major** premiss; the other premiss is called the **minor** premiss) but not in the conclusion.
- Hypothetical Proof **introduces** $\multimap$, i.e. there is an occurrence of $\multimap$ in the conclusion but not in the 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 $\Gamma \vdash A$, then we call the sequent $\langle \Gamma, A \rangle$ a **theorem** (in the present case, of LL).
- 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 $\langle \Gamma, A \rangle$.
- By a proof tree we mean an ordered tree whose nodes are labelled by sequents, such that
  - the label of each leaf node is 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 is instantiated.

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 an axiom, e.g.:

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

- This just means that any formula is deducible from itself.
- Although this doesn't sound very exciting, it turns out that an elaborated form of such axioms come into play in hypothetical reasoning in syntax, the categorial-grammar analog of wh-movement, quantifier raising, focus constructions, etc.

A More Interesting Proof Tree

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

- This is an instance of the *derived rule* of Type Raising (TR) to be introduced below.
- The statement in the root sequent is the tecto we called QP.
- This enables an ordinary (i.e. nonquantificational) NP to have the 'higher' type of a QP, e.g. in coordinate stuctures such as *Pedro and some donkey*.

Derived Rules (1/2)

- In natural deduction, we say that an inference rule is **derivable** if we *could have* proved the conclusion if the premiss(es) had been provable.
- In other words, we derive an inference rule by presenting a proof tree where
  - the root sequent is the conclusion of the rule, and
  - we allow the premisses of the rule, in addition to the usual axioms, to label the leaves.
- *Example:* The following derived rule is the Converse of Hypothetical Proof (i.e. the premiss and the conclusion are switched):

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

Derived Rules (2/2)

- CHP is LL-derivable as follows:

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

- More useful derived rules:

Hypothetical Syllogism (also called Composition)

$$\frac{\Gamma \vdash B \multimap C \qquad \Delta \vdash A \multimap B}{\Gamma, \Delta \vdash A \multimap C}\text{HS}$$

Generalized Contraposition

$$\frac{\Gamma \vdash A \multimap B}{\Gamma \vdash (B \multimap C) \multimap A \multimap C}\text{GC}$$

Type Raising

$$\frac{\Gamma \vdash A}{\Gamma \vdash (A \multimap B) \multimap B}\text{TR}$$

- Once derived, a rule can be used in any proof just as if it were one of the original rules of the proof system.

Relating Rules and Theorems

- In LL, for any formulas A and B, $A \vdash B$ is a theorem iff the rule schema

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

  is derivable.

- For example, the derived rules GC and TR could just as well be expressed, respectively, as the theorems

$$A \multimap B \vdash (B \multimap C) \multimap A \multimap C$$

$$A \vdash (A \multimap B) \multimap B$$

Positive Intuitionistic Propositional Logic (PIPL, 1/2)

- PIPL is like LL but with more connectives and more axioms and rules.
- The connectives of PIPL are
  - The 0-ary connective T (read 'true'), and
  - the three binary connectives $\rightarrow$ (intuitionistic implication), $\wedge$ (conjunction), and $\vee$ (disjunction).

Positive Intuitionistic Propositional Logic (PIPL, 2/2)

- With the addition of negation, PIPL can be extended to *intuitionistic* or *classical* propositional logic, depending on what rules are adopted for negation. These in turn can be extended to **first-order** logics with the addition of universal and existential quantifers and corresponding rules.
- PIPL also underlies the *type system* of typed lambda calculus (TLC) and higher order logic (HOL), which are widely used for theorizing about meaning, and in curryesque categorial frameworks for theorizing about phenogrammar.

Axioms of PIPL

- Like LL, PIPL has the Hypothesis schema

$$A \vdash A$$

- In addition, it has the True axiom

$$\vdash \mathrm{T}$$

Intuitively, T is usually thought of corresponding to an arbitrary necessary truth.

This can also be thought of as a nullary introduction rule for T.

Rules of PIPL

- Introduction and elimination rules for implication
- Introduction and elimination rules for conjunction
- Introduction and elimination rules for disjunction
- 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:

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

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

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

PIPL Rules for Conjunction

The rules for conjunction include *two* elimination rules (for eliminating the left and right disjunct 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 Rules for Disjunction

The rules for disjunction include *two* introduction rules (for introducing the left and right conjunct respectively):

$\vee$-Elimination:

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

$\vee$-Introduction 1:

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \vee\mathrm{I1}$$

$\vee$-Introduction 2:

$$\frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \vee\mathrm{I2}$$

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!

Extensions of PIPL

- By adding two more connectives—F (false), and ¬ (negation)—and corresponding rules/axioms to PIPL we get full intutionistic propositional logic (IPL).
- With the addition of one more rule we get classical propositional logic (CPL).
- And with the addition of rules for (universal and existential) quantification, we get (classical) first-order logic (FOL).

The Axiom for False (F)

- The False Axiom

$$F \vdash A$$

  is traditionally called EFQ (*ex falso quodlibet*).
- Intuitively, F is usually thought of corresponding to an arbitrary impossibility (necessary falsehood).
- EFQ is easily shown to be equivalent to the following rule:

  F-Elimination:

$$\frac{\Gamma \vdash F}{\Gamma \vdash A}\, FE$$

Rules for Negation

If we think of $\neg A$ as shorthand for $A \to \mathrm{F}$, then these rules are just special cases of Modus Ponens and Hypothetical Proof and needn't be explicitly stated:

¬-Elimination:

$$\frac{\Gamma \vdash \neg A \qquad \Delta \vdash A}{\Gamma, \Delta \vdash \mathrm{F}} \neg \mathrm{E}$$

¬-Introduction, or Proof by Contradiction

$$\frac{\Gamma, A \vdash \mathrm{F}}{\Gamma \vdash \neg A} \neg \mathrm{I}$$

Another name for ¬I is *Indirect Proof*.

Classical Propositional Logic (CPL)

CPL is obtained from IPL by the addition of any one of the following, which can be shown to be equivalent:

Reductio ad Absurdum:

$$\frac{\Gamma, \neg A \vdash \mathrm{F}}{\Gamma \vdash A}\,\mathrm{RAA}$$

Double Negation Elimination:

$$\frac{\Gamma \vdash \neg(\neg A)}{\Gamma \vdash A}\,\mathrm{DNE}$$

Law of Excluded Middle (LEM):

$$\vdash A \vee \neg A$$

Peirce's Law

In IPL, each of the preceding three rules/axioms is equivalent to Peirce's Law, which doesn't mention F or $\neg$:

$$\vdash ((A \to B) \to A) \to A$$

Rules for Quantifiers

- The following rules can be thought of as counterparts of those for $\wedge$ and $\vee$ where, instead of just two "juncts", there is one for each individual in the domain of quantification.
- These rules can be added to either IPL or CPL to obtain either intuitionistic or classical versions of FOL.

Rules for the Universal Quantifier

∀-Elimination, or Universal Instantiation (UI):

$$\frac{\Gamma \vdash \forall x A}{\Gamma \vdash A[x/t]} \forall\text{E}$$

Note: here '$A[x/t]$' is the formula resulting from replacing all free occurrences of x in A by the term t. This is only permitted if t is "free for x in A", i.e. the replacement does not cause any of the free variables of t to become bound.

∀-Introduction, or Universal Generalization (UG)

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall x A} \forall\text{I}$$

Note: here the variable x is not permitted to be free in any of the hypotheses in Γ.

Rules for the Existential Quantifier

∃-Elimination:

$$\frac{\Gamma \vdash \exists x A \qquad \Delta, A[x/y] \vdash C}{\Gamma, \Delta \vdash C} \exists\text{E}$$

Note: here y must be free for x in A and not free in A.

∃-Introduction, or Existential Generalization (EG):

$$\frac{\Gamma \vdash A[x/t]}{\Gamma \vdash \exists x A} \exists\text{I}$$

Note: here t must be free for x in A.