Introduction to Linear Grammar

Carl Pollard

Department of Linguistics
Ohio State University

December 1, 2011

LG Overview

- An LG for an NL is a sequent-style ND system that recursively defines a set of ordered triples called **signs**, each of which is taken to represent an expression of the NL.
- Signs are notated in the form

$$a : A; B; c : C$$

  where

  - $a : A$ is a typed term of a HO theory (the **pheno theory**), called the **pheno term**, or simply the **pheno**
  - B is a formula of a LL (the **tecto** logic) called the **tecto type**, or simply the **tecto**
  - $c : C$ is a typed term of a HO theory (the **semantic** theory), called the **semantic term**, or simply the **semantics**

The Pheno Theory

- There is a basic type s (strings (of phonological words))
- The nonlogical constants are:
  - **e** : s, which denotes the **null** string
  - a large number of string constants which denote phenos of lexical signs, such as it, rained, chiquita, pedro, maria, every, some, farmer, donkey, brayed, saw, believed, that, etc.
  - $\cdot : \mathrm{s} \to \mathrm{s} \to \mathrm{s}$, which denotes concatenation (written infix)
- We have the following nonlogical axioms (here $s, t, u : \mathrm{s}$):

  $$\vdash \forall_{stu}.(s \cdot t) \cdot u = s \cdot (t \cdot u)$$
  $$\vdash \forall_{s}.(\mathbf{e} \cdot s) = s$$
  $$\vdash \forall_{s}.(s \cdot \mathbf{e}) = s$$

  These axioms say that the set of strings forms a monoid with concatenation as the associative operation and the null string as the identity element.

The Tecto Logic

This is (implicative intuitionistic propositional) LL, with basic tecto types (i.e. atomic formulas) such as NP, It, S, $\bar{\mathrm{S}}$, N, etc.

Note: We abbreviate the type $(\mathrm{NP} \multimap \mathrm{S}) \multimap \mathrm{S}$ by QP (mnemonic for ‘quantifcational NP).

The Semantic Theory (1/2)

- There is a basic type e (entities), and types p (propositions) and w (worlds).
- Here we don't commit to which of p and w is basic (Wittgenstein/Lewis vs. Kripke/Montague).
- For convenience, we abbreviate certain types as follows:

  a. $\mathrm{p}_{\mathit{0}} =_{\mathrm{def}} \mathrm{p}$
  b. $\mathrm{p}_{n+\mathit{1}} =_{\mathrm{def}} \mathrm{e} \rightarrow \mathrm{p}_{n}$

- Nonlogical constants include the following:

  a. $@ : \mathrm{p} \rightarrow \mathrm{w} \rightarrow \mathrm{t}$ ('true at', written infix)
  b. a large number of constants which denote meanings of lexical signs, to be given below.

The Semantic Theory (2/2)

The nonlogical axioms ('meaning postulates') describe relationships between meanings, or between meanings and their extensions. For example, we could have the following axioms about the meanings of *and*, *every*, and *some* respectively (here the variables x, y, z have type e, P, Q have type p_{1}, p, q have type p, and w has type w)

$$\vdash \forall_{pqw}.(p \ \mathsf{and} \ q)@w \leftrightarrow (p@w \wedge q@w)$$

$$\vdash \forall_{PQw}.(\mathsf{every} \ P \ Q)@w \leftrightarrow \forall_{x}.(P \ x)@w \rightarrow (Q \ x)@w$$

$$\vdash \forall_{PQw}.(\mathsf{some} \ P \ Q)@w \leftrightarrow \exists_{x}.(P \ x)@w \wedge (Q \ x)@w$$

LG Architecture

In its simplest form, an LG consists of:

- Two kinds of **axioms**:
  - **logical** axioms, called **traces**
  - **nonlogical** axioms, called **lexical entries**
- Two rule schemas:
  - Modus Ponens
  - Hypothetical Proof

Before considering the precise form of the axioms and rules, we need to discuss the form of LG **sequents**.

LG Sequents

- A sign is called **hypothetical** provided its pheno and semantics are both variables.
- An LG **sequent** is an ordered pair whose first component (the **context**) is a finite multiset of hypothetical signs, and whose second component (the **statement**) is a sign.
- The hypothetical sign occurrences in the context are called the **hypotheses** or **assumptions** of the sequent.
- We require that no two hypotheses have the same pheno variable, and that no two hypotheses have the same semantic variable.
- So the contexts are actually just finite **sets**.

Notational convention: we often omit the types of tecto and semantic terms when no confusion will result.

The Trace Axiom Schema

Full form:

$$x : A; B; z : C \vdash x : A; B; z : C$$

Short form (when types of variables are known):

$$x; B; z \vdash x; B; z$$

Two Lexical Entries to Get Started

$\vdash \text{it}; \text{It}; *$ (dummy pronoun *it*)

Recall that $*$ is the logical constant of type T!

$\vdash \lambda_s.s \cdot \text{rained}; \text{It} \multimap \text{S}; \lambda_o.\mathsf{rain}$

Here o is of type T, and the constant rain is of type p.

The Two LG Rule Schemas (Full Form)

- Modus Ponens

$$\frac{\Gamma \vdash f : A \rightarrow D; B \multimap E; g : C \rightarrow F \qquad \Delta \vdash a : A; B; c : C}{\Gamma, \Delta \vdash f\ a : D; E; g\ c : F}$$

- Hypothetical Proof

$$\frac{\Gamma, x : A; B; z : C \vdash d : D; E; f : F}{\Gamma \vdash \lambda_x.d : A \rightarrow D; B \multimap E; \lambda_z.f : C \rightarrow F}$$

The Two LG Rule Schemata (Short Form)

These forms are used when the types of the terms are known.

- Modus Ponens

$$\frac{\Gamma \vdash f; B \multimap E; g \qquad \Delta \vdash a; B; c}{\Gamma, \Delta \vdash f\ a; E; g\ c}$$

- Hypothetical Proof

$$\frac{\Gamma, x; B; z \vdash d; E; f}{\Gamma \vdash \lambda_x.d; B \multimap E; \lambda_z.f}$$

An LG Proof

Here both axiom instances are lexical entries, and the only rule instance is Modus Ponens.

Unsimplified:

$$\frac{\vdash \lambda_s.s \cdot \text{rained}; \text{It} \multimap \text{S}; \lambda_o.\mathsf{rain} \qquad \vdash \text{it}; \text{It}; *}{\vdash (\lambda_s.s \cdot \text{rained})\ \text{it}; \text{S}; (\lambda_o.\mathsf{rain})\ *}$$

Simplified:

$$\frac{\vdash \lambda_s.s \cdot \text{rained}; \text{It} \multimap \text{S}; \lambda_o.\mathsf{rain} \qquad \vdash \text{it}; \text{It}; *}{\vdash \text{it} \cdot \text{rained}; \text{S}; \mathsf{rain}}$$

We use TLC term equivalences and meaning postulates to simplify terms in intermediate conclusions before using them as premisses for later rule instances.

More Nonlogical Constants for Lexical Semantics

- $\vdash \mathsf{p} : \mathrm{e}$ (Pedro)
- $\vdash \mathsf{c} : \mathrm{e}$ (Chiquita)
- $\vdash \mathsf{m} : \mathrm{e}$ (Maria)
- $\vdash \mathsf{donkey} : \mathrm{p}_{\mathit{1}}$
- $\vdash \mathsf{farmer} : \mathrm{p}_{\mathit{1}}$
- $\vdash \mathsf{bray} : \mathrm{p}_{\mathit{1}}$
- $\vdash \mathsf{see} : \mathrm{p}_{\mathit{2}}$
- $\vdash \mathsf{give} : \mathrm{p}_{\mathit{3}}$
- $\vdash \mathsf{believe} : \mathrm{e} \rightarrow \mathrm{p} \rightarrow \mathrm{p}$
- $\vdash \mathsf{persuade} : \mathrm{e} \rightarrow \mathrm{e} \rightarrow \mathrm{p} \rightarrow \mathrm{p}$
- $\vdash \mathsf{every} : \mathrm{p}_{\mathit{1}} \rightarrow \mathrm{p}_{\mathit{1}} \rightarrow p$
- $\vdash \mathsf{some} : \mathrm{p}_{\mathit{1}} \rightarrow \mathrm{p}_{\mathit{1}} \rightarrow p$

More Lexical Entries

$\vdash$ pedro; NP; p

$\vdash$ chiqita; NP; c

$\vdash$ maria; NP; m

$\vdash$ donkey; N; donkey

$\vdash$ farmer; N; farmer

$\vdash \lambda_s.s \cdot \text{brayed}; \text{NP} \multimap \text{S}; \mathsf{bray}$

$\vdash \lambda_{st}.s \cdot \text{saw} \cdot t; \text{NP} \multimap \text{NP} \multimap \text{S}; \mathsf{see}$

$\vdash \lambda_{st}.s \cdot \text{gave} \cdot t; \text{NP} \multimap \text{NP} \multimap \text{NP} \multimap \text{S}; \mathsf{give}$

$\vdash \lambda_{st}.s \cdot \text{believed} \cdot t; \text{NP} \multimap \bar{\text{S}} \multimap \text{S}; \mathsf{believe}$

$\vdash \lambda_{stu}.s \cdot \text{persuaded} \cdot t \cdot u; \text{NP} \multimap \text{NP} \multimap \bar{\text{S}} \multimap \text{S}; \mathsf{believe}$

Note: The finite verb entries are written to combine the verb first with the subject, then with the complements (the reverse of how things are traditionally done!)

Still More Lexical Entries

$\vdash \lambda_s.\text{that} \cdot s; \mathrm{S} \multimap \bar{\mathrm{S}}; \lambda_p.p$ (complementizer *that*)

$\vdash \lambda_{fs}.s \cdot \text{that} \cdot (f\ \mathbf{e}); (\mathrm{NP} \multimap \mathrm{S}) \multimap \mathrm{N} \multimap \mathrm{N}; \lambda_{PQx}.(Q\ x)\ \mathsf{and}\ (P\ x)$ (relativizer *that*)

$\vdash \lambda_{sf}.f\ (\text{every} \cdot s); \mathrm{N} \multimap \mathrm{QP}; \mathsf{every}$

$\vdash \lambda_{sf}.f\ (\text{some} \cdot s); \mathrm{N} \multimap \mathrm{QP}; \mathsf{some}$

Another LG Proof

$$\frac{\vdash \lambda_s.s \cdot \text{brayed}; \text{NP} \multimap \text{S}; \mathsf{bray} \qquad \vdash \text{chiqita}; \text{NP}; \mathsf{c}}{\vdash \text{chiqita} \cdot \text{brayed}; \text{S}; \mathsf{bray\ c}}$$

Yet Another LG Proof

$$\dfrac{\dfrac{\vdash \lambda_{st}.s \cdot \text{saw} \cdot t; \text{NP} \multimap \text{NP} \multimap \text{S}; \mathsf{see4} \qquad \vdash \text{pedro}; \text{NP}; \mathsf{p}}{\lambda_t.\text{pedro} \cdot \text{saw} \cdot t; \text{NP} \multimap \text{S}; \mathsf{see\ p}} \qquad \vdash \text{chiqita}; \text{NP}; \mathsf{c}}{\text{pedro} \cdot \text{saw} \cdot \text{chiquita}; \text{S}; \mathsf{see\ p\ c}}$$

Note that we had to shrink this to tiny to fit it on the slide! This approach of course has its limits.

The Same Proof with Semantics Omitted

Alternatively, if we are not concerned about semantics, we can sometimes overcome the space problem by omitting the semantics components of the signs:

$$\dfrac{\dfrac{\vdash \lambda_{st}.s \cdot \mathrm{saw} \cdot t; \mathrm{NP} \multimap \mathrm{NP} \multimap \mathrm{S} \qquad \vdash \mathrm{pedro}; \mathrm{NP}}{\lambda_{t}.\mathrm{pedro} \cdot \mathrm{saw} \cdot t; \mathrm{NP} \multimap \mathrm{S}} \qquad \vdash \mathrm{chiqita}; \mathrm{NP}}{\mathrm{pedro} \cdot \mathrm{saw} \cdot \mathrm{chiquita}; \mathrm{S}}$$

Of course this approach also has its limits.

An Oversized LG Proof

$$\vdash \lambda_s.s \cdot \text{brayed}; \text{N}$$

$$\vdash \lambda_{st}.s \cdot \text{believed} \cdot t; \text{NP} \multimap \bar{\text{S}} \multimap \text{S} \qquad \vdash \text{pedro}; \text{NP} \qquad \vdash \lambda_s.\text{that} \cdot s; \text{S} \multimap \bar{\text{S}} \qquad \vdash \text{chic}$$

$$\vdash \lambda_t.\text{pedro} \cdot \text{believed} \cdot t; \bar{\text{S}} \multimap \text{S} \qquad \vdash \text{that} \cdot \text{chiquita} \cdot \text{brayed}; \bar{\text{S}}$$

$$\vdash \text{pedro} \cdot \text{believed} \cdot \text{that} \cdot \text{chiquita} \cdot \text{brayed}; \text{S}$$

Another Solution to the Space Problem

[1]:

$$\frac{\vdash \lambda_{st}.s \cdot \text{believed} \cdot t; \text{NP} \multimap \bar{\text{S}} \multimap \text{S}; \mathsf{believe} \qquad \vdash \text{pedro}; \text{NP}; \mathsf{p}}{\vdash \lambda_{t}.\text{pedro} \cdot \text{believed} \cdot t; \bar{\text{S}} \multimap \text{S}; \mathsf{believe\ p}}$$

[2]:

$$\frac{\vdash \lambda_{s}.\text{that} \cdot s; \text{S} \multimap \bar{\text{S}}; \lambda_p.p \qquad \dfrac{\vdash \lambda_{s}.s \cdot \text{brayed}; \text{NP} \multimap \text{S}; \mathsf{bray} \qquad \vdash \text{chiqita}; \text{NP}; \mathsf{c}}{\vdash \text{chiquita} \cdot \text{brayed}; \text{S}; \mathsf{bray\ c}}}{\vdash \text{that} \cdot \text{chiquita} \cdot \text{brayed}; \bar{\text{S}}; \mathsf{bray\ c}}$$

$$\frac{[1] \qquad [2]}{\vdash \text{pedro} \cdot \text{believed} \cdot \text{that} \cdot \text{chiquita} \cdot \text{brayed}; \text{S}; \mathsf{believe\ p\ (bray\ c)}}$$