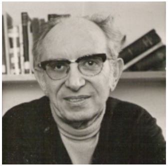
Lambek Calculus

K. Kogkalidis

Logic & Language 2020

Categorial Grammars: History

Kazimierz Ajdukiewicz



Yehoshua Bar-Hillel

AB Grammars

An AB Grammar is a tuple $(\Sigma, \mathcal{A}, S, L)$

- Σ a finite set of symbols
- $\mathcal{A}$ a finite set of primitives, deriving:

$$\mathcal{T}_{\mathcal{A}} := \mathcal{A} \mid \mathcal{T}_{\mathcal{A}}/\mathcal{T}_{\mathcal{A}} \mid \mathcal{T}_{\mathcal{A}}\backslash\mathcal{T}_{\mathcal{A}}$$

- S a distinguished type, $S \in \mathcal{T}_{\mathcal{A}}$
- L a mapping $\Sigma \to \mathcal{T}_{\mathcal{A}}$

Inference Rules

$$X \longleftarrow X/Y, Y$$

$$X \longleftarrow Y, Y\backslash X$$

AB Grammars & Constituency Parsing

Consider a grammar where

$\mathcal{A}$:= $\{S, N, NP\}$

Σ a (simple) lexicon of english

L a mapping from:

common nouns to n

proper nouns to np

determiners to np/n

adjectives to n/n

intrasitive verbs to $np\backslash s$

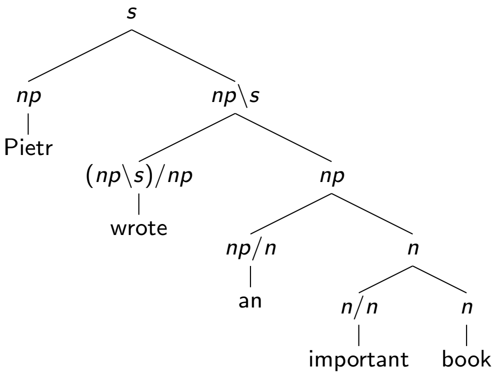
transitive verbs to $(np\backslash s)/np$

...

AB Grammars & Constituency Parsing

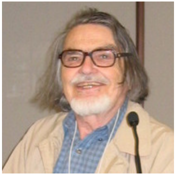
Consider a grammar where

- $\mathcal{A}$ $:= \{S, N, NP\}$
- Σ a (simple) lexicon of english
- L a mapping from:
  - common nouns to n
  - proper nouns to np
  - determiners to np/n
  - adjectives to n/n
  - intrasitive verbs to $np\backslash s$
  - transitive verbs to $(np\backslash s)/np$
  - ...



Pietr (wrote (an (important book)))

Refinement: Lambek Calculus L



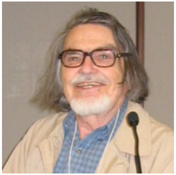
Joachim Lambek

The Mathematics of Sentence Structure (1958):

$$\mathcal{T}_{\mathcal{A}} := \mathcal{A} \mid \mathcal{T}_{\mathcal{A}}/\mathcal{T}_{\mathcal{A}} \mid \mathcal{T}_{\mathcal{A}}\backslash\mathcal{T}_{\mathcal{A}} \mid \mathcal{T}_{\mathcal{A}} \otimes \mathcal{T}_{\mathcal{A}}$$

- $/$ 'right' division (*over*)
- $\backslash$ 'left' division (*under*)
- $\otimes$ concatenation (*with*)

Refinement: Lambek Calculus L



Joachim Lambek

The Mathematics of Sentence Structure (1958):

$$\mathcal{T}_{\mathcal{A}} := \mathcal{A} \mid \mathcal{T}_{\mathcal{A}}/\mathcal{T}_{\mathcal{A}} \mid \mathcal{T}_{\mathcal{A}}\backslash\mathcal{T}_{\mathcal{A}} \mid \mathcal{T}_{\mathcal{A}} \otimes \mathcal{T}_{\mathcal{A}}$$

- $/$ 'right' division (*over*)
- $\backslash$ 'left' division (*under*)
- $\otimes$ concatenation (*with*)

$\leadsto$ $\mathsf{ILL}_{\multimap}$ without Exchange:

$$\frac{\Gamma \vdash B/A \quad \Delta \vdash A}{\Gamma, \Delta \vdash B}\ /E \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash B/A}\ /I$$

$$\frac{\Gamma \vdash A \quad \Delta \vdash A\backslash B}{\Gamma, \Delta \vdash B}\ \backslash E \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A\backslash B}\ \backslash I$$

$$\frac{\Gamma \vdash A \otimes B \quad \Delta, A, B, \Theta \vdash C}{\Delta, \Gamma, \Theta \vdash C}\ \otimes E \qquad \frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ \otimes I$$

Lambek Calculus L

The Lambek Calculus L

- is the grammar of strings, being order-sensitive
- is a substructural logic coinciding with the non-commutative fragment of multiplicative intuitionistic linear logic $\mathrm{ILL}_{\otimes,/,\backslash}$

  $\Longrightarrow$ assumptions of L are no longer multisets, but sequences
- has equal generative capacity to AB- and CF-grammars

Further Refinement: NL

The , of L assumptions still hides an implicit structural rule:

Further Refinement: NL

The , of L assumptions still hides an implicit structural rule: associativity

On the Calculus of Syntactic Types (1961):
Assumptions are now bracketed structures $\mathcal{S} := \mathcal{T}_{\mathcal{A}} \mid (\mathcal{S}, \mathcal{S})$

Further Refinement: NL

The , of L assumptions still hides an implicit structural rule: associativity

On the Calculus of Syntactic Types (1961):
Assumptions are now bracketed structures $\mathcal{S} := \mathcal{T}_{\mathcal{A}} \mid (\mathcal{S}, \mathcal{S})$

$$\frac{\Gamma \vdash B/A \quad \Delta \vdash A}{(\Gamma, \Delta) \vdash B} /E \qquad \frac{(\Gamma, A) \vdash B}{\Gamma \vdash B/A} /I$$

$$\frac{\Gamma \vdash A \quad \Delta \vdash A\backslash B}{(\Gamma, \Delta) \vdash B} \backslash E \qquad \frac{(A, \Gamma) \vdash B}{\Gamma \vdash A\backslash B} \backslash I$$

$$\frac{\Gamma \vdash A \otimes B \quad \Delta[(A, B)] \vdash C}{\Delta[\Gamma] \vdash C} \otimes E \qquad \frac{\Gamma \vdash A \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A \otimes B} \otimes I$$

where $\Gamma[\Delta]$: Δ a sub-structure of Γ

from NL one can recover L via explicit associativity:

$$\frac{\Gamma[(\Delta, (\Theta, \Phi))] \vdash C}{\Gamma[((\Delta, \Theta), \Phi)] \vdash C} A$$

Non-Associative Lambek Calculus NL

The N/A Lambek Calculus NL

- is the grammar of trees, being order- and constituency-sensistive
- is a substructural logic coinciding with the non-commutative non-associative fragment of $\mathrm{ILL}_{\otimes,/,\backslash}$

$\Longrightarrow$ assumptions of NL are no longer sequences, but binary trees

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in L

$$\frac{}{B/C \vdash (A/B)\backslash(A/C)}$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in L

$$\dfrac{\overline{A/B, B/C \vdash A/C}}{B/C \vdash (A/B)\backslash(A/C)}\ \backslash I$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in L

$$\dfrac{\dfrac{\overline{A/B, B/C, C \vdash A}}{A/B, B/C \vdash A/C}\,/I}{B/C \vdash (A/B)\backslash(A/C)}\,\backslash I$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in L

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A/B \vdash A/B}\,Ax \qquad \dfrac{}{B/C, C \vdash B}}{A/B, B/C, C \vdash A}\,/E}{A/B, B/C \vdash A/C}\,/I}{B/C \vdash (A/B)\backslash(A/C)}\,\backslash I$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in L

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A/B \vdash A/B}\,Ax \quad \dfrac{\dfrac{}{B/C \vdash B/C}\,Ax \quad \dfrac{}{C \vdash C}\,Ax}{B/C, C \vdash B}\,/E}{A/B, B/C, C \vdash A}\,/E}{A/B, B/C \vdash A/C}\,/I}{B/C \vdash (A/B)\backslash(A/C)}\,\backslash I$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in NL

$$\dfrac{}{B/C \vdash (A/B)\backslash(A/C)}$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in NL

$$\dfrac{\overline{(A/B, B/C) \vdash A/C}}{B/C \vdash (A/B)\backslash(A/C)} \backslash I$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in NL

$$\dfrac{\dfrac{\overline{((A/B, B/C), C) \vdash A}}{(A/B, B/C) \vdash A/C}\ /I}{B/C \vdash (A/B)\backslash(A/C)}\ \backslash I$$

Example: L vs NL

$$B/C \vdash (A/B)\backslash(A/C)$$

Derivation in NL

$$\begin{array}{cl}
\ldots & \\
\hline
(A/B, (B/C, C)) \vdash C & \\
\hline
((A/B, B/C), C) \vdash A & {\color{red}A} \\
\hline
(A/B, B/C) \vdash A/C & /I \\
\hline
B/C \vdash (A/B)\backslash(A/C) & \backslash I
\end{array}$$

Parsing as Deduction

For categorial grammars, syntactic parsing becomes equated with a logical deduction process, proving the well-formedness of a sentence and finding its structure

$$
\dfrac{\dfrac{\text{Pietr}}{\text{np}} \quad \dfrac{\dfrac{\text{wrote}}{(np\backslash s)/s} \quad \dfrac{\dfrac{\text{an}}{np/n} \quad \dfrac{\dfrac{\text{important}}{\text{n}/\text{n}} \quad \dfrac{\text{book}}{\text{n}}}{(\text{important} \cdot \text{book}) \vdash n}/E}{(\text{an} \cdot (\text{important} \cdot \text{book})) \vdash np}/E}{(\text{wrote} \cdot (\text{an} \cdot (\text{important} \cdot \text{book}))) \vdash np\backslash s}/E}{(\text{Pietr} \cdot (\text{wrote} \cdot (\text{an} \cdot (\text{important} \cdot \text{book})))) \vdash s}\backslash E
$$

Ambiguity

Reading 1:

$$
\dfrac{\dfrac{\mathrm{I}}{\mathrm{np}} \qquad \dfrac{\dfrac{\mathrm{saw}}{(np\backslash s)/s} \qquad \dfrac{\dfrac{\mathrm{the}}{\mathrm{np}/\mathrm{n}} \qquad \dfrac{\dfrac{\mathrm{man}}{n} \qquad \dfrac{\dfrac{\mathrm{with}}{(n\backslash n)/np} \qquad \dfrac{\dfrac{\mathrm{the}}{np/n} \qquad \dfrac{\mathrm{binoculars}}{n}}{(\mathrm{the} \cdot \mathrm{binoculars}) \vdash np}}{(\mathrm{with} \cdot (\mathrm{the} \cdot \mathrm{binoculars})) \vdash n\backslash n}}{(\mathrm{man} \cdot (\mathrm{with} \cdot (\mathrm{the} \cdot \mathrm{binoculars}))) \vdash n}}{(\mathrm{the} \cdot (\mathrm{man} \cdot (\mathrm{with} \cdot (\mathrm{the} \cdot \mathrm{binoculars})))) \vdash np}}{\mathrm{saw} \cdot (\mathrm{the} \cdot (\mathrm{man} \cdot (\mathrm{with} \cdot (\mathrm{the} \cdot \mathrm{binoculars})))) \vdash np\backslash s}}{(\mathrm{I} \cdot (\mathrm{saw} \cdot (\mathrm{the} \cdot (\mathrm{man} \cdot (\mathrm{with} \cdot (\mathrm{the} \cdot \mathrm{binoculars})))))) \vdash s}
$$

Ambiguity

Reading 2:

$$\frac{?}{(\mathsf{I} \cdot ((\mathsf{saw} \cdot (\mathsf{the} \cdot \mathsf{man})) \cdot (\mathsf{with} \cdot (\mathsf{the} \cdot \mathsf{binoculars})))) \vdash s}$$

Ambiguity

Reading 2:

$$\frac{?}{(\mathrm{I} \cdot ((\mathrm{saw} \cdot (\mathrm{the} \cdot \mathrm{man})) \cdot (\mathrm{with} \cdot (\mathrm{the} \cdot \mathrm{binoculars}))) \vdash s}$$

Need an alternative type for “with”

- with (producing noun modifier): $(n\backslash n)/np$
- with (producing verb-phrase modifier): $((np\backslash s)\backslash(np\backslash s))/np$

Ambiguity

Reading 2:

$$\frac{?}{(\mathsf{I} \cdot ((\mathsf{saw} \cdot (\mathsf{the} \cdot \mathsf{man})) \cdot (\mathsf{with} \cdot (\mathsf{the} \cdot \mathsf{binoculars}))) \vdash s}$$

Need an alternative type for "with"

- with (producing noun modifier): $(n\backslash n)/np$
- with (producing verb-phrase modifier): $((np\backslash s)\backslash(np\backslash s))/np$

Syntactic/structural ambiguity becomes lexical ambiguity
contrapose: $VP \to VP\ PP$ vs. $N \to N\ PP$

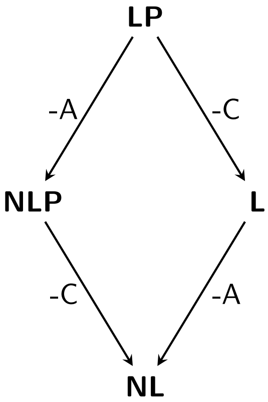
lexicaly ambiguous types can be treated with the & connective

The Full Substructural Picture

$$\begin{tikzcd}[column sep=small] \textbf{LP} \arrow[dr, "-C"] & \\ & \textbf{L} \arrow[dl, "-A"] \\ \textbf{NL} & \end{tikzcd}$$

logic	struct	assoc	commut
LP	multiset	✓	✓
L	string	✓	-
NL	tree	-	-

The Full Substructural Picture



logic	struct	assoc	commut
LP	multiset	✓	✓
L	string	✓	-
NL	tree	-	-
NLP	mobile	-	✓

Comparison with CFGs

- More "formal"
  The Lambek Calculus defines a substructural logic and an algebra.
- More general
  Rule size constant with vocabulary size. Lexicalization happens on the lexicon, assigning a type to each "type" of word.
- Natural syntax-semantics interface
  Connection to I(L)L allow easy translation from syntactic to semantic calculus (tbd . . .)