Non-Classical Logics for Natural Language: Introduction to Formal Semantics, Lambda Calculus and Curry-Howard Correspondence

Raffaella Bernardi

KRDB, Free University of Bozen-Bolzano

e-mail: bernardi@inf.unibz.it

Contents

1 Main points 4
2 Lambek Calculi 5
2.1 Lambek Calculus: Model Theory 6
2.1.1 Lambek Calculus: Kripke Models 7
2.1.2 Frame Constraints: Structural Rules 8
2.1.3 Modes of the Relation 8
2.1.4 Lambek Calculus as a Logical Grammar (I) 9
2.2 Lambek Calculus: Proof Theory 10
2.2.1 Full Residuated System: NL($\Diamond$) 11
2.3 Lambek Calculus as a Logical Grammar (II) 14
2.4 Residuated Logical Grammar complexity 21
3 Conclusion: NL as Logical Grammar 22
4 Formal Semantics: Main questions 24
4.1 Logical Approach 25
4.2 Formal Semantics: What and How 26
4.2.1 Example 27
4.2.2 Quantified NP 28

4.3 Set-Theoretical and Functional Perspectives 31
5 Models, Domains, Interpretation 32
5.1 Lambda Calculus in Linguistics 33
5.1.1 Application and Abstraction 34
5.1.2 Quantified NPs 37
6 Syntax-Semantics Interface 38
6.1 Curry-Howard Correspondence 39
6.2 Example 40
6.3 Lifting 41
6.4 Remarks 42
7 Limitations and Advantages 43
7.1 My current research on this 44

1. Main points

Formal Linguistics is the study of natural language. Formal Linguists aim to

- formally define **grammaticality** of sentences
- understand how syntactic structures are built
- formally define the **meaning** of sentences
- understand how semantic structures are built
- model **syntax-semantic** interface
- find the universal **core** of all natural languages
- find natural language variations

2. Lambek Calculi

- Jim Lambek: Lambek Calculus (L) Non-associative Lambek Calculus (NL) (1958, 1961)
- Dick Oehrle: in the early '80s brought the Lambek'58 paper back to the attention of linguists and logicians (Frans Zwarts and Johan van Benthem)
- Michael Moortgat and Dick Oehrle: used modes of operators (1986) to control structural rules.
- Michael Moortgat and Natasha Kurtonina: extended NL with unary (modal) operators (1995) (NL($\Diamond$)).

2.1. Lambek Calculus: Model Theory

- Originally, Lambek used algebraic models.
- Van Benthem and Kurtonina looked at these calculi as Modal Logics.

Definition 2.1 (Formulas of NL($\Diamond$)) Given a set ATOM of atomic propositional symbols, the logical language of NL($\Diamond$) is as below

$$\mathsf{FORM} \quad ::= \quad \mathsf{ATOM} \mid \mathsf{FORM}/\mathsf{FORM} \mid \mathsf{FORM}\backslash\mathsf{FORM} \mid \mathsf{FORM} \bullet \mathsf{FORM} \mid \Diamond\mathsf{FORM} \mid \blacksquare\mathsf{FORM}.$$

Definition 2.2 (NL($\Diamond$)) The system NL($\Diamond$) is defined by the axioms below. Given $A, B, C \in \mathsf{FORM}$

$$\begin{array}{rl} [\text{REFL}] & A \Rightarrow A, \\ [\text{TRANS}] & \text{If } A \Rightarrow B \text{ and } B \Rightarrow C, \text{ then } A \Rightarrow C, \\ [\text{RES}_2] & A \Rightarrow C/B \text{ iff } A \bullet B \Rightarrow C \text{ iff } B \Rightarrow A\backslash C. \\ [\text{RES}_1] & \Diamond A \Rightarrow C \text{ iff } A \Rightarrow \blacksquare C \end{array}$$

2.1.1. Lambek Calculus: Kripke Models Standard models for modal logics are Kripke models, or relational structures. These structures are rather simple, they only consist of a set together with a collection of relations on that set,

Definition 2.3 (Kripke Models) *A model for* $\mathsf{NL}(\Diamond)$ *is a tuple* $\mathcal{M} = (W, R^3_\bullet, R^2_\Diamond, V)$ *where* W *is a non-empty set,* $R^3_\bullet \subseteq W^3$, $R^2_\Diamond \subseteq W^2$, *and* V *is a valuation* $V : \mathsf{ATOM} \to \mathcal{P}(W)$. *The* $R^3_\bullet$ *relation governs the residuated triple* $(\backslash, \bullet, /)$, *the* $R^2_\Diamond$ *relation governs the residuated pair* $(\Diamond, \blacksquare)$. *Given a model* $\mathcal{M} = (W, R, V)$ *and* $x, y \in W$, *the* **satisfiability relation** *is inductivly defined as follows.*

$$
\begin{array}{rcl}
\mathcal{M}, x \Vdash A & \textit{iff} & x \in V(A) \quad \textit{where } A \in \mathsf{ATOM}. \\
\mathcal{M}, x \Vdash \Diamond A & \textit{iff} & \exists y[R_\Diamond xy \ \&\ \mathcal{M}, y \Vdash A]. \\
\mathcal{M}, y \Vdash \blacksquare A & \textit{iff} & \forall x[R_\Diamond xy \to \mathcal{M}, x \Vdash A]. \\
\mathcal{M}, x \Vdash A \bullet B & \textit{iff} & \exists y \exists z[R_\bullet xyz \ \&\ \mathcal{M}, y \Vdash A \ \&\ \mathcal{M}, z \Vdash B]. \\
\mathcal{M}, y \Vdash C/B & \textit{iff} & \forall x \forall z[(R_\bullet xyz \ \&\ \mathcal{M}, z \Vdash B) \to \mathcal{M}, x \Vdash C]. \\
\mathcal{M}, z \Vdash A\backslash C & \textit{iff} & \forall x \forall y[(R_\bullet xyz \ \&\ \mathcal{M}, y \Vdash A) \to \mathcal{M}, x \Vdash C].
\end{array}
$$

2.1.2. Frame Constraints: Structural Rules In order to maintain completeness in the presence of structural rules, one has to impose restrictions on the interpretation of the accessibility relations $R_\bullet$ and $R_\diamond$. The completeness result is extended to stronger logics by restricting the attention to the relevant classes of frames.

A useful class of structural rules with pleasant completeness properties is characterized by Weak Sahlqvist structural rules. (Kurtonina'95)

Example : $\forall x, y, z, u \in W$

$$(\text{Ass}) \quad (A \bullet B) \bullet C \Leftrightarrow A \bullet (B \bullet C)$$

$$\exists t.Rtxy \& Rutz \leftrightarrow \exists v.Rvyz \& Ruxv$$

2.1.3. Modes of the Relation In modal logics, we can have Relations of the same arity and of different modes marked by means of indexes. These relations can have different properties (e.g. R^3_i could be commutative whereas R^3_j could be associative, etc.). They correspond to families of operators.

2.1.4. Lambek Calculus as a Logical Grammar (I)

Formulas as syntactic categories

$$\mathsf{ATOM} := np \mid n \mid s \quad \text{for (noun phrase, noun, sentence)}$$

Complex formulas are eg. $(np\backslash s), ((np\backslash s)/np), (np/n) \ldots$

Derivability

A derives B iff in all the worlds where A is true, B is true too:

$A \Rightarrow B$ iff $World(A) \subseteq World(B)$ *i.e.* $Expression(CAT_A) \subseteq Expression(CAT_B)$

Example

- np is true in "John", "the student", "the nice student", etc.
- $np\backslash s$ is true in "left", "knows Lori", etc.
- $((np\backslash s)/np) \bullet np$ is true "knows Lori", etc.
- $np \bullet np\backslash s$ is true in "John left", "the student left", "the nice student left", etc.

$((np\backslash s)/np) \bullet np \Rightarrow np\backslash s$

2.2. Lambek Calculus: Proof Theory

$$\frac{}{A \vdash A}\ (\text{axiom})$$

$$\frac{\Gamma[(A, B)] \vdash C}{\Gamma[A \bullet B] \vdash C}\ (\bullet\text{L}) \qquad \frac{\Gamma \vdash A \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A \bullet B}\ (\bullet\text{R})$$

$$\frac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[(B/A, \Delta)] \vdash C}\ (/\text{L}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash B/A}\ (/\text{R})$$

$$\frac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[(\Delta, A\backslash B)] \vdash C}\ (\backslash\text{L}) \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A\backslash B}\ (\backslash\text{R})$$

Recall Classical Logic Sequents.

2.2.1. Full Residuated System: NL($\diamond$)

$$\frac{}{A \vdash A}\ (\text{axiom})$$

$$\frac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[(B/A, \Delta)] \vdash C}\ (/\text{L}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash B/A}\ (/\text{R})$$

$$\frac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[(\Delta, A\backslash B)] \vdash C}\ (\backslash\text{L}) \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A\backslash B}\ (\backslash\text{R})$$

$$\frac{\Gamma[(A, B)] \vdash C}{\Gamma[A \bullet B] \vdash C}\ (\bullet\text{L}) \qquad \frac{\Gamma \vdash A \quad \Delta \vdash B}{(\Gamma, \Delta) \vdash A \bullet B}\ (\bullet\text{R})$$

$$\frac{\Gamma[B] \vdash C}{\Gamma[\langle \blacksquare B \rangle] \vdash C}\ (\blacksquare\text{L}) \qquad \frac{\langle \Gamma \rangle \vdash B}{\Gamma \vdash \blacksquare B}\ (\blacksquare\ \text{R})$$

$$\frac{\Gamma[\langle B \rangle] \vdash C}{\Gamma[\diamond B] \vdash C}\ (\diamond\text{L}) \qquad \frac{\Gamma \vdash B}{\langle \Gamma \rangle \vdash \diamond B}\ (\diamond\text{R})$$

Think of $\diamond$ as a binary $\bullet$, and of $\blacksquare$ as a binary $\backslash$.

Roughly, $\diamond$ corresponds to $A \bullet \cdot$, and $\blacksquare$ to $A\backslash\cdot$

Residuation in the Sequents The Lambek Calculus is also known as the pure calculus of residuation. It's logical rules encode the principle of residuation which is the minimum principle shared by all substructural logics.

(Areces and Bernardi '00) showed how the residuation principle is compiled in the sequent calculus. And used the explained method to extend the logic with other operators governed by similar principles. Their method is based on the work by Goré on Display Logic.

Recall:

$$[RES_1] \quad \forall x \in A, y \in B \left(\begin{array}{c} f(x) \leq_2 y \\ \text{iff} \\ x \leq_1 g(y) \end{array} \right) \qquad \frac{\langle \Gamma \rangle \vdash B}{\Gamma \vdash \blacksquare B} \ (\blacksquare \text{ R})$$

$$[RES_2] \quad \forall x \in A, y \in B, z \in C \left(\begin{array}{c} x \leq_1 h(z, y) \\ \text{iff} \\ f(x, y) \leq_3 z \\ \text{iff} \\ y \leq_2 g(x, z) \end{array} \right) \qquad \frac{(\Gamma, A) \vdash B}{\Gamma \vdash B/A} \ (/\text{R})$$

The other half of the "iff" relation is compiled in the left rules.

The composition of the two functions f, g:

$$f(x, g(x, z) \leq_3 z$$

is encoded in the (\L).

$$\frac{\Delta \vdash B \quad \Gamma[A] \vdash C}{\Gamma[(\Delta, B \backslash A)] \vdash C}\ (\backslash\mathrm{L}) \qquad\qquad B \bullet B \backslash A \vdash A$$

The composition function for unary operators (we will intro them later):

$$f(g(y)) \leq_2 y$$

is the equivalent of the ($\blacksquare$L)

$$\frac{\Gamma[A] \vdash C}{\Gamma[\langle \blacksquare A \rangle] \vdash C}\ (\blacksquare\mathrm{L}) \qquad\qquad \Diamond \blacksquare A \vdash A$$

2.3. Lambek Calculus as a Logical Grammar (II)

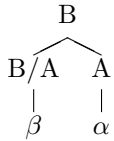
```
CFG Lexicon          Rules                    S
    NP --> john      S --> NP VP            /   \
    IV --> left      VP --> IV             /     VP
    TV --> knows     VP --> TV NP         /     /   \
    DTV --> gives    VP --> DTV NP NP    NP    TV    NP
                                         Lori knows Mary
```

CG Lexicon (Categorization):

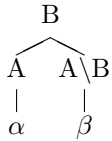
Lori:	np	knows:	$(np\backslash s)/np$
left:	$np\backslash s$	gives:	$((np\backslash s)/np)/np$

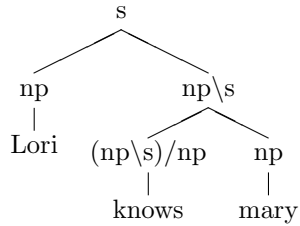
CG Rules (Assembly):

[/ L]



[\ L]





Local dependency: Application $[[[\text{the}]_{det}[\text{student}]_{n}]_{np}[[\text{knows}]_{v}[\text{Lori}]_{np}]_{vp}]_{s}$

Lexicon

Lori	np	left	$np\backslash s$
student	n	knows	$(np\backslash s)/np$
the	np/n		

Gentzen Sequents:

$$\dfrac{\dfrac{}{\underbrace{np/n}_{\text{the}},\ \underbrace{n}_{\text{student}}\ \vdash np/n} \quad \dfrac{\dfrac{n \vdash n \quad np \vdash np}{np/n, n \vdash np}\,(/\text{L}) \qquad s \vdash s}{(np/n, n), np\backslash s \vdash s}\,(\backslash\text{L})}{(\underbrace{np/n}_{\text{the}},\ \underbrace{n}_{\text{student}}), (\underbrace{(np\backslash s)/np}_{\text{knows}},\ \underbrace{np}_{\text{Lori}}) \vdash s}\,(/\text{L})$$

$$\dfrac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[(B/A, \Delta)] \vdash C}\,(/\text{L}) \qquad \dfrac{\Delta \vdash A \quad \Gamma[B] \vdash C}{\Gamma[(\Delta, A\backslash B)] \vdash C}\,(\backslash\text{L})$$

Abstraction: left-branch extraction (proof) Extraction is accounted for as below.

$$\underbrace{\text{The student who } [[\ldots]\text{knows Lori}]_s}_{np}\ \underbrace{\text{left}}_{np\backslash s}$$

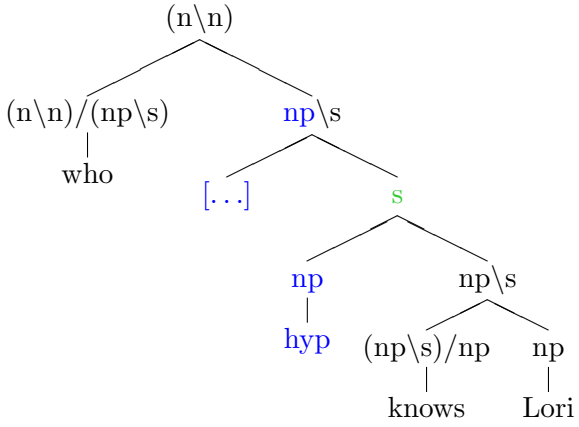
Lexicon

Lori	np	left	$np\backslash s$
student	n	knows	$(np\backslash s)/np$
the	np/n	who	$(n\backslash n)/(np\backslash s)$

$$\dfrac{\dfrac{\dfrac{\dfrac{np \vdash np \quad \dfrac{np \vdash np \quad s \vdash s}{np, np\backslash s \vdash s}\,(\backslash\text{L})}{np, ((np\backslash s)/np, np) \vdash s}\,(/\text{L})}{(np\backslash s)/np, np \vdash np\backslash s}\,(\backslash\text{R})}{\underbrace{(n\backslash n)/(np\backslash s)}_{who}, \underbrace{(np\backslash s)/np}_{knows}, \underbrace{np}_{Lori} \vdash n\backslash n} \quad n\backslash n \vdash n\backslash n}\,(/\text{L})$$

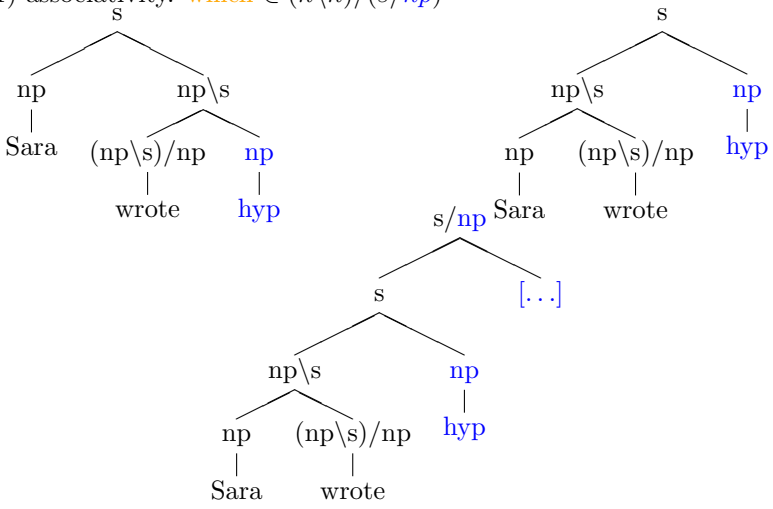
Note, how (\R) decomposes the built structure by removing/abstracting the np.

Abstraction: left-branch extraction (tree) The parse tree is as below



where abstraction can happen since the hypothesis is in a peripheral (accessible) position.

Abstraction: right-branch extraction (tree) Instead, e.g., "which Sara wrote [...]" requires (some form of) associativity. which $\in (n\backslash n)/(s/np)$



Abstraction: Right-branch extraction (proof) The derivation is as below

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{np \vdash np \quad s \vdash s}{np, np\backslash s \vdash s}\,(\backslash\mathrm{L})}{np, ((np\backslash s)/np, np) \vdash s}\,(/\mathrm{L})}{(np, (np\backslash s)/np), np \vdash s}\,(\mathrm{Ass})}{np, (np\backslash s)/np \vdash s/np}\,(/\mathrm{R}) \qquad n\backslash n \vdash n\backslash n}{\underbrace{(n\backslash n)/(s/np)}_{which}, (\underbrace{np}_{Sara}, \underbrace{(np\backslash s)/np}_{wrote}) \vdash n\backslash n}\,(/\mathrm{L})$$

But

- global structural rules are "unsound" when reasoning with natural language (e.g. which sara [wrote [...] in Pescara]).
- The logical grammar will overgenerate proving as grammatical also ungrammatical sentence.

Residuated unary operators (NL($\Diamond$)) Unary operators have been used in several ways.

- Morphological Agreement [D. Heylen (1999)]
- Control of structural rules to avoid e.g. global associativity: Long-Distance Dependency [M. Moortgat 1999,]; Word order [M. Moortgat and R. Oehrle (1996), E. Kraak (1998), W. Vermaat (2005)], i.e. language diversities; Cross-dependencies.
- Classification: Scope distribution [R. Bernardi and R. Moot (2000), R. Bernardi (2002), O. Nilsen (2002)], Licensing Relations: [R. Bernardi (2002)]

2.4. Residuated Logical Grammar complexity

Complexity w.r.t. Chomsky Hierarchy We are interested in the problem of determining whether a string is in the language recognized by a grammar of a certain type.

- For Context Free Language the problem is polynomial.
- the same holds for Mildly CFL.
- whereas, for Context Sensitive Languages the problem is PSPACE-complete

Residuated Logics Complexity The languages recognized by Lambek calculi are as below:

- NL (non associative Lambek Calculus) is n^6
- L (Associative Lambek Calculus) is NP-complete (Pentus'03)
- LP (Associative and Permutative Lambek Calculus) is NP
- NL($\Diamond$) (NL with unary operators) is polynomial.
- NL($\Diamond$) + a class of structural rules (non expanding) is PSPACE (see Moot'00)
- NL($\Diamond$) + structural rules outside that class is undecidable. (see Carpenter '99)
- NL($\Diamond$) + a class of structural rules obtained by embedding LTAG is polynomial. (see Moot'00).

3. Conclusion: NL as Logical Grammar

We have seen that Lambek Calculi can

- prove whether a linguistic structure with local dependency is **grammatical**.
- both compose (assembly) and decompose (extraction) linguistic structures.

The main idea of the approach is that

- The principle of residuation is the **core** of natural languages structure assembly
- Structural rules (frame constraints) capture natural language **diversities.**

We have focused attention on the core calculus of residuation and shown its limitation w.r.t.

- long distance dependencies

$\mathsf{NL}(\Diamond)$ can deal with these phenomena too, by means of structural rules lexically anchored –i.e. that apply only to those structures marked by means of unary operators.

We will now look at how Lambek Calculi

- compositionally account for the assembly of meaning representation.
- have problems with non local scope construal

We will mention how $\mathsf{NL}(\Diamond)$ has been extended to overcome this limitation.

4. Formal Semantics: Main questions

The main questions are:

1. What does a given sentence mean?
2. How is its meaning built?
3. How do we infer some piece of information out of another?

The first and last questions are closely connected.

In fact, since we are ultimately interested in understanding, explaining and accounting for the entailment relation holding among sentences, we can think of **the meaning of a sentence as its truth value**.

4.1. Logical Approach

To tackle these questions we will use Logic, since using Logic helps us answering the above questions at once.

1. Logics have a precise semantics in terms of **models** —so if we can translate/represent a natural language sentence S into a logical formula ϕ, then we have a precise grasp on at least part of the meaning of S.
2. Important **inference problems** have been studied for the best known logics, and often good **computational implementations** exists. So translating into a logic gives us a handle on inference.

When we look at these problems from a computational perspective, i.e. we bring in the implementation aspect too, we move from Formal Semantics to **Computational Semantics**.

4.2. Formal Semantics: What and How

- **What does a given sentence mean?**

  The meaning of a sentence is its **truth value**. Hence, this question can be rephrased in "Which is the meaning representation of a given sentence to be evaluated as true or false?"

- **How is the meaning of a sentence built?**

  Meaning flows from the **lexicon**. The meaning representation of a sentence is built **compositionally** starting from the meaning representations of its atomic expressions, the words.

Hence, we are interested in the meaning (representations) of lexical entries.

4.2.1. Example Let our model be based on the set of entities $E = \{\texttt{lori}, \texttt{ale}, \texttt{sara}, \texttt{pim}\}$ which represent **Lori, Ale, Sara** and **Pim**, respectively. Assume that they all know themselves, plus **Ale** and **Lori** know each other, but they do not know **Sara** or **Pim**; **Sara** does know **Lori** but not **Ale** or **Pim**. The first three are students whereas **Pim** is a professor, and both **Lori** and **Pim** are tall. This is easily expressed set theoretically. Let $[\![\texttt{w}]\!]$ indicate the interpretation of **w**:

$$
\begin{array}{lcl}
[\![\texttt{sara}]\!] & = & \text{sara;} \\
[\![\texttt{pim}]\!] & = & \text{pim;} \\
[\![\texttt{lori}]\!] & = & \text{lori;} \\
[\![\texttt{know}]\!] & = & \{\langle\text{lori, ale}\rangle, \langle\text{ale,lori}\rangle, \langle\text{sara, lori}\rangle, \\
 & & \langle\text{lori, lori}\rangle, \langle\text{ale, ale}\rangle, \langle\text{sara, sara}\rangle, \langle\text{pim, pim}\rangle\}; \\
[\![\texttt{student}]\!] & = & \{\text{lori, ale, sara}\}; \\
[\![\texttt{professor}]\!] & = & \{\text{pim}\}; \\
[\![\texttt{tall}]\!] & = & \{\text{lori, pim}\}.
\end{array}
$$

which is nothing else to say that, for example, the relation **know** is the **set of pairs** $\langle\alpha, \beta\rangle$ where α knows β; or that 'student' is the set of all those elements which are a student.

4.2.2. Quantified NP

a) Every Mexican student of the EM in CL attends the Comp Ling course.

b) No Mexican student of the EM in CL attend the Logic course.

What is the interpretation of "every Mexican student" and of "no Mexican student"?

Individual constants used to denote specific individuals cannot be used to denote quantified expressions like "every man", "no student", "some friends".

Quantified-NPs like "every man", "no student", "some friends" are called **non-referential**.

Quantifier Phrases (QPs) Quantifier Phrases (QP) have been interpreted as sets of properties, i.e. **sets of sets-of-individuals**.

For instance, "every man" denotes the set of properties that every man has. The property of "walking" is in this set iff every man walks. For instance,

$$\begin{array}{lcl} [\![\texttt{man}]\!] & = & \{a, b, c\}; \\ [\![\texttt{fat}]\!] & = & \{a, b, c, d\}; \\ [\![\texttt{dog}]\!] & = & \{d\}; \\ [\![\texttt{run}]\!] & = & \{a, b\}; \\ [\![\texttt{jump}]\!] & = & \{b, c, d\}; \\ [\![\texttt{laugh}]\!] & = & \{b, d\}; \end{array}$$

Which is the interpretation of "every man"?

$$[\![\texttt{every man}]\!] = \{X | [\![\texttt{man}]\!] \subseteq X\} = \{\{a, b, c\}, \{a, b, c, d\}\}.$$

Generalized Quantifiers other generalized quantifiers are:

$$\begin{array}{lcl} [\![\texttt{no man}]\!] & = & \{X \subseteq E \mid [\![\texttt{man}]\!] \cap X = \emptyset\}. \\ [\![\texttt{some man}]\!] & = & \{X \subseteq E \mid [\![\texttt{man}]\!] \cap X \neq \emptyset\}. \\ [\![\texttt{man which VP}]\!] & = & [\![\texttt{man}]\!] \cap [\![\texttt{VP}]\!]. \end{array}$$

Therefore, determiners are as below:

$$\begin{array}{lcl} [\![\texttt{no N}]\!] & = & \{X \subseteq E \mid [\![\texttt{N}]\!] \cap X = \emptyset\}. \\ [\![\texttt{some N}]\!] & = & \{X \subseteq E \mid [\![\texttt{N}]\!] \cap X \neq \emptyset\}. \\ [\![\texttt{every N}]\!] & = & \{X \subseteq E \mid [\![\texttt{N}]\!] \subseteq X\}. \\ [\![\texttt{N which VP}]\!] & = & [\![\texttt{N}]\!] \cap [\![\texttt{VP}]\!]. \end{array}$$

Generalized quantifiers have attracted the attention of many researchers working on the interaction between logic and linguistics.

4.3. Set-Theoretical and Functional Perspectives

Alternatively, one can assume a functional perspective and interpret, for example, **student** as a function from individual (entities) to truth values, $student(monika) = 1$, $student(raffaella) = 0$.

The shift from the set-theoretical to the functional perspective is made possible by the fact that the **sets and their characteristic functions amount to the same thing**:

> if f_X is a function from Y to $\{0, 1\}$, then $X = \{y \mid f_X(y) = 1\}$. In other words, the assertion '$y \in X$' and '$f_X(y) = 1$' are equivalent.

Therefore, the two notations $y(z)(u)$ and $y(u, z)$ are equivalent.

Natural language expressions can be seen as **functions**.

A formal language to represent function is the Lambda calculus.

$$\Downarrow$$

We use FOL augmented with **lambda operators** to represent the meaning of natural language expressions.

5. Models, Domains, Interpretation

In order to interpret meaning representations expressed in FOL augmented with λ, the following facts are essential:

- **Sentences**: Sentences can be thought of as referring to their truth value, hence they denote in the the domain $D_t = \{1, 0\}$.
- **Entities**: Entities can be represented as constants denoting in the domain D_e, e.g. $D_e = \{\texttt{john}, \texttt{vincent}, \texttt{mary}\}$
- **Functions**: The other natural language expressions can be seen as incomplete sentences and can be interpreted as **boolean functions** (i.e. functions yielding a truth value). They denote on functional domains $D_b^{D_a}$ and are represented by functional terms of type $(a \to b)$.

  For instance "walks" misses the subject (of type e) to yield a sentence (t).

  - denotes in $D_t^{D_e}$
  - is of type $(e \to t)$,
  - is represented by the term $\lambda x_e(\texttt{walk}(x))_t$

5.1. Lambda Calculus in Linguistics

The pure lambda calculus is a theory of functions as rules invented around 1930 by Church. It has more recently been applied in Computer Science for instance in "Semantics of Programming Languages".

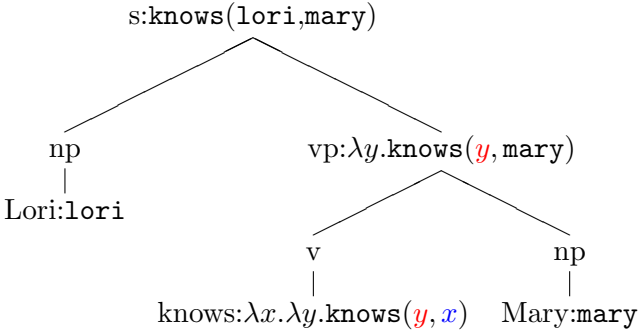
- Here, we are mostly interested in lambda conversion and abstraction. Moreover, we work only with typed-lambda calculus and even more, only with a fragment of it.
- When dealing with linguistic structures, we can also think of function application and λ-abstraction as a way to compose and decompose structures, respectively.
- When considering the "parse as deduction" approach the interest in the λ-calculus is further motivated by its correspondence with intuitionistic implicational logic, known as **Curry-Howard Correspondence**.

5.1.1. Application and Abstraction

While studying the syntax of natural language, we have seen that important concepts to account for are local and long-distance dependencies.

The λ-operator gives us (more or less) a way to represent this link semantically.

For instance, in $\lambda x.\lambda y.\texttt{knows}(y, x)$ we express that the dependency of the subject and object from the verb. Function application and lambda conversion account for local dependencies.



Relative Pronouns Abstraction is used when long distance dependencies occur. For instance, "which John read [...]":

"John" : `john`

"read": $\lambda x.y.\texttt{read}(y, x)$.

What is the role of "which" in e.g. "the book which John read is read"?

The term representing "which" has to express the fact that it is replacing the role of a noun phrase in subject (or object position) within a subordinate sentence while being the subject (object) of the main sentence:

$$\lambda X.\lambda Y.\lambda z.Y(z) \wedge X(z)$$

The double role of "which" is expressed by the double occurrence of z.

Recall :

$$[\![\texttt{N which VP}]\!] \quad = \quad [\![\texttt{N}]\!] \cap [\![\texttt{VP}]\!].$$

Recall,

$$\lambda X.\lambda Y.\lambda z.Y(z) \wedge X(z) \quad [\![\texttt{N which VP}]\!] \quad = \quad [\![\texttt{N}]\!] \cap [\![\texttt{VP}]\!].$$

i. read u: $\lambda y(\texttt{read}(y, u))$ ii. John read u: $\texttt{read}(\texttt{j}, u)$
iii. John read: $\lambda u.\texttt{read}(\texttt{j}, u)$ iv. which John read: $\lambda Y.\lambda z.Y(z) \wedge \texttt{read}(\texttt{j}, z)$

- at the syntactic level we said that the relative pronoun "which" plays the role of the verb's object and it leaves a gap in the object position.
- Semantically, the gap is represented by the u on which the relative pronoun forces the abstraction [iii.] before taking its place.

5.1.2. Quantified NPs

Recall :

$[\![\texttt{some man}]\!] = \{X \subseteq E \mid [\![\texttt{man}]\!] \cap X \neq \emptyset\}.$

$[\![\texttt{every man}]\!] = \{X \subseteq E \mid [\![\texttt{man}]\!] \subseteq X\}$

some man: $\lambda X.\exists z.\texttt{Man}(z) \wedge X(z)$

every man: $\lambda X.\forall z.\texttt{Man}(z) \Rightarrow X(z)$

For uniformity (proper names and QPs can occur in the same context syntactically), proper names have been interpreted as the set of those properties which are true of the person. Eg.

$[\![\texttt{werner}]\!] = \{\texttt{german}, \texttt{prof}, \texttt{man}, ..\}$

$[\![\texttt{werner}]\!] = \{X | [\![X(\texttt{werner})]\!] = 1\}$

werner: $\lambda X.X(\texttt{werner})$

6. Syntax-Semantics Interface

- We have seen how semantic structures are built, the last issue was how to model **syntax-semantic** interface.
- We have seen that if we use CFG there is no tied connection between the two aspects.
- Similarly, this holds for other formal grammars, e.g. HPSG, TAG, etc.
- The Lambek Calculus is in a (formal) correspondence with Lambda Calculus.
- Hence, while proving the grammaticality of a structure (parsing), it builds the corresponding meaning representations.

Definition 6.1 (Categories and Types) *Let us define a function* $\texttt{type} : \mathit{CAT} \rightarrow \mathit{TYPE}$ *which maps syntactic categories to semantic types.*

$$\begin{array}{ll} \texttt{type}(np) = e; & \texttt{type}(A/B) = (\texttt{type}(B), \texttt{type}(A)); \\ \texttt{type}(s) = t; & \texttt{type}(B\backslash A) = (\texttt{type}(B), \texttt{type}(A)); \\ \texttt{type}(n) = (e, t). & \end{array}$$

6.1. Curry-Howard Correspondence

The Curry-Howard correspondence tells us that every proof in the natural deduction calculus for intuitionistic implicational logic can be encoded by a typed λ-term and vice versa.

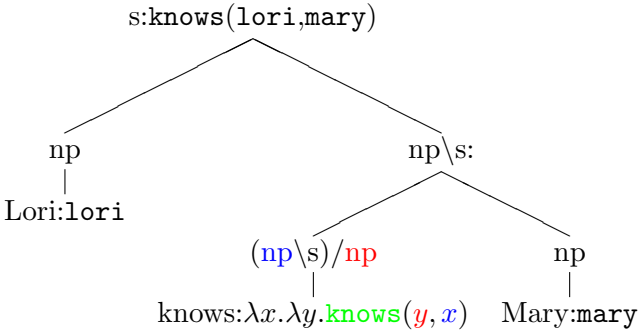
Thanks to this correspondence, the logical rules of this logic or of fragments of it can be labeled with lambda terms as shown below for the non-associative product free Lambek calculus, by means of example.

$$\frac{}{x : A \vdash x : A}\ (\text{axiom})$$

$$\frac{\Delta \vdash t : B \quad \Gamma[x : A] \vdash u : C}{\Gamma[(y : A/B, \Delta)] \vdash u[x := y\ t] : C}\ (/\text{L}) \qquad \frac{\Gamma, x : B \vdash t : A}{\Gamma \vdash \lambda x.t : A/B}\ (/\text{R})$$

$$\frac{\Delta \vdash t : B \quad \Gamma[x : A] \vdash u : C}{\Gamma[(\Delta, y : B\backslash A)] \vdash u[x := y\ t] : C}\ (\backslash\text{L}) \qquad \frac{x : B, \Gamma \vdash t : A}{\Gamma \vdash \lambda x.t : B\backslash A}\ (\backslash\text{R})$$

6.2. Example



$$\frac{np : Z \vdash np : Z \quad \dfrac{np : X \vdash np : X \quad s : Y \vdash s : Y}{np : X, np\backslash s : V \vdash s : Y[V(X)/Y]}}{np : X, ((np\backslash s)/np : P, np : Z) \vdash s : V(X)[P(Z)/V]}$$

Hence, $(P(Z))(X)$ where P, Z, X are the meaning representations of "knows", "Mary" and "Lori", respectively. Relative pronoun.

6.3. Lifting

We said before that for uniformity "werner" is also considered the set of the properties that uniquely identify him.

But we have assigned to proper names the syntactic category: np.

The correspondence guarantees there is no mismatch between categories and lambda.

$$\dfrac{\dfrac{np : \texttt{werner} \vdash np : \texttt{werner} \quad s : z \vdash s : z}{np : \texttt{werner}, X : np\backslash s \vdash s : z[X(\texttt{werner})/z]}\,(\backslash\mathrm{L})}{np : \texttt{werner} \vdash s/(np\backslash s) : \lambda X.X(\texttt{werner})}\,(/\mathrm{R})$$

Which is the category assigned to QPs, too. Since they are of type $((e \to t) \to t)$ –functions from properties to truth values.

6.4. Remarks

The Lambek calculi are fragments of intuitionistic implicational logic. Consequently, the lambda terms computed by it forms a fragment of the full language of lambda terms.

First of all, since empty antecedents are not allowed and the Lambek calculi are resource sensitive, viz. each assumption is used exactly once (no weakening and no contraction), the system reasons about lambda terms with specific properties:

(i) each subterm contains a free variable; and

(ii) no multiple occurrences of the same variable are present.

7. Limitations and Advantages

Limitations

- Long distance dependency via structural rules controlled by means of unary operators. Is this a good solution?
- What is the (linguistic) meaning of the unary operators?
- $\mathsf{NL}(\diamond)$ does not account for non local scope construals.
- Which is the real linguistic coverage of the grammar?

Advantages

- simple principle at the heart of Logic and of natural language (residuation)
- syntax-semantic interface captured formally
- Grail: Automated Proof System

Method extend the logic of the minimum ingredients needed to increase its expressivity, but without reaching overgeneration problems and loosing the correspondence with meaning representation.

7.1. My current research on this

- with Moortgat, Goré and Kurtonina, we are looking at dual-Lambek Calculi –considering structures on the right, and communication postulates between Lambek and dual-lambek (**Bi-Lambek**). Bi-Lambek properly models non local scope construal problem. It is complete with respect to the same class of models of NL. It seems to be polynomial. But we have no (final) result on the Curry-Howard correspondence... yet ;)
- with Anna Szabolsci we are studying semantic information that effect syntactic distribution and how the grammar can handle them.
- Future (?): look for linguistically based semantic interpretation of the unary operators (Anna Szabolsci and Natasha Kurtonina).