Lambek’s computational approach to conjugation

Silvia Ghilezan

University of Novi Sad

LAP 2014, IUC, Dubrovnik, September 22-26, 2014
Journal, Jim, Lambek
1922, Leipzig - 2014, Montréal

“For more than 60 years, Jim Lambek has been a profoundly inspirational mathematician, with groundbreaking contributions to algebra, category theory, linguistics, theoretical physics, logic and proof theory... Jim Lambek’s ideas keep inspiring upcoming generations of scholars.”

(Festschrift on the occasion of Lambek’s 90th birthday, 2014)
Kosta Došen
Djordje Ćubrić
Zoran Petrić
Predrag Tanović
S.G.
Duško Pavlović
To celebrate his life

Jim Lambek was one of my postdoc supervisors. He would ask a trivial question, and 3 months later my research would advance to the point where I would understand what he meant. He made me understand that science is a conversation between people. He would take me to a quiet sushi restaurant and teach me how to be rational, and how to use chopsticks. Life makes sense because we meet such people.

Duško Pavlović
I have come to know Jim Lambek during my visit and work at the Seminar for Mathematical Linguistics in the spring of 1993 at McGill University, Montréal.

It was an inspiring project ran by Jim Lambek which linked between linguists and mathematicians (Ed Kennan, distinguished linguist from UCLA)
Overview

- This is an overview of Lambek’s work on formal grammars for verb conjugation in several languages English, French, Latin, Turkish, Arabic, Hebrew and partly Serbian and Croatian.

- We will focus on the production grammar of conjugation of simple tenses in Serbian and Croatian, which is along the lines of Lambek’s early work on Latin and French conjugation.

- His work was further developed and extended by Lambek and his co-authors to Turkish, Arabic and (Biblical) Hebrew.

- Lambek’s approach was lately applied to Japanese.
Fundamental papers

- J. Lambek,
  A mathematician looks and French conjugation.

- J. Lambek,
  A mathematician looks and Latin conjugation.

- S. Ghilezan,
  Conjugation in SerboCroatian.
The Verb - Simple Tenses

- Inflected languages are languages that changes the form or ending of some words when the way in which they are used in sentences changes.
- Latin, Greek, Sanskrit, Polish, Serbian, Croatian, Hungarian and Finnish are all highly inflected languages.
- English is weakly inflected

Lambek’s method considers only simple tenses and disregards compound tenses.

<table>
<thead>
<tr>
<th>Language</th>
<th>Inflected forms</th>
<th>Patterns</th>
<th>Simple tenses $\times$ Persons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latin</td>
<td>90</td>
<td>3</td>
<td>$5 \times 6$</td>
</tr>
<tr>
<td>French</td>
<td>42</td>
<td>1</td>
<td>$7 \times 6$</td>
</tr>
<tr>
<td>Serbian</td>
<td>24</td>
<td>1(2)</td>
<td>$4 \times 6$</td>
</tr>
<tr>
<td>Hebrew B</td>
<td>140</td>
<td>7</td>
<td>$2 \times 10$</td>
</tr>
<tr>
<td>Spanish</td>
<td>54</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Lambek’s production grammar

- Lambek’s production grammar is a simple computational method for generating these conjugational forms step by step.

- The mathematical structure involved is:
  - finitely generated partially ordered semi-group or
  - semi-Thue system in mathematics
  - rewriting system in computer science
  - production grammar or
  - Chomsky’s Type zero language in linguistics
Lambek’s production grammar

- With each verb \( V \), there is associated a \( n \times m \times p \) matrix of conjugational verb-forms, \( C_{ij}^k(V) \)
  - \( i = 1, \ldots, n \) represents the (simple) tense
  - \( j = 1, \ldots, m \) represents the person-number
  - \( k = 1, \ldots, p \) represents the pattern

- A production grammar, in general, provides a method for calculating \( C_{ij}^k(V) \) for a given \((i, j, k, V)\).
Aspects - one pattern

Most verbs in Serbian and Croatian have two aspects:

▶ **imperfective** - the action is still in progress or it is being repeated
▶ **perfective** - the action has been completed, or that it is limited

Example. The two aspects of the verb *gledati*:

▶ *gledati* - “to look” (imperfective),
▶ *pogledati* - “to have a look” (perfective).

Interconnection:

▶ imperfective may become perfective by means of a prefix.
▶ perfective may become imperfective by:
  ▶ changing the tone (4 tones not written)
  ▶ dropping the prefix and prolonging and changing the root of the word.

One pattern for conjugation - both aspects use the same pattern.
Four groups of rules

1. Stem
2. Tense
3. Person-Number
4. Morphographemic
1. The Stem

$Stem_1$ and $Stem_2$ are operations which assign to each verb its present and infinitive (aorist) stem.

- $Stem_1$ assigns to each verb the third person singular of the present tense
- $Stem_2$ of a verb is obtained from the first person singular of the aorist of the verb by leaving out the ending $oh$ or $h$ when preceded by another vowel.
- $Stem_3$, $Stem_4$.

<table>
<thead>
<tr>
<th>Stem</th>
<th>$Stem_1$</th>
<th>$Stem_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>gledati</td>
<td>gleda</td>
<td>gleda (to look)</td>
</tr>
<tr>
<td>zvati</td>
<td>zove</td>
<td>zva (to call)</td>
</tr>
<tr>
<td>voleti</td>
<td>voli</td>
<td>vole (to love)</td>
</tr>
</tbody>
</table>
2. The Tense Signs

Each aspect has the following simple tenses:

\[ T_1 = \text{present tense}, \]
\[ T_2 = \text{aorist}, \]
\[ T_3 = \text{imperfect}, \]
\[ T_4 = \text{future}. \]
In order to obtain the infix corresponding to the appropriate tense we stipulate the following 4 rules (2a-2d):

2a. $T_1 \rightarrow \emptyset$,

2b. $T_2 \rightarrow \begin{cases} os_1, & \text{after a consonant,} \\ s_2, & \text{otherwise} \end{cases}$

2c. $T_3 \rightarrow \begin{cases} *\text{as}, & \text{after } a, \\ *\text{jās}, & \text{after } ne, je \text{ and } i, \\ *\text{ijās}, & \text{otherwise} \end{cases}$

2d. $T_4 \rightarrow \dagger\text{c}$. 
3. The Person-Number Signs

We present the six persons as follows:

Singular

\[ P_1 = \text{first person} \]
\[ P_2 = \text{second person} \]
\[ P_3 = \text{third person} \]

Plural

\[ P_4 = \text{first person} \]
\[ P_5 = \text{second person} \]
\[ P_6 = \text{third person} \]

In order to obtain the suffix corresponding to the appropriate person-number we stipulate the following 6 rules (3a-3f):
3a. $P_1 \rightarrow \begin{cases} 
    m, & \text{after a vowel,} \\
    u, & \text{after } \acute{c} \text{ or } g, \\
    \dagger h, & \text{otherwise.}
\end{cases}$

3b. $P_2 \rightarrow \begin{cases} 
    \check{\acute{s}}, & \text{after a vowel,} \\
    +e\check{\acute{s}}, & \text{after } \acute{c} \text{ or } g, \\
    \oplus(e), & \text{otherwise.}
\end{cases}$

3c. $P_3 \rightarrow \begin{cases} 
    \emptyset, & \text{after a vowel,} \\
    \oplus(e), & \text{otherwise.}
\end{cases}$

3d. $P_4 \rightarrow \begin{cases} 
    +\text{emo}, & \text{after } \acute{c} \text{ or } g, \\
    \text{mo, otherwise.}
\end{cases}$

3e. $P_5 \rightarrow \begin{cases} 
    +\text{ete}, & \text{after } \acute{c} \text{ or } g, \\
    \text{te, otherwise.}
\end{cases}$

3f. $P_6 \rightarrow \begin{cases} 
    *u, & \text{after } a, e, s \text{ or } g, \\
    *e, & \text{otherwise}
\end{cases}$
In order to take into consideration the morphographemic aspect of the language needed for generation, we stipulate the following 4 groups of rules (4a-4d):

4b. The second group of rules:

\[
\begin{align*}
\ & s^+ e \to \acute{s}e, \\
\ & \acute{c}^+ u \to ku, \\
\ & k^+ e \to \acute{c}e, \\
\ & g^+ e \to \zh e.
\end{align*}
\]
Verb Forms and One Word Sentences

The form $C_{ij}$ of a verb can be generated by the following rule:

$$C_{ij}(V) \rightarrow Stem_i(V) T_i P_j.$$  

We proceed from left to right by the application of rules.

As in Latin, each conjugational verb form can be regarded as one-word sentence: “plivam” means “I swim”.

These sentences $S$ are generated in the following way:

$$S \rightarrow C_{ij}(V).$$  

This rule converts a sentence $S$ into a verb form via $Stem_i(V) T_i P_j$. 
\[ C_{16}(brinuti) \rightarrow Stem_1(brinuti)T_1P_6 \]
\[ \rightarrow brineT_1P_6 \quad (1a) \]
\[ \rightarrow brineP_6 \quad (2a) \]
\[ \rightarrow brine \ast u \quad (3f) \]
\[ \rightarrow brinu \quad (4a) \]
Implementation

- A **Prolog** implementation program of the present work has been developed by Miroslav Martinović.

- The main predicate (logic grammar symbol)

  \[\text{conjugate} \,(\text{Inf, Tense, Per-Num, Form})\]

  \text{Inf, Tense, Per-Num} \text{ are given.}
  \text{Form} \text{ is to be derived.}

- A sort of lexicon needed for this program also contains \('Stem_1'\) and \('Stem_2'\) forms of the verbs. A version of this program is envisaged in which both of these operations under consideration will be implemented.
Further developments of Lambek’s method

D. Bargelli, J. Lambek,
A Computational View of Turkish Conjugation

D. Bargelli, J. Lambek,
A Computational Approach to Arabic Conjugation

C. Casadio, B. Coecke, M. Moortgat, P. Scott, P. (Eds.)
Categories and Types in Logic, Language, and Physics
Essays dedicated to Jim Lambek on the Occasion of his 90th Birthday.
Lecture Notes in Computer Science 8222 Subseries:

Kumi Cardinal
An algebraic study of Japanese grammar