# Computational Linguistics: Formal Semantics, Compositionality (Abstraction)

Raffaella Bernardi

University of Trento

# Contents

# 1. Seen to be done

We have seen how to

- represent linguistic expressions as functions. Viz, $\lambda$-terms
- compose linguistic expressions, viz. $\lambda$-calculus. Function application and $\beta$ conversion

Today we are going to see how to:

- obtain MR of structure containing long-distance dependency.
- obtain MRs of ambiguos sentences.

# 2. Recall: Lambda Calculus

Lambda calculus was introduced by Alonzo Church in the 1930s as part of an investigation into the foundations of mathematics.

- It has a **variable binding operator** $\lambda$. Occurrences of variables bound by $\lambda$ should be thought of as place-holders for missing information: they explicitly mark where we should substitute the various bits and pieces obtained in the course of semantic construction.
- Function can be applied to argument (**Function application**)
- An operation called $\beta$**-conversion** performs the required substitutions.
- Variables can be abstracted from a term (**Abstraction**)

# 3. Relative Pronouns

For instance, "which John read [. . .]":

We know how to represent the noun phrase "John" and the verb "read", namely, as $\texttt{john}$ and $\lambda x.\lambda y.\texttt{read}(y, x)$.

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

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.

The meaning of "student who lori knows" is a set of entities.

"who" creates the intersection between the set of students and the set of those people who lori knows:

$$[\![N\ who\ VP]\!] = [\![\texttt{N}]\!] \cap [\![\texttt{VP}]\!]$$

## 3.1. Relative Pronouns

$$
\begin{array}{lcl}
[\![\texttt{lori}']\!] & = & \text{lori; } \ldots \\
[\![\lambda x.\texttt{student}'(x)]\!] & = & \{\text{lori, alex, sara}\}; \\
[\![\lambda x.\lambda y.\texttt{know}'(y,x)]\!] & = & \{(\text{lori, alex}), (\text{lori, pim}), (\text{sara, alex})\} \\
[\![\lambda \texttt{y}.\texttt{know}'(\texttt{y},\texttt{alex}')]\!] & = & \{\text{lori, sara}\} \\
[\![\lambda x.\texttt{know}'(\texttt{lori}',\texttt{x})]\!] & = & \{\text{alex, pim}\}
\end{array}
$$

The meaning of "student who lori knows" is a set of entities: {alex}.

"who" creates the intersection between the set of students and the set of those people who lori knows:

$$[\![N \ who \ VP]\!] = [\![\texttt{N}]\!] \cap [\![\texttt{VP}]\!]$$

"who": $\lambda VP.\lambda N.\lambda x.N(x) \wedge VP(x)$

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

The double role of "who" (or "which") is expressed by the double occurrence of $z$.

# 4. Quantifiers

A Generalized Quantifier (GQ) is a set of properties, i.e. **a set 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"? What do we know that holds for every man?

"Every man is man", "Every man is fat"

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

## 4.1. Generalized Quantifiers

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

Therefore, determiners are as below:

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

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

Which is the lambda term representing quantifiers like “nobody”, “everybody”, “a man”or “every student” or a determiner like “a”, “every” or “no” ?

## 4.2. Determiners (Cont'd)

Let's start from what we have, namely "man" and "loves Mary":

$\lambda y.man(y)$, $\lambda x.love(x, mary)$.

Hence, the term representing "a" is:

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

Try to obtain the meaning representation for "a man", and the "a man loves Mary".

By $\beta$-conversion twice we obtain that "a man" is $\lambda Y.\exists z.\texttt{Man}(z) \wedge Y(z)$, and then $\exists z.\texttt{Man}(z) \wedge love(z, mary)$

## 4.3. Lambda terms and types

| word | type | term | meaning |
|---|---|---|---|
| "lori" | $e$ | $\mathtt{l}$ | lori |
| "walks" | $(e \to t)$ | $\lambda x_e.(\mathtt{walks}(x))_t$ | {lori} |
| "teases" | $(e \to (e \to t))$ | $\lambda y_e.(\lambda x_e.(\mathtt{teases}(x, y))_t)$ | {(lori, alex)} |

Complete the table with "which", "everyone" and "every".

# 5. Exercise

Build the MR for the following sentences:

- Someone left
- Everyone left
- None left
- Some student left
- Every student left
- No student left
- Some smart student left
- Some student who knows Mary left

# 6. Abstraction

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

1. read u: $\lambda y(\texttt{read}(y, u))$

2. John read u: $\texttt{read}(\texttt{j}, u)$

3. John read: $\lambda u.\texttt{read}(\texttt{j}, u)$ Abstraction

4. which John read:
   $\lambda Y.\lambda z.Y(z) \wedge \texttt{read}(\texttt{j}, z)$

5. book which John read $\lambda z.\texttt{Book}(z) \wedge \texttt{read}(\texttt{j}, z)$

Recall:

$$[\![Book]\!] \cap [\![\text{John read}]\!]$$

# 7. Scope ambiguity

"All students passed an exam"

- One syntactic tree.
- Two meaning representations.
  - $Q_{obj}\ \lambda x.Q_{sub}\ \lambda y.TV(y,x)$ $\qquad OBJ > SUBJ$
  - $Q_{sub}\ \lambda y.Q_{obj}\ \lambda x.TV(y,x)$ $\qquad SUB > OBJ$

by replacing the place holders ($Q_{obj}$, $Q_{sub}$, $TV$) with their MR, one obtains the MR of the sentence.

## 7.1. $SUB > OBJ$

All students passed an exam

1. an exam: $\lambda X.\exists x.Exam(x) \wedge X(x)$
2. all students: $\lambda X.\forall x.Student(x) \rightarrow X(x)$
3. passed u: $\lambda y.\texttt{passed}(y, u)$
4. z passed u: $\texttt{passed}(z, u)$

5. z passed: $\lambda u.\texttt{passed}(z, u)$ Abs. obj
6. z passed an exam: $\exists x.Exam(x) \wedge passed(z, x)$
7. passed an exam $\lambda z.\exists x.Exam(x) \wedge passed(z, x)$ Abs. sub
8. all student passed an exam: $\forall x.Student(x) \rightarrow \exists y.Exam(y) \wedge passed(x, y)$

$\forall\exists$.

## 7.2. $OBJ > SUB$

All students passed an exam

1. an exam: $\lambda X.\exists x.Exam(x) \wedge X(x)$
2. all students: $\lambda X.\forall x.Student(x) \rightarrow X(x)$
3. passed u: $\lambda y.\texttt{passed}(y, u)$
4. z passed u: $\texttt{passed}(z, u)$
5. passed u: $\lambda z.\texttt{passed}(z, u)$ Abs. sub
6. all students passed u: $\forall x.Student(x) \rightarrow passed(x, u)$
7. all students passed $\lambda u.\forall x.Student(x) \rightarrow passed(x, u)$ Abs. obj
8. all student passed an exam: $\exists y.Exam(y) \wedge \forall x.Student(x) \rightarrow passed(x, y)$

$\exists\forall$ More on this with RZ.