Computational Linguistics: Syntax-Semantics II

Raffaella Bernardi

KRDB, Free University of Bozen-Bolzano

P.zza Domenicani 3, Room: 2.19, e-mail:

bernardi@inf.unibz.it

Contents

1 Recall: Formal Semantics Main questions 4
1.1 Building Meaning Representations 5
1.2 Lambda-calculus: Functional Application 6
2 Extending the lexicon 7
2.1 Quantified NP 8
2.2 Generalized Quantifiers 9
2.3 Generalized Quantifiers 10
2.4 Determiners (Cont'd) 11
3 Dependencies 12
3.1 Relative Pronouns 13
3.2 Relative Pronoun (Cont'd) 14
4 Ambiguities 15
4.1 Scope Ambiguities 16
5 Summing up: Constituents and Assembly 17
6 Syntax-Semantics: Parallel vs. Non-parallel 18
6.0.1 Advantages 19
6.1 Montague Universal Grammar 20
7 Practical info 21

1. Recall: 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?

1.1. Building Meaning Representations

To build a meaning representation we need to fulfill three tasks:

Task 1 Specify a reasonable **syntax** for the natural language fragment of interest.

Task 2 Specify semantic representations for the **lexical items**.

Task 3 Specify the **translation** of constituents **compositionally**. That is, we need to specify the translation of such expressions in terms of the translation of their parts, parts here referring to the substructure given to us by the syntax.

Moreover, when interested in Computational Semantics, all three tasks need to be carried out in a way that leads to computational implementation naturally.

1.2. Lambda-calculus: Functional Application

Summing up:

- FA has the form: Functor(Argument). E.g. $(\lambda x.love(x, mary))(john)$
- FA triggers a very simple operation: Replace the λ-bound variable by the argument. E.g. $(\lambda x.love(x, mary))(john) \Rightarrow love(john, mary)$

Exercise 1 and 2.

2. Extending the lexicon

Before we have left open the question of what does an expression like “a” contribute to?

FOL does not give us the possibility to express its meaning representation.

We will see now that instead lambda terms provide us with the proper expressivity.

2.1. Quantified NP

a) Every female student of the EM in LCT attends the Comp Ling course.

b) No female student of the EM in LCT attend the Logic course.

a) means that if Quynh constitute the set of the female students of the EM in LCT, then it is true that she attend the Comp. Ling course.

b) means that for none of the individual members of the set of female students of the EM in LCT it is true that she attends the Logic course.

What is the interpretation of "every female student" and of "no female 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.

2.2. Generalized 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"?

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

2.3. Generalized Quantifiers

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

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.

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

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

3. Dependencies

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 $\lambda x.\lambda y.like(y, x)$ we express that the dependency of the subject and object from the verb.

But the calculus gives us also a natural way to handle long-distance dependencies: eg. relative pronouns.

3.1. Relative Pronouns

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

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

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:

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

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

3.2. Relative Pronoun (Cont'd)

Recall,

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

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

- 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.

Note, we use another operation of the λ-calculus: **Abstraction**!

4. Ambiguities

How many meanings has the sentence "John didn't read a book."?

Starting from:

john: $\mathtt{j}$	book: $\lambda x(\mathtt{book}(x))$
read: $\lambda x.\lambda y.(\mathtt{read}(y,x))$	didn't: $\lambda X.\lambda y.\neg X(y)$
a: $\lambda X.\lambda Y(\exists x.X(x) \wedge Y(x))$	

build the meaning representation for "John didn't read a book".

a. $\exists x.\mathtt{book}(x) \wedge \neg\mathtt{read}(\mathtt{j},x)$ [A > NOT]

b. $\neg\exists x.\mathtt{B}(x) \wedge \mathtt{read}(\mathtt{j},x)$ [NOT > A]

- **Scope**: In a. the quantifier phrase (QP), "a book", has scope over "didn't" [A > NOT], whereas in b. it has narrow scope [NOT > A].
- **Binding**: the variable x is bound by "a book" in "John didn't read a book".

4.1. Scope Ambiguities

Can you think of other expressions that may cause scope ambiguity?

John **think** a student left

Does the student exist or not?

a. $\exists x.think(j, left(x))$

b. $think(j, \exists x.left(x))$

Every student passed an exam?

5. Summing up: Constituents and Assembly

Let's go back to the points where FOL fails, i.e. constituent representation and assembly. The λ-calculus succeeds in both:

Constituents: each constituent is represented by a lambda term.

John: $\mathtt{j}$ knows: $\lambda xy.(\mathtt{know}(x))(y)$ read john: $\lambda y.\mathtt{know}(y, \mathtt{j})$

Assembly: function application ($\alpha(\beta)$) and abstraction ($\lambda x.\alpha[x]$) capture composition and decomposition of meaning representations.

6. Syntax-Semantics: Parallel vs. Non-parallel

We could build the meaning representation of an expression either

(a) in parallel with the construction of its syntactic structure, or

(b) after having built the syntactic analysis.

(a) is the method followed by most formal grammar frameworks as Categorial Grammar (CG), Head-Driven Phrase Structure Grammar (HPSG), Lexical Functional Grammar (LFG), Tree-Adjoining Grammar (TAG).

(b) is used by the Government and Binding Theory and the Minimalist Program (both due to Chomsky).

6.0.1. Advantages The reasons for preferring the first approach are the following:

Psycholinguistic works suggest that human processing proceeds incrementally through the simultaneous application of syntactic, semantics, and phonological constraints to resolve syntactic ambiguity. (Though, note that these systems are models of linguistic competence rather than performance. Hence, these results could not provide direct support of either of the approaches.)

Computational approach requires a way to rule out a semantically ill-formed phrase as soon as it is encountered. Therefore, (a) offers a more efficient architecture for implementing constraint satisfaction. For instance,

1. The delegates met for an hour.
2. The committee met for an hour.
3. *The woman met for an hour.

The use of "met" as intransitive verb requires a subject denoting a plural entity.

6.1. Montague Universal Grammar

The rule-to-rule and lambda techniques are used in the approach to natural language semantics developed by Richard Montague. In his theory, there are

- **syntactic rules** which show how constituents maybe combined to form other constituents.
- **translation rules** (associated with each such syntax rule) which show how the logical expressions for the constituents have to be joined together to form the logical form of the whole.

For instance, the syntactic and semantics rule for composing and NP with and IV:

S2: If $\delta \in P_{IV}$ and $\alpha \in P_{NP}$, then $F_1(\alpha, \delta) \in P_S$ and $F_1(\alpha, \delta) = \alpha\delta'$, where δ' is the result of replacing the main verb in δ by its third-person singular present form.

T2: If $\delta \in P_{IV}$ and $\alpha \in P_{NP}$ and $\delta | \rightarrow \delta'$ and $\alpha | \rightarrow \alpha'$, then $F_1(\alpha, \delta) | \rightarrow \alpha'(\delta')$.

As grammar, he used Categorial Grammar. We will look at it next time.

7. Practical info

Next meetings:

- 28/10/10, 10:30-12:30 Lexical Semantics, with Elena. (watch for the room!)
- 03/11/10, 17:00-18:00 WordNet, with Elena.
- 04/11/10, 10:30-12:30 Textual Entailiment, with Elena. (watch for the room!)
- 10/11/10, 17:00-18:00, Reading group on TE, with Elena.
- 11/11/10, 10:30-12:20, Critiques by students!!!

We will come back to the topics of today on the 17/11/10: Lab on lambda calculus.