Computational Linguistics: Syntax-Semantics Interface

RAFFAELLA BERNARDI

KRDB, Free University of Bozen-Bolzano

P.zza Domenicani, Room: 2.28, e-mail: bernardi@inf.unibz.it
$\textbf{Contents}$

1  The Syntax-Semantic Interface ...........................................  4
   1.1 Parallel vs. Non-parallel ...........................................  5
       1.1.1 Advantages .............................................  6
   1.2 Compositionally vs. Non-compositionally .........................  7
2  Lambda terms and DCG ...................................................  8
   2.1 Augumenting DCG with terms .......................................  9
       2.1.1 Exercise ................................................  9
   2.2 Quantified NP: class ............................................. 11
   2.3 Quantified NP: terms and syntactic rules ......................... 12
   2.4 Quantified NP and Proper Nouns .................................. 13
3  Model .................................................................. 14
   3.1 Quantified NP .................................................... 15
   3.2 Relational and Functional Perspectives .......................... 16
   3.3 Summing up ..................................................... 17
4  Extending the fragment of English .................................... 18
5  Historical Introduction: Meaning Assembly .......................... 19
6  Montague Universal Grammar ........................................... 20
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Montague Grammar (Cont’d)</td>
<td>21</td>
</tr>
<tr>
<td>6.2</td>
<td>Syntactic and Translation Rules: Example</td>
<td>22</td>
</tr>
<tr>
<td>6.3</td>
<td>Quantified NP in Montague Grammar</td>
<td>23</td>
</tr>
<tr>
<td>6.4</td>
<td>Quantified NP: Example</td>
<td>24</td>
</tr>
<tr>
<td>6.5</td>
<td>Montague Grammar: Key Points</td>
<td>26</td>
</tr>
<tr>
<td>7</td>
<td>Categorial Grammar</td>
<td>27</td>
</tr>
<tr>
<td>8</td>
<td>CG: Syntactic Rules</td>
<td>28</td>
</tr>
<tr>
<td>9</td>
<td>CG Lexicon: Toy Fragment</td>
<td>29</td>
</tr>
<tr>
<td>10</td>
<td>Classical Categorial Grammar</td>
<td>30</td>
</tr>
<tr>
<td>11</td>
<td>Classical Categorial Grammar. Examples</td>
<td>31</td>
</tr>
<tr>
<td>11.1</td>
<td>Relative Pronoun</td>
<td>32</td>
</tr>
<tr>
<td>11.2</td>
<td>CFG and CG</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>CG: syntax-semantics interface</td>
<td>35</td>
</tr>
<tr>
<td>12.1</td>
<td>Mapping: types-categories</td>
<td>36</td>
</tr>
<tr>
<td>12.2</td>
<td>CG: categories and terms</td>
<td>37</td>
</tr>
<tr>
<td>13</td>
<td>Next Time</td>
<td>38</td>
</tr>
</tbody>
</table>
1. The Syntax-Semantic Interface

In the first part of the course, we have spoken of syntax and semantics of natural language as two distinct and separate levels. However, as we know from our everyday use of NL these levels are tiedely connected.

Today, we will look at the interface between syntax and semantic.

Recall, from syntax we know that phrases are composed out of words, and from semantics we know that meaning flows from the lexicon.

1.1. 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).
1.1.1. 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.

The use of “met” as intransitive verb requires a subject denoting a plural entity.
1.2. Compositionally vs. Non-compositionally

- In **compositional** semantics theory the relation between the meaning of an expression and the meaning of its constituents is a **function**: to each distinct syntactic structure correspond a distinct interpretation.

- In **underspecification** theory this relation is systematic but it’s **not a function**: an expression analyzed by a single syntactic structure can be associated with a set of alternative interpretations rather than with a unique semantic value. Sentences are assigned underspecified representation containing parameters whose value can be defined in several distinct ways. Constraints apply to filter the possible combinations of values for the sent of parameters in such a schematic representation.

Reference: For underspecified semantics see BB1.
2. Lambda terms and DCG

We will look at the compositional approach to the syntax-semantics interface and build the meaning representation in parallel to the syntactic tree. This reduces to have a rule-to-rule system, i.e. each syntactic rule correspond to a semantic rule.

**Syntactic Rule 1** \( S \rightarrow NP \ VP \)

**Semantic Rule 1** If the logical form of the \( NP \) is \( \alpha \) and the logical form of the \( VP \) is \( \beta \) then the logical form for the \( S \) is \( \beta(\alpha) \).

**Syntactic Rule 2** \( VP \rightarrow TV \ NP \)

**Semantic Rule 2** If the logical form of the \( TV \) is \( \alpha \) and the logical form of the \( NP \) is \( \beta \) then the logical form for the \( VP \) is \( \alpha(\beta) \).
2.1. Augumenting DCG with terms

That can also be abbreviated as below where $\gamma, \alpha$ and $\beta$ are the meaning representations of $S, NP$ and $VP$, respectively.

$$S(\gamma) \rightarrow NP(\alpha) \ VP(\beta) \ \gamma = \beta(\alpha)$$

This implies that lexical entries must now include semantic information. For instance, a way of writing this information is as below.

$$TV(\lambda x.\lambda y.\text{wrote}(y, x)) \rightarrow [\text{wrote}] :$$

2.1.1. Exercise (a) Write the semantic rules for the following syntactic rules:

- $s \rightarrow np \ vp$
- $vp \rightarrow iv$
- $vp \rightarrow tv \ np$
- $np \rightarrow det \ n$
n --> adj n
n --> student
det --> a
adj --> tall

(b) apply these labeled rules to built the partial labeled parse trees for “A student” and “A tall student”.
2.2. **Quantified NP: class**

In attempting to extend the technique of compositional semantics we run into problems with e.g. the rule for quantified noun phrases (QP).

QP should belong to the same category of noun phrases, as suggested by the substitution test or the coordination test. E.g.

```
1. I will bring here *every student* and Mary.
2. I will bring here *John* and Mary.
```
2.3. Quantified NP: terms and syntactic rules

We have seen that the term of a quantifier like “every student” is $\lambda Y.\forall z.\text{Student}(z) \rightarrow Y(z)$, which is of type $(e \rightarrow t) \rightarrow t$. Hence the sentence,

Every student left.

is obtained by applying the quantified noun phrase to the verb. In other words, if “Everybody” is of category $NP$ we need the rule below:

$$S(\gamma) \rightarrow NP(\alpha) \ VP(\beta) \ \gamma = \alpha(\beta)$$
2.4. Quantified NP and Proper Nouns

This has brought semanticists to change the meaning representation of the noun phrase too, since they have to be of the same “sort”. E.g, “John” could be represented as

\[ \lambda X. X(john) \]

a function of the same type as the quantified NP, i.e. \((e \rightarrow t) \rightarrow t\).

But how is this possible? Are we free of changing the meaning representation of words?
3. Model

Let our model be based on the set of entities $E = \{\text{lori, ale, sara, pim}\}$ 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 $[w]$ indicate the interpretation of $w$:

- $[\text{sara}] = \text{sara}$;
- $[\text{pim}] = \text{pim}$;
- $[\text{lori}] = \text{lori}$;
- $[\text{know}] = \{\langle\text{lori, ale}\rangle, \langle\text{ale, lori}\rangle, \langle\text{sara, lori}\rangle,$
  $\langle\text{lori, lori}\rangle, \langle\text{ale, ale}\rangle, \langle\text{sara, sara}\rangle, \langle\text{pim, pim}\rangle\}$;
- $[\text{student}] = \{\text{lori, ale, sara}\}$;
- $[\text{professor}] = \{\text{pim}\}$;
- $[\text{tall}] = \{\text{lori, pim}\}$.

which is nothing else to say that, for example, the relation know is the set of pairs $\langle\alpha, \beta\rangle$ where $\alpha$ knows $\beta$; or that ‘student’ is the set of all those elements which are a student.
3.1. Quantified NP

Logical constants are interpreted by using set theoretical operations as illustrated below.

By evaluating the lambda expressions in a model, one obtains the interpretations below:

\[
\begin{align*}
    \text{[no N]} & = \{X \subseteq E \mid [N] \cap X = \emptyset\}. \\
    \text{[some N]} & = \{X \subseteq E \mid [N] \cap X \neq \emptyset\}. \\
    \text{[every N]} & = \{X \subseteq E \mid [N] \subseteq X\}. \\
    \text{[N which VP]} & = [N] \cap [VP].
\end{align*}
\]

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

Alternatively, one can assume a functional perspective and interpret, for example, \texttt{know} as a function \( f : Dom_e \rightarrow (Dom_e \rightarrow Dom_t) \).

The shift from the relational to the functional perspective is made possible by the fact that the \textbf{sets and their characteristic functions amount to the same thing}:

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

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

**Question** So, what would it mean that e.g. “John” is represented as \( \lambda X. X (\text{john}) \)? Which would be its domain of interpretation? Hence, what would be its interpretation in the relational perspective?
3.3. Summing up

Summarizing, when trying to formalize natural language semantics, at least two sorts of objects are needed to start with: the set of *truth values* $t$, and the one of *entities* $e$.

Moreover, we spoke of more complex objects as well, namely functions. More specifically, we saw that the kind of functions we need are *truth-valued functions* (or boolean functions).

Furthermore, we have illustrated how one can move back and forwards between a *set/relational and a functional perspective*. The former can be more handy and intuitive when reasoning about entailment relations among expressions; the latter is more useful when looking for lexicon assignments.

References: Keenen 85.
4. Extending the fragment of English

**Question** How did we build the meaning representation of sentences with a quantifier in object position? Can we augment the DCG rules so to handle these cases too?

We could consider $NP$ (both proper names and quantified NP as function of type $((e \rightarrow t) \rightarrow t)$ and transitive verbs as function: $((e \rightarrow t) \rightarrow t) \rightarrow (e \rightarrow t)$.

Still this solution won’t give us the reading with quantifier in object position having wide scope, e.g. Every woman loves one man. $\exists x. M(x) \land (\forall y. W(y) \rightarrow L(y, x))$. 
5. Historical Introduction: Meaning Assembly

We have seen that besides syntactic concatenation of strings, a fundamental aspect of natural language is meaning assembly.

► **Who:** Montague (1973)

► **Aim:** To build a syntax-semantic interface.

► **How:** Concatenation of strings is seen as function application; syntactic categories are mapped into semantic types/typed lambda terms, and syntactic rules are mapped into semantic rules.
6. **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.
6.1. Montague Grammar (Cont’d)

Whereas a syntactic rule will deal with left-to-right order of items, verb-agreement, and other grammatical matters, the translation rule will define how the corresponding 'semantic' values have to be operated upon.

The syntactic rules used by Montague are more powerful than simple PSG rules, in that they are allowed to perform virtually any computation in constructing their results, including the substituting of values for variables.

As for the syntactic rules, Montague uses the idea at the heart of Categorial Grammar of considering syntactic categories as function that are in a many-to-one correspondence to types. (We will come back to this later.)
6.2. Syntactic and Translation Rules: Example

For instance, the syntactic rule for composing a term (e.g. “John”) with an intransitive verb is:

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

the semantic representation is built by means of the corresponding translation rule

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

\[ S7: \text{If } \delta \in P_{TV} \text{ and } \alpha \in P_{NP}, \text{ then } F_6(\delta, \alpha) \in P_{IV} \text{ and } F_6(\delta, \alpha) = \delta\alpha', \text{ where } \alpha' \text{ is the accusative form of } \alpha \text{ if } \alpha \text{ is a syntactic variable; otherwise } \alpha' = \alpha. \]

\[ T7: \text{If } \delta \in P_{TV} \text{ and } \alpha \in P_{NP} \text{ and } \delta \rightarrow \delta' \text{ and } \alpha \rightarrow \alpha', \text{ then } F_6(\delta, \alpha) \rightarrow \delta'(\alpha'). \]

Notice, Montague considers a TV to be of type \(((e \rightarrow t) \rightarrow t) \rightarrow (e \rightarrow t)\)
6.3. **Quantified NP in Montague Grammar**

In order to deal with scope ambiguities, Montague proposed a rule of quantification that uses syntactic variables: expression of the form $he_n$ and category $NP$.

So to build a sentence with a quantifier, the quantification rule will start from sentences built by means of syntactic variables first and then quantified.

$S_8$: If $\alpha \in P_{NP}$ and $\phi \in P_S$, then $F_n(\alpha, \phi) \in P_S$, and $F_n(\alpha, \phi) = \phi'$, where $\phi'$ is the result of the following substitution in $\phi$:

i. If $\alpha$ is not a syntactic variable $he_k$, then replace the first occurrence of $he_n$ or $him_n$ with $\alpha$, and the other occurrences of $he_n$ or $him_n$ with appropriate anaphoric pronouns;

ii. if $\alpha = he_k$, then replace every occurrence of $he_n$ with $he_k$ and of $him_n$ with $him_k$.

$T_8$: If $\alpha \in P_{NP}$ and $\phi \in P_S$ and $\alpha \rightarrow \alpha'$ then $F_n(\alpha, \phi) \rightarrow \alpha'(\lambda x_n.\phi')$. 
6.4. Quantified NP: Example

EVERY > ONE

Every woman loves one man, S, S2
    / \
every woman, NP, S3  loves one man, IV, S7
     |  /
woman, N  love, TV  one man, NP, S6
      | 
      man, N

If you add the lambda terms to this tree by means of the translation rules corresponding to the syntactic rules in each node, you will obtain the meaning representation of “Every woman loves one man” with “one man” having narrow scope.
If you add the lambda terms to this tree by means of the translation rules corresponding to the syntactic rules in each node, you will obtain the meaning representation of “Every woman loves one man” with “one man” having **wide scope**.
6.5. Montague Grammar: Key Points

- Correspondence between syntactic categories and semantics types.
- Correspondence between syntactic rules and semantics rules.
- Functional application play a crucial role both at the syntactic and semantic level.
- Both at syntactic and semantic level there is the need of “abstract” a variable from a structure.
7. Categorial Grammar

▶ **Who:** Lesniewski (1929), Ajdukiewicz (1935), Bar-Hillel (1953).

▶ **Aim:** To build a language recognition device.

▶ **How:** Linguistic strings are seen as the result of concatenation obtained by means of syntactic rules starting from the categories assigned to lexical items. The grammar is known as Classical Categorial Grammar (CG).

▶ **Connection with Type Theory:** The syntax of type theory closely resembles the one of categorial grammar. The links between types (and lambda terms) with models, and types (and lambda terms) with syntactic categories, gives an interesting framework in which syntax and semantic are strictly related. (We will come back on this later.)

**Categories:** Given a set of basic categories \( p \), the set of categories \( \text{CAT} \) is the smallest set such that:

\[
\text{CAT} := p \mid \text{CAT}\text{\setminus}\text{CAT} \mid \text{CAT}/\text{CAT}
\]
8. **CG: Syntactic Rules**

Categories can be composed by means of the syntactic rules below

**[BA]** If $\alpha$ is an expression of category $A$, and $\beta$ is an expression of category $A/B$, then $\alpha\beta$ is an expression of category $B$.

**[FA]** If $\alpha$ is an expression of category $A$, and $\beta$ is an expression of category $B/A$, then $\beta\alpha$ is an expression of category $B$.

where [FA] and [BA] stand for Forward and Backward Application, respectively.

![Diagram of syntactic rules](image-url)
9. **CG Lexicon: Toy Fragment**

Let $\mathbf{p}$ be \{n, s, np\} (for nouns, sentences and noun phrases, respectively) and \textbf{LEX} as given below. Recall PSG rules: \( np \rightarrow \text{det} \ n, s \rightarrow np \ vp, vp \rightarrow v \ np \ldots \)

**Lexicon**

- Sara  \( np \)
- the  \( np/n \)
- student  \( n \)
- walks  \( np/s \)
- wrote  \( (np/s)/np \)

Sara walks $\in s$? $\leadsto$ \( np, np/s \in s ? \) Yes

simply [BA]

```
\( s \)
\( \text{np} \quad \text{np/s} \)
\( \text{Sara} \quad \text{walks} \)
```
10. Classical Categorial Grammar

Alternatively the rules can be thought of as Modus Ponens rules and can be written as below.

\[
\begin{align*}
B/A, A & \Rightarrow B & \text{MP}_r \\
A, A \backslash B & \Rightarrow B & \text{MP}_l
\end{align*}
\]

\[
\begin{align*}
\frac{B/A}{A} & (\text{MP}_r) \\
\frac{A}{A \backslash B} & (\text{MP}_l)
\end{align*}
\]
11. Classical Categorial Grammar. Examples

Given $p = \{np, s, n\}$, we can build the following lexicon:

**Lexicon**

- John, Mary $\in np$
- student $\in n$
- walks $\in np/s$
- sees $\in (np/s)/np$

**Analysis**

- John walks $\in s$?
  - $\rightsquigarrow np, np/s \Rightarrow s$?
  - Yes
  - $np \quad np/s$
  - $s$ (MP$_1$)

- John sees Mary $\in s$?
  - $\rightsquigarrow np, (np/s)/np, np \Rightarrow s$?
  - Yes
  - $(np/s)/np \quad np$
  - $np \quad np/s$
  - $s$ (MP$_r$)
  - $s$ (MP$_1$)
11.1. Relative Pronoun

**Question** Which would be the syntactic category of a relative pronoun in subject position? E.g. “the student **who** knows Lori”

\[
\text{[the [[student]_n [who [knows Lori]_{np\_s}]]_n} \]

\[
\text{who knows Lori} \in n\_n? \quad \sim\Rightarrow \quad (n\_n)/(np\_s), (np\_s)/np, np \Rightarrow n\_n? \\
\]

\[
\frac{\text{who}}{(n\_n)/(np\_s)} \quad \frac{\text{knows}}{np\_s} \quad \frac{\text{Lori}}{np} \quad (\text{MP}_r) \\
\]

\[
\frac{\text{who}}{(n\_n)/(np\_s)} \quad \frac{\text{knows}}{np\_s} \quad \frac{\text{Lori}}{np} \quad (\text{MP}_r) \\
\]

\[
\frac{\text{who}}{(n\_n)/(np\_s)} \quad \frac{\text{knows}}{np\_s} \quad \frac{\text{Lori}}{np} \quad (\text{MP}_r) \\
\]
(n\n)/(np\s) who

(np\s)/np np

knows

Lori
11.2. **CFG and CG**

Below is an example of a simple CFG and an equivalent CG:

**CFG**

S --> NP VP  
VP --> TV NP  
N --> Adj N

**Lexicon:**  
Adj --> poor  
NP --> john  
TV --> kisses

**CG Lexicon:**  
John: \( np \)  
kisses: \( (np\backslash s)/np \)  
poor: \( n/n \)
12. **CG: syntax-semantics interface**

Summing up, CG specifies a language by describing the combinatorial possibilities of its lexical items directly, without the mediation of phrase-structure rules. Consequently, two grammars in the same system differ only in the lexicon.

The close relation between the syntax and semantics comes from the fact that the two syntactic rules are application of a functor category to its argument that corresponds to functional application of the lambda calculus.

We have to make sure that the lexical items are associated with semantic terms which correspond to the syntactic categories.
12.1. Mapping: types-categories

To set up the form-meaning correspondence, it is useful to build a language of semantic types in parallel to the syntactic type language.

Definition 12.1 (Types) Given a non-empty set of basic types $\text{Base}$, the set of types $\text{TYPE}$ is the smallest set such that

1. $\text{Base} \subseteq \text{TYPE}$;
2. $(a \rightarrow b) \in \text{TYPE}$, if $a$ and $b \in \text{TYPE}$.

Note that this definition closely resembles the one of the syntactic categories of CG. The only difference is the lack of directionality of the functional type $(a, b)$. A function mapping the syntactic categories into $\text{TYPE}$ can be given as follows.

Definition 12.2 (Categories and Types) Let us define a function $\text{type} : \text{CAT} \rightarrow \text{TYPE}$ which maps syntactic categories to semantic types.

- $\text{type}(np) = e$;
- $\text{type}(s) = t$;
- $\text{type}(n) = (e \rightarrow t)$;
- $\text{type}(A/B) = (\text{type}(B) \rightarrow \text{type}(A))$;
- $\text{type}(B\setminus A) = (\text{type}(B) \rightarrow \text{type}(A))$. 
12.2. CG: categories and terms

Modus ponens corresponds to functional application.

\[
\begin{align*}
B/A : t & \quad A : r \\
\frac{}{B : t(r)} & \quad (\text{MP}_r) \\
\end{align*}
\]

\[
\begin{align*}
A : r & \quad A \backslash B : t \\
\frac{}{B : t(r)} & \quad (\text{MP}_l)
\end{align*}
\]

Example

\[
\begin{align*}
np : \text{john} & \quad np \backslash s : \text{walk} \\
\frac{}{s : \text{walk}(\text{john})} & \quad (\text{MP}_l)
\end{align*}
\]

\[
np \backslash s : \lambda x.\text{walk}(x) & \quad (\lambda x.\text{walk}(x))(\text{john}) \sim_{\lambda-\text{conv.}} \text{walk}(\text{john})
\]

\[
\begin{align*}
(np \backslash s)/np : \text{know} & \quad np : \text{mary} \\
\frac{}{np : \text{john} \quad np \backslash s : \text{know}(\text{mary})} & \quad (\text{MP}_r) \\
\frac{}{s : \text{know}(\text{mary})(\text{john})} & \quad (\text{MP}_l)
\end{align*}
\]
13. **Next Time**

While working with the lambda-terms we have seen we need abstraction to, e.g. to account for the different ways quantified NP can scope.

But in Categorial Grammar there is no way to abstract from a built structure.

Next week we will see the missing ingredient (abstraction at syntactic level) allows us to move from a formal grammar to a logic (a **logical grammar**).

We will look at Lambek Calculi (or more generally, Categorial Type Logic) and their application to NL.