IA165 Combinatory Logic for Computational Semantics

Spring 2012

Juyeon Kang

gkang@fi.muni.cz

B410, Faculty of Informatics, Masaryk University, Brno, Czech Rep.

Application of the combinators to natural language analysis: basic

example of the formal semantic analysis some structures: cooridination

Summing up: last lecture

- · How to apply the combinators to natural language analysis
 - 1) using introduction and elimination rules by beta-reduction of combinators: control heurstic of combinatorial application and bracketing
 - 2) using a syntactic tool for controlling the application of combinators
 - : CCG assumes the preliminary steps to find a well-structured normal form, that is, a formal semantic structure

Example: use the combinators as a logical tool of semantic analysis

- a. apply directly the β -reduction rules of combinators
- b. use the CCG types and rules by integrating the β -reduction rules of combinators into the CCG rules

Remind 1... METHOD

- <u>Goal:</u> Formal semantic structure in term of operator and of operand
 bracketed expression written by convention: (operator(operand))
- Two methods of application of the combinators: both can be useful
 a. useful for defining some specific operators: passivisation,
 reflexivisation, quantification...
 - b. useful for general formal semantic analysis; can handle the analysis at the syntactic level and semantic level in one representation; informatic implementation more systemactic \rightarrow see the nexe slides "Remind 2 & 3"

Remind 2... TYPES

· CCG types

primitive types: S for sentence, NP for noun phrase, N for noun derived types: S/NP, N/N, N/N, (S/NP)/NP, NP/N...

Directionality: / (over) and \ (under)

a/b: a applies to b, a\b: b is applied to a

→ direction of application of operator to operand

Remind 3... RULES

1. Forward(>) and backward (<) functional application rules

2. Function composition (FC) rules with the combinator B

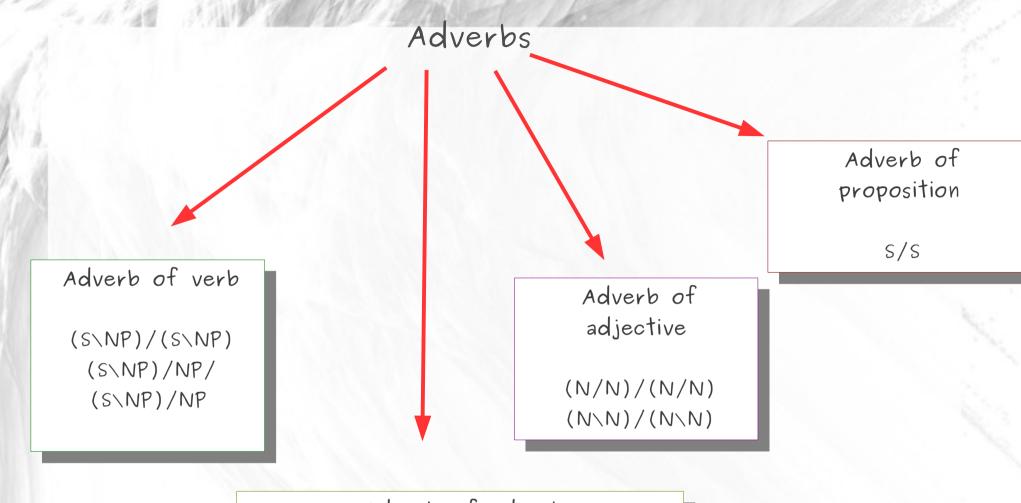
```
(S\NP)/NP:likes (NP/N):a
-----> (>B)
(S\NP)/N: (B likes a)
```

3. Type-raising rules with the combinator C*

For NP subj
e1:John
----> (>C*)
S/(S\NP):C*John

Some preliminary exercises to introduce the combinators:

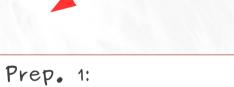
Finding a ccg type adequate to the given lexique



Adverb of adverb

 $(S\NP)/(S\NP)/(S\NP)/(S\NP)$ $(S\NP)/NP/(S\NP)/NP/$ $(S\NP)/NP$





constructor of adverbial phrase

(S\NP)\(S\NP)/NP (S/S)/NP (S/S)/N Prep. 2: constructor of adjectival phrase

 $(N \setminus N) / NP$ $(N \setminus N) / N$

Example: Dictionary of typed words

Syntactic categories	Syntactic types	Lexical entries
Nom.	Ν	Olivia, apple
Completed nom.	NP	an apple, the school
Pron.	NP	She, he
Adj.	(N/N), (N\N)	pretty woman,
Adv.	(N/N)/(N/N), (S\NP)\(S\NP)	very delicious,
Vb	(S\NP), (S\NP)/NP	run, give
Prep.	(S\NP)\(S\NP)/NP	run in the park,
	(NP\NP)/NP	book of John,
Relative	(S\NP)/S	I believe that

Some structures of natural language: coordination

- > Cecila picks the pear.
- > Cecilia eats the pear.
- > Cecilia <u>picks</u> and <u>eats</u> <u>the pear</u>.
- Any logical form of this kind must of course express that the arguments appear in two predications.

```
((and (picks (the pears)) (eats (the pears))) Cecilia)
```

Coordination (Φ)

X CONJ
$$X\Rightarrow_{\Phi} X$$
 (Coordination Φ)

x:e1 CONJ x:e2

-----> (> Φ)

x: Φ CONJ e1 e2

Follow the next steps.

- (1) Attribute first the CCG types to each linguistic expression
- (2) calculate theses types to obtain the syntactic analysis by applying the CCG rules.
- (3) eliminate the applied combinators with respect to each b-reduction of combinators
- (4) check if your semantic representation is well-structured normal form.

· Example : Cecilia picks and eats the pear.

```
1/[NP:Cecilia]-[(S\NP)/NP: picks]-[CONJ:and]-[(S\NP)/NP: eats]-[NP: the pear]
2/[(S/(S\NP):C*Cecilia]-[(S\NP)/NP: picks]-[CONJ:and]-[(S\NP)/NP: eats]-[NP: the pear] (>C*)
3/[(S/(S\NP):C*Cecilia]-[(S\NP)/NP: Φ and picks eats]-[NP: the pear] (>Φ)
4/[(S/(S\NP):C*Cecilia]-[(S\NP)/NP: Φ and picks eats]-[NP: the pear] (>B)
5/[(S/NP:B(C*Cecilia) (Φ and picks eats)]-[NP: the pear] (>B)
6/[(S:(B(C*Cecilia) (Φ and picks eats)) (the pear)] (>)
```

```
7/(B(C*Cecilia) (Φ and picks eats))(the pear)
8/(C*Cecilia) ((Φ and picks eats))(the pear)) (elimination of B)
9/((Φ and picks eats))(the pear))(Cecilia) (elimination of C*)
10/((and (picks(the pear)) (eats(the pear)))(Cecilia) ) (elimination of Φ)
```

Next week ...

• Continue about the application of the combinators to natural language analysis: extraction asymmetries, subordinative structure and relative