Combinatory Categorial Grammar

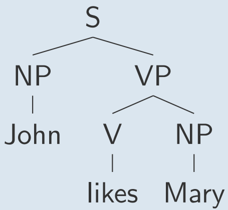
Weiwei Sun

Institute of Computer Science and Technology
Peking University

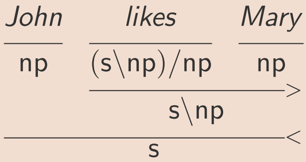
March 12, 2019

Simple vs. complex category

Phrase Structure Grammar



Today's lecture: Categorial Grammar



Discussion

What is the meaning of the symbols in this analysis?

$$\cfrac{\cfrac{\mathit{John}}{\mathsf{np}} \qquad \cfrac{\cfrac{\mathit{likes}}{\mathsf{(s\backslash np)/np}} \qquad \cfrac{\mathit{Mary}}{\mathsf{np}}}{\mathsf{s\backslash np}}{>}}{\mathsf{s}}{<}$$

Categorial Grammar: Overview

Complex category

In a CG, all constituents—and in particular the lexical elements—are associated with a very specific *category* which define their syntactic behaviour.

Simple *phrase-structure* rules

A set of universal rules defines how words and other constituents can be combined according to their categories.

Syntax vs. Semantics

Syntactic and semantic descriptions are tightly connected $\rightarrow$ CG is popular amongst logicians and semanticists.

Outline

Ideas of Categorial Grammar

- Category
- Rule Schemata
- Semantics

Non-local Dependency Constructions

Combinatory Categorial Grammar

Categories

Definition

The set of syntactic categories $\mathcal{C}$ is defined recursively:

- **Atomic categories**: the grammar for each language is assumed to define a finite set of atomic categories, usually $\mathrm{s}, \mathrm{np}, \mathrm{n}, \mathrm{pp}, \ldots \in \mathcal{C}$
- **Complex categories**: if X and Y $\in \mathcal{C}$, then $X/Y, X\backslash Y \in \mathcal{C}$

Complex categories X/Y or $X\backslash Y$ are functors

- X: a result
- Y: an argument
- $/$: arguments to the right of the functor
- $\backslash$: arguments to the left of the functor

Lexical categories

Complex categories encode subcategorisation information

- intransitive verb: s\np ▷ walked
- transitive verb: (s\np)/np ▷ respected
- ditransitive verb: ((s\np)/np)/np ▷ gave

(s\np)/np

- the verb takes a noun phrase to its right, and
- another noun phrase to its left to form a sentence.

There is no explicit difference made between phrases and words: An intransitive verb is described in the same way as a verb phrase with an object: s\np.

Lexical categories

Complex categories encode subcategorisation information

- intransitive verb: s\np ▷ walked
- transitive verb: (s\np)/np ▷ respected
- ditransitive verb: ((s\np)/np)/np ▷ gave

(s\np)/np

- the verb takes a noun phrase to its right, and
- another noun phrase to its left to form a sentence.

There is no explicit difference made between phrases and words: An intransitive verb is described in the same way as a verb phrase with an object: s\np.

Lexical categories

Complex categories encode subcategorisation information

- intransitive verb: s\np ▷ walked
- transitive verb: (s\np)/np ▷ respected
- ditransitive verb: ((s\np)/np)/np ▷ gave

(s\np)/np

- the verb takes a noun phrase to its right, and
- another noun phrase to its left to form a sentence.

There is no explicit difference made between phrases and words: An intransitive verb is described in the same way as a verb phrase with an object: s\np.

Lexical categories

Modification

In CG, adjuncts have the following general form: $X\backslash X$ or X/X.

Example

- PP nominal: $(np\backslash np)/np$
- PP verbal: $((s\backslash np)\backslash(s\backslash np))/np$

Lexicalization (1)

Lexicalization

In a lexicalized grammar, each element of the grammar contains at least one lexical item (terminal).

- G1: S→ SS, S→ a
- G2: S→ aS, S→ a

Grammar or Lexicon

In a CG,

- the lexicon specifies the categories that the words of a language can take;
- lexical entries do most of the grammatical work of mapping the strings of the language to their interpretations.

Lexicalization (2)

The Principle of Lexical Head Government

Both bounded and unbounded syntactic dependencies are specified by the lexical syntactic type of their head.

Example

(1)
a. *John* $\vdash$ np ▷ *John* is a noun phrase.
b. *shares* $\vdash$ np ▷ *shares* is a noun phrase.
c. *buys* $\vdash$ (s\np)/np ▷ *buy* is a transitive verb.
d. *sleeps* $\vdash$ s\np ▷ *sleeps* is an intransitive verb.
e. *well* $\vdash$ (s\np)/(s\np) ▷ *well* can modify a s\np-like thing.

Lexicalization (3)

The Principle of Head Categorial Uniqueness

A single nondisjunctive lexical category for the head of a given construction specifies both the bounded dependencies that arise when its complements are in canonical position and the unbounded dependencies that arise when those complements are displaced under relativization, coordination, and the like.

admire $\vdash$ (s\np)/np

(2) a. John admires Mary.

b. the man that I believe that John admires.

c. I believe that John admires and you believe that he dislikes, the woman in the skinny skirt.

AB categorial grammar (1)

Variants of categorial grammar differ in the rules they allow.

- The system defined by Ajdukiewicz (1935) and Bar-Hillel (1953) forms the basis for all variants of categorial grammar.
- In AB categorial grammar, categories can only combine through function application.

Forward application

$$X/Y \quad Y \quad \Rightarrow \quad X \qquad (>)$$

Backward application

$$Y \quad X\backslash Y \quad \Rightarrow \quad X \qquad (<)$$

AB categorial grammar (2)

Deriving a string

- A string α is grammatical if each word in the string can be assigned a category (as defined by the lexicon) so that the lexical categories of the words in α can be combined (according to the grammar rules) to form a constituent.
- The process of combining constituents in this manner is called a derivation.

Example

$$
\dfrac{\dfrac{\textit{John}}{\mathsf{np}} \quad \dfrac{\dfrac{\textit{buys}}{(\mathsf{s}\backslash\mathsf{np})/\mathsf{np}} \quad \dfrac{\textit{shares}}{\mathsf{np}}}{\mathsf{s}\backslash\mathsf{np}}>}{\mathsf{s}}<
$$

Adjuncts

Modification

In CG, adjuncts have the following general form: X\X or X/X.

Example

The	*small*	*cat*	*chased Mary*	*quickly*	*round the garden*
np/n	n/n	n	s\np	(s\np)/(s\np)	(s\np)/(s\np)

- *small* (n/n) + *cat* (n) → n (>)
- *The* (np/n) + n → np (>)
- *chased Mary* (s\np) + *quickly* ((s\np)/(s\np)) → s\np (<)
- s\np + *round the garden* ((s\np)/(s\np)) → s\np (<)
- np + s\np → s (<)

Not all X\X nor X/X are modifiers.

More examples

我 | 病 | 了
np | s\np | (s\np)\(s\np)

———————— >
s\np

———————— <
s

我们 | 参观 | 了 | 北京大学
np | (s\np)/np | ((s\np)/np)\((s\np)/np) | np

———————— >
(s\np)/np

———————— >
s\np

———————— <
s

Lexicalization (4)

L. Bloomfield, *Language*

The lexicon is really an appendix of the grammar, a list of basic irregularities. This is all the more evident if meanings are taken into consideration, since the meaning of each morpheme belongs to it by an arbitrary tradition.

CG's view

- If this is the case, nothing in the lexicon is predictable, hence we do not need a theory of the lexicon.
- CG argues that this dichotomy gets in the way of our understanding of how syntax can shape possible lexicons.
  - Any combinatory difference must be lexically specifiable.

Interpretation and predicate–argument structure (1)

Architecture

Syntactic structure (CG derivation) + Lexical interpretation

$\Downarrow$

Meaning representation

buy $\vdash$ (s\np$_1$)/np$_2$: **buy**$\rightarrow_A$np$_1$ $\wedge$ **buy**$\rightarrow_P$np$_2$

- Syntactic category: (s\np)/np
- Semantic type (intuitive idea): the np indexed with "2" is the Patient of *buy*; the np indexed with "1" is the Agent of *buy*.

Interpretation and predicate–argument structure (2)

buy $\vdash (s\backslash np_1)/np_2$: **buy**$\rightarrow_A np_1 \wedge$ **buy**$\rightarrow_P np_2$

- Syntactic category: $(s\backslash np)/np$
- Semantic type (intuitive idea): the np indexed with "2" is the Patient of *buy*; the np indexed with "1" is the Agent of *buy*.

Using a dependency interpretation

John	*buys*	*shares*
np	$(s\backslash np_1)/np_2$	np
	$s\backslash np_1$ (>)	
s (<)		

Interpretation and predicate–argument structure (2)

buy $\vdash (s\backslash np_1)/np_2$: **buy**$\rightarrow_A np_1 \wedge$ **buy**$\rightarrow_P np_2$

- Syntactic category: $(s\backslash np)/np$
- Semantic type (intuitive idea): the np indexed with "2" is the Patient of *buy*; the np indexed with "1" is the Agent of *buy*.

Using a dependency interpretation

John	*buys*	*shares*
np : john	$(s\backslash np_1)/np_2$: buy$\rightarrow_A np_1 \wedge$ buy$\rightarrow_P np_2$	np : shares
	$s\backslash np_1$ (>)	
s (<)		

Interpretation and predicate–argument structure (2)

buy $\vdash (s\backslash np_1)/np_2$: **buy**$\rightarrow_A np_1 \wedge$ **buy**$\rightarrow_P np_2$

- Syntactic category: $(s\backslash np)/np$
- Semantic type (intuitive idea): the np indexed with "2" is the Patient of *buy*; the np indexed with "1" is the Agent of *buy*.

Using a dependency interpretation

$$\dfrac{\dfrac{John}{np : john} \quad \dfrac{\dfrac{buys}{(s\backslash np_1)/np_2 : buy\rightarrow_A np_1 \wedge buy\rightarrow_P np_2} \quad \dfrac{shares}{np : shares}}{s\backslash np_1 : buy\rightarrow_A np_1 \wedge buy \rightarrow_P shares}>}{s}<$$

Interpretation and predicate–argument structure (2)

buy $\vdash (s\backslash np_1)/np_2$: **buy**$\rightarrow_A np_1 \wedge$ **buy**$\rightarrow_P np_2$

- Syntactic category: $(s\backslash np)/np$
- Semantic type (intuitive idea): the np indexed with "2" is the Patient of *buy*; the np indexed with "1" is the Agent of *buy*.

Using a dependency interpretation

John	*buys*	*shares*
np : john	$(s\backslash np_1)/np_2$: buy$\rightarrow_A np_1 \wedge$ buy$\rightarrow_P np_2$	np : shares

$>$: $s\backslash np_1$: buy$\rightarrow_A np_1 \wedge$ buy $\rightarrow_P$ shares

$<$: s : buy $\rightarrow_A$ john $\wedge$ buy$\rightarrow_P$shares

The principle of type transparency

The principle of Categorial Type Transparency

For a given language, the semantic type of the interpretation together with a number of language-specific directional parameter settings uniquely determines the syntactic category of a category.

The inverse of Type Transparency

For any category, the semantic type is a function of the syntactic type.

The Principle of Combinatory Type Transparency

All syntactic combinatory rules are type-transparent versions of one of a small number simple semantic operations over functions.

CG vs. CFG

- CGs put into the lexicon most of the information that is captured in CFG rules.
- In CGs, all constituents and lexical elements are associated with a syntactic "category."
- In CGs, syntactic information is tightly related to semantic information.

Example

$$\begin{array}{lcl} S & \rightarrow & NP\ VP \\ VP & \rightarrow & TV\ NP \\ TV & \rightarrow & \text{married}|\text{finds}|\ldots \end{array}$$

married := (s\np)/np.

Outline

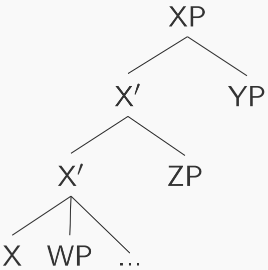
Local dependencies

A head generally realizes its dependents locally within its head domain

- Arguments:
  - Verbs take arguments: subject, object, complements, ...
  - Heads subcategorize for their arguments
- Adjuncts/Modifiers:
  - adjectives modify nouns,
  - adverbs modify VPs or adjectives,
  - PPs modify NPs or VPs
  - Heads do not subcategorize for their modifiers

These are all local dependencies that can typically be expressed in a CFG.

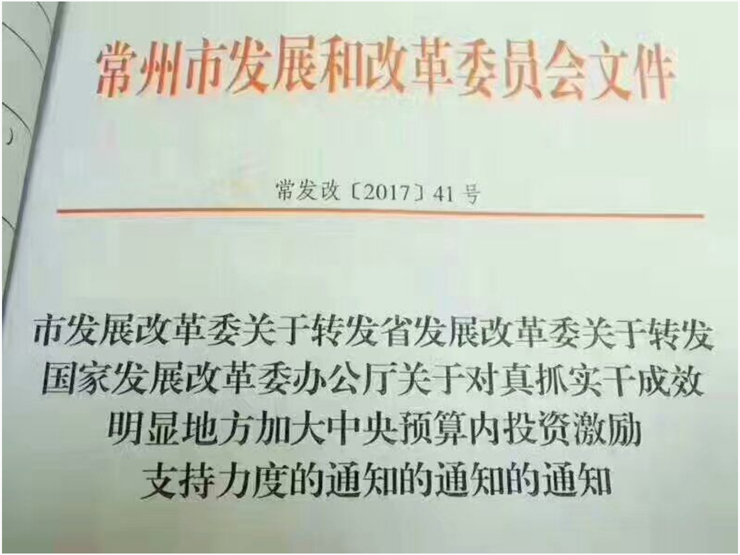
Center embedding



VP⇒V NP NP

(3) a. 我给了那个人一本书
b. 我给了站在那儿的那个人一本书
c. 我给了站在那儿正在扫二维码的那个人一本书
d. 我给了站在那儿正在扫微信二维码买水喝的那个人一本书

Center embedding



Long-distance dependencies

Certain kind of constructions resist this generalization

- Some sentences exhibit phrases that appear "out of place" based on simple head-argument or head-modifier constraints.
- The distance from the position of the "dislocated" phrase to its "natural home" can be quite far.

wh-question

(4) a. Who do you think _ writes well about human sadness?
    b. Who do you think the cops are going to believe _?

Long-distance dependencies

Non-local

A syntactic theory needs a mechanism for expressing these *non-local*/*long-distance*/*long-range* dependencies.

- How can the non-local relation between a head and such argument be licensed?
- How can their properties be captured?

In Transformational Grammar

Non-local dependencies are analyzed as results of movement.

Wh-movement

Move a *wh*-phrase to the specifier of CP to check a [+WH] feature in C.

Long-distance dependencies

Long-distance dependencies

Bounded long-distance dependencies:

- Locally mediated dependencies
- Limited distance between the head and argument

Unbounded long-distance dependencies:

- Arbitrary distance (within the same sentence)

Bounded dependencies

Raising

(5) He **seems** to sleep in class.

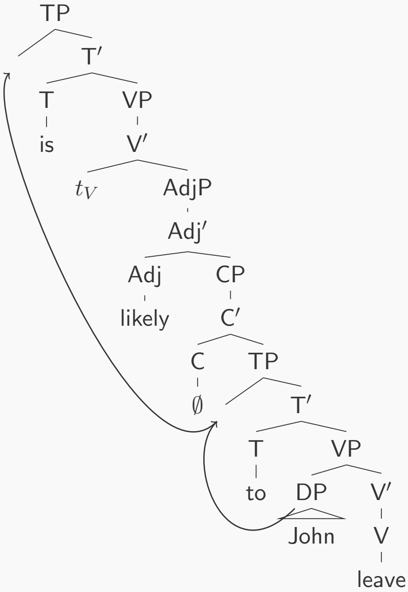
(Subject/Object) Control

(6) a. He **wants** to sleep in class.

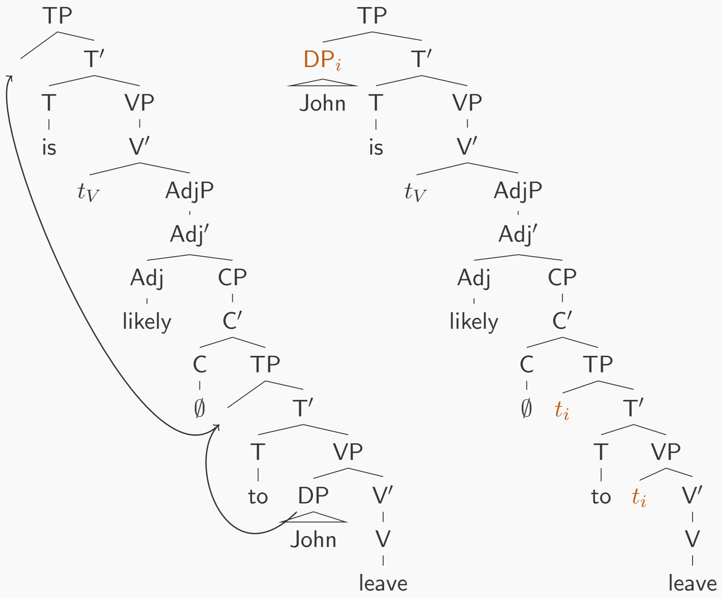
b. He **promises** her not to sleep in class.

c. She **persuades** him not to sleep in class.

DP movement


TP
T′
T
VP
is
V′
t_V
AdjP
Adj′
Adj
CP
likely
C′
C
TP
∅
T′
T
VP
to
DP
V′
John
V
leave

DP movement


TP
T′
T
VP
is
V′
t_V
AdjP
Adj′
Adj
CP
likely
C′
C
TP
∅
T′
T
VP
to
DP
V′
John
V
leave
TP
DP_i
T′
John
T
VP
is
V′
t_V
AdjP
Adj′
Adj
CP
likely
C′
C
TP
∅
t_i
T′
T
VP
to
t_i
V′
V
leave

No transformation in CG derivation

Raising

John	*is likely*	*to*	*leave*
np_l	$(s\backslash np_i)/(s\backslash np_i)$	$(s\backslash np_j)/(s\backslash np_j)$	$s\backslash np_k$

$$\text{to leave} \Rightarrow_{>} s\backslash np : j = k$$

$$\text{is likely to leave} \Rightarrow_{>} s\backslash np : i = j = k$$

$$\text{John is likely to leave} \Rightarrow_{<} s : \text{leave'} \rightarrow_A \text{john'}(k = l)$$

Control

She	*persuades*	*him*	*not to sleep in class*
np	$((s\backslash np)/(s\backslash np_i))/np_i$	np_j	$s\backslash np_k$

$$\text{persuades him} \Rightarrow_{>} (s\backslash np)/(s\backslash np_i) : i{=}j$$

$$\text{persuades him not to sleep in class} \Rightarrow_{>} s\backslash np : \text{sleep'} \rightarrow_A \text{him}(j{=}k)$$

$$\text{She persuades him not to sleep in class} \Rightarrow_{<} s$$

Unbounded dependency constructions (UDC)

Wh-movement

(7) a. Who do you think Bob saw?
b. Who do you think Bob said he saw?
c. Who do you think Bob said he imagined that he saw?

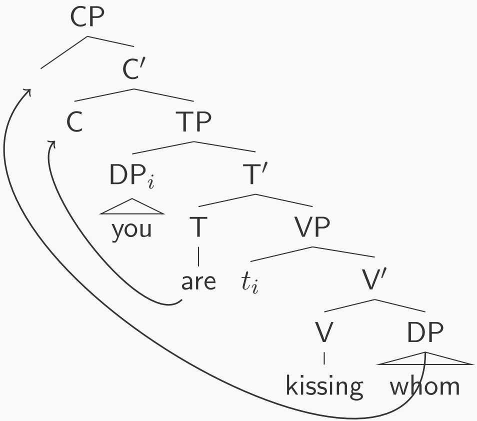
Topicalization

(8) That guy, [I believe Peter told me you thought] you like.

Clefts

(9) It's that guy that I believe Peter told me you thought] you like

Wh-movement



Coordination

Right-node raising

(10) [[she would have bought] and [he might sell]] shares.

Argument-cluster coordination

(11) I give [[you an apple] and [him a pear]].

Gapping

(12) [She likes sushi], and [he sashimi].

Outline

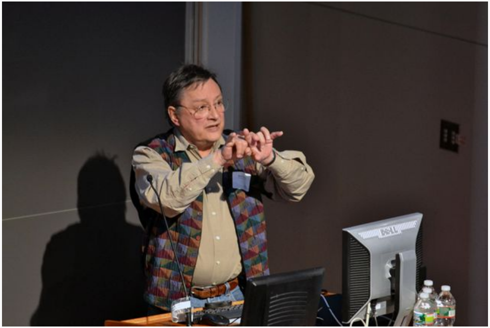
Combinatory Categorial Grammar: Overview

Extending AB Grammar

- CCG extends AB categorial grammar by a set of rule schemata based on the combinators of combinatory logic.
- CCG facilitates the recovery of the non-local dependencies
  - Syntactically, they allow analyses of extraction and coordinate constructions which use the same lexical categories for the heads of such constructions as in the *canonical case*.
  - Semantically, they guarantee that non-local dependencies fill the same argument slots as local dependencies.

- The weak generative power of AB Grammar and CFG are is equivalent.
- Extra combinatory rules increase the weak generative power to mildly context-sensitivity.

Mark Steedman



- *Surface Structure and Interpretation*
- *The Syntactic Process*

Coordination

Simplified coordination rule

$$X \quad CONJ \quad X^* \Rightarrow \quad X^\star \qquad (\Phi)$$

- X, X^* and $X^\star$ are categories of the same type but different interpretations.

Example

Anna	*met*	*and*	*married*	*Manny*
np_0	$(s_1\backslash np_2)/np_3$	CONJ	$(s_4\backslash np_5)/np_6$	np_7

$(s_1\backslash np_2)/np_3$ CONJ $(s_4\backslash np_5)/np_6$ $\xrightarrow{<\Phi>}$ $(s_8\backslash np_9)/np_{10}$

$(s_8\backslash np_9)/np_{10}$ np_7 $\xrightarrow{>}$ $s_8\backslash np_9$

np_0 $s_8\backslash np_9$ $\xrightarrow{<}$ s_8

Semantics

Coordination

Simplified coordination rule

$$X \quad \text{CONJ} \quad X^{*} \Rightarrow \quad X^{\star} \qquad (\Phi)$$

- X, X^{*} and $X^{\star}$ are categories of the same type but different interpretations.

Example

Anna	*met*	*and*	*married*	*Manny*
np_0	$(s_1\backslash np_2)/np_3$	CONJ	$(s_4\backslash np_5)/np_6$	np_7

$$(s_1\backslash np_2)/np_3 \quad \text{CONJ} \quad (s_4\backslash np_5)/np_6 \Rightarrow (s_8\backslash np_9)/np_{10} \quad <\Phi>$$

$$(s_8\backslash np_9)/np_{10} \quad np_7 \Rightarrow s_8\backslash np_9 \quad >$$

$$np_0 \quad s_8\backslash np_9 \Rightarrow s_8 \quad <$$

Semantics

$np_0 = \text{anna}'$, $np_7 = \text{manny}'$

Coordination

Simplified coordination rule

$$\text{X} \quad \text{CONJ} \quad \text{X}^* \Rightarrow \quad \text{X}^\star \qquad (\Phi)$$

- X, X^* and $X^\star$ are categories of the same type but different interpretations.

Example

Anna	*met*	*and*	*married*	*Manny*
np_0	$(s_1\backslash np_2)/np_3$	CONJ	$(s_4\backslash np_5)/np_6$	np_7

met CONJ $married$ $\Rightarrow_{<\Phi>}$ $(s_8\backslash np_9)/np_{10}$

$(s_8\backslash np_9)/np_{10}$ np_7 $\Rightarrow_{>}$ $s_8\backslash np_9$

np_0 $s_8\backslash np_9$ $\Rightarrow_{<}$ s_8

Semantics

meet' $\rightarrow_A$ np_2, meet' $\rightarrow_P$ np_3,

Coordination

Simplified coordination rule

$$X \quad \text{CONJ} \quad X^{*} \Rightarrow \quad X^{\star} \qquad (\Phi)$$

- X, X^* and $X^\star$ are categories of the same type but different interpretations.

Example

Anna	*met*	*and*	*married*	*Manny*
np_0	$(s_1\backslash np_2)/np_3$	CONJ	$(s_4\backslash np_5)/np_6$	np_7

$$\frac{(s_1\backslash np_2)/np_3 \quad \text{CONJ} \quad (s_4\backslash np_5)/np_6}{(s_8\backslash np_9)/np_{10}} <\Phi>$$

$$\frac{(s_8\backslash np_9)/np_{10} \quad np_7}{s_8\backslash np_9} >$$

$$\frac{np_0 \quad s_8\backslash np_9}{s_8} <$$

Semantics

marry' $\rightarrow_A$ np_5, marry' $\rightarrow_P$ np_6,

Coordination

Simplified coordination rule

$$X \quad \text{CONJ} \quad X^{*} \Rightarrow \quad X^{\star} \qquad (\Phi)$$

- X, X^{*} and $X^{\star}$ are categories of the same type but different interpretations.

Example

Anna	*met*	*and*	*married*	*Manny*
np_0	$(s_1\backslash np_2)/np_3$	CONJ	$(s_4\backslash np_5)/np_6$	np_7

$$\text{met and married} \Rightarrow_{<\Phi>} (s_8\backslash np_9)/np_{10}$$

$$(s_8\backslash np_9)/np_{10} \;\; np_7 \Rightarrow_{>} s_8\backslash np_9$$

$$np_0 \;\; s_8\backslash np_9 \Rightarrow_{<} s_8$$

Semantics

$s_8 = s_1 = s_4,\ np_9 = np_2 = np_5,\ np_{10} = np_3 = np_6$

Coordination

Simplified coordination rule

$$\mathrm{X} \quad \mathrm{CONJ} \quad \mathrm{X}^* \Rightarrow \quad \mathrm{X}^\star \qquad (\Phi)$$

- X, X* and X⋆ are categories of the same type but different interpretations.

Example

Anna	*met*	*and*	*married*	*Manny*
np_0	$(s_1\backslash np_2)/np_3$	CONJ	$(s_4\backslash np_5)/np_6$	np_7

met CONJ $married$ $\Rightarrow_{<\Phi>}$ $(s_8\backslash np_9)/np_{10}$

$(s_8\backslash np_9)/np_{10}$ np_7 $\Rightarrow_{>}$ $s_8\backslash np_9$

np_0 $s_8\backslash np_9$ $\Rightarrow_{<}$ s_8

Semantics

$np_{10} = np_7,$

Coordination

Simplified coordination rule

$$X \quad CONJ \quad X^* \Rightarrow \quad X^\star \qquad (\Phi)$$

- X, X^* and $X^\star$ are categories of the same type but different interpretations.

Example

$$\dfrac{\dfrac{Anna}{np_0} \quad \dfrac{\dfrac{\dfrac{met}{(s_1\backslash np_2)/np_3} \quad \dfrac{and}{CONJ} \quad \dfrac{married}{(s_4\backslash np_5)/np_6}}{(s_8\backslash np_9)/np_{10}}{<\Phi>} \quad \dfrac{Manny}{np_7}}{s_8\backslash np_9}{>}}{s_8}{<}$$

Semantics

$np_0 = np_9$,

Coordination

Simplified coordination rule

$$X \quad \text{CONJ} \quad X^{*} \Rightarrow \quad X^{\star} \qquad (\Phi)$$

- X, X^{*} and $X^{\star}$ are categories of the same type but different interpretations.

Example

$$\dfrac{\dfrac{\textit{Anna}}{np_0} \quad \dfrac{\dfrac{\dfrac{\textit{met}}{(s_1\backslash np_2)/np_3} \quad \dfrac{\textit{and}}{\text{CONJ}} \quad \dfrac{\textit{married}}{(s_4\backslash np_5)/np_6}}{(s_8\backslash np_9)/np_{10}}{<\Phi>} \quad \dfrac{\textit{Manny}}{np_7}}{s_8\backslash np_9}{>}}{s_8}{<}$$

Semantics

meet'(anna', manny'), marry'(anna', manny')

Composition

Forward composition

X/Y Y/Z $\Rightarrow_{\mathbf{B}}$ X/Z $(>\mathbf{B})$

Example (Abbreviation: vp: s\np)

Anna	*met*	*and*	*might*	*marry*	*Manny*
np	(s\np)/np	CONJ	(s\np)/vp	vp/np	np

might + *marry*: (s\np)/vp vp/np $>\mathbf{B}$ (s\np)/np

met *and* *might marry*: $<\Phi>$ (s\np)/np

met and might marry + *Manny*: $>$ s\np

Anna + *met and might marry Manny*: $<$ s

Semantics:

- *marry* $\vdash$ (s\np$_1$)/np$_2$: s = marry' $\wedge$ marry' $\rightarrow_A$ np$_1$ $\wedge$ marry' $\rightarrow_P$ np$_2$
- *might* $\vdash$ (s$_1$\np$_1$)/(s$_2$\np) : may' $\rightarrow_A$ np$_1$ $\wedge$ may' $\rightarrow_M$ s$_2$

Composition

Generalized forward composition

$$X/Y \quad (Y/Z)/\$_1 \quad \Rightarrow_{\mathbf{B}^n} \quad (X/Z)/\$_1 \qquad (>\mathbf{B}^n)$$

Example

I	*offered,*	*and*	*may*	*give,*	*a flower*	*to Mary*
np	((s\np)/pp)/np	CONJ	(s\np)/vp	((s\np)/pp)/np	np	pp

may give, $\Rightarrow$ ((s\np)/pp)/np ($>\mathbf{B}^2$)

offered, and may give, $\Rightarrow$ ((s\np)/pp)/np ($<\Phi>$)

offered, and may give, a flower $\Rightarrow$ (s\np)/pp ($>$)

offered, and may give, a flower to Mary $\Rightarrow$ s\np ($>$)

I offered, and may give, a flower to Mary $\Rightarrow$ s ($<$)

Type raising

Forward type raising

$\mathsf{X} \quad \Rightarrow_{\mathbf{T}} \quad \mathsf{T/(T\backslash X)}$ $\qquad (>\mathbf{T})$

$\mathsf{T/(T\backslash X)}$ is a parametrically licensed category for the language.

Extraction out of a relative clause

The	*company*	*which*	*Google*	*bought*
np/n	n	(np\np)/(s/np)	np	(s\np)/np

- *The company*: np/n, n $\Rightarrow_{>}$ np
- *Google*: np $\Rightarrow_{>\mathbf{T}}$ s/(s\np)
- *Google bought*: s/(s\np), (s\np)/np $\Rightarrow_{>\mathbf{B}}$ s/np
- *which Google bought*: (np\np)/(s/np), s/np $\Rightarrow_{>}$ np\np
- *The company which Google bought*: np, np\np $\Rightarrow_{<}$ np

Semantics:

- *which* $\vdash (\mathsf{np_1\backslash np_2})/(\mathsf{s/np_3}) : \mathsf{np_1 = np_2 = np_3}$
- *Type raising* $\mathsf{np_1} \Rightarrow_{\mathrm{T}} \mathsf{s/(s\backslash np_2)} : \mathsf{np_1 = np_2}$

Forward composition and type-raising

Right-node raising

Mary	*ordered*	*and*	*John*	*ate*	*the tapas*
np	(s\np)/np	CONJ	np	(s\np)/np	np

- *Mary*: np $>\mathbf{T}$ s/(s\np)
- s/(s\np) + (s\np)/np $>\mathbf{B}$ s/np
- *John*: np $>\mathbf{T}$ s/(s\np)
- s/(s\np) + (s\np)/np $>\mathbf{B}$ s/np
- s/np CONJ s/np $<\Phi>$ s/np
- s/np + np $>$ s

Maximally incremental left-to-right processing

John	*buys*	*shares*
np	(s\np)/np	np

- *John*: np $>\mathbf{T}$ s/(s\np)
- s/(s\np) + (s\np)/np $>\mathbf{B}$ s/np
- s/np + np $>$ s

Backward composition and type-raising

Backward composition

$\mathsf{Y\backslash Z} \quad \mathsf{X\backslash Y} \quad \Rightarrow_{\mathbf{B}} \quad \mathsf{X\backslash Z} \qquad (<\mathbf{B})$

Backward type raising

$\mathsf{X} \quad \Rightarrow_{\mathbf{T}} \quad \mathsf{T\backslash(T/X)} \qquad (<\mathbf{T})$

$\mathsf{T\backslash(T/X)}$ is a parametrically licensed category for the language.

Argument cluster (tv: vp/np, dtv: (vp/np)/np)

give	*John*	*an apple*	*and*	*Mary*	*a flower*
dtv	tv\dtv ($<\mathbf{T}$)	vp\tv ($<\mathbf{T}$)	CONJ	tv\dtv ($<\mathbf{T}$)	vp\tv ($<\mathbf{T}$)

- *John* + *an apple*: tv\dtv, vp\tv $\Rightarrow$ vp\dtv ($<\mathbf{B}$)
- *Mary* + *a flower*: tv\dtv, vp\tv $\Rightarrow$ vp\dtv ($<\mathbf{B}$)
- vp\dtv CONJ vp\dtv $\Rightarrow$ vp\dtv ($<\Phi>$)
- dtv vp\dtv $\Rightarrow$ vp ($<$)

Backward crossed substitution

Forward crossing composition

$$Y/Z \quad (X\backslash Y)/Z \quad \Rightarrow_{\mathbf{S}} \quad X/Z \qquad (<\mathbf{S}_{\times})$$

Example

which	*I will*	*file*	*without*	*reading*
$(n\backslash n)/(s/np)$	s/vp	vp/np	$(vp\backslash vp)/vping$	$vping/np$

$$(vp\backslash vp)/vping \quad vping/np \Rightarrow (vp\backslash vp)/np \quad (>\mathbf{B})$$

$$vp/np \quad (vp\backslash vp)/np \Rightarrow vp/np \quad (<\mathbf{S}_{\times})$$

$$s/vp \quad vp/np \Rightarrow s/np \quad (>\mathbf{B})$$

$$(n\backslash n)/(s/np) \quad s/np \Rightarrow n\backslash n \quad (>)$$

Crossing composition

Forward crossing composition

$$X/Y \quad Y\backslash Z \quad \Rightarrow_{\mathbf{B}} \quad X\backslash Z$$

Backward crossing composition

$$Y/Z \quad X\backslash Y \quad \Rightarrow_{\mathbf{B}} \quad X/Z$$

Cross-serial dependency construction

dat	*ik*	*Cecilia*	*de nijlpaarden*	*zag*	*voeren*
	np_1	np_2	np_3	$((s\backslash np_1)\backslash np_2)/vp$	$vp\backslash np_3$

$$((s\backslash np_1)\backslash np_2)/vp \quad vp\backslash np_3 \; \Rightarrow_{>\mathbf{B}_\times} \; ((s\backslash np_1)\backslash np_2)\backslash np_3$$

$$np_3 \quad ((s\backslash np_1)\backslash np_2)\backslash np_3 \; \Rightarrow_{<} \; (s\backslash np_1)\backslash np_2$$

$$np_2 \quad (s\backslash np_1)\backslash np_2 \; \Rightarrow_{<} \; s\backslash np_1$$

$$np_1 \quad s\backslash np_1 \; \Rightarrow_{<} \; s$$

Constraints on combinatory rules

The Principle of Adjacency

Combinatory rules may only apply to finitely many phonologically realized and string-adjacent entities.

The Principle of Consistency

All syntactic combinatory rules must be consistent with the directionality of the principal function.

Principal function: the function among the input functions which determines the range of the result.

Example

- $X\backslash Y \quad Y \quad \not\Rightarrow \quad X$

Constraints on combinatory rules

The Principle of Inheritance

If a category that results from the application of a combinatory rule is a function category, then the slash defining directionality for a given argument in that category will be the same as the one(s) defining directionality for the corresponding argument(s) in the input function(s).

Example

- $X/Y \quad Y/Z \quad \not\Rightarrow \quad X\backslash Z$
- $X/Y \quad \text{CONJ} \quad X\backslash Y \not\Rightarrow \quad X/Y$

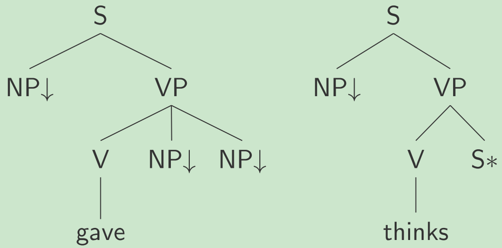
Any language is free to restrict combinatory rules to certain categories, or to entirely exclude a given rule type.

LTAG vs. CCG

Lexicalized TAGs are similar to CCGs

For each lexical item the elementary tree(s) which is (are) anchored on that lexical item can be regarded as the (structured) category (categories) associated with that item.

Example

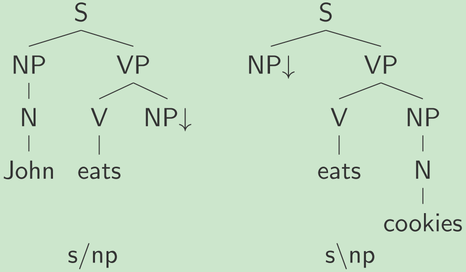


((s/np)/np)\np   (s/s)\np

LTAG vs. CCG (cont)

By combining elementary trees with substitution or adjunction, we can assign a structured category (the derived tree) and a functional interpretation to sequences of lexical items even in the cases when the sequence is discontinuous or when it does not define a constituent in the conventional sense.

Example



Summary

- Like other CG's, CCG has a transparent syntax-semantics interface. If we know the syntax of a sentence, we also know its meaning.
- CCG has a flexible constituent structure:
  - Simple, unified treatment of extraction and coordination
  - Psycholinguistic motivation: allows incremental processing
- CCG is mildly context-sensitive: CCG can capture crossing dependencies.
- CCG is non-transformational

Reading & homework

- §3 *The Syntactic Process*

Homework

- 自选10个汉语句子，试用CCG进行分析，并谈谈你对使用CCG分析汉语的感想。
- 对比你的LTAG和CCG分析，比较两种分析方法的差别