NLP

Introduction to NLP

Combinatory Categorial Grammar (CCG)

Combinatory Categorial Grammar (CCG)

- Complex types
  - E.g., *X/Y* and *X|Y*
  - These take an argument of type *Y* and return an object of type *X*.
  - X/Y – means that Y should appear on the right
  - X\Y – means that Y should appear on the left

Structure of CCG

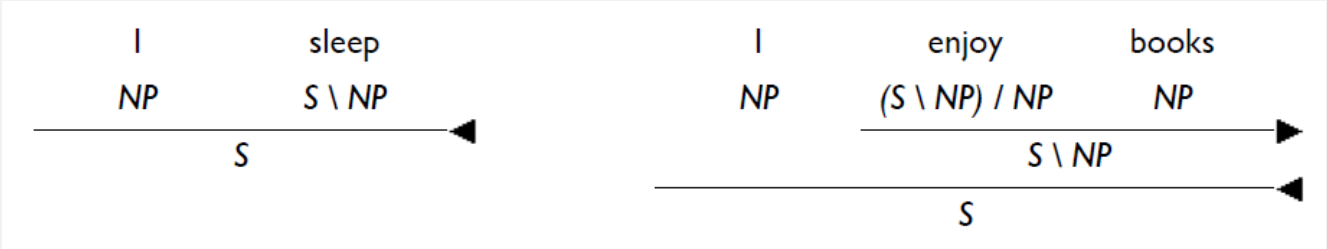
- Categories
- Combinatory rules
- Lexicon

CCG Rules

- Function composition
  - X/Y Y/Z -> X/Z
  - X\Y Z\X -> Z\Y
  - X/Y Y\Z -> X\Z
  - X/Y Z\X -> Z/Y
- Type raising
  - X -> Y/(Y\X)
  - X -> Y\(Y/X)
- Coordination

Example

I	NP
books	NP
sleep	S \ NP
enjoy	(S \ NP) / NP



Example from Jonathan Kummerfeld, Aleka Blackwell, and Patrick Littell

Expressive power

- CCGs can generate the language $a^n b^n c^n d^n$, $n>0$
- Interesting examples:
  - I like New York
  - I like and hate New York
  - I like and would rather be in New York
  - I gave a book to Chen and a laptop to Jorge
  - I want Chen to stay and Jorge to leave
  - I like and Chen hates, New York
    - Where are the verb phrases?

Examples from Steedman 1996

(6) *Forward Application:* (>)

$$X/Y : f \quad Y : a \quad \Rightarrow \quad X : fa$$

(7) *Backward Application:* (<)

$$Y : a \quad X\backslash Y : f \quad \Rightarrow \quad X : fa$$

They yield derivations like the following:

(8)

$$\dfrac{\text{Mary} \qquad \dfrac{\text{likes} \qquad \text{musicals}}{(S\backslash NP_{3s})/NP : like' \quad NP : musicals'}>}{S\backslash NP_{3s} : like'\, musicals'}$$

$$\dfrac{NP_{3sm} : mary' \qquad S\backslash NP_{3s} : like'\, musicals'}{S : like'\, musicals'\, mary}<$$

Examples from Steedman 1996

(9) *Coordination:* (< & >)

$$X \quad conj \quad X \Rightarrow X$$

(10)

I	loathe	and	detest	opera
NP	$(S\backslash NP)/NP$	$CONJ$	$(S\backslash NP)/NP$	NP
			$(S\backslash NP)/NP$ (<&>)	
			$S\backslash NP$ (>)	
		S (<)		

(13) *Forward Composition:* (> B)

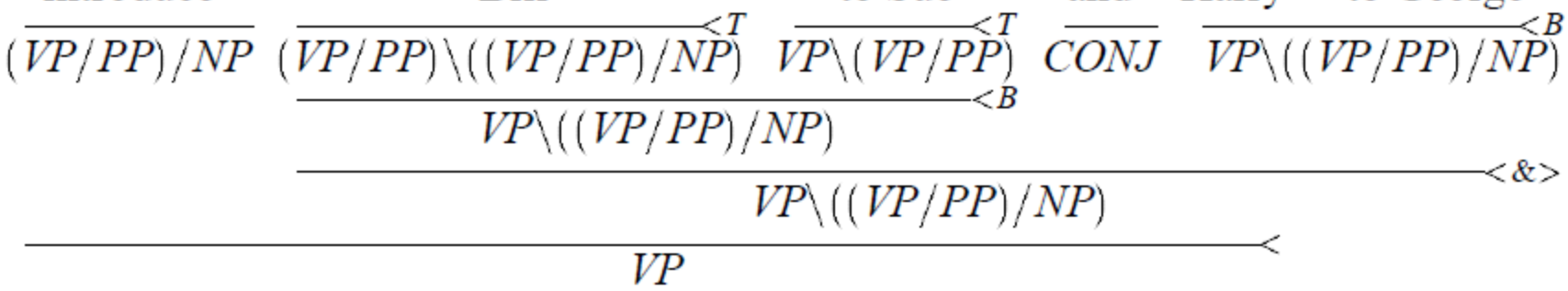
$$X/Y : f \quad Y/Z : g \Rightarrow X/Z : \lambda x.f(gx)$$

(14)

I	requested	and	would	prefer	musicals
NP	$(S\backslash NP)/NP$	$CONJ$	$(S\backslash NP)/VP : will'$	$VP/NP : prefer'$	NP
			$(S\backslash NP)/NP : \lambda x.\lambda y.will'(prefer'x)y$ (>B)		
			$(S\backslash NP)/NP$ (<&>)		
			$S\backslash NP$ (>)		
		S (<)			

Examples from Steedman 1996

(18)



CCG in NLTK

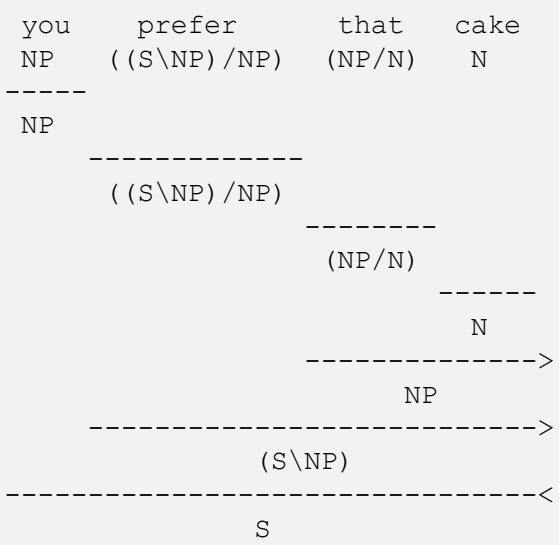
```
from nltk.ccg import chart, lexicon
lex = lexicon.parseLexicon('''
:- S, NP, N, VP
Det :: NP/N
Pro :: NP
Modal :: S\\NP/VP
TV :: VP/NP
DTV :: TV/NP
the => Det
that => Det
that => NP
I => Pro
you => Pro
we => Pro
chef => N
cake => N
children => N
dough => N
```

```
will => Modal
should => Modal
might => Modal
must => Modal
and => var\\.,var/.,var
to => VP[to]/VP
without => (VP\\VP)/VP[ing]
be => TV
cook => TV
eat => TV
cooking => VP[ing]/NP
give => DTV
is => (S\\NP)/NP
prefer => (S\\NP)/NP
which => (N\\N)/(S/NP)
persuade => (VP/VP[to])/NP
''')
```

http://www.nltk.org/howto/ccg.html

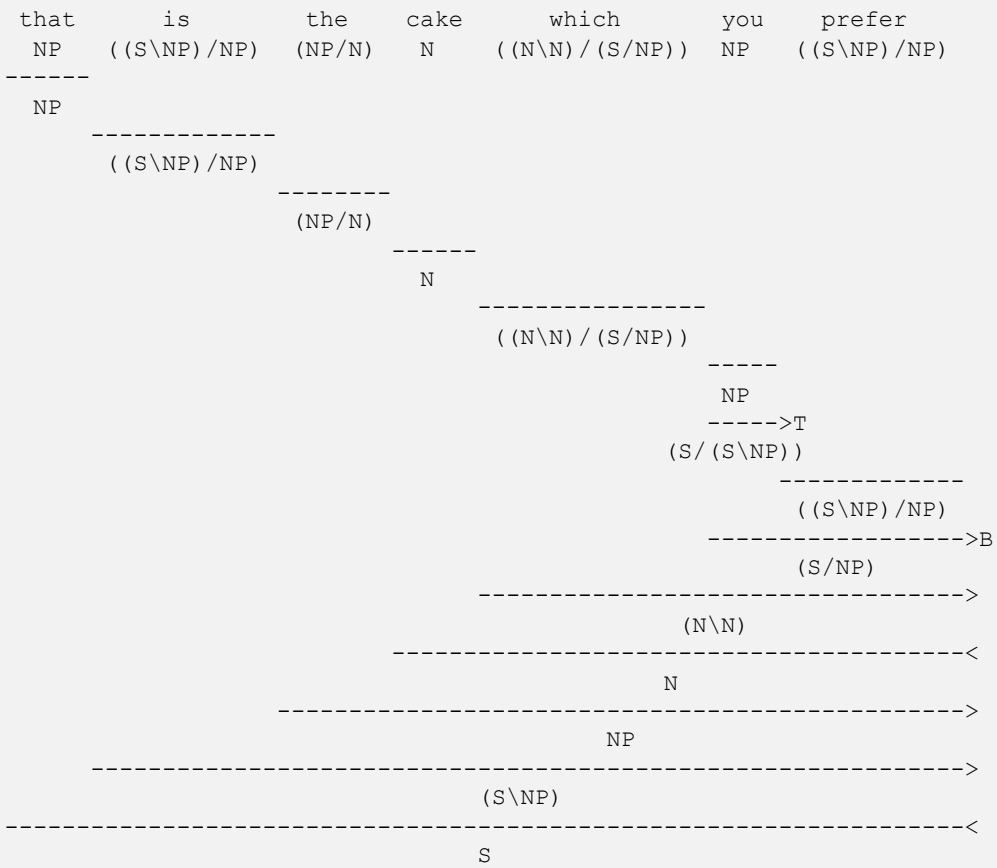
CCG in NLTK

```
parser = chart.CCGChartParser(lex, chart.DefaultRuleSet)
for parse in parser.parse("you prefer that cake".split()):
    chart.printCCGDerivation(parse)
    break
```



http://www.nltk.org/howto/ccg.html

```
for parse in parser.parse("that is the cake which you prefer".split()):
    chart.printCCGDerivation(parse)
    break
```



http://www.nltk.org/howto/ccg.html

CCG

- NACLO problem from 2014
- Authors: Jonathan Kummerfeld, Aleka Blackwell, and Patrick Littell
- http://www.nacloweb.org/resources/problems/2014/N2014-O.pdf
- http://www.nacloweb.org/resources/problems/2014/N2014-OS.pdf
- http://www.nacloweb.org/resources/problems/2014/N2014-P.pdf
- http://www.nacloweb.org/resources/problems/2014/N2014-PS.pdf

CCG problem in NACLO

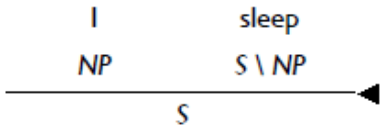
One way for computers to understand language is by forming a structure that represents the relationships between words using a technique called Combinatorial Categorial Grammar (CCG). Computer scientists and linguists can use CCG to parse sentences (that is, try to figure out their structure) and then extract meaning from the structure.

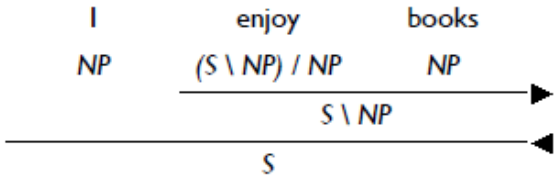
As the name suggests, Combinatorial Categorial Grammar parses sentences by combining categories. Each word in a sentence is assigned a particular category; note that / and \ are two different symbols:

I	NP
books	NP
sleep	S \ NP
enjoy	(S \ NP) / NP

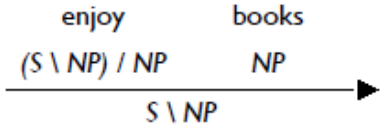
CCG

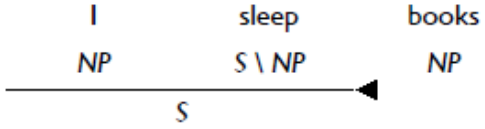
These categories are then combined in systematic ways. We will not explain how, but we will give you two successful parses…

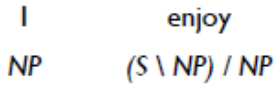


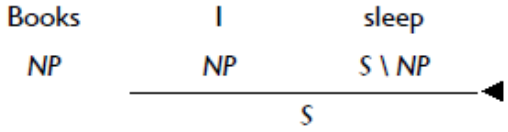


...and four unsuccessful parses…









If a parse is successful, the sentence is declared "grammatical"; if not, the sentence is declared "ungrammatical".

CCG

O1. Using the above examples as evidence, figure out how CCG parses sentences, and describe it briefly here:

O2. In the sentence "I enjoy long books", list all of the categories that, if assigned to "long", make the sentence have a successful parse.

O3. Not every grammatical sentence of English will be declared "grammatical" by the process above. Using only the words "I", "books", "sleep", and "enjoy", form a grammatically correct English sentence that will fail to parse given the categories above. You don't have to use all four of the words.

Answer

O1. CCG assigns a category to each word and constructs a parse by combining pairs of categories to form an S. Not all pairs of categories can combine. A pair is allowed to combine if one category (e.g. A) is contained within the category next to it (e.g. B / A) and lies on the side indicated by the slash (\ for left, / for right). When two categories combine, the result is a new category, taken from the left of the slash (B in this example).

O2. There are four categories that 'long' could have that would create a successful parse of 'I enjoy long books':

1. NP / NP
2. ((S \ NP) \ ((S \ NP) / NP)) / NP
3. ((S \ NP) / NP) \ ((S \ NP) / NP)
4. ((S / NP) \ NP) \ ((S \ NP) / NP)

The first of these is probably the most appropriate. Some possible reasons:

- It is by far the simplest. (After all, all our other categories are relatively simple.)
- It keeps the existing structure of the sentence (where "enjoy" combines with what follows it and then with what precedes it).
- "Long" describes "books" and not "enjoy", so it might make sense to keep them together.
- The first would be the only one to work if "long books" were in any other position.

O3. Possible answers: "I enjoy sleep", topicalized object sentences like "Books I enjoy" and "Sleep I enjoy".

CCG

This problem is a follow-up to problem O and has to be solved after that problem. Tok Pisin (also referred to as New Guinea Pidgin or Melanesian Pidgin) is a creole language spoken in the northern mainland of Papua New Guinea and surrounding islands. It is an official language and the mostly widely used language in the country, spoken by over 5 million people.

Many Tok Pisin words come originally from English – its name comes from "talk" and "pidgin"[1] -- but Tok Pisin isn't just English. It has a distinct grammar and uses these words in different (but systematic!) ways.

P1. Below are sentences in Tok Pisin with a scrambled list of English translations. Match each sentence to its English equivalent.

1.	Brata bilong em i stap rit.	
2.	Ol i stap dringim wara.	
3.	Ol i ken ritim buk bilong mi.	
4.	Em i ritim buk pinis.	
5.	Em i laik rit.	
6.	Susa bilong em i ken rait.	
7.	Susa bilong mi i boilim wara.	
8.	Wara i boil pinis.	

A.	He has read the book.
B.	My sister boils the water.
C.	They can read my book.
D.	His sister can write.
E.	His brother is reading.
F.	The water has boiled.
G.	He wants to read.
H.	They are drinking water.

CCG

P2. Translate the following Tok Pisin sentence into English:

Brata bilong mi i stap ritim buk bilong susa bilong mi.

_____

P3. Translate the following English sentence into Tok Pisin:

Their sister wants to write a book.

_____

CCG

P4. Describing these words in terms of their CCG categories (introduced in Problem O) highlights that these aren't English words combined according to English rules, but are Tok Pisin words combined according to Tok Pisin rules.

Match each Tok Pisin word to its CCG category. Some categories will be used more than once. The symbol S_b is short for 'Bare Clause'.

1.	bilong	
2.	brata	
3.	boil	
4.	boilim	
5.	buk	
6.	dringim	
7.	em	
8.	i	
9.	ken	
10.	laik	

11.	mi	
12.	ol	
13.	pinis	
14.	stap	
15.	raitim	
16.	rit	
17.	ritim	
18.	susa	
19.	wara	

A.	NP
B.	(NP \ NP) / NP
C.	(S \ NP) / (S_b \ NP)
D.	(S_b \ NP)
E.	(S_b \ NP) / NP
F.	(S_b \ NP) \ (S_b \ NP)
G.	(S_b \ NP) / (S_b \ NP)

P5. Explain your answer.

CCG

P1.

1.	*Brata bilong em i stap rit.*	E
2.	*Ol i stap dringim wara.*	H
3.	*Ol i ken ritim buk bilong mi.*	C
4.	*Em i ritim buk pinis.*	A
5.	*Em i laik rit.*	G
6.	*Susa bilong em i ken rait.*	D
7.	*Susa bilong mi i boilim wara.*	B
8.	*Wara i boil pinis.*	F

A.	He has read the book.
B.	My sister boils the water.
C.	They can read my book.
D.	His sister can write.
E.	His brother is reading.
F.	The water has boiled.
G.	He wants to read.
H.	They are drinking water.

P2. My brother is reading my sister's book.

P3. Susa bilong ol i laik raitim buk.

CCG

P4.

1.	bilong	B
2.	brata	A
3.	boil	D
4.	boilim	E
5.	buk	A
6.	dringim	E
7.	em	A
8.	i	C
9.	ken	G
10.	laik	G

11.	mi	A
12.	ol	A
13.	pinis	F
14.	stap	G
15.	raitim	E
16.	rit	D
17.	ritim	E
18.	susa	A
19.	wara	A

A.	NP
B.	(NP \ NP) / NP
C.	(S \ NP) / (S_b \ NP)
D.	(S_b \ NP)
E.	(S_b \ NP) / NP
F.	(S_b \ NP) \ (S_b \ NP)
G.	(S_b \ NP) / (S_b \ NP)

CCG

P5. A. Any noun or pronoun is category A (NP) because they can be used as a noun.

B. The word "bilong" shows possession of the preceding NP by the following NP; therefore, it is (NP\NP)/NP. Also, the phrase [NP bilong NP] yields a noun phrase (NP).

C. The word "i" is necessary for a grammatical sentence, so it is (S\NP)/(S_b\NP). It wants a following verb phrase (indicated by (S_b\NP)) and a preceding noun phrase (NP). NP+i+(S_b\NP) forms a sentence.

D. Each intransitive verb (boil and rit) can stand on its own as S_b\NP, forming the verb phrase.

E. Transitive verbs (boilim, dringim, raitim, ritim; the ones ending in -im), need a following NP, so they are categorized as (S_b\NP)/NP, a verb phrase followed by a noun phrase.

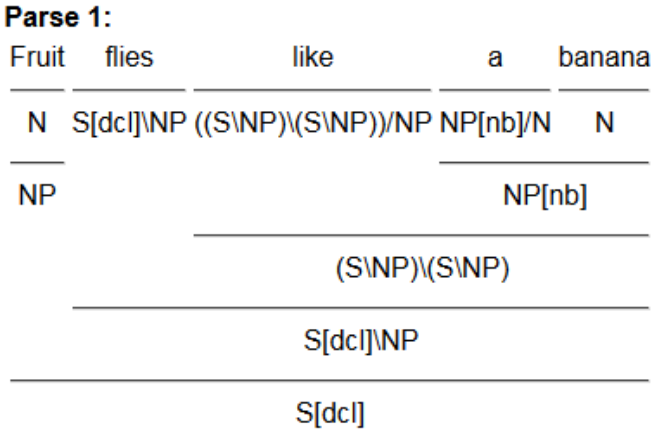
F. The verbs "stap," "ken," and "laik" precede the primary verb phrase and need another verb phrase to create an S_b\NP, so they are the category (S_b\NP)/(S_b\NP).

G. The verb "pinis" comes after the main verb, so it is of the category (S_b\NP)\(S_b\NP) which requires a (S_b\NP) to precede it.

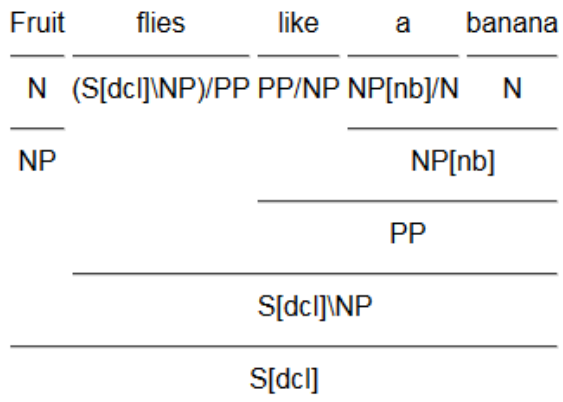
CCG Parsing

- CKY works fine
- http://openccg.sourceforge.net/

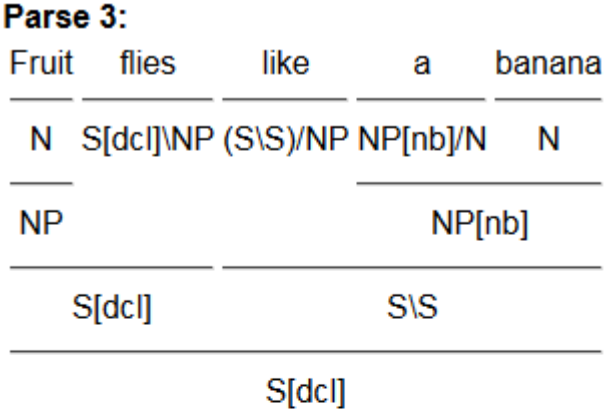
Parse 1:



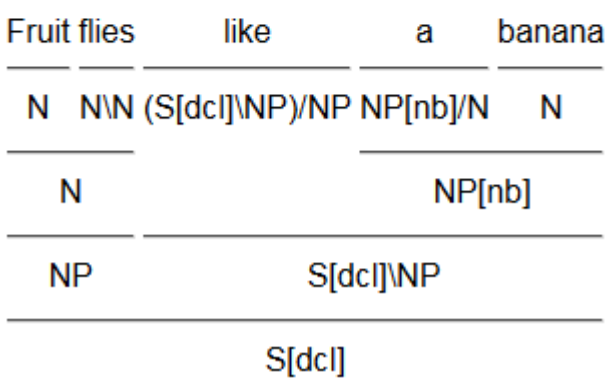
Parse 2:



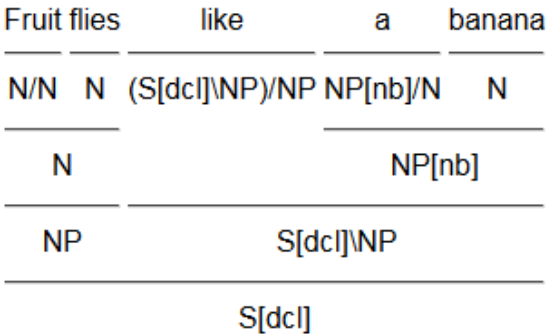
Parse 3:



Parse 4:



Parse 5:



http://4.easy-ccg.appspot.com/do_parse?sentence=Fruit+flies+like+a+banana&nbest=5

Exercise

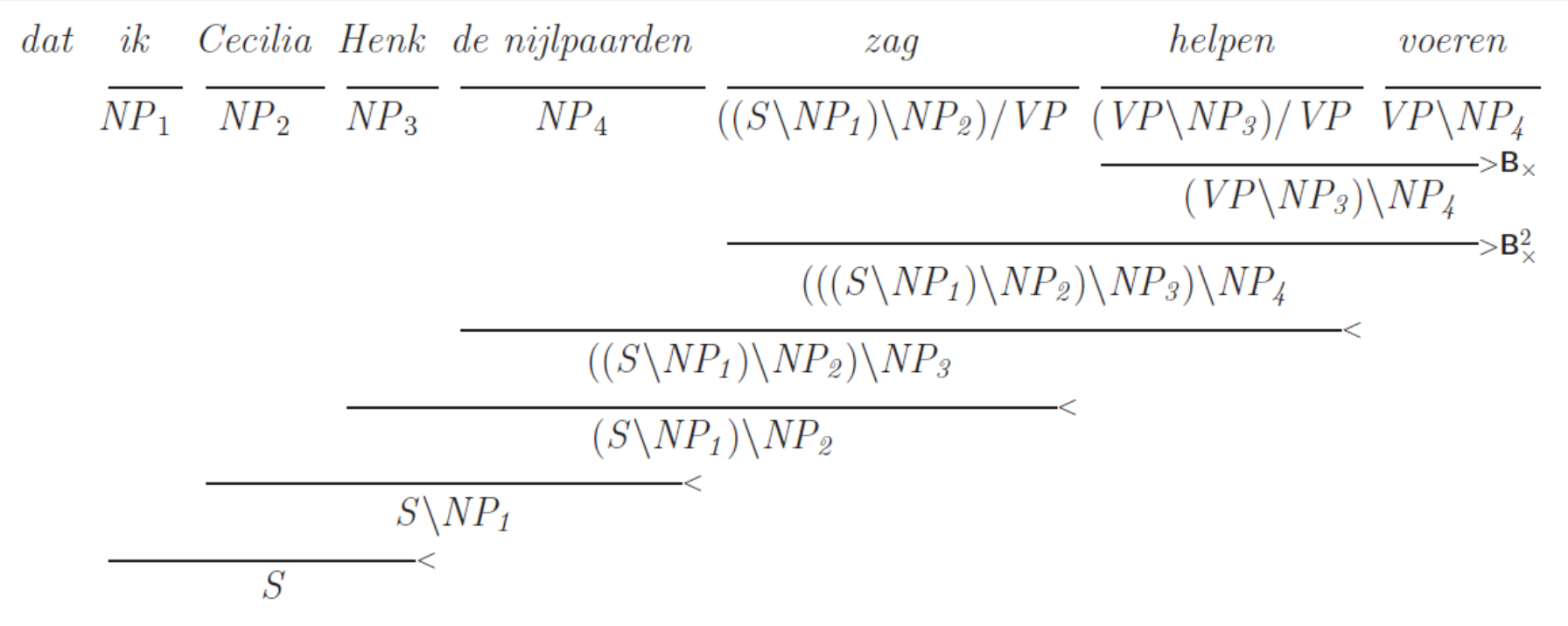
- How do you represent the following categories in CCG
  - Nouns
  - Adjectives
  - Articles
  - Prepositions
  - Transitive verbs
  - Intransitive verbs

Exercise

- How do you represent the following categories in CCG
  - Nouns N
  - Adjectives N/N
  - Articles NP/N
  - Prepositions (NP\NP)/NP
  - Transitive verbs (S\NP)/NP
  - Intransitive verbs

CCG for Dutch Cross-Serial Dependencies

… because I_1 Cecilia$_2$ Henk$_3$ the hippopotamuses$_4$ saw$_1$ help$_2$ feed$_{3,4}$.



[Example from Stephen Clark]

Notes

- CCG is mildly context-sensitive
  - It can handle some phenomena that are not CFG
  - But it can be parsed efficiently
- CCG rules are monotonic
  - Movement is not allowed
- Complexity
  - $O(n^3)$ – with restricted type raising
  - Unbounded – with unrestricted type raising
- CCG gets closer to semantics and lambda calculus

CCGBank

- Hockenmaier and Steedman (2005)

Sentence 1

```
{S[dcl] {S[dcl] {NP {NP {NP {NP {N {N/N  Pierre}
                                   {N  Vinken}}}
                             {,  ,}}
                       {NP\NP {S[adj]\NP {NP {N {N/N  61}
                                                {N  years}}}
                                         {(S[adj]\NP)\NP  old}}}}
                {,  ,}}
          {S[dcl]\NP {(S[dcl]\NP)/(S[b]\NP)  will}
                     {S[b]\NP {S[b]\NP {(S[b]\NP)/PP {((S[b]\NP)/PP)/NP  join}
                                                     {NP {NP[nb]/N  the}
                                                         {N  board}}}
                                              {PP {PP/NP  as}
                                                  {NP {NP[nb]/N  a}
                                                      {N {N/N  nonexecutive}
                                                         {N  director}}}}}
                              {(S\NP)\(S\NP) {((S\NP)\(S\NP))/N  Nov.}
                                             {N  29}}}}}
   {.  .}}
```

Pierre	**(N/N)**	Vinken		
61	**(N/N)**	years		
old	**((S[adj]\NP)\NP)**	*Vinken*	years	
will	**((S[dcl]\NP)/(S[b]\NP))**	Vinken	join	
join	**(((S[b]\NP)/PP)/NP)**	*Vinken*	as	board
the	**(NP[nb]/N)**	board		
as	**(PP/NP)**	director		
a	**(NP[nb]/N)**	director		
nonexecutive	**(N/N)**	director		
Nov.	**(((S\NP)\(S\NP))/N)**		join	29

NLP