Lecture 6
CFGs – Good for your health or not?
Lord of the Loops:
Three loops to rule them all

Professor Robert C. Berwick
berwick@csail.mit.edu

Ambiguity is…everywhere

• Part of speech tag: dog/dog N/V
Syntactic Ambiguity & Human processing

- I told Bill that my dog died yesterday
- I told Bill that my dog will die yesterday

What are the possible syntactic relationships in a sentence?

What other relationships are needed?
Warlpiri: a ‘free word order language’ but…

Ngajulu-rlu ka-rna-rla punta-mi kurdu-ku karli
‘I am taking the boomerang from the child’
‘I tense-3-1 take the child-from boomerang’
A remarkable story about a remarkable person
The return of Dr. No?

Figure 4: Schematic view of activation pattern for the main effect of hierarchy in the language domain. For Artificial Grammar I and II, the main effect of hierarchy was found in Broca’s area (BA 44/45) (Friederici et al. 2006, Bahlmann et al. 2008). For the natural grammar, the main effect of hierarchy was located in BA 44 (Makuuchi et al. 2009) and in the posterior superior temporal gyrus (pSTG) extending into the superior temporal sulcus (Friederici et al. 2009).

Key: BA = Brodmann Area; CS = central sulcus; IFS = inferior frontal sulcus; STG = superior temporal gyrus.

Activation areas are different for ‘normal’ language as opposed to ‘puzzle solving’

Maria comprè le no caffè

Maria comprè le caffè
What about engineered parsers?

A CFG challenge

JOHN GAVE MARY AND BILL TO FRED BOOKS THAT LOOKED REMARKABLY SIMILAR
JOHN HUMMED AND MARY SANG AT EQUAL VOLUMES
JOHN HUMMED AND MARY SANG THE same tune
I have been wondering whether but would not want to state that your theory is correct
I can tell you when but I can not tell you why bill left me
I like but bill does not like to visit new cities
John has claimed but I do not believe that bill is a communist
Mary used to be and bill Still is very suspicious
Mary said she would and bill actually did eat a raw eggplant
Bill may be and John certainly is a werewolf
Bill offered and Mary gave John a cadillac
Bill caught and Mary killed the rabid dog
The woman who was here believed that the man was ill
The woman believed that the guy who was here was ill
The woman believed that the guy was ill who was here
TO WHICH CITY AND WHICH CONFERENCE DID BILL GO TO
WHICH CITY AND TO WHICH CONFERENCE DID BILL GO TO
WHICH CITY AND WHICH CONFERENCE DID BILL GO TO
TO WHICH CITY AND WHICH CONFERENCE DID BILL GO
TO WHICH CITY AND TO WHICH CONFERENCE DID BILL GO
ON WHICH TABLE AND UNDER WHICH FLOWER POT DID MARY PUT THE KEYS
WHERE AND WHEN DID BILL PUT THE BOOK
WHICH BOOK AND WHICH PENCIL DID MARY BUY
MARY ASKED WHO AND WHAT BOUGHT
MARY ASKED WHO AND WHERE BILL HAD SEEN
I WONDER WHO MARY LIKES AND HOPES WILL WIN
I WONDER WHO BILL SAW AND LIKED MARY
I WONDER WHO SEEN BILL AND LIKED MARY
I WONDER WHO BILL SAW AND MARY LIKED
Phrase structure rules can be eliminated!

Let’s do this one step at a time.
First: Why do we have the first rule but not the second?

\[
\begin{align*}
\text{NP} & \rightarrow \ldots \text{NN}\ldots \\
\text{NP} & \rightarrow \ldots \text{V}\ldots 
\end{align*}
\]

Both are equally ‘costly’ in terms of ‘minimum description length’
But the second one is impossible, it seems
1970

- Whenever a phrase has an XP, it has an X (1970)
- Whenever a phrase has an X, it has an XP (1982)
- (together, a bi-conditional)
- There are no phrases, just lexical items, with there properties (which properties?) projected

The short shelf life of context-free grammars

- Introduced in 1963 to generate the ‘deep structure’ of a language (ie, predicate-argument relations)
- But by 1965, they half-disappear…and by 1970.. they’re nearly history…by 1989, they are gone!
- What happened and why?
What happened?

CFG has rules like this:
VP → V₁ NP (*eat ice-cream*)
VP → V₃ PP(*talk about*)
VP → V₄ S (*think that John likes me*)
...

Dictionary (lexicon) has rules English learner has internalized like this:

*see*: ___NP  
*think*: ___ S  
*solve*: ___ NP

talk: ___ PP but crucially individual verbs include other idiosyncratic information! (eg, can *solve a problem*, but not *solve a hotdog*....)

What’s the point?

- The existence of rules like:
  VP → V NP
  VP → V PP
  VP → V S

*Implies* the existence of a lexical entry like *see*: ___NP
Why? Because if there were no entry like this, the rule would never have a chance to operate. (If no verb took a direct object, how would speakers ever know?)

But this means the rules & lexicon are redundant

If we have to choose between getting rid of the rules, and getting rid of the lexicon, we must get rid of the rules, because we don’t want to get rid of idiosyncratic facts. Deep point: CFG rules are entirely superfluous
But let’s work with CFGs anyway…

- How can we parse sentences efficiently with CFGs?
- Top-down or bottom-up?
- How to avoid the bottomless recursive pit?
- How to avoid computing the same thing more than once?
- How to integrate top-down and bottom-up information sources?

\[
NP \rightarrow NP \; NP \\
NP \rightarrow \text{natural} | \text{language} | \text{processing} | \text{book}
\]
A toy grammar example

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>→</td>
</tr>
<tr>
<td>S</td>
<td>→</td>
</tr>
<tr>
<td>VP</td>
<td>→</td>
</tr>
<tr>
<td>VP</td>
<td>→</td>
</tr>
<tr>
<td>VP</td>
<td>→</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
</tr>
<tr>
<td>NP</td>
<td>→</td>
</tr>
<tr>
<td>PP</td>
<td>→</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Vi</td>
<td>→</td>
</tr>
<tr>
<td>Vt</td>
<td>→</td>
</tr>
<tr>
<td>NN</td>
<td>→</td>
</tr>
<tr>
<td>NN</td>
<td>→</td>
</tr>
<tr>
<td>NN</td>
<td>→</td>
</tr>
<tr>
<td>NN</td>
<td>→</td>
</tr>
<tr>
<td>DT</td>
<td>→</td>
</tr>
<tr>
<td>IN</td>
<td>→</td>
</tr>
<tr>
<td>IN</td>
<td>→</td>
</tr>
</tbody>
</table>

0 the 1 dog 2 saw 3 the 4 man 5

Lord of the Loops:
Order of operations – CKY algorithm

Loop 1: Go Across columns, word by word, index j

Loop 2: Go Up a column, row by row, index i
Lord of the Loops:
Order of operations – CKY algorithm

Loop 1: Go Across columns, word by word, index j

Loop 2: Go Up a column, row by row, index i

Loop 3: paste 2 previous entries (subtrees) together, index k

CKY: diagonal elements = the words

the
dog
saw
the
man