CS 294-5: Statistical Natural Language Processing

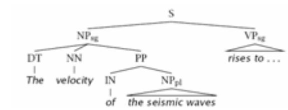
Syntax, Ambiguities, and Parsing

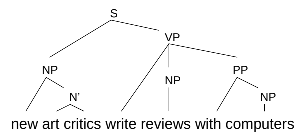
Lecture 13: 10/19/05

Slides from Manning, Sarkar, Lascarides, Xu

Phrase Structure Parsing

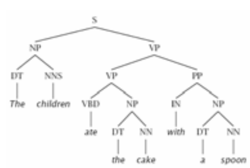
- Phrase structure parsing organizes syntax into *constituents* or *brackets*
- In general, this involves nested trees
- Linguists can, and do, argue about details
- Lots of ambiguity
- Not the only kind of syntax…





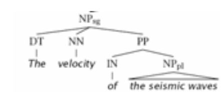
Constituency Tests

- How do we know what nodes go in the tree?
- Classic constituency tests:
  - Substitution by *proform*
  - Question answers
  - Semantic reference
  - Dislocation
- Cross linguistic arguments, too



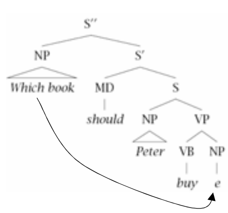
Conflicting Tests

- Constituency isn't always clear
  - Units of transfer:
    - think about ~ penser à
    - talk about ~ hablar de
  - Phonological reduction:
    - I will go → I'll go
    - I want to go → I wanna go
    - a le centre → au centre



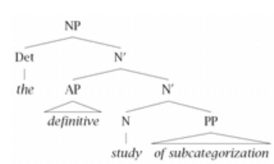
Non-Local Phenomena

- Dislocation / gapping
  - Why did the postman think that the neighbors were home?
  - A debate arose which continued until the election.
- Binding
  - Reference
    - The IRS audits itself
  - Control
    - I want to go
    - I want you to go

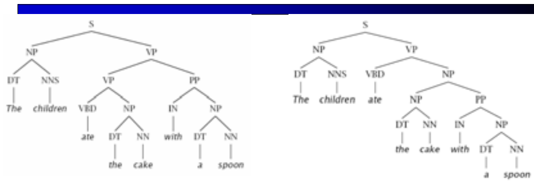


Regularity of Rules

- Argumentation
- Adjunction
- Coordination
- X' Theory



PP Attachment



PP Attachment

V	N1	P	N2	Attachment
join	board	as	director	V
is	chairman	of	N.V.	N
using	crocidolite	in	filters	V
bring	attention	to	problem	V
is	asbestos	in	products	N
making	paper	for	filters	N
including	three	with	cancer	N

Method	Accuracy
Always noun attachment	59.0
Most likely for each preposition	72.2
Average Human (4 head words only)	88.2
Average Human (whole sentence)	93.2

Attachment is a Simplification

- I cleaned the dishes from dinner
- I cleaned the dishes with detergent
- I cleaned the dishes in the sink

Syntactic Ambiguities I

- Prepositional phrases:
  They cooked the beans in the pot on the stove with handles.
- Particle vs. preposition:
  A good pharmacist dispenses with accuracy.
  The puppy tore up the staircase.
- Complement structures
  The tourists objected to the guide that they couldn't hear.
  She knows you like the back of her hand.
- Gerund vs. participial adjective
  Visiting relatives can be boring.
  Changing schedules frequently confused passengers.

Syntactic Ambiguities II

- Modifier scope within NPs
  impractical design requirements
  plastic cup holder
- Multiple gap constructions
  The chicken is ready to eat.
  The contractors are rich enough to sue.
- Coordination scope:
  Small rats and mice can squeeze into holes or cracks in the wall.

Treebank Sentences

```
( (S (NP-SBJ The move)
     (VP followed
         (NP (NP a round)
             (PP of
                 (NP (NP similar increases)
                     (PP by
                         (NP other lenders))
                     (PP against
                         (NP Arizona real estate loans)))))
         ,
         (S-ADV (NP-SBJ *)
                (VP reflecting
                    (NP (NP a continuing decline)
                        (PP-LOC in
                                (NP that market))))))
     .))
```

Human Processing

- Garden pathing:

  the man who hunts ducks out on weekends

  the cotton shirts are made from grows in Mississippi

  the old train the young

  the daughter of the king's son loves himself

- Ambiguity maintenance

  Have the police … eaten their supper?
  come in and look around.
  taken out and shot.

Context-Free Grammars

- A context free grammar is a tuple <*N, T, S, R*>
  - *N* : the set of non-terminals
    - Phrasal categories: S, NP, VP, ADJP, etc.
    - Parts-of-speech (pre-terminals): NN, JJ, DT, VB
  - *T* : the set of terminals (the words)
  - *S* : the start symbol
    - Often written as ROOT or TOP
    - *Not* usually the sentence non-terminal S
  - *R* : the set of rules
    - Of the form $X \rightarrow Y_1\ Y_2 \ldots Y_k$, with $X, Y_i \in N$
    - Examples: S → NP VP, VP → VP CC VP
    - Also called rewrites, productions, or local trees

Example CFG

- Can just write the grammar (rules with non-terminal LHSs) and lexicon (rules with pre-terminal LHSs)

Grammar		Lexicon
ROOT → S	NP → NNS	JJ → new
S → NP VP	NP → NN	NN → art
VP → VBP	NP → JJ NP	NNS → critics
VP → VBP NP	NP → NP NNS	NNS → reviews
VP → VP PP	NP → NP PP	NNS → computers
PP → IN NP		VBP → write
		IN → with

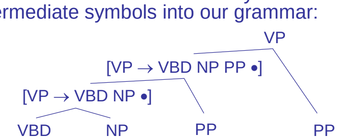
N-Ary Rules

- Often we want to write grammar rules like

  VP → VBD NP PP PP

  which are not binary.

- We can work with these rules by introducing new intermediate symbols into our grammar:

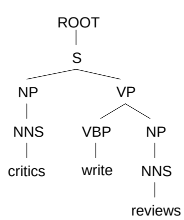


- We'll hear much more about this kind of thing when we talk about grammar representation later in the course.

Top-Down Generation from CFGs

- A CFG *generates* a language
- Fix an order: apply rules to leftmost non-terminal

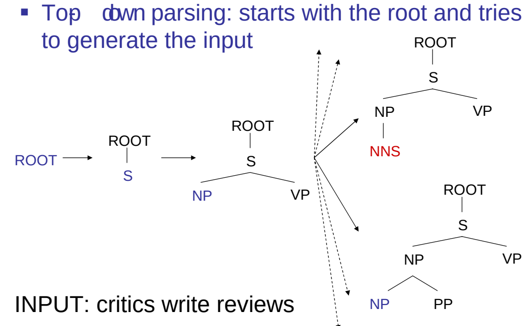
  ROOT
  S
  NP VP
  NNS VP
  critics VP
  critics VBP NP
  critics write NP
  critics write NNS
  critics write reviews



- Gives a *derivation* of a tree using rules of the grammar

Parsing as Search: Top-Down

- Top down parsing: starts with the root and tries to generate the input



How Top-Down Fails

- Big problem 1: search isn't guided by the input

INPUT: critics write reviews

NP VP → NP PP VP → NP NNS PP VP → JJ NP NNS PP VP

- Big problem 2: separate ambiguities create redundant work

new art critics write reviews with computers

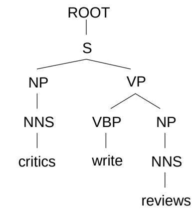
NP VP → [NP] NNS VP → [new art] critics VP → [new art] critics VBP NP / [new art] critics VBP NP PP

NP VP → [JJ NP] VP → new [art critics] VP → new [art critics] VBP NP / new [art critics] VBP NP PP

Parsing as Search: Bottom-Up

- Bottom-up parsing: input drives the search (shift-reduce parsing)

critics write reviews
NNS write reviews
NP write reviews
NP VBP reviews
NP VBP NNS
NP VBP NP
NP VP
S
ROOT



How Bottom-Up Fails

- Big Problem 1: ambiguities still create redundant work

new art critics write reviews with computers

new art critics write reviews with computers → [art critics] NP VBP NP PP → NP VBP NP / NP VP PP

new art critics write reviews with computers → [new art] NP VBP NP PP → NP VBP NP / NP VP PP

- Little Problem: partial analyses which can't be completed

art critics write reviews with art computers

[NP] [VP] computers

A Simple Chart Parser

- Chart parsers are sparse dynamic programs
- Ingredients:
  - Nodes: positions between words
  - Edges: spans of words with labels, represent the set of trees over those words rooted at x
  - A chart: records which edges we've built
  - An agenda: a holding pen for edges (a queue)
- We're going to figure out:
  - What edges can we build?
  - All the ways we built them.

PP

0 critics 1 write 2 reviews 3 with 4 computers 5

Word Edges

- An edge found for the first time is called discovered. Edges go into the agenda on discovery.
- To initialize, we discover all word edges.

AGENDA

critics[0,1], write[1,2], reviews[2,3], with[3,4], computers[4,5]

CHART [EMPTY]

0 1 2 3 4 5

critics write reviews with computers

Unary Projection

- When we pop an word edge off the agenda, we check the lexicon to see what tag edges we can build from it

critics[0,1]	write[1,2]	reviews[2,3]	with[3,4]	computers[4,5]
NNS[0,1]	VBP[1,2]	NNS[2,3]	IN[3,4]	NNS[3,4]

0 critics 1 write 2 reviews 3 with 4 computers 5

critics write reviews with computers

The "Fundamental Rule"

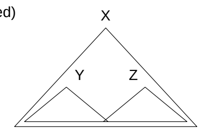
- When we pop edges off of the agenda:
  - Check for unary projections (NNS → critics, NP → NNS)

  Y[i,j] with X → Y forms X[i,j]

  - Combine with edges already in our chart (this is sometimes called the fundamental rule)

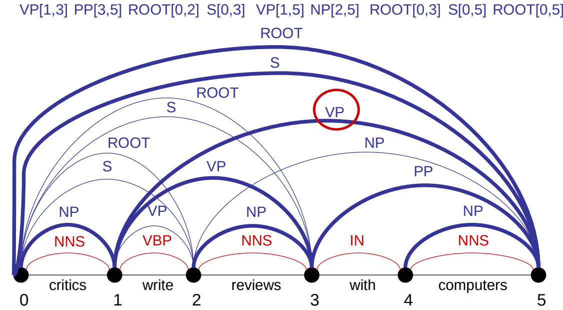
  Y[i,j] and Z[j,k] with X → Y Z form X[i,k]

  - Enqueue resulting edges (if newly discovered)
  - Record backtraces (called traversals)
  - Stick the popped edge in the chart
- Queries a chart must support:
  - Is edge X:[i,j] in the chart?
  - What edges with label Y end at position j?
  - What edges with label Z start at position i?



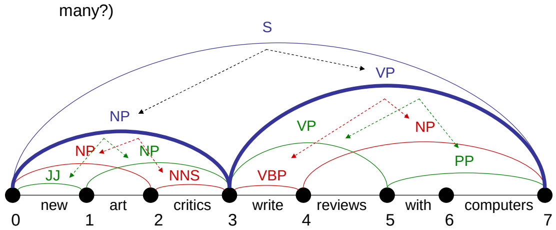
An Example

NNS[0,1] VBP[1,2] NNS[2,3] IN[3,4] NNS[3,4] NP[0,1] VP[1,2] NP[2,3] NP[4,5] S[0,2] VP[1,3] PP[3,5] ROOT[0,2] S[0,3] VP[1,5] NP[2,5] ROOT[0,3] S[0,5] ROOT[0,5]



Exploiting Substructure

- Each edge records all the ways it was built (locally)
  - Can recursively extract trees
  - A chart may represent too many parses to enumerate (how many?)



Order Independence

- A nice property:
  - It doesn't matter what policy we use to order the agenda (FIFO, LIFO, random).
  - Why? Invariant: before popping an edge:
    - Any edge X[i,j] that can be directly built from chart edges and a single grammar rule is either in the chart or in the agenda.
    - Convince yourselves this invariant holds!
  - This will not be true once we get weighted parsers.

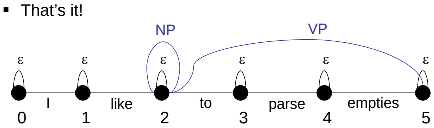
Empty Elements

- Sometimes we want to posit nodes in a parse tree that don't contain any pronounced words:

  I want John to parse this sentence

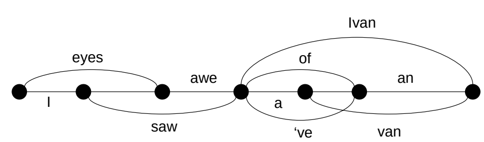
  I want [ ] to parse this sentence

- These are easy to add to our chart parser!
  - For each position i, add the "word" edge ε:[i,i]
  - Add rules like NP → ε to the grammar
  - That's it!



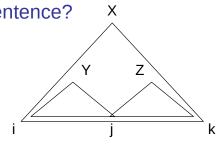
(Speech) Lattices

- There was nothing magical about words spanning exactly one position.
- When working with speech, we generally don't know how many words there are, or where they break.
- We can represent the possibilities as a lattice and parse these just as easily.



Runtime: Theory

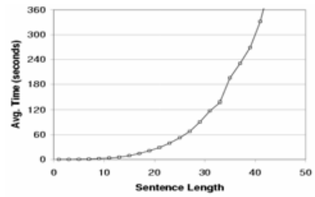
- How long does it take to parse a sentence?
- Depends on:
  - Sentence length
  - Grammar size (and structure)
  - Specific input sentences



- Asymptotically:
  - Do we do constant work per edge pop?
  - No, because one edge may combine with many others.
  - We do constant work per traversal (edge-edge combination).
  - How many traversals?
  - Form of traversal: Y[i,j] + Z[j,k] form Z[i,k].
  - So there are $O(n^3)$ traversals – cubic time.

Runtime: Practice

- Let's take the treebank grammar and go parsing!



~ 20K Rules

(not an optimized parser!)

Observed exponent:

3.6

- Yikes! Why's it worse in practice?
  - Longer sentences "unlock" more of the grammar
  - All kinds of systems issues don't scale