

Statistical NLP

Spring 2007

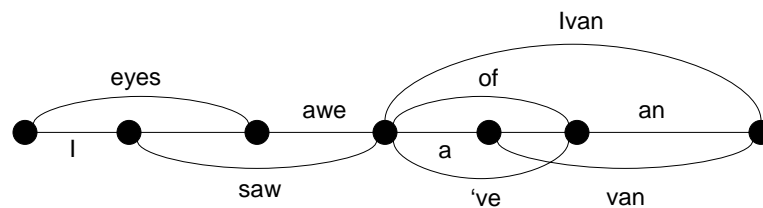


Lecture 16: PCFGs

Dan Klein – UC Berkeley

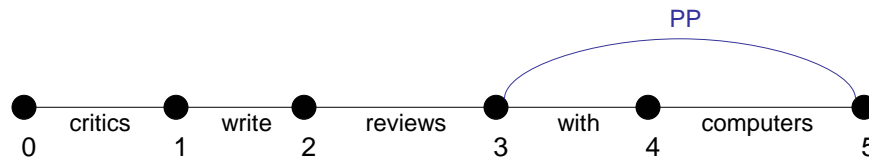
(Speech) Lattices

- 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.



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.



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]



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]



The “Fundamental Rule”

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

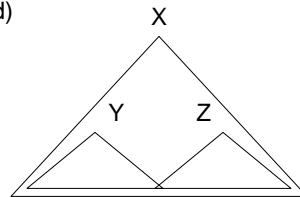
$Y[i,j]$ with $X \rightarrow Y$ forms $X[i,j]$

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

$Y[i,j]$ and $Z[j,k]$ with $X \rightarrow Y Z$ form $X[i,k]$

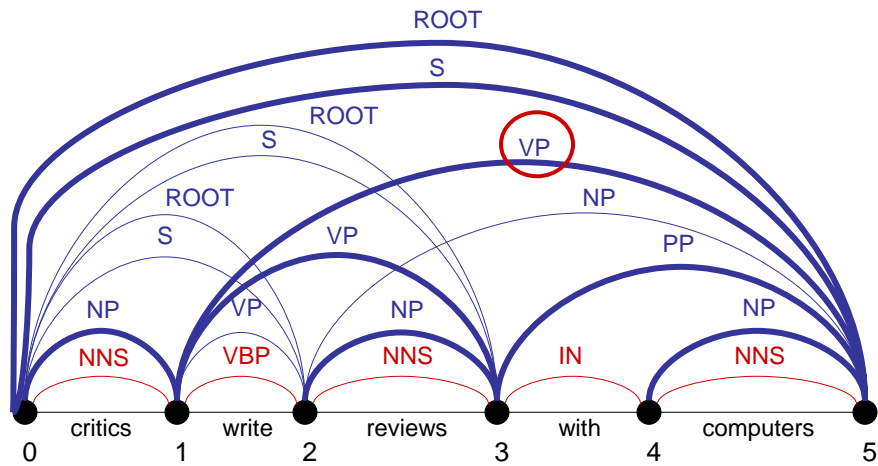
- Enqueue resulting edges (if newly discovered)
- Record backtraces (called traversals)
- Stick the popped edge in the chart

- 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 ?



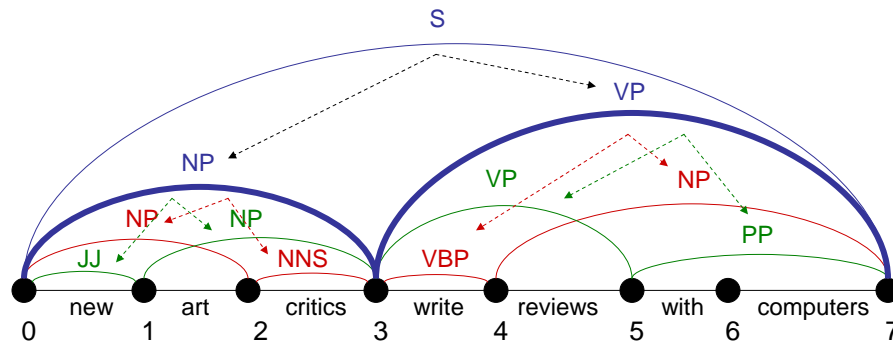
An Example

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.

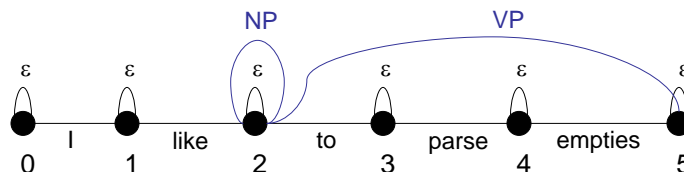
Empty Elements

- 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 $\varepsilon:[i,i]$
 - Add rules like $NP \rightarrow \varepsilon$ to the grammar
 - That's it!

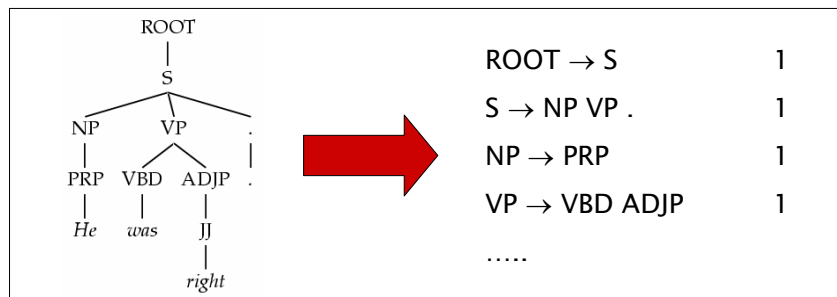


Treebank Sentences

```
( (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))))))
  .))
```

Treebank Parsing in 20 sec

- Need a PCFG for broad coverage parsing.
- Can take a grammar right off the trees (doesn't work well):



- Better results by enriching the grammar (e.g., lexicalization).
- Can also get reasonable parsers without lexicalization.

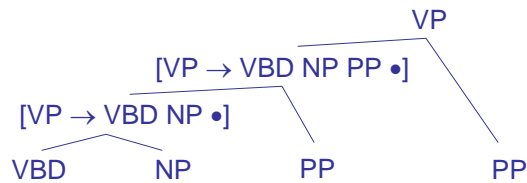
N-Ary Rules, Grammar States

- Often we want to write grammar rules like

$VP \rightarrow VBD\ NP\ PP\ PP$

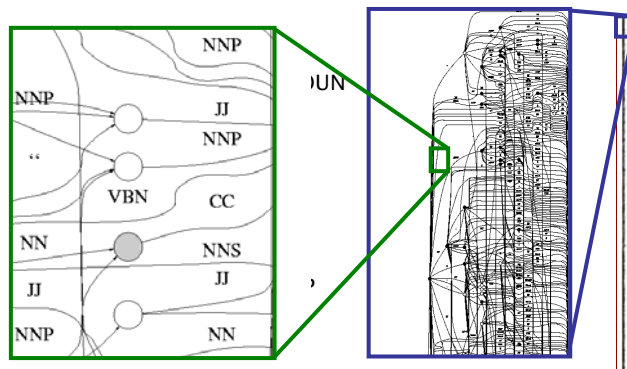
which are not binary.

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



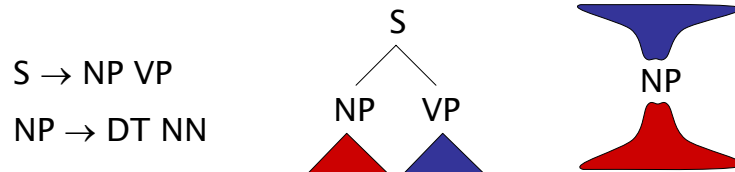
Treebank Grammar Scale

- Treebank grammars can be enormous!
 - As a set of FSTs, the raw grammar has ~10K states (why?).
 - Better parsers usually make the grammars larger, not smaller.



PCFGs and Independence

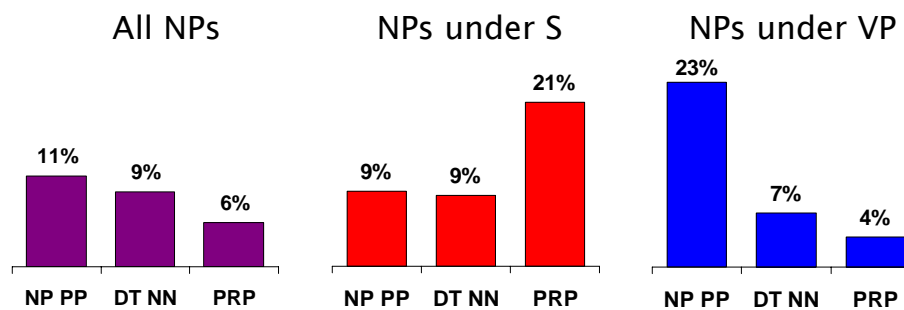
- Symbols in a PCFG define independence assumptions:



- At any node, the material inside that node is independent of the material outside that node, given the label of that node.
- Any information that statistically connects behavior inside and outside a node must flow through that node.

Non-Independence I

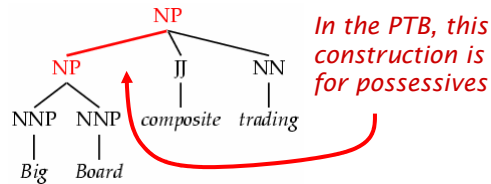
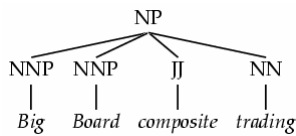
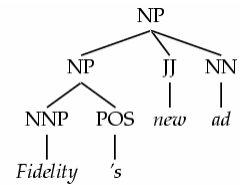
- Independence assumptions are often too strong.



- Example: the expansion of an NP is highly dependent on the parent of the NP (i.e., subjects vs. objects).
- Also: the subject and object expansions are correlated!

Non-Independence II

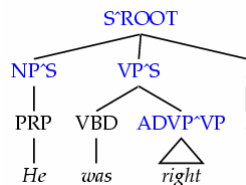
- Who cares?
 - NB, HMMs, all make false assumptions!
 - For **generation**, consequences would be obvious.
 - For **parsing**, does it impact accuracy?
- Symptoms of overly strong assumptions:
 - Rewrites get used where they don't belong.
 - Rewrites get used too often or too rarely.



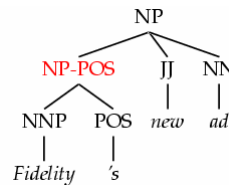
Breaking Up the Symbols

- We can relax independence assumptions by encoding dependencies into the PCFG symbols:

Parent annotation
[Johnson 98]



Marking
possessive NPs



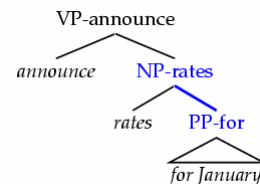
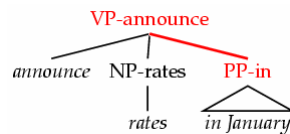
- What are the most useful "features" to encode?

Annotations

- Annotations split the grammar categories into sub-categories (in the original sense).
- Conditioning on history vs. annotating
 - $P(\text{NP}^{\text{S}} \rightarrow \text{PRP})$ is a lot like $P(\text{NP} \rightarrow \text{PRP} \mid \text{S})$
 - $P(\text{NP-POS} \rightarrow \text{NNP POS})$ isn't history conditioning.
- Feature / unification grammars vs. annotation
 - Can think of a symbol like $\text{NP}^{\text{NP-POS}}$ as $\text{NP} [\text{parent:NP}, +\text{POS}]$
- After parsing with an annotated grammar, the annotations are then stripped for evaluation.

Lexicalization

- Lexical heads important for certain classes of ambiguities (e.g., PP attachment):
- Lexicalizing grammar creates a much larger grammar. (cf. next week)
 - Sophisticated smoothing needed
 - Smarter parsing algorithms
 - More data needed
- How necessary is lexicalization?
 - Bilexical vs. monolexical selection
 - Closed vs. open class lexicalization



Unlexicalized PCFGs

- What is meant by an “unlexicalized” PCFG?
 - Grammar not systematically specified to the level of lexical items
 - NP [stocks] is not allowed
 - NP^S-CC is fine
 - Closed vs. open class words (NP^S [the])
 - Long tradition in linguistics of using function words as features or markers for selection
 - Contrary to the bilexical idea of semantic heads
 - Open-class selection really a proxy for semantics
- It's kind of a gradual transition from unlexicalized to lexicalized (but heavily smoothed) grammars.

Typical Experimental Setup

- Corpus: Penn Treebank, WSJ

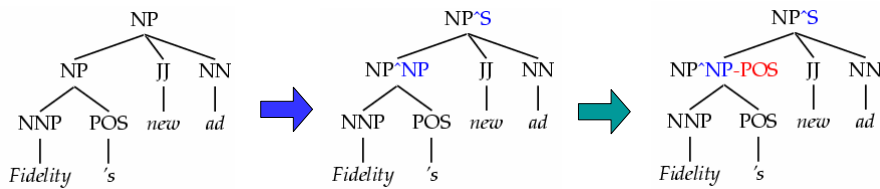


Training:	sections	02-21
Development:	section	22 (here, first 20 files)
Test:	section	23

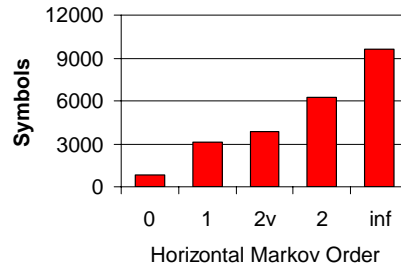
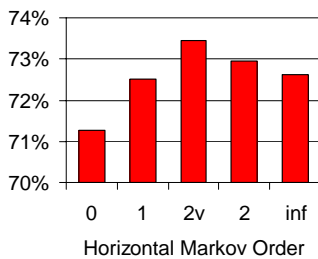
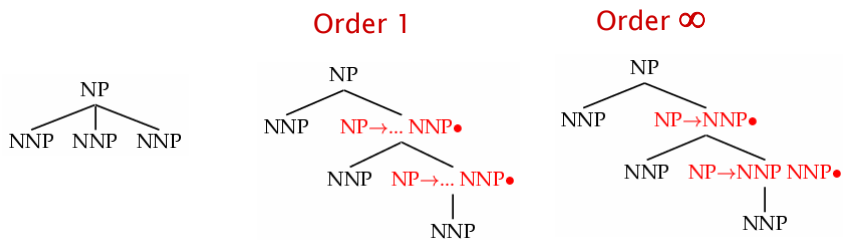
- Accuracy – F1: harmonic mean of per-node labeled precision and recall.
- Here: also size – number of symbols in grammar.
 - Passive / complete symbols: NP, NP^S
 - Active / incomplete symbols: NP → NP CC •

Multiple Annotations

- Each annotation done in succession
 - Order does matter
 - Too much annotation and we'll have sparsity issues

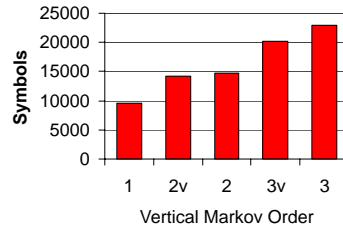
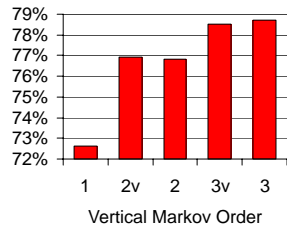
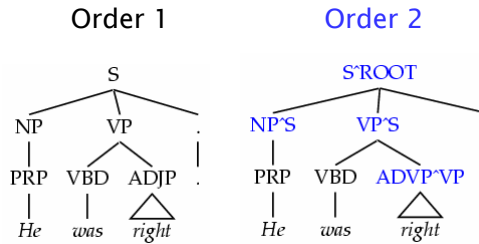


Horizontal Markovization

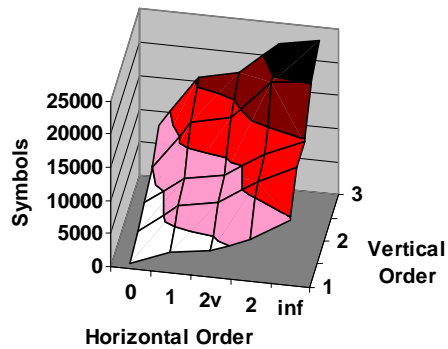
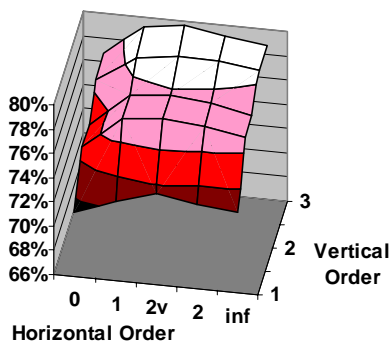


Vertical Markovization

- Vertical Markov order: rewrites depend on past k ancestor nodes. (cf. parent annotation)



Vertical and Horizontal

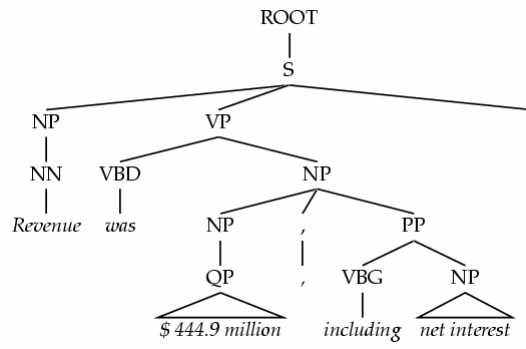


- Examples:
 - Raw treebank: $v=1, h=\infty$
 - Johnson 98: $v=2, h=\infty$
 - Collins 99: $v=2, h=2$
 - Best F1: $v=3, h=2v$

Model	F1	Size
Base: $v=h=2v$	77.8	7.5K

Unary Splits

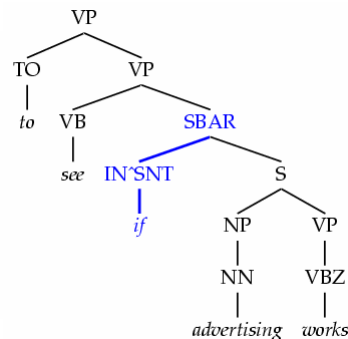
- Problem: unary rewrites used to transmute categories so a high-probability rule can be used.
- Solution: Mark unary rewrite sites with -U



Annotation	F1	Size
Base	77.8	7.5K
UNARY	78.3	8.0K

Tag Splits

- Problem: Treebank tags are too coarse.
- Example: Sentential, PP, and other prepositions are all marked IN.
- Partial Solution:
 - Subdivide the IN tag.



Annotation	F1	Size
Previous	78.3	8.0K
SPLIT-IN	80.3	8.1K

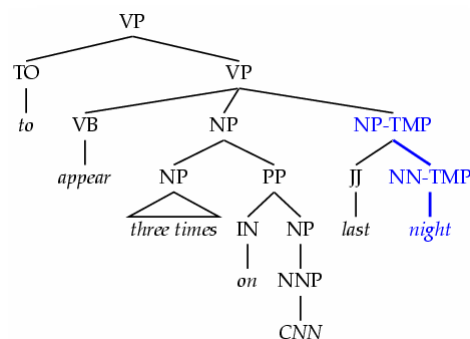
Other Tag Splits

- UNARY-DT: mark demonstratives as DT^U (“the X” vs. “those”)
- UNARY-RB: mark phrasal adverbs as RB^U (“quickly” vs. “very”)
- TAG-PA: mark tags with non-canonical parents (“not” is an RB^VP)
- SPLIT-AUX: mark auxiliary verbs with –AUX [cf. Charniak 97]
- SPLIT-CC: separate “but” and “&” from other conjunctions
- SPLIT-%: “%” gets its own tag.

F1	Size
80.4	8.1K
80.5	8.1K
81.2	8.5K
81.6	9.0K
81.7	9.1K
81.8	9.3K

Treebank Splits

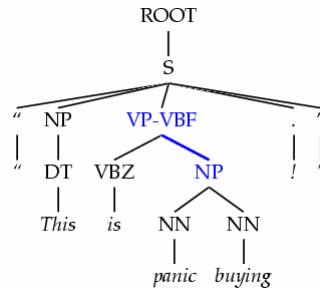
- The treebank comes with some annotations (e.g., -LOC, -SUBJ, etc).
 - Whole set together hurt the baseline.
 - One in particular is very useful (NP-TMP) **when pushed down to the head tag (why?)**.
 - Can mark gapped S nodes as well.



Annotation	F1	Size
Previous	81.8	9.3K
NP-TMP	82.2	9.6K
GAPPED-S	82.3	9.7K

Yield Splits

- Problem: sometimes the behavior of a category depends on something inside its future yield.

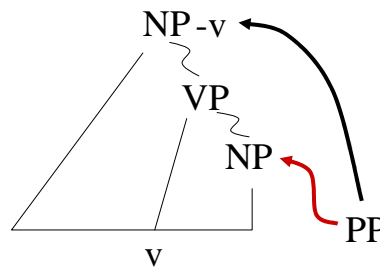


- Examples:
 - Possessive NPs
 - Finite vs. infinite VPs
 - Lexical heads!
- Solution: annotate future elements into nodes.
 - Lexicalized grammars do this (in very careful ways – why?).

Annotation	F1	Size
Previous	82.3	9.7K
POSS-NP	83.1	9.8K
SPLIT-VP	85.7	10.5K

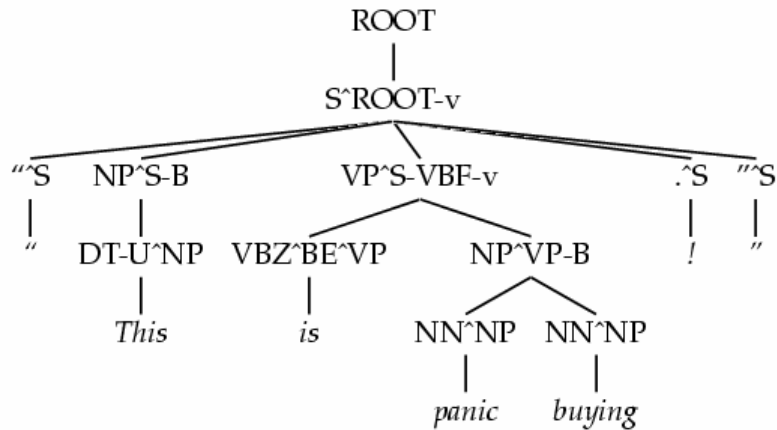
Distance / Recursion Splits

- Problem: vanilla PCFGs cannot distinguish attachment heights.
- Solution: mark a property of higher or lower sites:
 - Contains a verb.
 - Is (non)-recursive.
 - Base NPs [cf. Collins 99]
 - Right-recursive NPs



Annotation	F1	Size
Previous	85.7	10.5K
BASE-NP	86.0	11.7K
DOMINATES-V	86.9	14.1K
RIGHT-REC-NP	87.0	15.2K

A Fully Annotated (Unlex) Tree

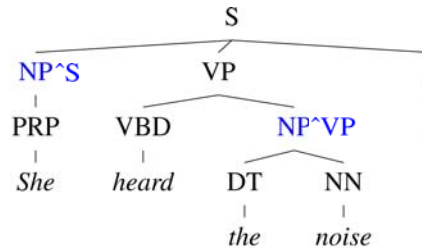


Some Test Set Results

Parser	LP	LR	F1	CB	0 CB
Magerman 95	84.9	84.6	84.7	1.26	56.6
Collins 96	86.3	85.8	86.0	1.14	59.9
Unlexicalized	86.9	85.7	86.3	1.10	60.3
Charniak 97	87.4	87.5	87.4	1.00	62.1
Collins 99	88.7	88.6	88.6	0.90	67.1

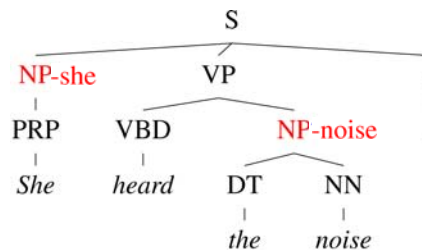
- Beats "first generation" lexicalized parsers.
- Lots of room to improve – more complex models next.

The Game of Designing a Grammar



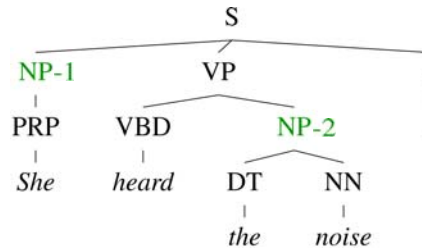
- Annotation refines base treebank symbols to improve statistical fit of the grammar
 - Parent annotation [Johnson '98]

The Game of Designing a Grammar



- Annotation refines base treebank symbols to improve statistical fit of the grammar
 - Parent annotation [Johnson '98]
 - Head lexicalization [Collins '99, Charniak '00]

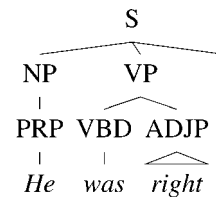
The Game of Designing a Grammar



- Annotation refines base treebank symbols to improve statistical fit of the grammar
 - Parent annotation [Johnson '98]
 - Head lexicalization [Collins '99, Charniak '00]
 - Automatic clustering?

Manual Annotation

- Manually split categories
 - NP: subject vs object
 - DT: determiners vs demonstratives
 - IN: sentential vs prepositional



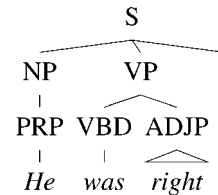
- Advantages:
 - Fairly compact grammar
 - Linguistic motivations
- Disadvantages:
 - Performance leveled out
 - Manually annotated

<i>Model</i>	<i>F1</i>
Naïve Treebank Grammar	72.6
Klein & Manning '03	86.3

Automatic Annotation Induction

- Advantages:

- Automatically learned:
 - Label *all* nodes with latent variables.
 - Same number k of subcategories for all categories.



- Disadvantages:

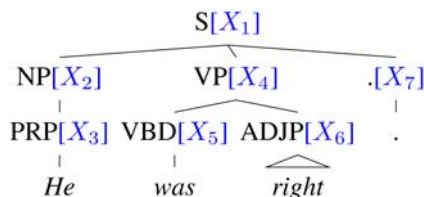
- Grammar gets too large
- Most categories are oversplit while others are undersplit.

Model	F1
Klein & Manning '03	86.3
Matsuzaki et al. '05	86.7

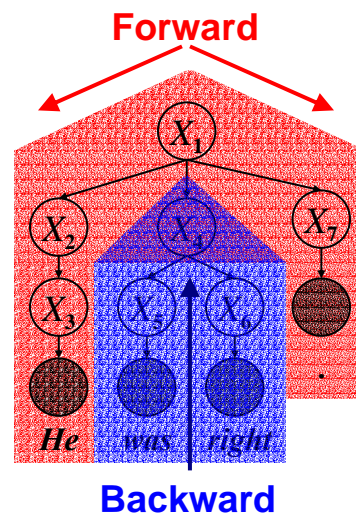
Learning Latent Annotations

EM algorithm:

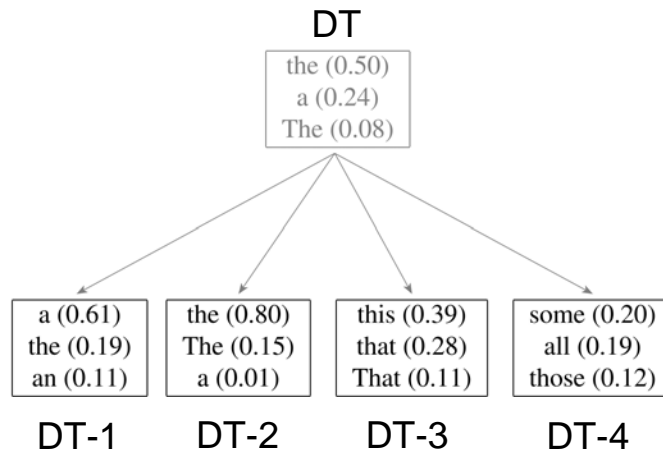
- Brackets are known
- Base categories are known
- Only induce subcategories



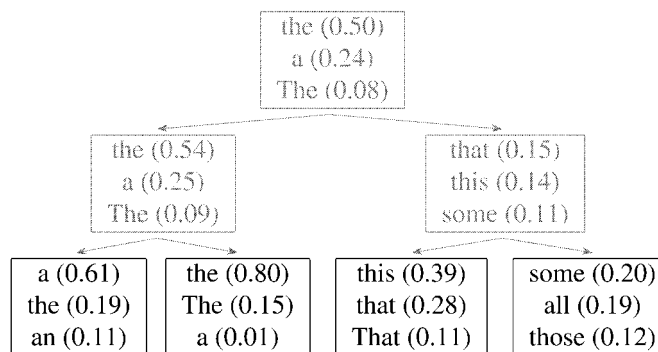
Just like Forward-Backward for HMMs.



Refinement of the DT tag

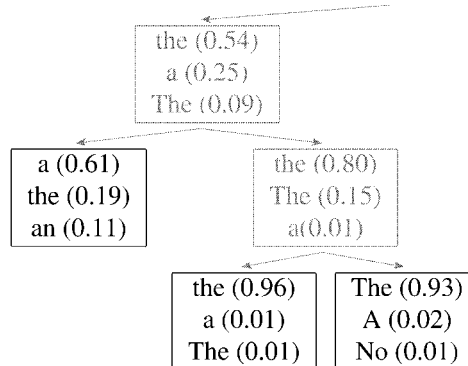


Hierarchical refinement



Adaptive Splitting

- Want to split complex categories more
- Idea: split everything, roll back splits which were least useful



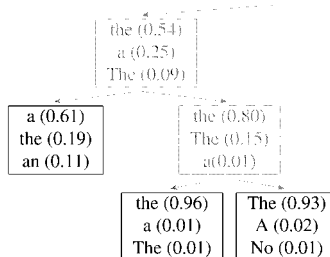
Adaptive Splitting

- Evaluate loss in likelihood from removing each split =

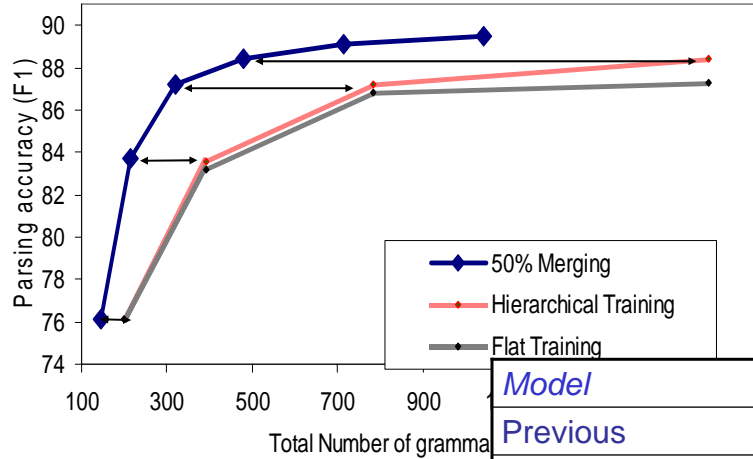
Data likelihood with split reversed

Data likelihood with split

- No loss in accuracy when 50% of the splits are reversed.

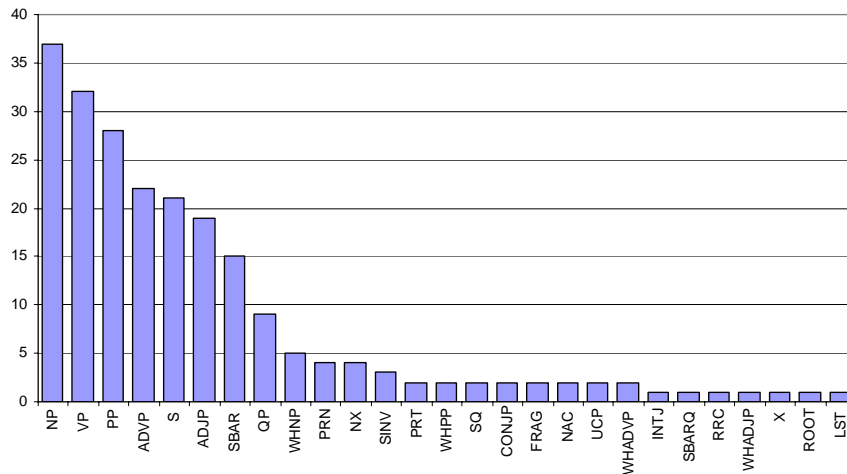


Adaptive Splitting Results

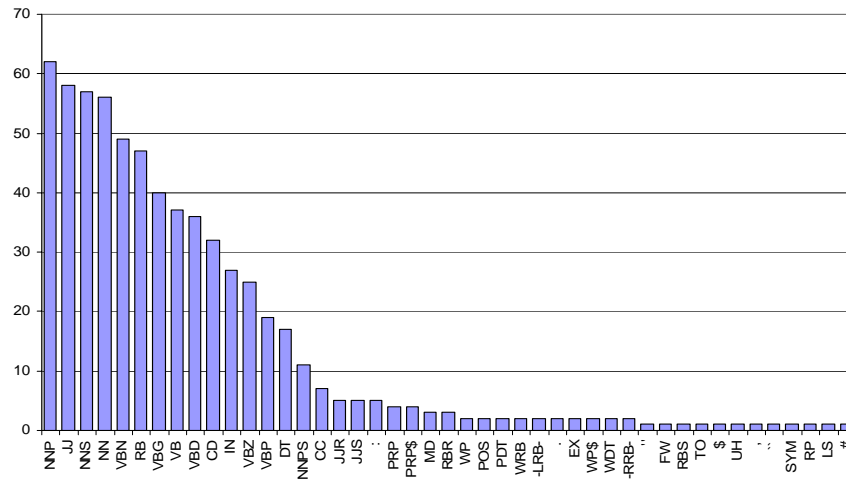


Model	F1
Previous	88.4
With 50% Merging	89.5

Number of Phrasal Subcategories



Number of Lexical Subcategories



Final Results

<i>Parser</i>	<i>F1 ≤ 40 words</i>	<i>F1 all words</i>
Klein & Manning '03	86.3	85.7
Matsuzaki et al. '05	86.7	86.1
Collins '99	88.6	88.2
Charniak & Johnson '05	90.1	89.6
Petrov et. al. 06	90.2	89.7

Learned Splits

- Proper Nouns (NNP):

NNP-14	Oct.	Nov.	Sept.
NNP-12	John	Robert	James
NNP-2	J.	E.	L.
NNP-1	Bush	Noriega	Peters
NNP-15	New	San	Wall
NNP-3	York	Francisco	Street

- Personal pronouns (PRP):

PRP-0	It	He	I
PRP-1	it	he	they
PRP-2	it	them	him

Learned Splits

- Relative adverbs (RBR):

RBR-0	further	lower	higher
RBR-1	more	less	More
RBR-2	earlier	Earlier	later

- Cardinal Numbers (CD):

CD-7	one	two	Three
CD-4	1989	1990	1988
CD-11	million	billion	trillion
CD-0	1	50	100
CD-3	1	30	31
CD-9	78	58	34