Lexicalization of PCFGs

Introduction

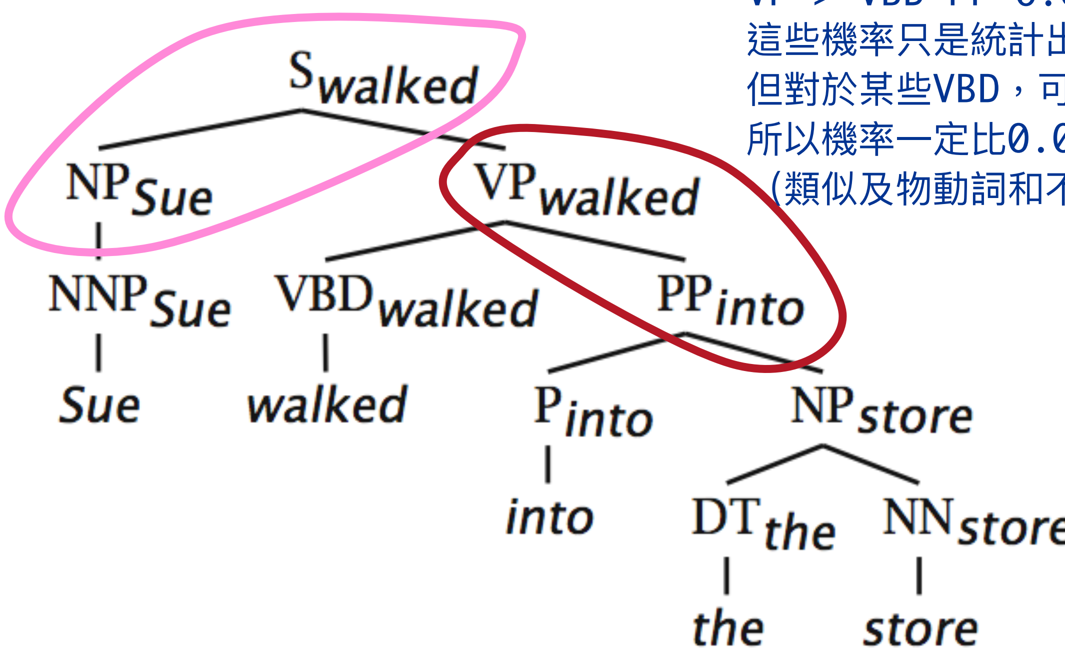
Christopher Manning

(Head) Lexicalization of PCFGs

[Magerman 1995, Collins 1997; Charniak 1997]

- The head word of a phrase gives a good representation of the phrase's structure and meaning
- Puts the properties of words back into a PCFG

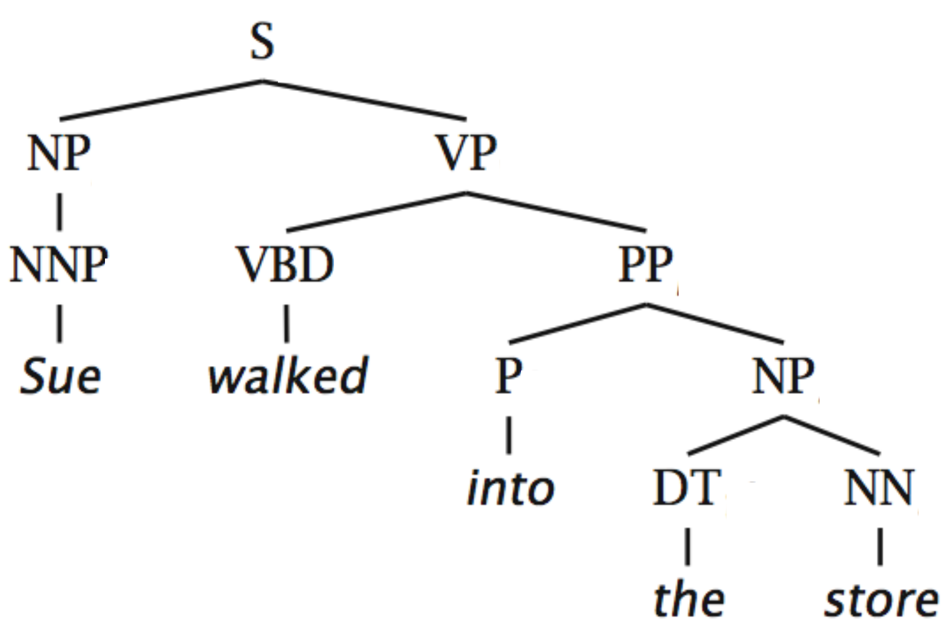
S -> NP VP 0.4
VP-> VBD PP 0.03
這些機率只是統計出來的整體機率
但對於某些VBD，可能後面就比較容易接PP
所以機率一定比0.03再高一些
(類似及物動詞和不及物動詞)



(Head) Lexicalization of PCFGs

[Magerman 1995, Collins 1997; Charniak 1997]

- The head word of a phrase gives a good representation of the phrase's structure and meaning
- Puts the properties of words back into a PCFG

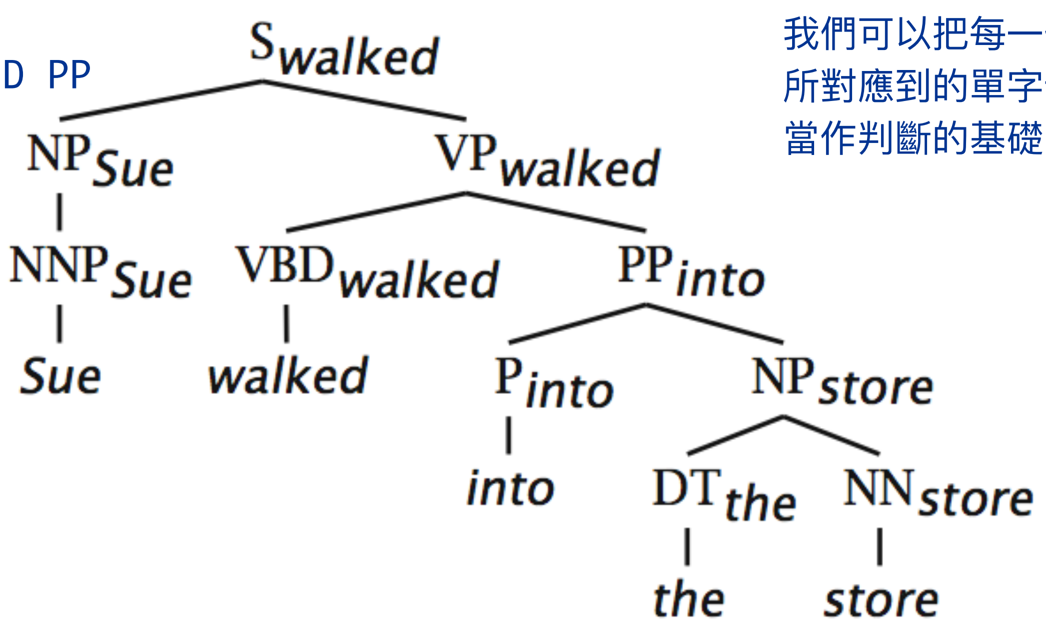


(Head) Lexicalization of PCFGs

[Magerman 1995, Collins 1997; Charniak 1997]

- The head word of a phrase gives a good representation of the phrase’s structure and meaning
- Puts the properties of words back into a PCFG

例如：
原本是VP->VBD PP
改寫成VP(walked)->VBD PP
可以解決很多歧義問題

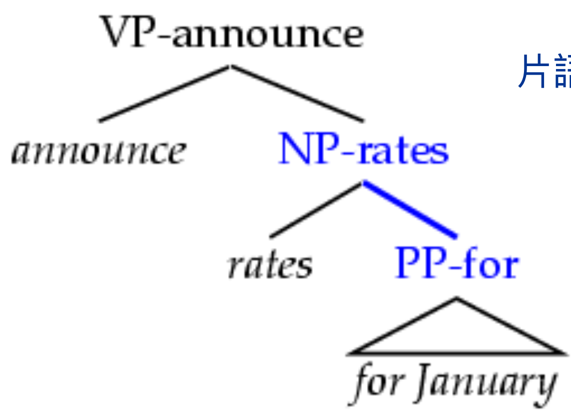


為了解決第二頁提到的問題
我們可以把每一個Terminal
所對應到的單字也加進Tree裡
當作判斷的基礎之一

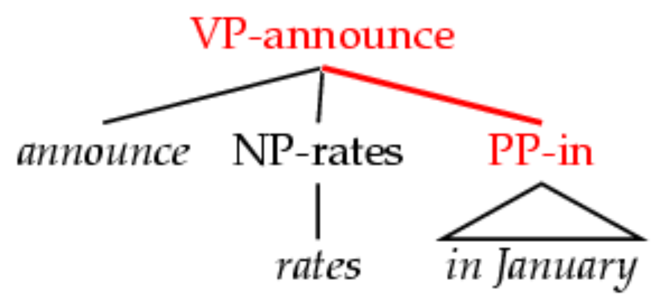
(Head) Lexicalization of PCFGs

[Magerman 1995, Collins 1997; Charniak 1997]

- Word-to-word affinities are useful for certain ambiguities
  - PP attachment is now (partly) captured in a local PCFG rule.
    - Think about: What useful information isn't captured?



片語的感覺



  - Also useful for: coordination scope, verb complement patterns

Lexicalized parsing was seen as *the* parsing breakthrough of the late 1990s

- Eugene Charniak, 2000 JHU workshop: “To do better, it is necessary to condition probabilities on the actual words of the sentence. This makes the probabilities much tighter:
  - p(VP → V NP NP) = 0.00151
  - p(VP → V NP NP | said) = 0.00001
  - p(VP → V NP NP | gave) = 0.01980 ”
- Michael Collins, 2003 COLT tutorial: “Lexicalized Probabilistic Context-Free Grammars … perform vastly better than PCFGs (88% vs. 73% accuracy)”

Lexicalization of PCFGs

Introduction

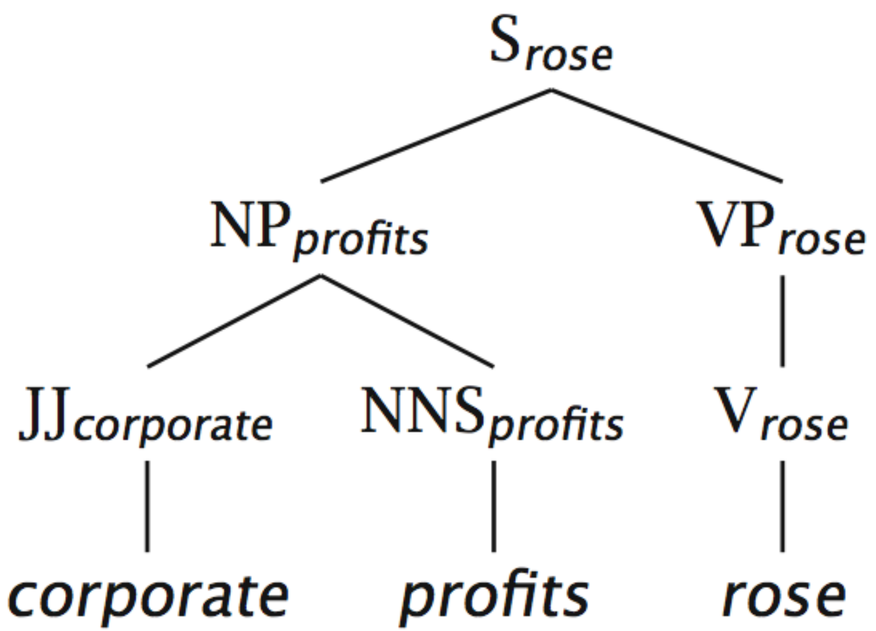
Christopher Manning

Lexicalization of PCFGs

The model of Charniak (1997)

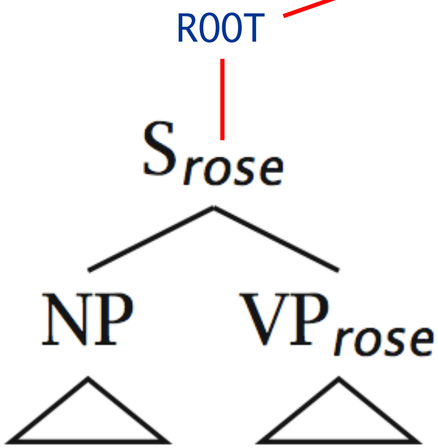
Charniak (1997)

- A very straightforward model of a lexicalized PCFG
- Probabilistic conditioning is “top-down” like a regular PCFG
  - But actual parsing is bottom-up, somewhat like the CKY algorithm we saw



對於ROOT的字，機率這樣算：
P(h=rose|c=ROOT), ph, pc不用管
P(r=S->NP VP|h=rose, c=S, pc=ROOT)

Charniak (1997) example



head word　　category

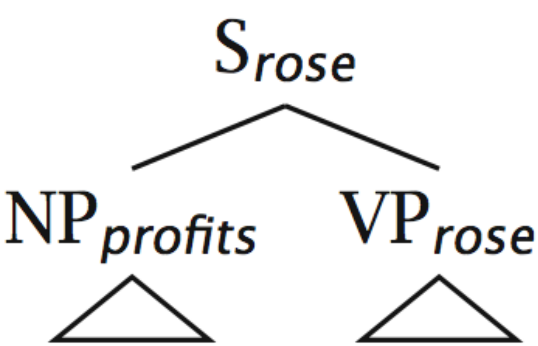
a. $h = \mathit{profits};\ c = \mathrm{NP}$

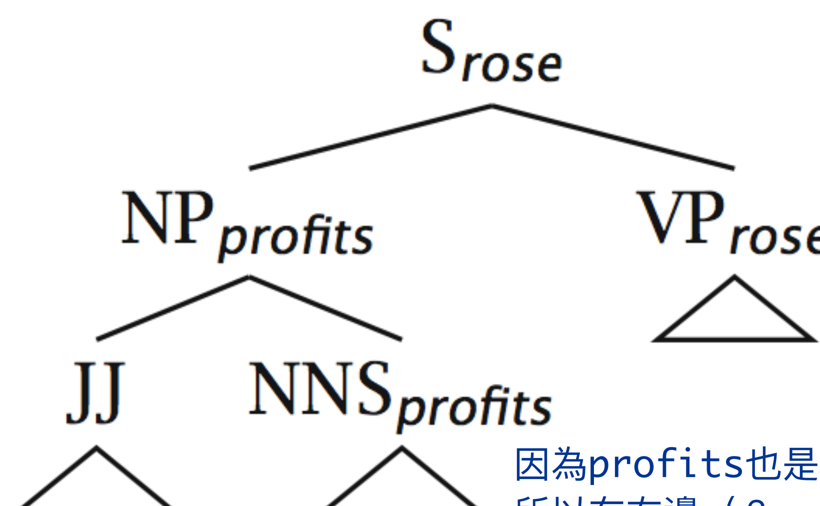
parent head word　parent category

b. $ph = \mathit{rose};\ pc = \mathrm{S}$

c. $P(h \mid ph, c, pc)$　head word(profits) 會取決於S, rose, NP

d. $P(r \mid h, c, pc)$　NP改寫的Rule會取決於 profits, NP, S





因為profits也是NNS
所以在右邊（？

Lexicalization models argument selection by sharpening rule expansion probabilities

head verb的不同
會產生VP的每個
Rules的不同機率

- The probability of different verbal complement frames (i.e., "subcategorizations") depends on the verb:

Local Tree	*come*	*take*	*think*	*want*
VP → V	9.5%	2.6%	4.6%	5.7%
VP → V NP	1.1%	32.1%	0.2%	13.9%
VP → V PP	34.5%	3.1%	7.1%	0.3%
VP → V SBAR	6.6%	0.3%	73.0%	0.2%
VP → V S	2.2%	1.3%	4.8%	70.8%
VP → V NP S	0.1%	5.7%	0.0%	0.3%
VP → V PRT NP	0.3%	5.8%	0.0%	0.0%
VP → V PRT PP	6.1%	1.5%	0.2%	0.0%

"monolexical" probabilities

Lexicalization sharpens probabilities: Predicting heads

"Bilexical probabilities"

- P(prices | n-plural) = .013
- P(prices | n-plural, NP) = .013
- P(prices | n-plural, NP, S) = .025
- P(prices | n-plural, NP, S, v-past) = .052
- P(prices | n-plural, NP, S, v-past, fell) = .146

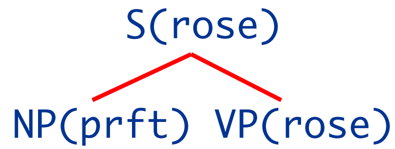
對於整個Rule的了解越多
就越能確認缺少的字應該填入哪一個字

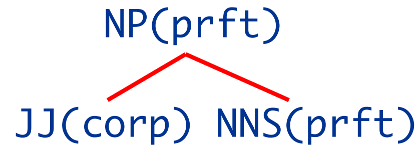
Charniak (1997) linear interpolation/ shrinkage

$$
\begin{aligned}
\hat{P}(h|ph, c, pc) \;=\; & \lambda_1(e) P_{\mathsf{MLE}}(h|ph, c, pc) \\
& + \lambda_2(e) P_{\mathsf{MLE}}(h|C(ph), c, pc) \\
& + \lambda_3(e) P_{\mathsf{MLE}}(h|c, pc) + \lambda_4(e) P_{\mathsf{MLE}}(h|c)
\end{aligned}
$$

- $\lambda_i(e)$ is here a function of how much one would expect to see a certain occurrence, given the amount of training data, word counts, etc.
- $C(ph)$ is semantic class of parent headword
- Techniques like these for dealing with data sparseness are vital to successful model construction

Christopher Manning





Charniak (1997) shrinkage example

	P(h|ph, c, pc)	P(h|ph, c, pc)
	$P(\mathrm{prft}\mid\mathrm{rose},\mathrm{NP},\mathrm{S})$	$P(\mathrm{corp}\mid\mathrm{prft},\mathrm{JJ},\mathrm{NP})$
$P(h\mid ph,c,pc)$	0	0.245
$P(h\mid C(ph),c,pc)$	0.00352	0.0150
$P(h\mid c,pc)$	0.000627	0.00533
$P(h\mid c)$	0.000557	0.00418

- Allows utilization of rich highly conditioned estimates, but smoothes when sufficient data is unavailable
- One can't just use MLEs: one commonly sees previously unseen events, which would have probability 0.

Lexicalization of PCFGs

The model of Charniak (1997)

Sparseness & the Penn Treebank

- The Penn Treebank – 1 million words of parsed English WSJ – has been a key resource (because of the widespread reliance on supervised learning)
- But 1 million words is like nothing:
  - 965,000 constituents, but only 66 WHADJP, of which only 6 aren't *how much* or *how many*, but there is an infinite space of these
    - *How clever/original/incompetent (at risk assessment and evaluation) …*
- Most of the probabilities that you would like to compute, you can't compute

Quiz question!

- Classify each of the italic red phrases as a:

  WHNP   WHADJP   WHADVP   WHPP

1. That explains *why* she is succeeding.
2. *Which student* scored highest on the assignment?
3. Nobody knows *how deep* the recession will be.
4. *During which class* did the slide projection not work?
5. *Whose iPhone* was stolen?

Sparseness & the Penn Treebank (2)

- Many parse preferences depend on bilexical statistics: likelihoods of relationships between pairs of words (compound nouns, PP attachments, …)
- Extremely sparse, even on topics central to the WSJ:
  - *stocks plummeted* 2 occurrences
  - *stocks stabilized* 1 occurrence
  - *stocks skyrocketed* 0 occurrences
  - #*stocks discussed* 0 occurrences
- There has been only modest success in augmenting the Penn Treebank with extra unannotated materials or using semantic classes – given a reasonable amount of annotated training data.
  - Cf. Charniak 1997, Charniak 2000
  - But McClosky et al. 2006 doing self-training and Koo and Collins2008 semantic classes are rather more successful!

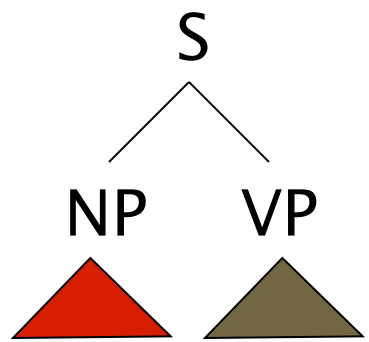
PCFG Independence Assumptions

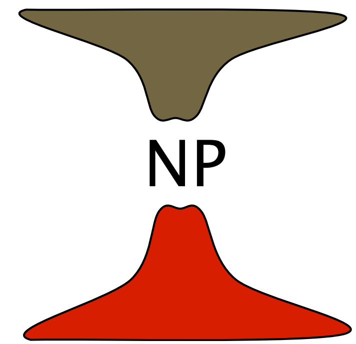
PCFGs and Independence

- The symbols in a PCFG define independence assumptions:

S → NP VP

NP → DT NN





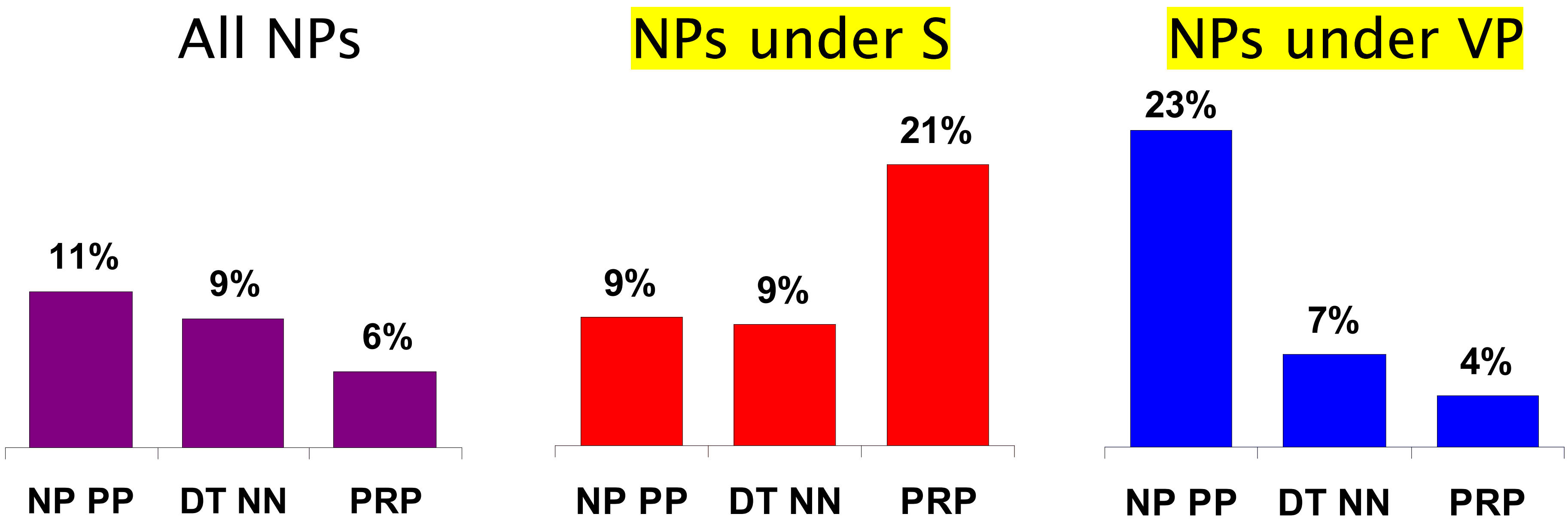
- At any node, the material inside that node is independent of the material outside that node, given the label of that node
- Any information that statistically connects behavior inside and outside a node must flow through that node's label

Independence Assumptions
代表你只需要知道你現在要分析的Non-terminal (例如NP)
就可以知道NP之前和NP之後的所有資訊
但這個假設真的正確嗎？ 看下一頁

Non-Independence I

這邊可以發現前一頁的Independence Assumptions是不太正確的
像NP改寫的Rules，就會受到NP的Parent所影響

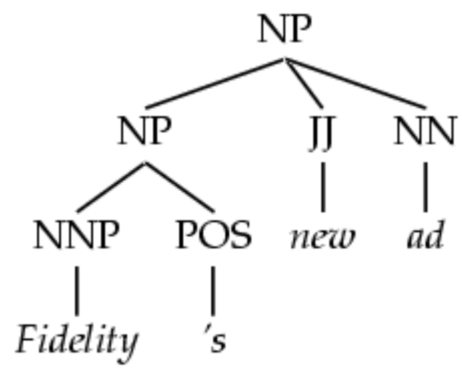
- The independence assumptions of a PCFG are often too strong

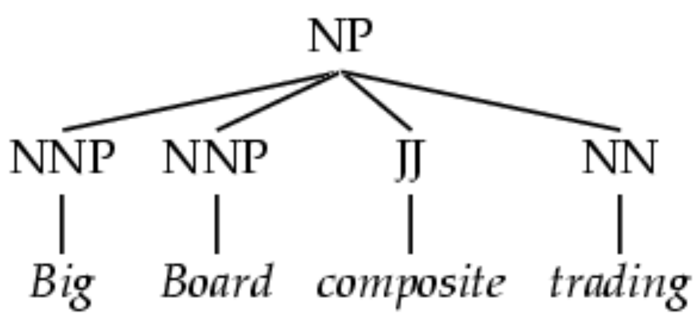


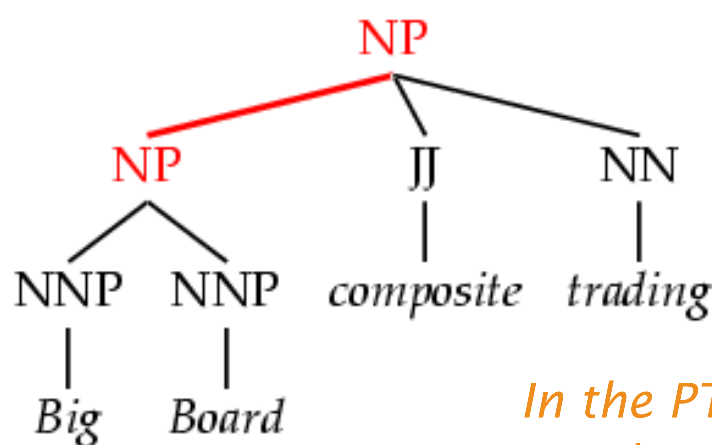
- Example: the expansion of an NP is highly dependent on the parent of the NP (i.e., subjects vs. objects)

Non-Independence II

- Symptoms of overly strong assumptions:
  - Rewrites get used where they don't belong







In the PTB, this construction is for possessives

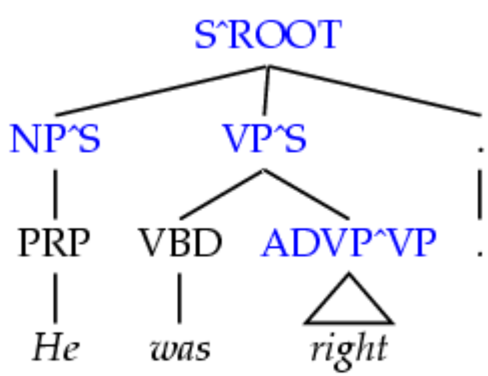
例如：
NP–>NNP NNP
這是給「所有格」名詞用的
所以如果你把最後改寫的單字納入考量
就會知道Big Board並沒有「所有格」
所以這句話就不會被改寫成NP–>NNP NNP

Refining the Grammar Symbols

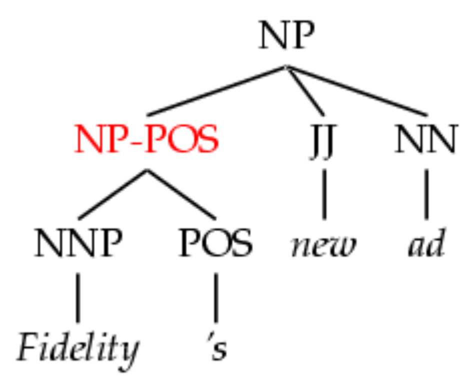
- We can relax independence assumptions by encoding dependencies into the PCFG symbols, by state splitting:

在每一個Non-terminal後面增加他的
Parent Non-terminal

Parent annotation
[Johnson 98]



Marking
possessive NPs



- Too much state-splitting → sparseness (no smoothing used!)
- What are the most useful features to encode?

PCFG Independence Assumptions

Annotations

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

The Return of Unlexicalized PCFGs

Accurate Unlexicalized Parsing

[Klein and Manning 1993]

- What do we mean by an "unlexicalized" PCFG?
  - Grammar rules are not systematically specified down to the level of lexical items
    - NP-stocks is not allowed
    - NP^S-CC is fine

```
unlexicalized PCDG
就是指改寫的Rule的第一層，無法改寫到單字，例如：
NP   -> stocks 不是unlexicalized PCFG
NP^S -> CC     就是unlexicalized PCFG
```

  - Closed vs. open class words
    - Long tradition in linguistics of using function words as features or markers for selection (VB-have, SBAR-if/whether)
    - Different to the bilexical idea of semantic heads
    - Open-class selection is really a proxy for semantics

- Thesis
  - Most of what you need for accurate parsing, and much of what lexicalized PCFGs actually capture *isn't* lexical selection between content words but just basic grammatical features, like verb form, finiteness, presence of a verbal auxiliary, etc.

Experimental Approach

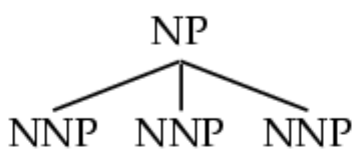
- Corpus: Penn Treebank, WSJ; iterate on small dev set

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

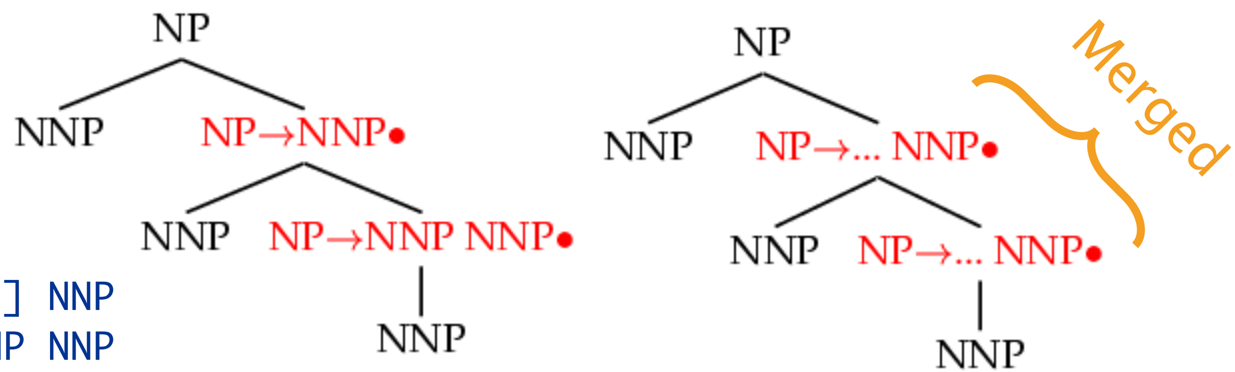
- Size – number of symbols in grammar.
  - Passive / complete symbols: NP, NP^S
  - Active / incomplete symbols: @NP_NP_CC [from binarization]
- We state-split as sparingly as possible
  - Highest accuracy with fewest symbols
  - Error-driven, manual hill-climb, one annotation at a time

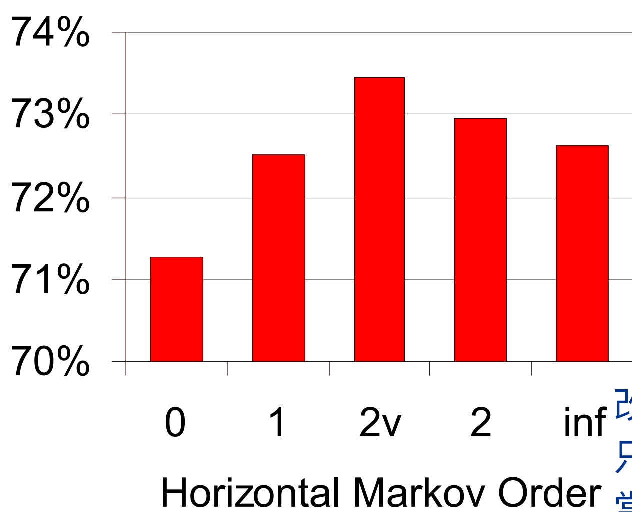
Horizontal Markovization

- Horizontal Markovization: Merges States

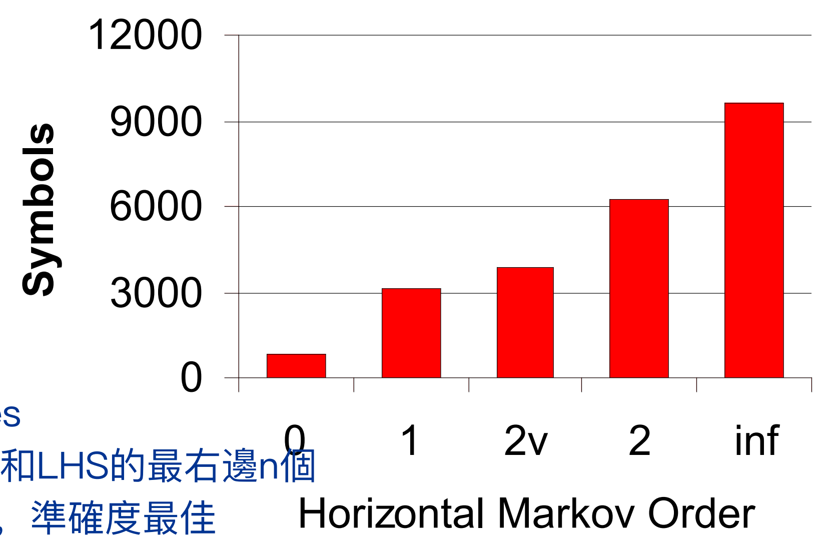


e.g.
n = 1: NP -> [ ] NNP
n = 2: NP -> NNP NNP





改寫Rules
只看LHS和LHS的最右邊n個
當n=2時，準確度最佳



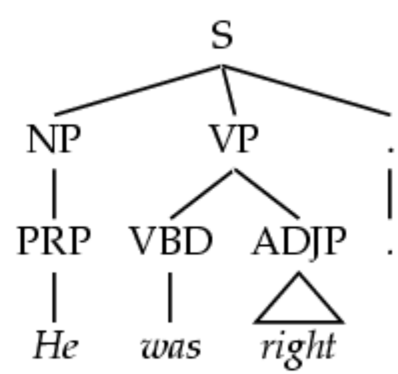
Vertical Markovization

- Vertical Markov order: rewrites depend on past *k* ancestor nodes.

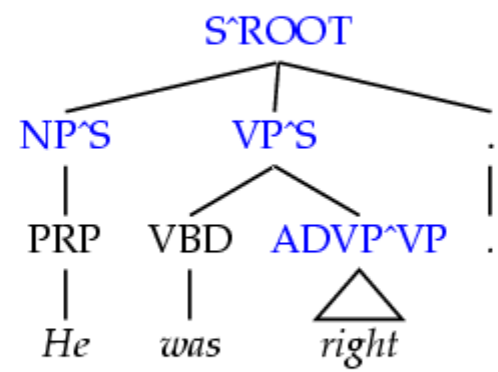
  (i.e., parent annotation)

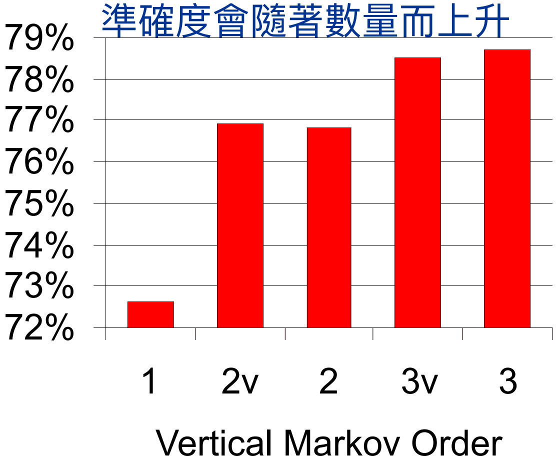
e.g.
1:只看VBD
2:看VBD^VP
3:看VBD^VP^S
準確度會隨著數量而上升

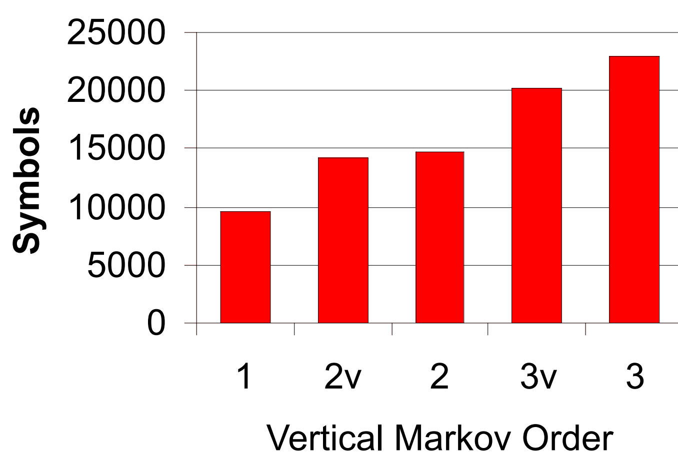
Order 1



Order 2







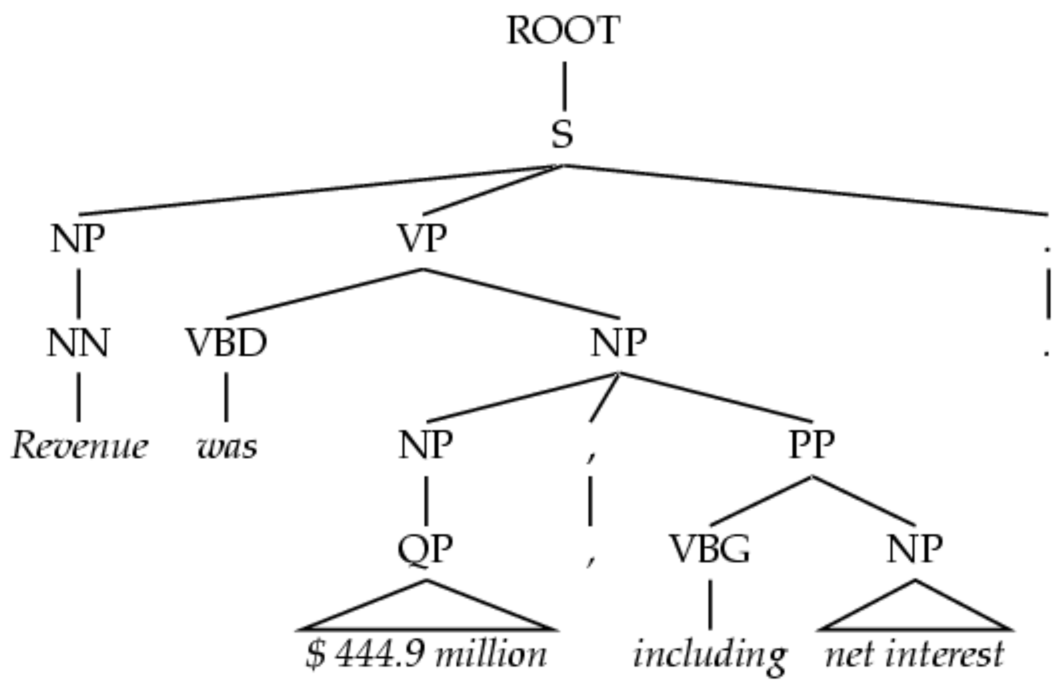
e.g.
1:只看VBD
2:看VBD^VP
3:看VBD^VP^S
需要計算的文法數隨數量上升

判斷一個Rule時
要把幾個Non-terminal納入考量

Model	F1	Size
v=h=2v	77.8	7.5K

Unary Splits

- Problem: unary rewrites are used to transmute categories so a high-probability rule can be used.



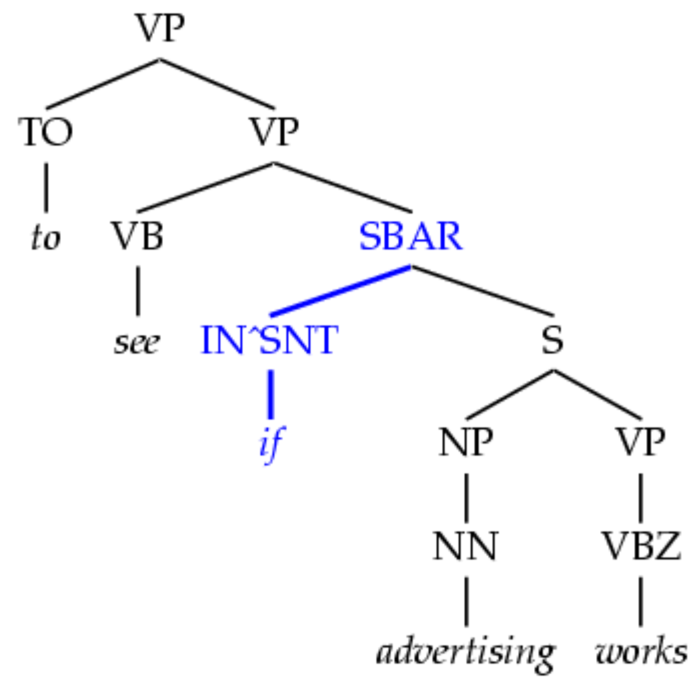
- Solution: Mark unary rewrite sites with -U

Annotation	F1	Size
Base	77.8	7.5K
UNARY	78.3	8.0K

Tag Splits

連接詞、助詞等，代表的意義可能大不相同
但卻都被標記為IN
可能會導致後續的標記出錯
所以要再對IN的標記做細分

- Problem: Treebank tags are too coarse.
- Example: SBAR sentential complementizers (*that, whether, if*), subordinating conjunctions (*while, after*), and true prepositions (*in, of, to*) are all tagged IN.
- Partial Solution:
  - Subdivide the IN tag.



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

Christopher Manning

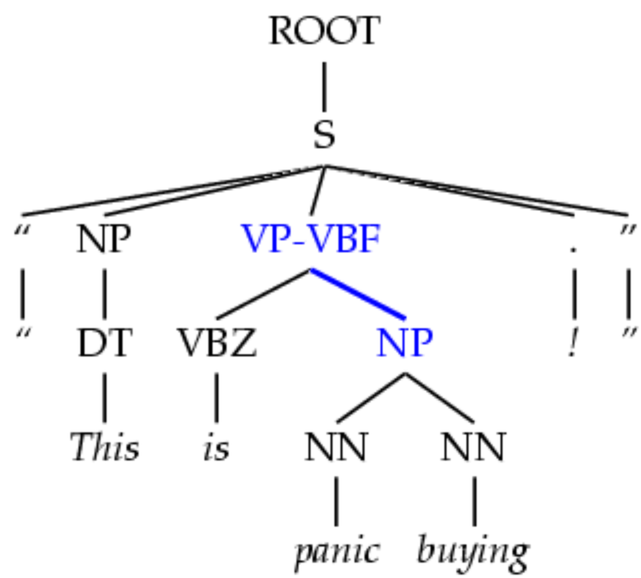
Yield Splits

動詞有分限定動詞(fin)和非限定動詞(inf)
會影響整句的文法結構
1. 限定動詞(1)現在式 (2)過去式
2. 非限定動詞(1) V-ing現在分詞 (2) p.p.過去分詞
(3) G (V-ing)動名詞(4) to + VR不定詞 (5) VR動詞原形

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

- Examples:
  - Possessive NPs
  - Finite vs. infinite VPs
  - Lexical heads!

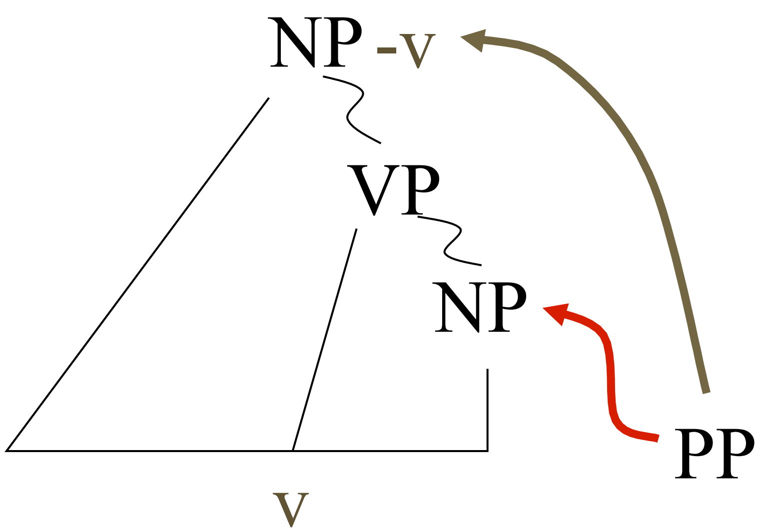
- Solution: annotate future elements into nodes.



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

Distance / Recursion Splits

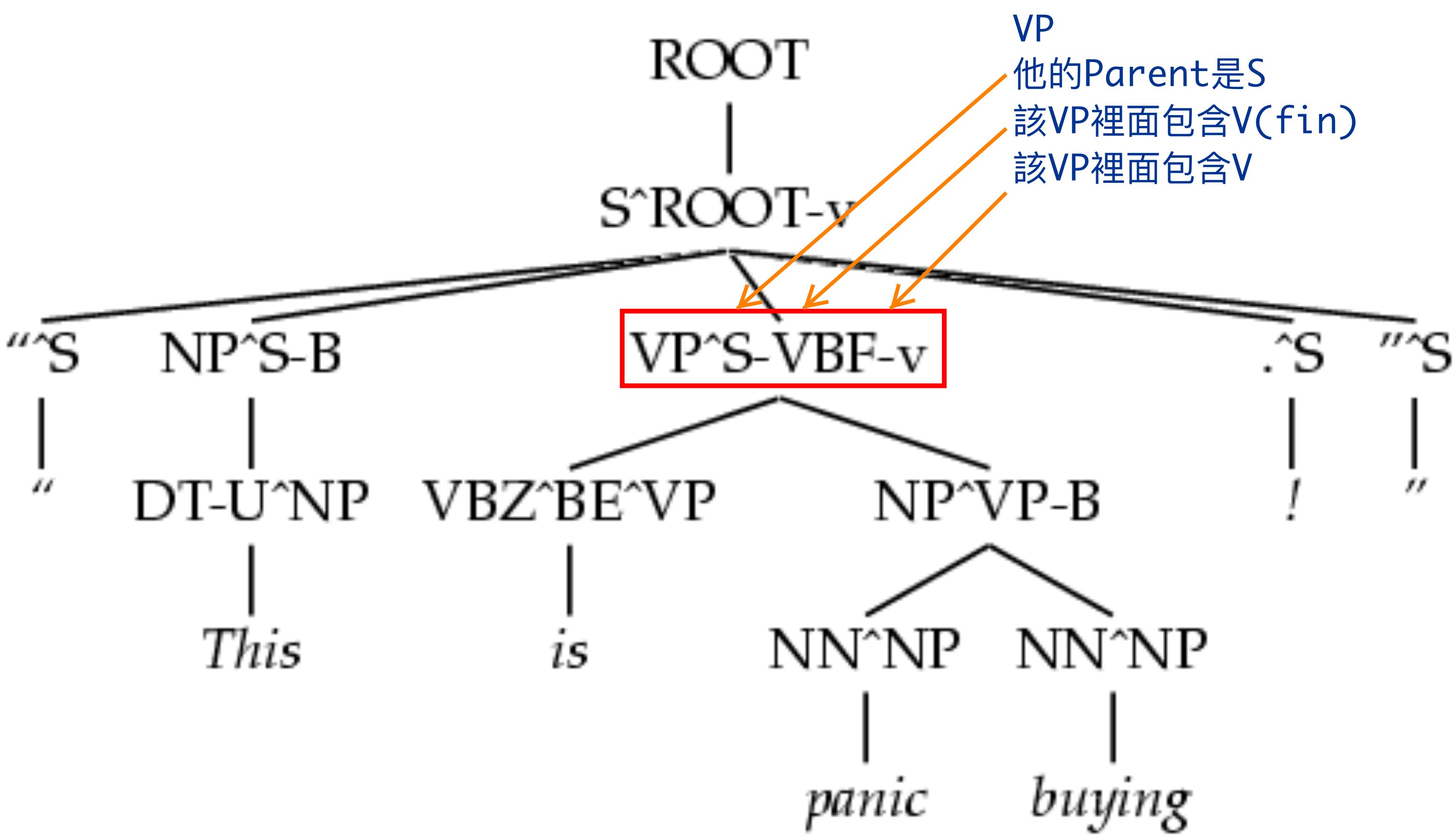
- Problem: vanilla PCFGs cannot distinguish attachment heights.
- Solution: mark a property of higher or lower sites:
  - Contains a verb.
  - Is (non)-recursive.
    - Base NPs [cf. Collins 99]
    - Right-recursive NPs



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 Tree

加入上述四個細節資訊，得到這顆最終的樹



Final Test Set Results

Parser	LP	LR	**F1**
Magerman 95	84.9	84.6	**84.7**
Collins 96	86.3	85.8	**86.0**
Klein & Manning 03	86.9	85.7	**86.3**
Charniak 97	87.4	87.5	**87.4**
Collins 99	88.7	88.6	**88.6**

- Beats "first generation" lexicalized parsers

The Return of Unlexicalized PCFGs

Latent Variable PCFGs

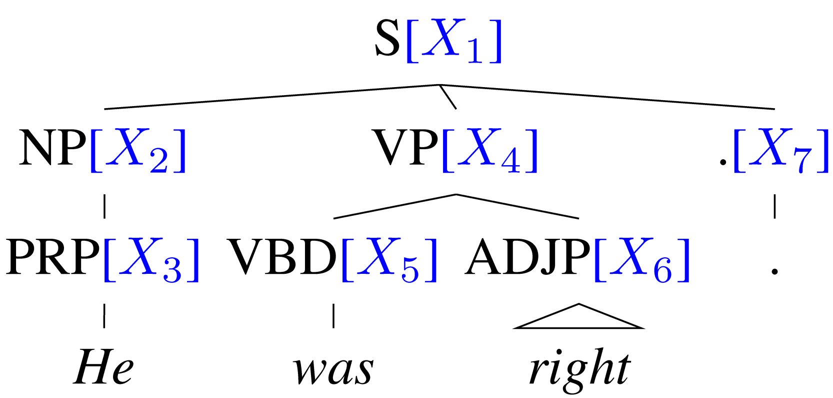
Extending the idea to induced syntactico-semantic classes

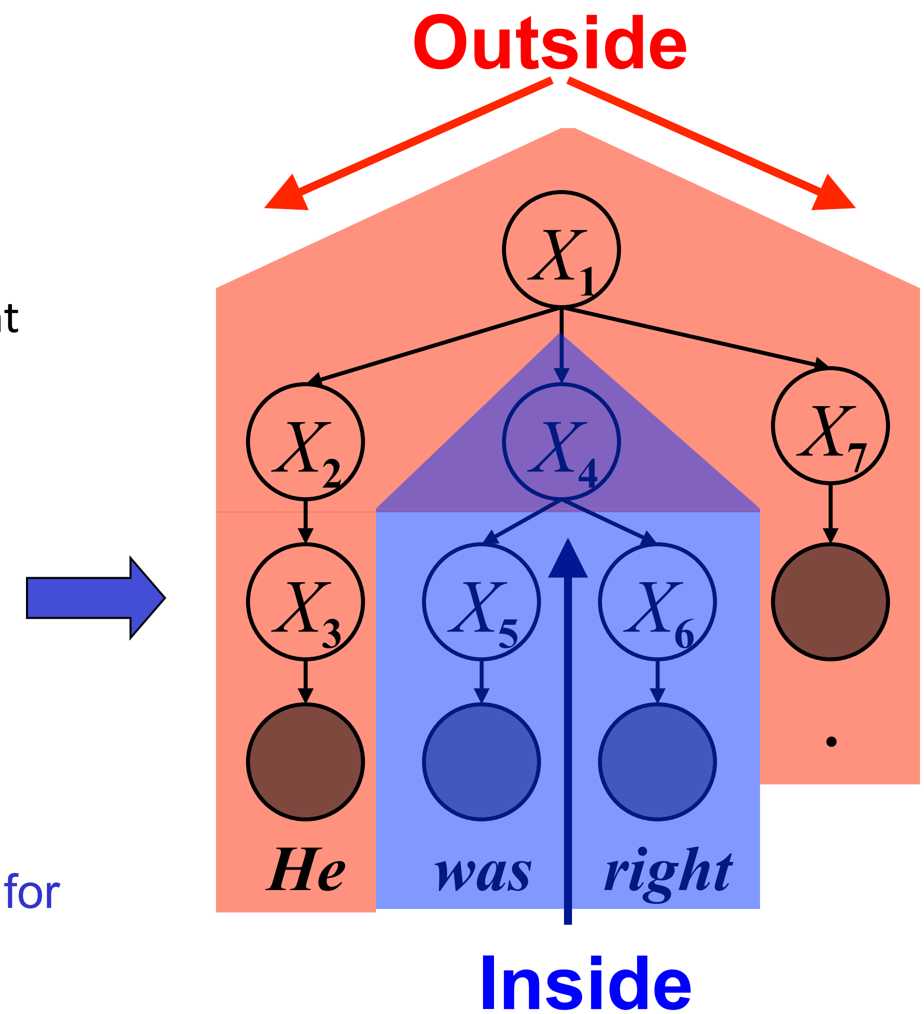
Learning Latent Annotations

[Petrov and Klein 2006, 2007]

Can you automatically find good symbols?

- Brackets are known
- Base categories are known
- Induce subcategories
- Clever split/merge category refinement





EM algorithm, like Forward-Backward for HMMs, but constrained by tree

POS tag splits' commonest words: effectively a semantic class-based model

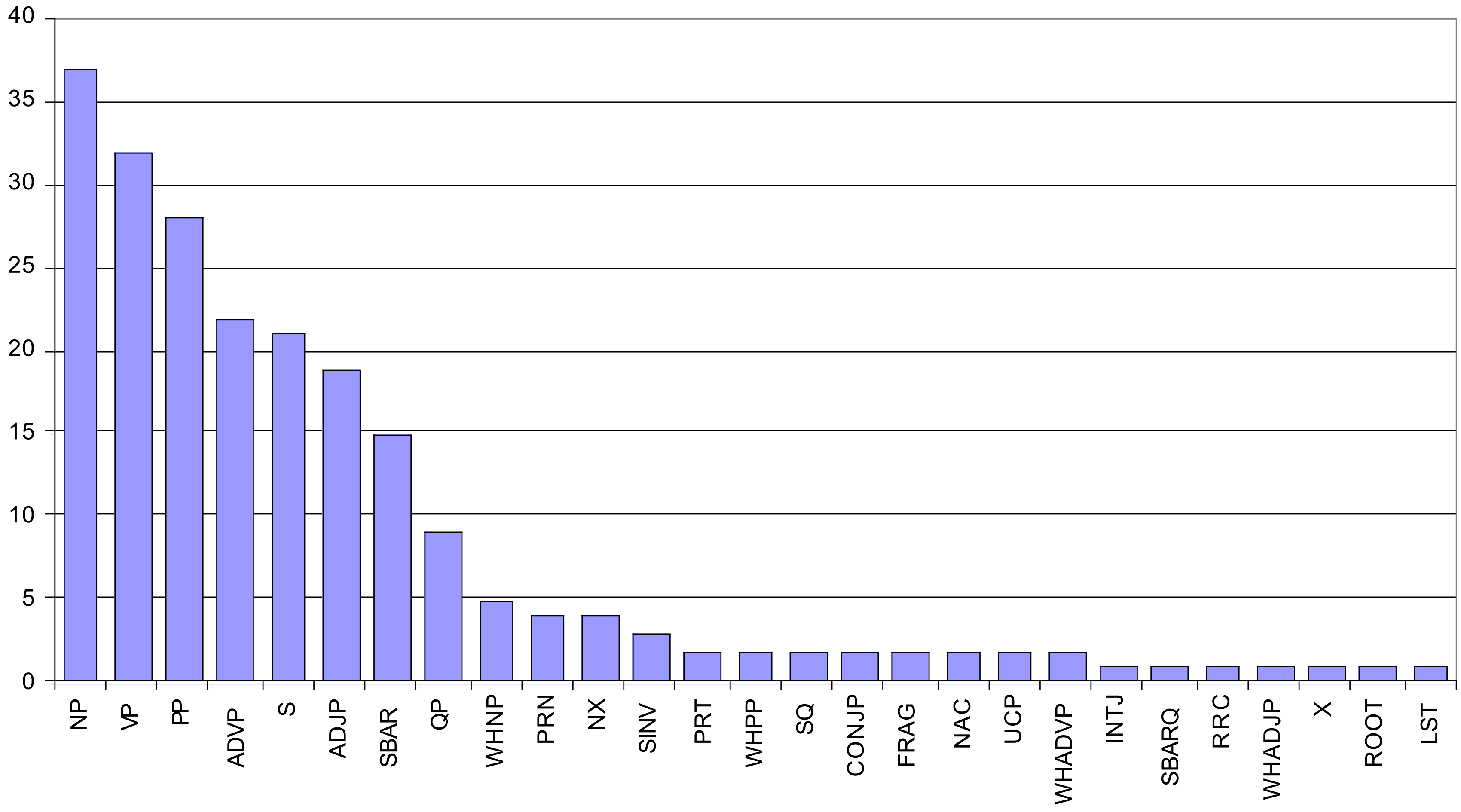
- Proper Nouns (NNP):

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

Number of phrasal subcategories



The Latest Parsing Results… (English PTB3 WSJ train 2-21, test 23)

Parser	*F1* *≤ 40 words*	*F1* *all words*
Klein & Manning unlexicalized 2003	86.3	85.7
Matsuzaki et al. simple EM latent states 2005	86.7	86.1
Charniak generative, lexicalized ("maxent inspired") 2000	90.1	89.5
Petrov and Klein NAACL 2007	90.6	90.1
Charniak & Johnson discriminative reranker 2005	92.0	91.4
Fossum & Knight 2009 combining constituent parsers		**92.4**

Latent Variable PCFGs

Extending the idea to induced syntactico-semantic classes