# CS114 Lecture 14

## Probabilistic Parsing Continued

March 17, 2014

Professor Meteer

Thanks for Jurafsky & Martin & Prof. Pustejovksy for slides

# Problems with PCFGs

- The probability model we're using is just based on the rules in the derivation…
  - Doesn't use the words in any real way
  - Doesn't take into account where in the derivation a rule is used

# Solution

- Add lexical dependencies to the scheme…
  - Infiltrate the predilections of particular words into the probabilities in the derivation
  - I.e. Condition the rule probabilities on the actual words

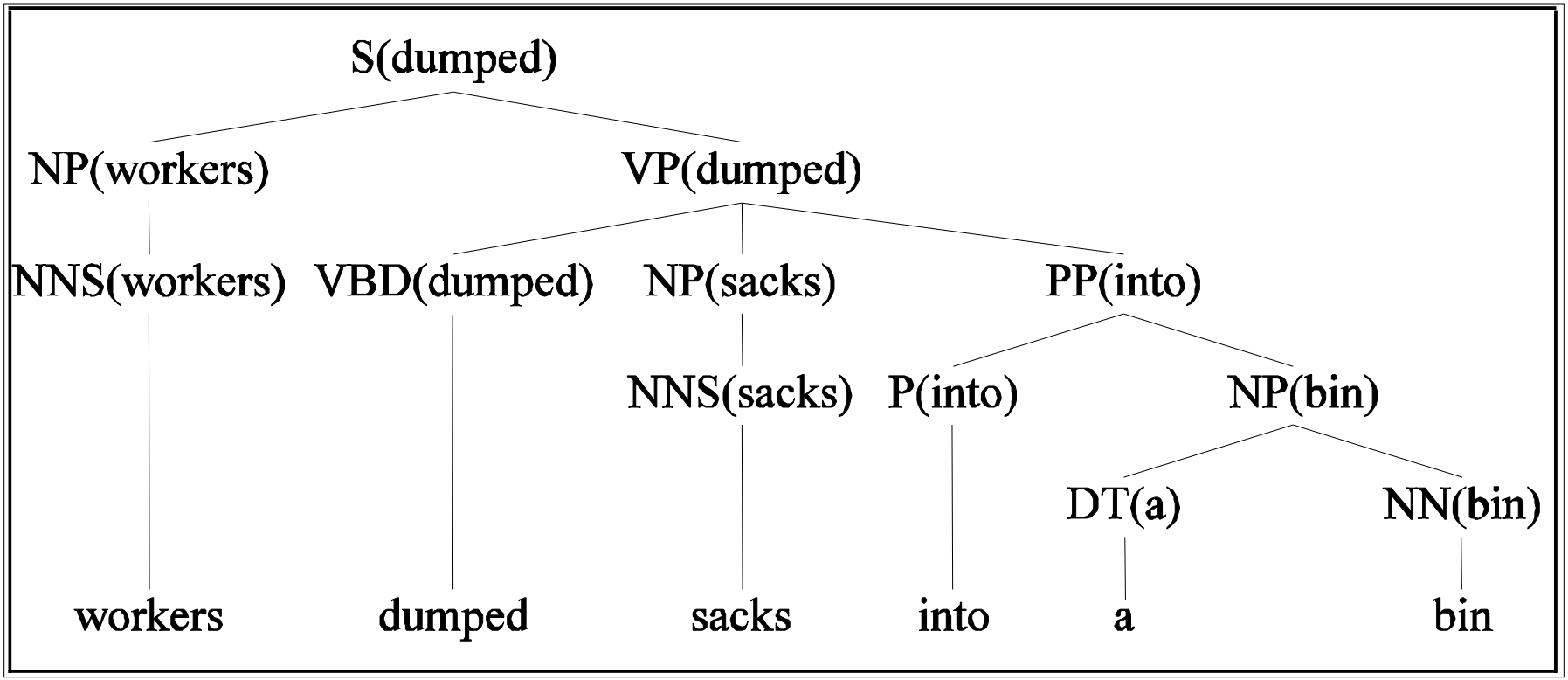
# Heads

- To do that we're going to make use of the notion of the head of a phrase
  - The head of an NP is its noun
  - The head of a VP is its verb
  - The head of a PP is its preposition

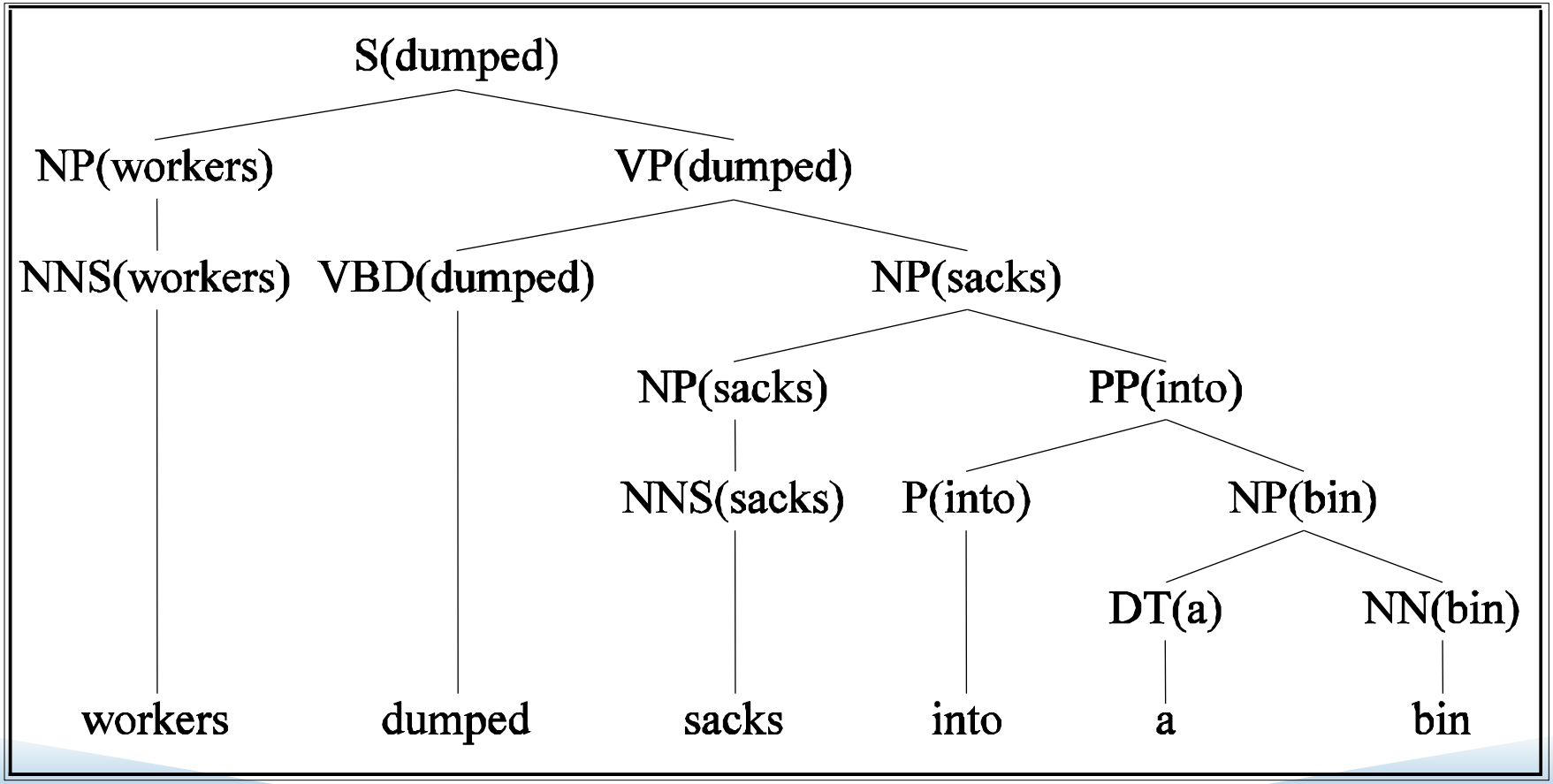
  (It's really more complicated than that but this will do.)

# Example (right)

Attribute grammar



# Example (wrong)



# How?

- We used to have
  - VP -> V NP PP  P(rule|VP)
    - That's the count of this rule divided by the number of VPs in a treebank
- Now we have
  - VP(dumped)-> V(dumped) NP(sacks)PP(in)
  - P(r|VP ^ dumped is the verb ^ sacks is the head of the NP ^ in is the head of the PP)
  - Not likely to have significant counts in any treebank

# Declare Independence

- When stuck, exploit independence and collect the statistics you can…
- We'll focus on capturing two things
  - Verb subcategorization
    - Particular verbs have affinities for particular VPs
  - Objects affinities for their predicates (mostly their mothers and grandmothers)
    - Some objects fit better with some predicates than others

# Subcategorization

- Condition particular VP rules on their head… so

  r: VP -> V NP PP P(r|VP)

  Becomes

  P(r | VP ^ dumped)

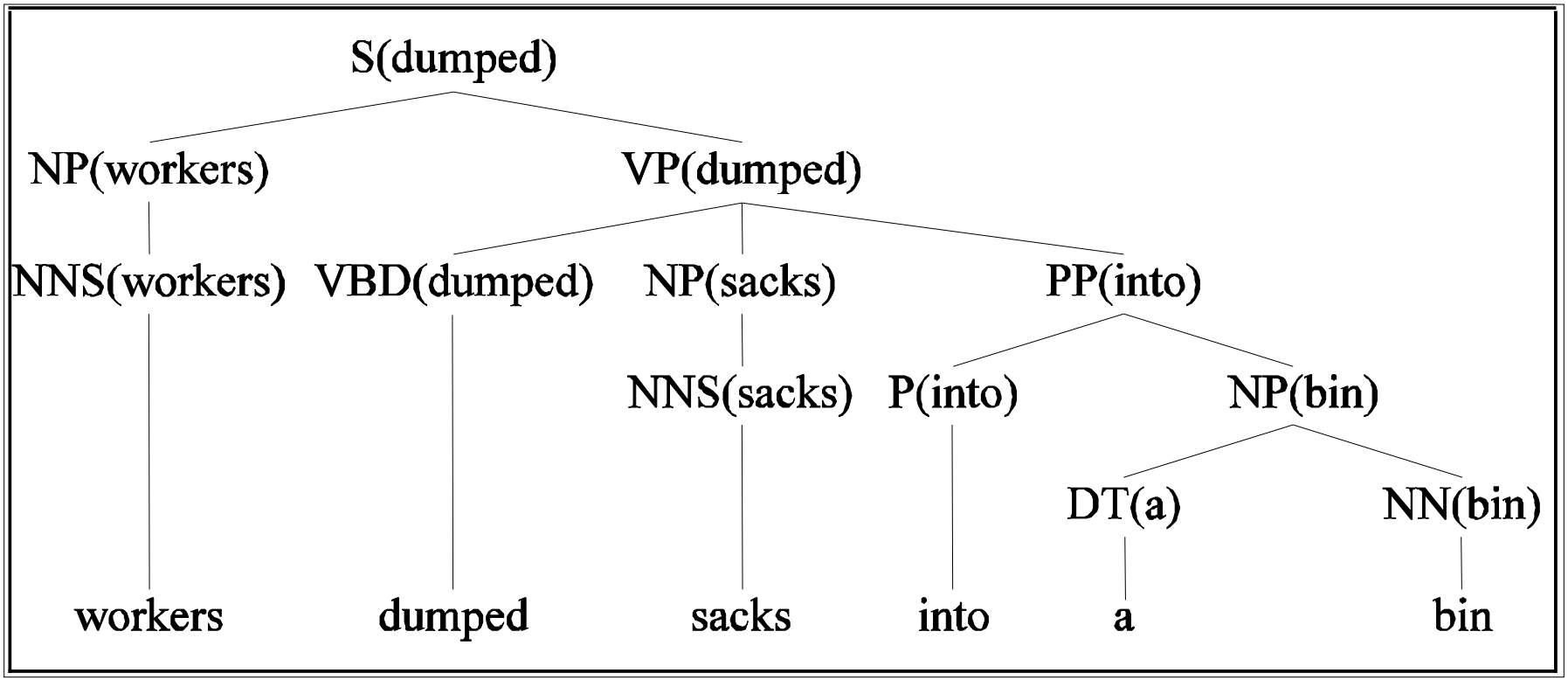
  What's the count?

  How many times was this rule used with (head) dump, divided by the number of VPs that dump appears (as head) in total

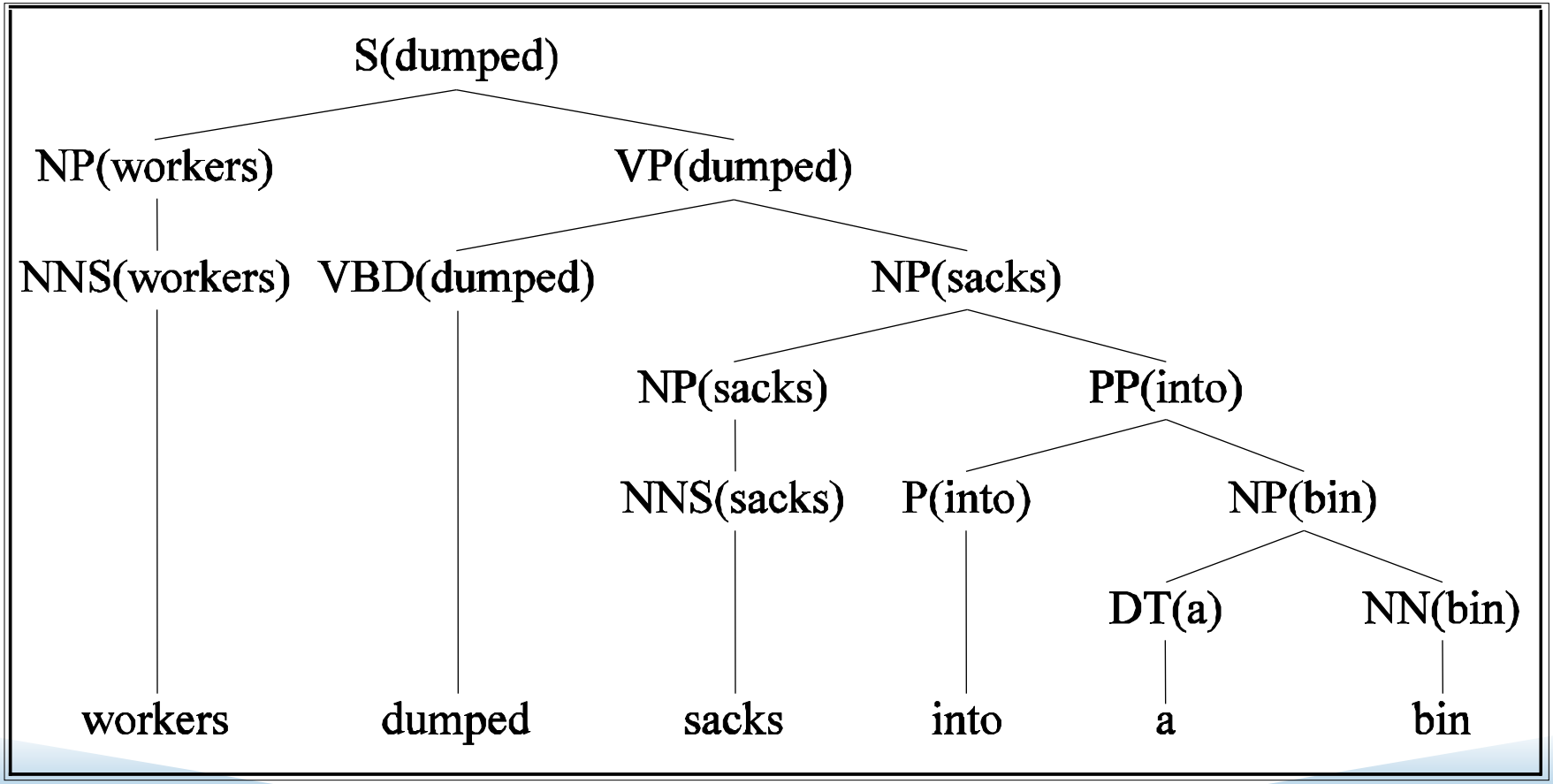
# Preferences

- Subcat captures the affinity between VP heads (verbs) and the VP rules they go with.
- What about the affinity between VP heads and the heads of the other daughters of the VP
- Back to our examples…

# Example (right)



# Example (wrong)



# Preferences

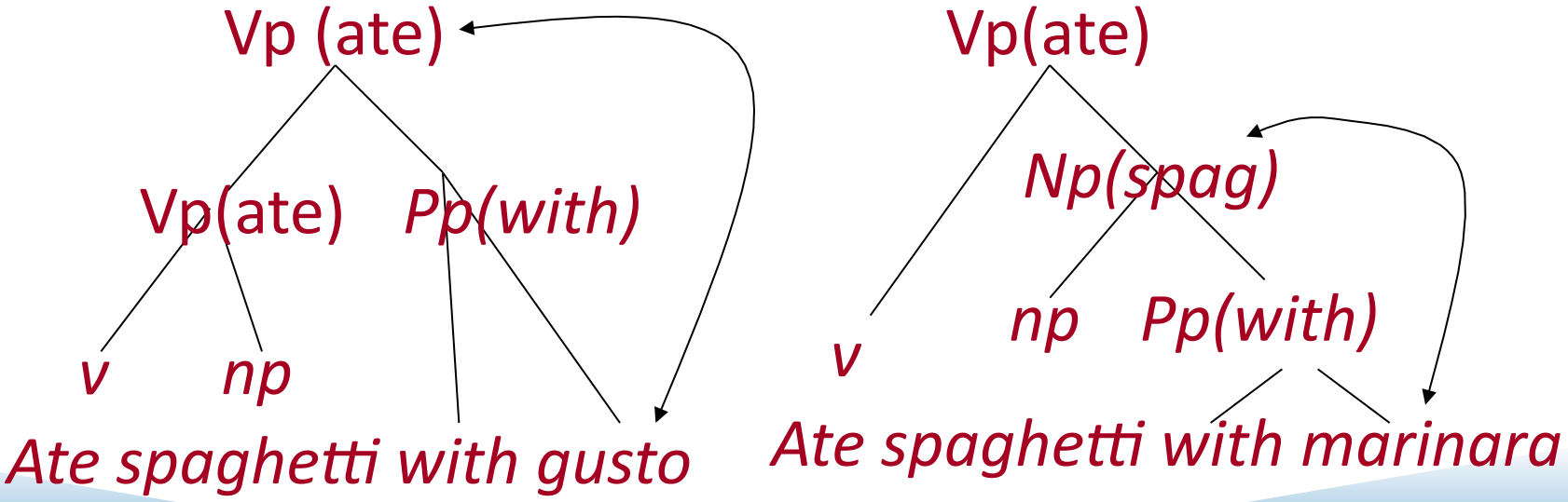
- The issue here is the attachment of the PP. So the affinities we care about are the ones between dumped and into vs. sacks and into.
- So count the places where dumped is the head of a constituent that has a PP daughter with into as its head and normalize
- Vs. the situation where sacks is a constituent with into as the head of a PP daughter.

# Preferences (2)

- Consider the VPs
  - Ate spaghetti with gusto
  - Ate spaghetti with marinara
- The affinity of gusto for eat is much larger than its affinity for spaghetti
- On the other hand, the affinity of marinara for spaghetti is much higher than its affinity for ate

# Preferences (2)

- Note the relationship here is more distant and doesn't involve a headword since gusto and marinara aren't the heads of the PPs.

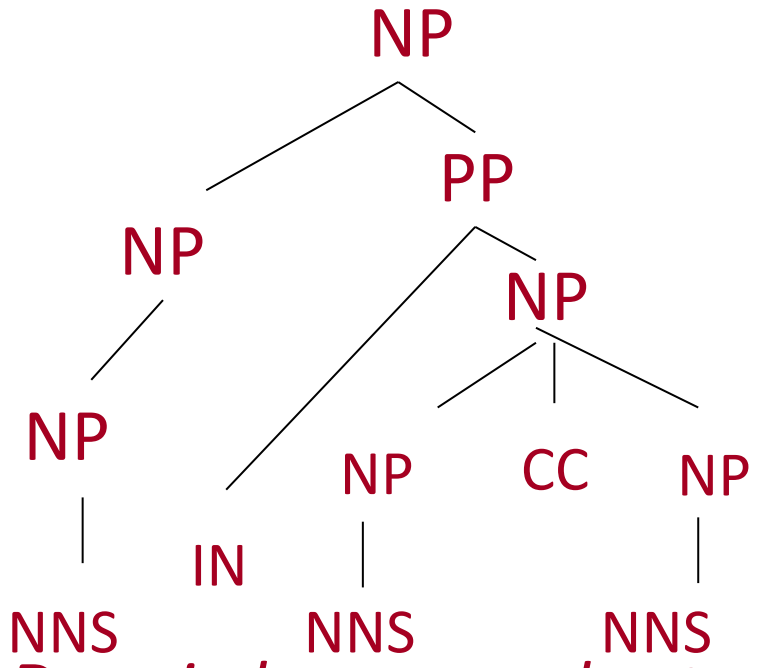


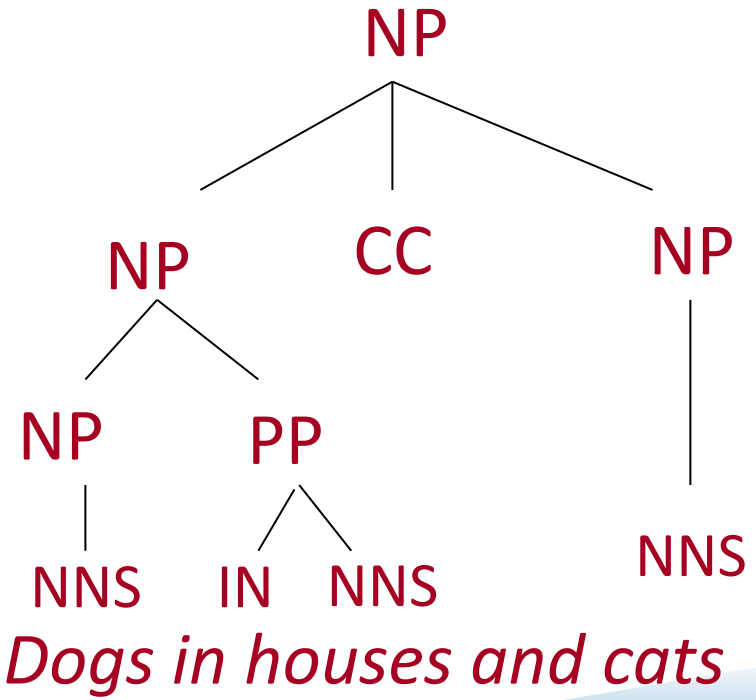
# Summary

- Context-Free Grammars
- Parsing
  - Top Down, Bottom Up Metaphors
  - Dynamic Programming Parsers: CKY. Earley
- Disambiguation:
  - PCFG
  - Probabilistic Augmentations to Parsers
  - Treebanks

# Other Issues with PCFGs

## A Case of Coordinated Ambiguity



# Conjunction

| Rules |
| --- |
| NP → NP CC NP |
| NP → NP PP |
| NP → NNS |
| PP → IN NP |
| NP → NNS |
| NP → NNS |
| NNS → dogs |
| N → in |
| NNS → houses |
| CC → and |
| NNS → cats |

| Rules |
| --- |
| NP → NP CC NP |
| NP → NP PP |
| NP → NNS |
| PP → IN NP |
| NP → NNS |
| NP → NNS |
| NNS → dogs |
| N → in |
| NNS → houses |
| CC → and |
| NNS → cats |

**Here the two parses have identical rules, and therefore have identical probability under any assignment of PCFG rule probabilities**

# Structural Preferences: Close Attachment

- Example:
  - A. President of (a company in Africa)
  - B. (President of a company) in Africa
- Both parses have the same rules, therefore receive same probability under a PCFG
- "Close attachment" (structure A) is twice as likely in Wall Street Journal text.

# Structural Preferences: Close Attachment

- **Previous example:**
  - John was believed to have been shot by Bill
- Here the low attachment analysis (Bill does the *shooting)* contains same rules as the high attachment analysis (Bill does the *believing),* so the two analyses receive same probability.

# Adding “Heads”

- Each context-free rule has one "special" child that is the head of the rule, e.g.,

  S => NP VP

  VP => Vt NP

  NP => DT NN NN

- A core idea in syntax

  (e.g., see X-bar Theory, Head-Driven Phrase Structure Grammar)

- Some intuitions:
  - The central sub-constituent of each rule.
  - The semantic predicate in each rule.

# Rules which Recover Heads:
## An Example for NPs

- If rule contains NN, NNS, or NNP
  - Choose rightmost NN, NNS or NNP
- Else if rule contains NP
  - Choose leftmost NP
- Else if rule contains a JJ
  - Choose rightmost JJ
- Else if rule contains a CD
  - Choose right most CD
- Else choose the rightmost child

**NP=> DT NNP NN**
**NP => DT NN NNP**
**NP => NP PP**
**NP => DT JJ**
**NP => CD**

# Parameters in a Lexicalized PCFG

- An example parameter in a PCFG:
- $q(S \rightarrow \text{NP VP})$
- An example parameter in a Lexicalized PCFG:
  ^(S(saw) →2 NP(man) VP(saw))

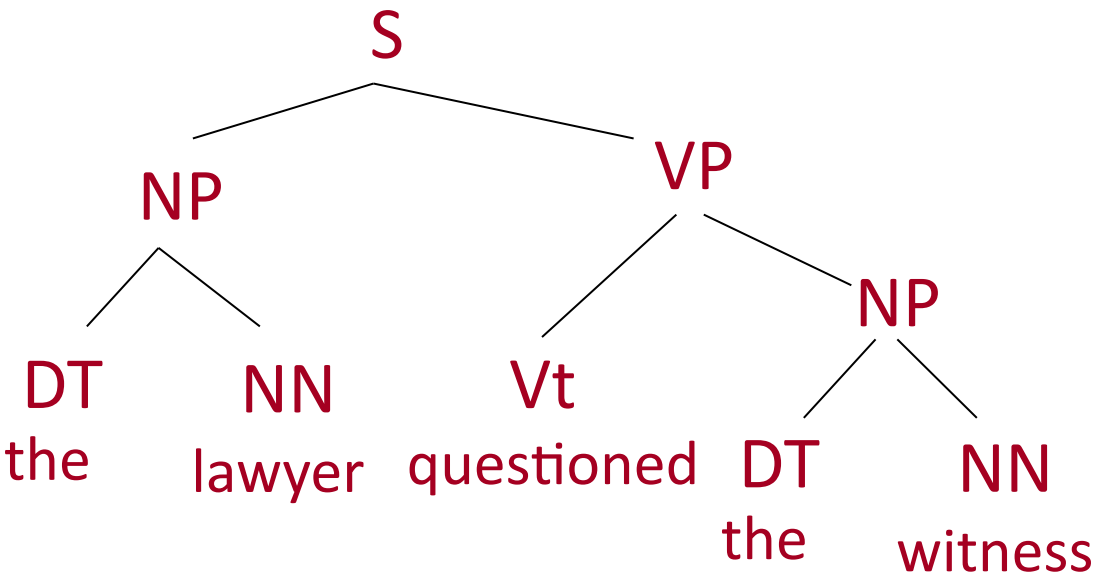
# Parsing with Lexicalized CFGs

- The new form of grammar looks just like a Chomsky normal form CFG, but with potentially huge set of words.
- Crucial observation: Any rules which contain a lexical item that is not one of $w_1 \ldots w_n$, can be safely discarded.
- The result: we can parse in $O(n^5|N|^3)$ time.
  - n: length of sentence
  - N: set of nonterminals

# Other Important Details

- Need to deal with rules with more than two children,
  VP(told)→V(told) NP(him) PP(on) SBAR(that)
- Need to incorporate parts of speech (useful in smoothing)
  VP-V(told) → V(told) NP-PRP(him) PP-IN(on) SBAR-COMP(that)
- Need to encode preferences for close attachment
  - John was believed to have been shot by Bill
- Further reading:
  - *Michael Collins. 2003.* Head-Driven Statistical Models for Natural Language Parsing. *In Computational Linguistics.*

# Evaluation:
## Representing Trees as Constituents




| Label | Start point | End point |
|---|---|---|
| NP | 1 | 2 |
| NP | 4 | 5 |
| VP | 3 | 5 |
| S | 1 | 5 |

# Precision and recall

| Label | Start point | End point |
|---|---|---|
| NP | 1 | 2 |
| NP | 4 | 5 |
| NP | 4 | 8 |
| PP | 6 | 8 |
| NP | 7 | 8 |
| VP | 3 | 8 |
| S | 1 | 8 |

| Label | Start point | End point |
|---|---|---|
| NP | 1 | 2 |
| NP | 4 | 5 |
| PP | 6 | 8 |
| NP | 3 | 8 |
| VP | 3 | 8 |
| S | 1 | 8 |

- G = number of constituents in gold standard = 7
- P = number in parse output = 6
- C = number correct = 6
- Recall = C/G = 6/7 = 87%
- Precision = C/P = 6/6 = 100%

# Results

- Training data: 40,000 sentences from the Penn Wall Street Journal treebank. Testing: around 2,400 sentences from the Penn Wall Street Journal treebank.
  - Results for a PCFG: 70.6% Recall, 74.8% Precision
  - Magerman (1994): 84.0% Recall, 84.3% Precision
- Results for a lexicalized PCFG: 88.1% recall, 88.3% precision (from Collins (1997, 2003))
- More recent results: 90.7% Recall/91.4% Precision (Carreras et a l., 2008); 91.7% Recall, 92.0% Precision (Petrov 2010); 91.2% Recall, 91.8% Precision (Charniak and Johnson, 2005)


# A hotly debated case: German

- Linguistic characteristics, relative to English
  - Ample derivational and inflectional morphology
  - Freer word order
  - Verb position differs in matrix/embedded clauses
  - Main ambiguities similar to English
- Most used corpus: Negra
  - ~400,000 words newswire text
  - Flatter phrase structure annotations (few PPs!)
  - Explicitly marked phrasal discontinuities
- Newer Treebank: TueBaDz
  - ~470,000 words newswire text (27,000 sentences)
  - [Not replacement; different group; different style]

Slides adapted from Christopher Manning

# German results

- Dubey and Keller [ACL 2003] present an unlexicalized PCFG outperforming Collins on NEGRA – and then get small wins from a somewhat unusual sister-head model, but…

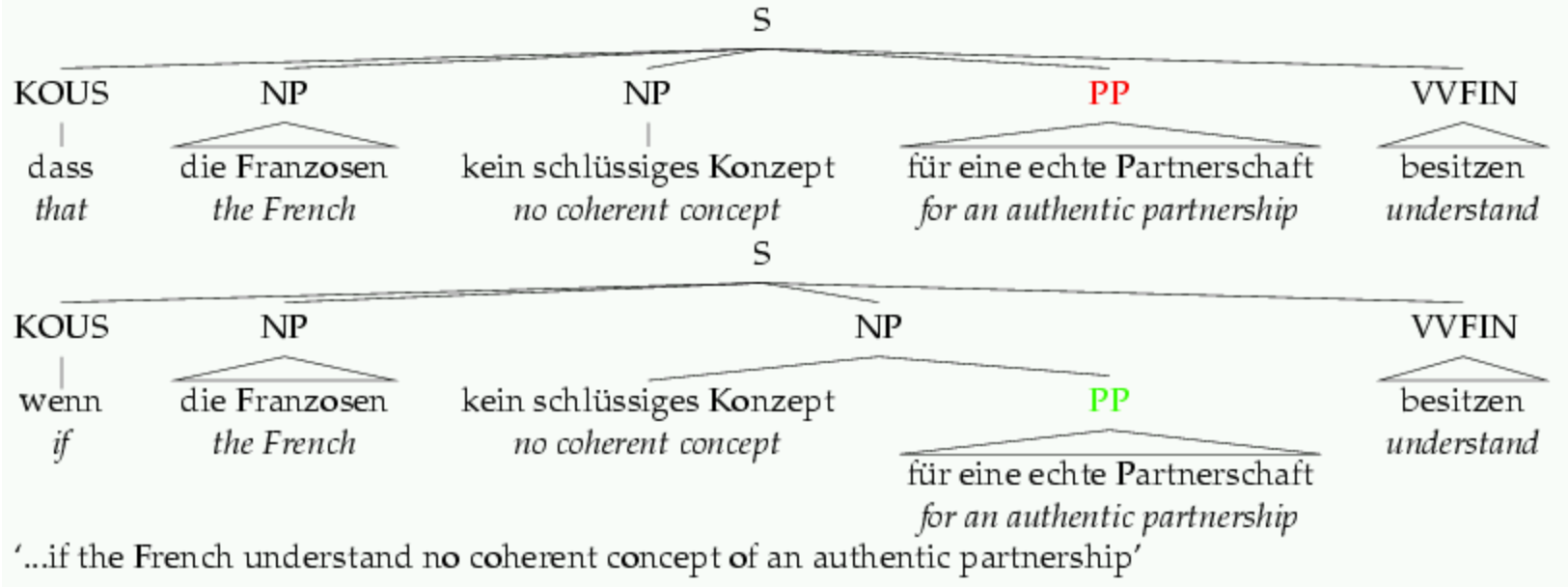
| | LPrec | LRec | F1 |
|---|---|---|---|
| D&K PCFG Baseline | 66.69 | 70.56 | 68.57 |
| D&K Collins | 66.07 | 67.91 | 66.98 |
| D&K Sister-head all | 70.93 | 71.32 | 71.12 |

| | LPrec | LRec | F1 |
|---|---|---|---|
| Stanford PCFG Baseline | 72.72 | 73.64 | 73.59 |
| Stanford Lexicalized | 74.61 | 76.23 | 75.41 |

- See also [Arun & Keller ACL 2005, Kübler & al. EMNLP 2006

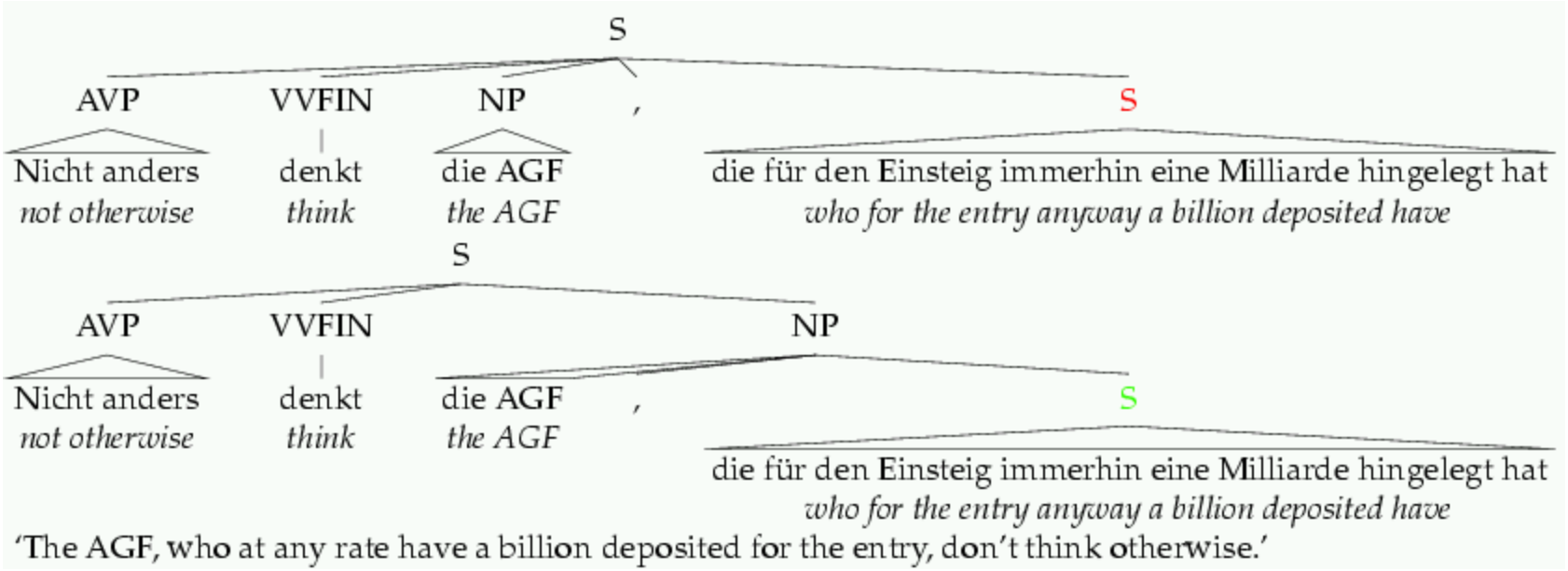
# Prominent ambiguities

- PP attachment



# Prominent ambiguities

- Sentential complement vs. relative clause

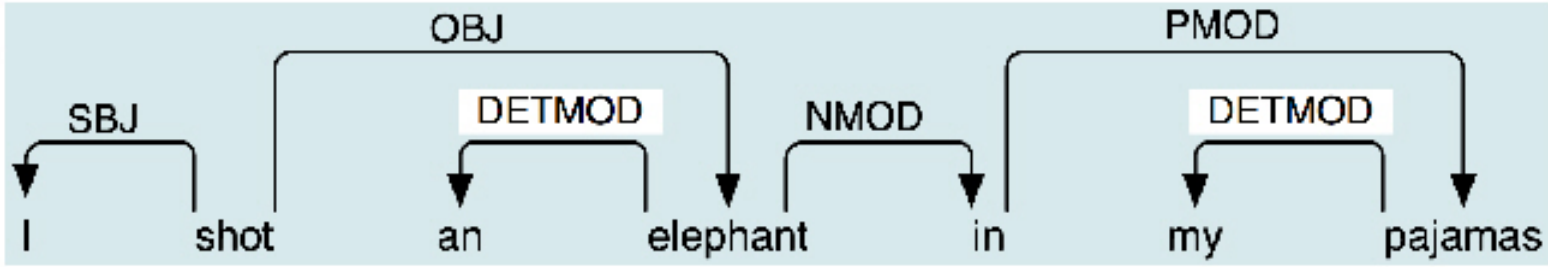


# Dependency grammars

- Phrase structure grammar is concerned with how words and sequences of words combine to form constituents.
- A distinct and complementary approach, dependency grammar, focuses instead on how words relate to other words
- Dependency is a binary asymmetric relation that holds between a head and its dependents.

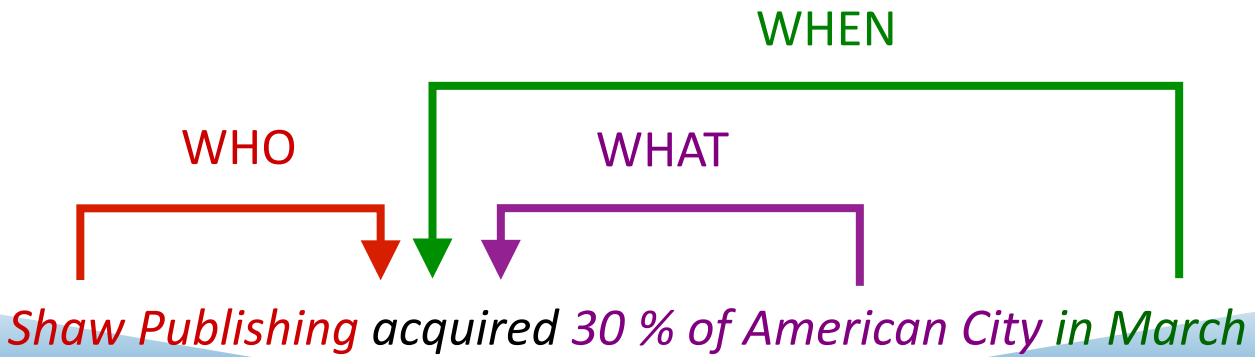
# Dependency grammars

- Dependency graph: labeled directed graph
  - nodes are the lexical items
  - labeled arcs represent dependency relations from heads to dependents
- Can be used to directly express grammatical functions as a type of dependency.

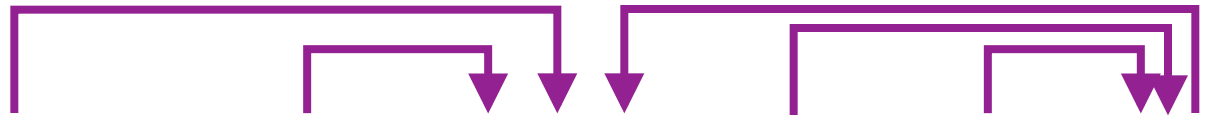


# Dependency grammars

- Dependency structure gives attachments.
- In principle, can express any kind of dependency
- How to find the dependencies?



# Idea: Lexical Affinity Models

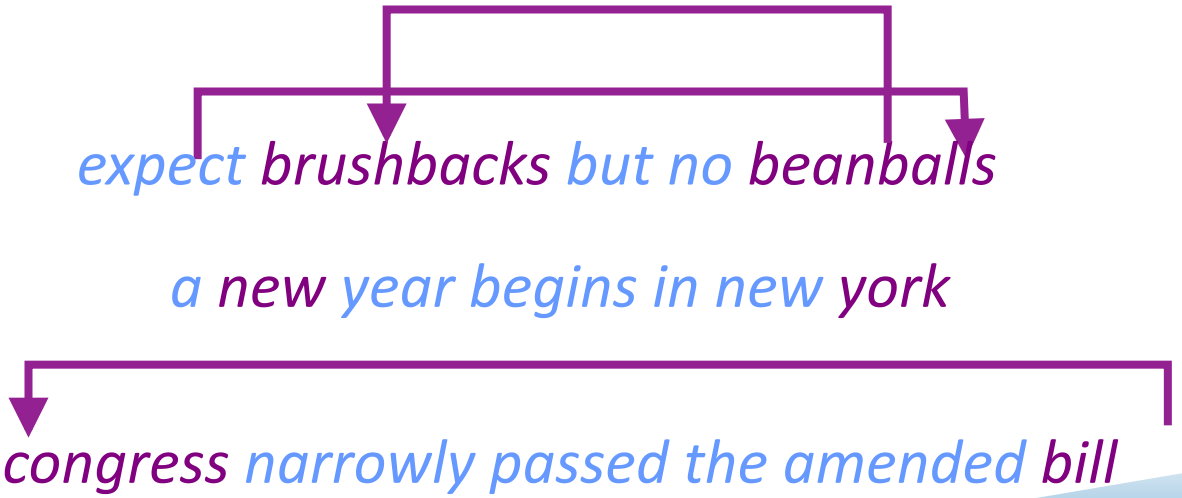


*congress narrowly passed the amended bill*

- Link up pairs with high mutual information
  - Mutual information measures how much one word tells us about another.
  - The doesn't tell us much about what follows
    - I.e. "the" and "red" have small mutual information
  - United ?

# Problem: Non-Syntactic Affinity

- Words select other words (also) on syntactic grounds
- Mutual information between words does not necessarily indicate syntactic selection.



# Idea: Word Classes

- Individual words like congress are entwined with semantic facts about the world.
- Syntactic classes, like NOUN and ADVERB are bleached of word-specific semantics.
- Automatic word classes more likely to look like DAYS-OF-WEEK or PERSON-NAME.
- We could build dependency models over word classes. [cf. Carroll and Charniak, 1992]

*congress narrowly passed the amended bill*

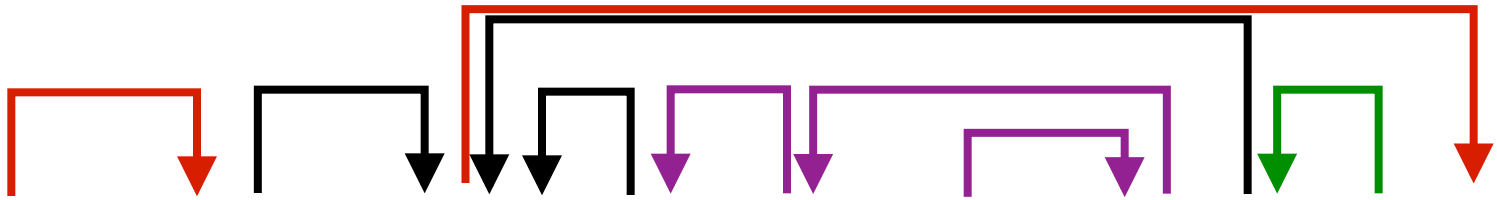
NOUN ADVERB VERB DET PARTICIPLE NOUN


# Dependency Grammar/Parsing

- A sentence is parsed by relating each word to other words in the sentence which depend on it.
- The idea of dependency structure goes back a long way
  - To Pāṇini's grammar (c. 5th century BCE)
- Constituency is a new-fangled invention
  - 20th century invention
- Modern work often linked to work of L. Tesniere (1959)
  - Dominant approach in "East" (Eastern bloc/East Asia)
- Among the earliest kinds of parsers in NLP, even in US:
  - David Hays, one of the founders of computational linguistics, built early (first?) dependency parser (Hays 1962)

Slides adapted from Christopher Manning

# Dependency structure



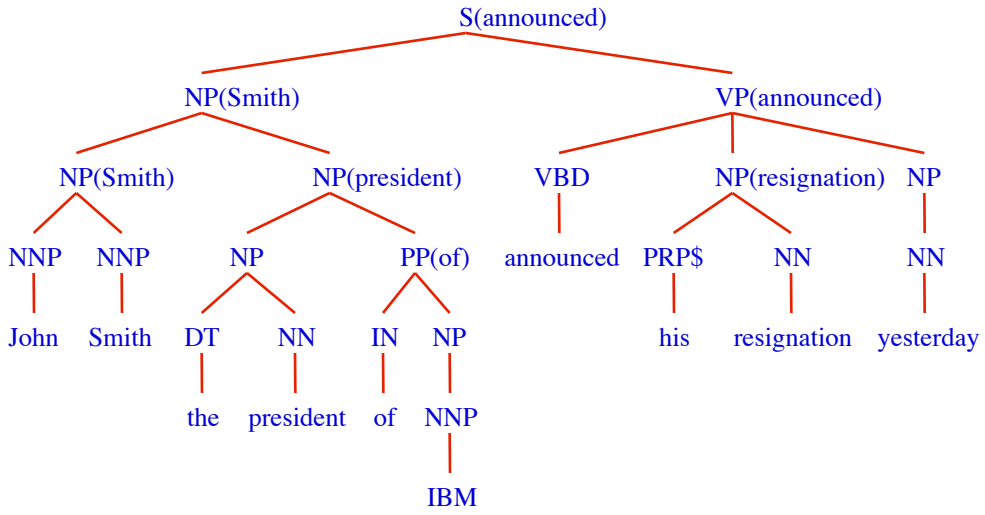
*Shaw Publishing acquired 30 % of American City in March $$*

- Words are linked from head (regent) to dependent
- Warning! Some people do the arrows one way; some the other way (Tesniere has them point from head to dependent…).
- Usually add a fake ROOT so every word is a dependent


# Relation between CFG to dependency parse

- A dependency grammar has a notion of a head
- Officially, CFGs don't
- But modern linguistic theory and all modern statistical parsers (Charniak, Collins, Stanford, …) do, via hand-written phrasal "head rules":
  - The head of a Noun Phrase is a noun/number/adj/…
  - The head of a Verb Phrase is a verb/modal/….
- The head rules can be used to extract a dependency parse from a CFG parse (follow the heads).
- A phrase structure tree can be got from a dependency tree, but dependents are flat (no VP!)

# Propagating head words

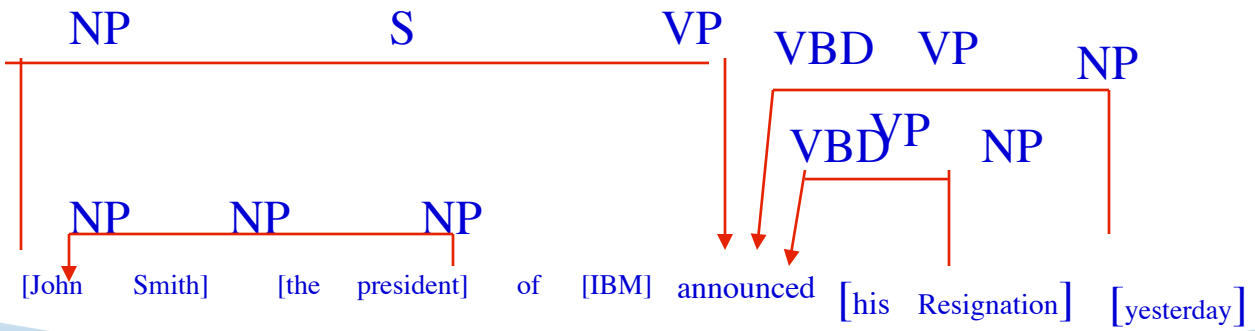



– Small set of rules propagate heads

# Extracted structure

**NB**. Not all dependencies shown here

- Dependencies are inherently untyped, though some work like Collins (1996) types them using the phrasal categories




# Dependency Conditioning Preferences

Sources of information:

- bilexical dependencies
- distance of dependencies
- valency of heads (number of dependents)

A word's **dependents** (adjuncts, arguments) tend to fall **near** it in the string.

These next 6 slides are based on slides by Jason Eisner and Noah Smith

# Probabilistic dependency grammar: generative model

1. Start with left wall $
2. Generate root $w_0$
3. Generate left children $w_{-1}$, $w_{-2}$, ..., $w_{-\ell}$ from the FSA $\lambda_{w_0}$
4. Generate right children $w_1$, $w_2$, ..., $w_r$ from the FSA $\rho_{w_0}$
5. Recurse on each $w_i$ for $i$ in {-ℓ, ..., -1, 1, ..., $r$}, sampling $\alpha_i$ (steps 2-4)
6. Return $\alpha_{\ell}...\alpha_{-1}w_0\alpha_1...\alpha_r$