Nothing Works in German

Talk Overview

- Probabilistic parsing in German
  - Lexicalized parsing
  - Parsing with simpler grammars
  - Parsing with probabilistic feature-based grammars
- Might be a bit long ...and a bit technical (yawn!)

Fundamental Trade-off in Parsing

Expressiveness	$\Longleftrightarrow$	Coverage
Grammars with complex feature structures	$\Longleftrightarrow$	Simpler grammars

Implicit vs. Explicit Expressiveness

Non-expressive $S \rightarrow NP\ V\ NP\ PP$

Implicitly expressive $S \rightarrow NP_{\text{she}}\ V_{\text{sells}}\ NP_{\text{shells}}\ PP_{\text{to}}$

Explicitly expressive
$$\begin{bmatrix} \text{sells} & \\ \textsc{tag} & \textsf{verb} \\ \textsc{lemma} & \textsf{to sell} \\ \textsc{tense} & \textsf{present} \\ \textsc{person} & \textsf{second} \\ \textsc{subcat} & \textsf{[NP, NP, PP]} \end{bmatrix}$$

Implicit vs. Explicit Expressiveness

- Implicit vs. explictly expressive grammars:
  - Some information you could recover (i.e. subcat frames) but not stored explicitly
  - Some information cannot be recovered without additional knowledge

Benefits of Implicit Expressiveness

- Do not sacrifice coverage
- Highly accurate
- Economize on grammar writer's time

My main argument:

- These are not general results!

Why?

- As most work has focused on English (and Chinese)
- Work on other languages not properly analyzed

A Closer Look at some Grammar Models

- Charniak (1997)
  - First to show that a CFG-like model can be lexicalized
  - Tested on English and German (Rooth et al. 1998)
- Collins (1997) (also appears as Collins 1999 and Collins 2004)
  - Well-documented
  - Tested on a variety of languages, including English, Czech (Collins et al. 1999), Chinese (Bikel and Chiang 2000) and Korean (Chung and Rim, 2004)

Experiment (Dubey & Keller, 2003)

- Compare Charniak and Collins models to an non-expressive grammar (an unlexicalized probabilistic CFG, or unlexicalized PCFG)
- Test on German data from the NEGRA corpus
- Result: PCFG wins!
- But: more expressive grammars require more data. Could sparse data be the reason why PCFG does better?
- No: learning curves show that the effect is not due to sparse data

What About Other Languages?

- Chinese
    - Results also quite bad
    - Did not compare to an unlexicalized baseline
    - But: other lexicalized models did much better...
- Czech
    - Different evaluation: word-word dependencies
    - Results appear much better... (80%)
    - ...but this is a scoring effect
    - Unlexicalized grammar gives similar results in NEGRA
- Korean
    - Similar story as Czech (83% on dependencies)
    - Again, lexicalization adds very little

What was Learned?

- Use the right baseline

- Lexicalization (i.e. using an implicitly expressive grammar) appears to help very little in languages with more productive morphologies

Postscript

- We also introduced a modification of the Collins model that appears to be more appropriate for NEGRA

- More data does seem to help

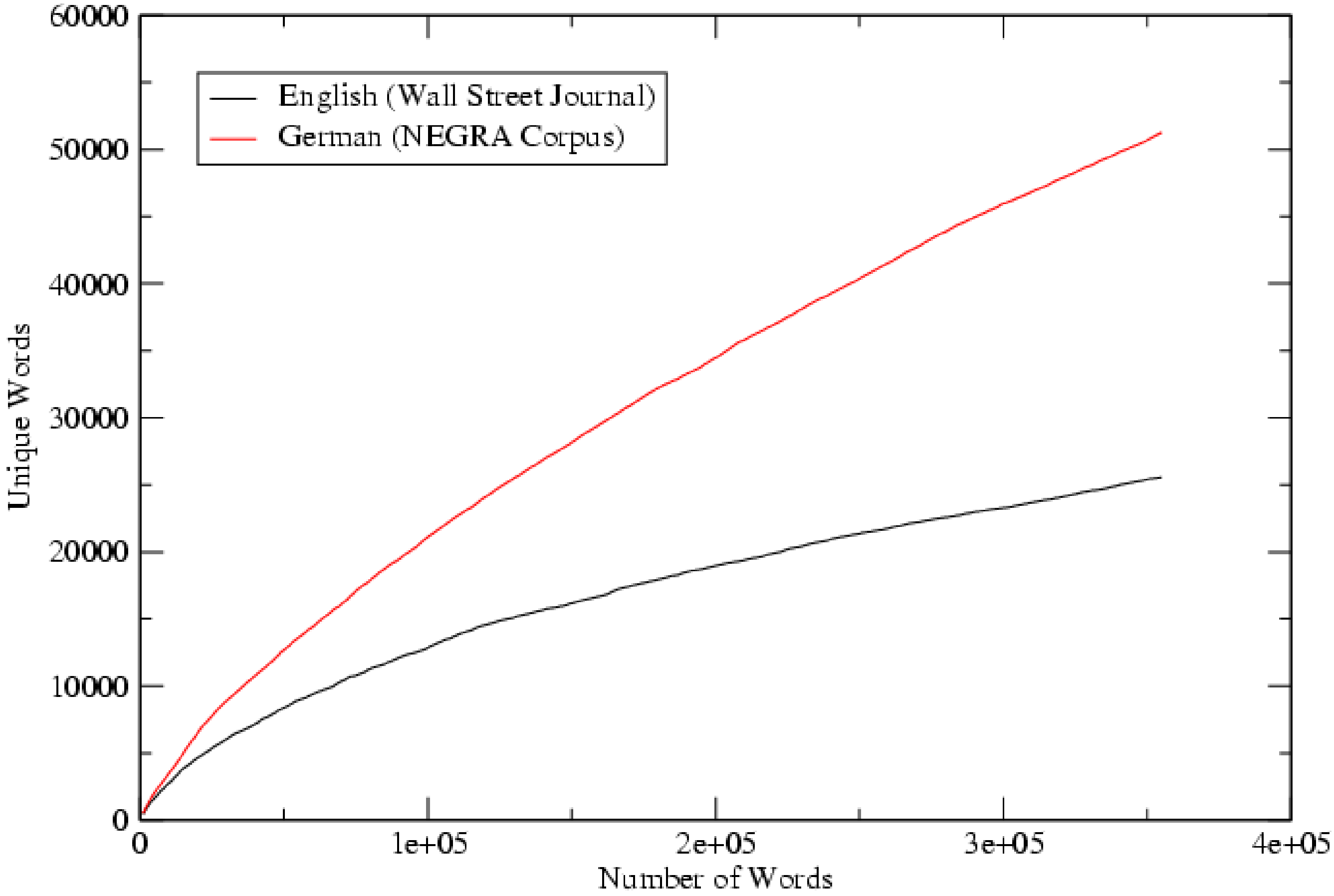
- BUT: lexicalization isn't the only trick that makes this model effective

  - We can turn lexicalization off, but use the other features

  - The model does quite well without lexicalization

  - Lexicalization does not help as much as it does in English

Why is the Effect of Lexicalization so Small?



English (Wall Street Journal)
German (NEGRA Corpus)
Unique Words
Number of Words
0
10000
20000
30000
40000
50000
60000
1e+05
2e+05
3e+05
4e+05

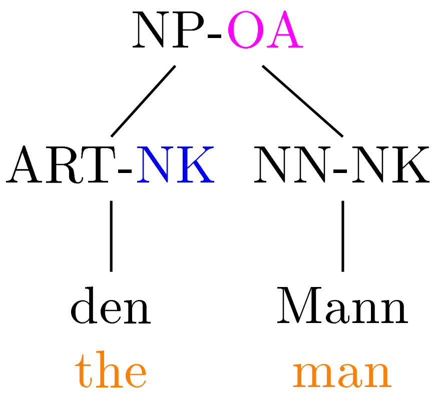
Another Option?

- We've tried "implicit expressiveness" due to lexicalization
- What about semi-automatically annotations to increase explicit expressiveness
  - Johnson (1998)
  - Klein & Manning (2003)
  - Last EGK meeting
- Final result:
  - Many of the modifications suggested by Johnson and Klein & Manning are specific to English and the corpus they used (Penn Treebank)
  - BUT: we identified some modifications which were more successful...

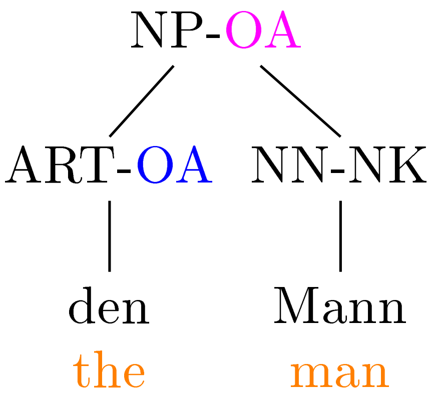
Grammatical Functions

- Mark things like subject, accusative or dative object, etc...
- Already annotated in NEGRA
- But they cannot be used directly...

Node Communication



Node Communication



Results

- "Lexical" communication helps almost as much as lexicalization
- (Yet to do: lexicalization + lexical communication)
    - It is possible both help in different ways
- With smoothing activated, the new grammar is more accurate than the lexicalized parsers...

Smoothing

- Some reasonable rules may not appear in the corpus
  - S $\rightarrow$ NP VVFIN ADV NP NP appears once in NEGRA;
  - S $\rightarrow$ NP VVFIN NP NP never appears.
- Rules which never appear have a probability of zero.
- But this is not always reasonable (is an ADV obligatory here?)
  - Use smoothing to ensure that all rules have non-zero probability
  - But smoothing introduces some new difficulties...

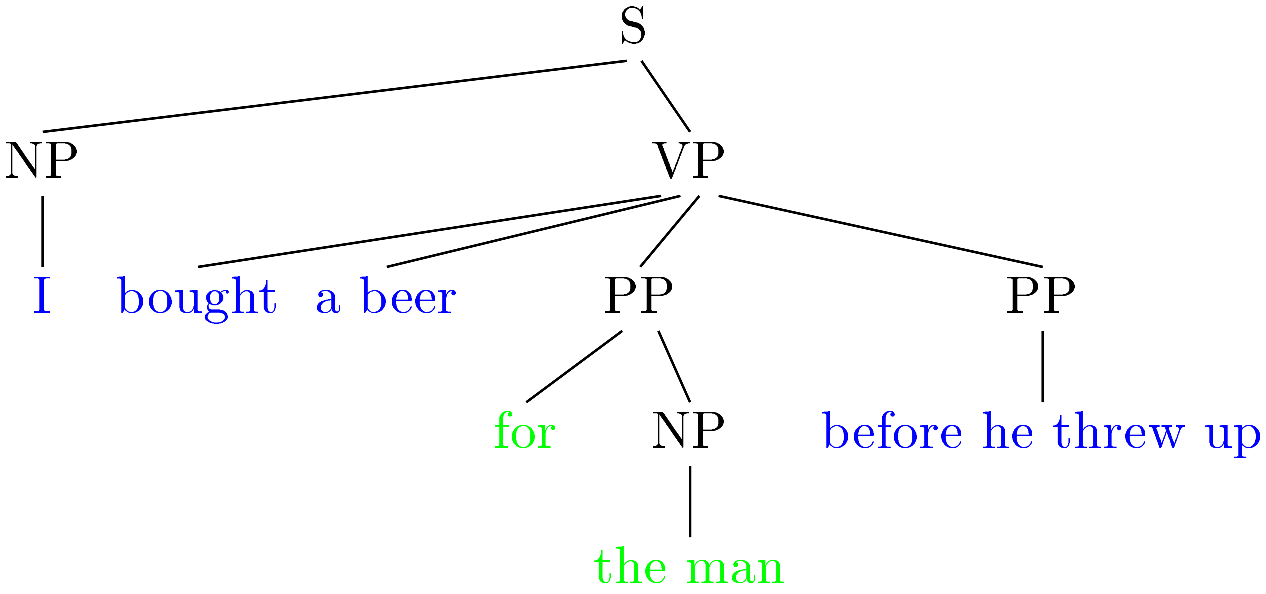
The Big Problem: Time & Memory Usage

- Parsing is very quick if you can fit chart/dynamic programming array in memory
- Size of array $= |\#\,\text{of rules}|^2 \cdot |\#\,\text{of words in the sentence}|$
- With smoothing, $|\#\,\text{of rules}|$ gets quite large
- Cannot use arrays
- Cannot parse exhaustively
    - Need to prune unwanted parses

Pruning (Goodman, 1997)

- “Fixed width” beam search
  - Keep track of n best partially constructed parses
  - Efficient, but parses are less accurate than exhaustive parsing
- “Variable width” beam search
  - Keep track of all partially constructed parses which are within a factor of the “best” parse so far
  - Parses are more accurate than exhaustive parsing, but parsing is VERY slow
- Multipass parsing

Inside & Outside Probabilities

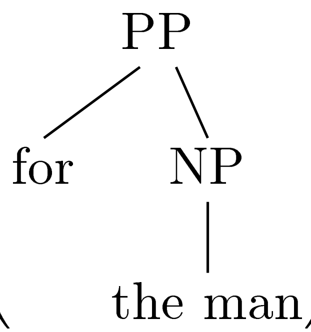


"I bought a beer for the man before he threw up"

Inside Probability

$$P(\mathrm{PP} \Rightarrow^{\star} \text{for the man})$$

$$= \quad P\left(\quad \right)$$



Outside Probability

$P(\text{I bought a beer PP before he threw up})$

Inside $\times$ Outside

$$\text{Inside(PP)} \times \text{Outside(PP)} = \text{Probability that PP spans the words ``for the man...'' in this sentence}$$

$$\frac{\text{Inside(PP)} \times \text{Outside(PP)}}{\text{Probability of Sentence}} = \text{Conditional probability that PP spans those words given this sentence}$$

- This last quantity is comparable across sentences
- So, we can define a cut-off value which works in all sentences
- We will call it the "node probability"

Multipass Parsing, Continued

- Parse exhaustively with a simple grammar
- Remember the node probabilities
- Parse again with the smoothed grammar, only considering nodes with a high enough node probability from the first pass

Why Bother?

- Lexicalization seems to work well
- And... it seems to work well with simple pruning strategies
- BUT:
  - Lexical communication seems like a powerful tool
  - Problem of more sparse data w/ lexicalization
  - Recent research suggests that unlexicalied grammars are unexpectedly useful (e.g. Klein & Manning 2003)
  - Smoothing would make the unlexicalized grammars even more useful
- The real question I want to explore is: how far can you take this idea?

To This Point...

- We have been using annotations available in NEGRA/TIGER
- Plus, some re-annotations to improve lexical communication
- Now: we try building on this by adding additional features
- Morphological analysis
  - Many readily available morph. analyzers available for German
  - Annotate trees w/ morphological information
  - Use this information in a new grammar

Difficulties

- What information do we include in the grammar?
- Words have ambiguous morphological tags
- Morphological taggers operate on a word-by-word basis
- Solution: hand-written rules to create graphs of disambiguated morphological tags
    - Use information from tree and surrounding words
    - Graphs 're-joined' to avoid exponential blow-up

Results

- Exhaustive parsing: low coverage
- With smoothing
  - Results slightly lower (but... some known "bugs" in hand-written rules)
  - ALSO: too slow to run on whole test corpus
- Additional benefits: gives a very complete description of parses
- NB: node labels are getting very complicated, but are still treated as atomic.
  - Is this a good idea?

Atomic Nodes vs. Features

- Not quite this complicated:

$$\begin{bmatrix} \text{sells} & \\ \textsc{tag} & \textsf{verb} \\ \textsc{lemma} & \textsf{to sell} \\ \textsc{tense} & \textsf{present} \\ \textsc{person} & \textsf{second} \\ \textsc{subcat} & \textsf{[NP, NP, PP]} \end{bmatrix}$$

- But still getting pretty bad: NP-PD.Akk.Neut.Sg
- Instead of treating the node as atomic, we can probabilistically model individual features separately
- Most recent work on probabilisitic feature grammars is based on ideas due to Abney (1997)...

Abney's Stochastic Attribute-Value Grammars

$$\begin{bmatrix} \text{S} \\ \\ \end{bmatrix} \rightarrow \begin{bmatrix} \text{A} \\ \boxed{1} \end{bmatrix} \begin{bmatrix} \text{A} \\ \boxed{1} \end{bmatrix}$$

$$\begin{bmatrix} \text{S} \\ \\ \end{bmatrix} \rightarrow \begin{bmatrix} \text{B} \\ \\ \end{bmatrix}$$

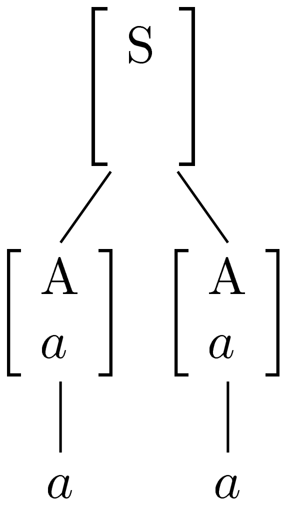
$$\begin{bmatrix} \text{A} \\ a \end{bmatrix} \rightarrow a$$

$$\begin{bmatrix} \text{A} \\ b \end{bmatrix} \rightarrow b$$

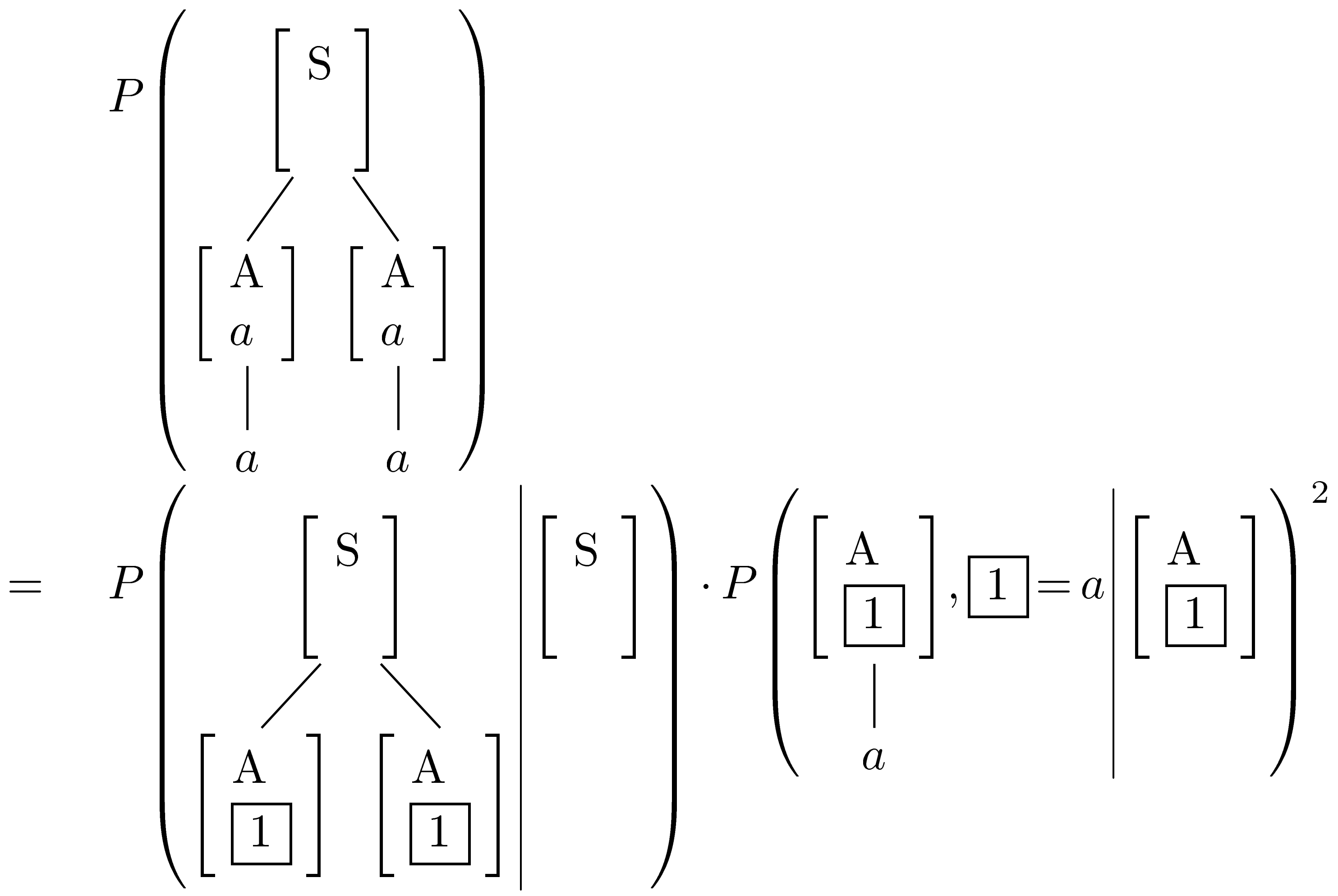
$$\begin{bmatrix} \text{B} \\ \\ \end{bmatrix} \rightarrow a\,a$$

$$\begin{bmatrix} \text{B} \\ \\ \end{bmatrix} \rightarrow b\,b$$

The Tree



The Tree's Probability



The Tree's Probability

$$P\left(\begin{array}{c} \begin{bmatrix}\text{S}\\ \\ \end{bmatrix} \\ \diagup \qquad \diagdown \\ \begin{bmatrix}\text{A}\\ a\end{bmatrix} \quad \begin{bmatrix}\text{A}\\ a\end{bmatrix} \\ | \qquad\quad | \\ a \qquad\quad a \end{array}\right)$$

$$= \quad P\left(\begin{array}{c} \begin{bmatrix}\text{S}\\ \\ \end{bmatrix} \\ \diagup \qquad \diagdown \\ \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix} \quad \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix} \end{array} \middle| \begin{bmatrix}\text{S}\\ \\ \end{bmatrix}\right) \cdot P\left(\begin{array}{c} \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix} \\ | \\ a \end{array}, \boxed{1} = a \middle| \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix}\right)^2$$

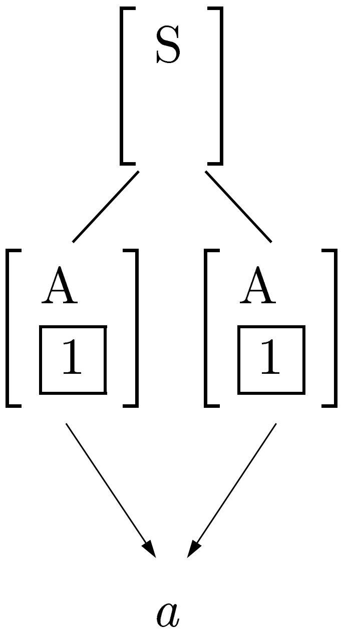
The Tree's Probability

$$P\left(\begin{array}{c} \begin{bmatrix}\text{S}\\ \\ \\ \end{bmatrix} \\ \diagup \qquad \diagdown \\ \begin{bmatrix}\text{A}\\ a\end{bmatrix} \quad \begin{bmatrix}\text{A}\\ a\end{bmatrix} \\ | \qquad\qquad | \\ a \qquad\qquad a\end{array}\right)$$

$$= \quad P\left(\begin{array}{c} \begin{bmatrix}\text{S}\\ \\ \\ \end{bmatrix} \\ \diagup \qquad \diagdown \\ \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix} \quad \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix}\end{array} \middle| \begin{bmatrix}\text{S}\\ \\ \\ \end{bmatrix}\right) \cdot P\left(\begin{array}{c} \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix} \\ | \\ a\end{array} \middle| \begin{bmatrix}\text{A}\\ \boxed{1}\end{bmatrix}\right)^2 \cdot P(\boxed{1} = a)$$

Abney's Solution

- Tree + variables = branches re-merge



Abney's Solution

- Idea: instead of calculating probability of tree from rules alone, use rules + re-merged branches
- But: Abney does not provide an algorithm
- Rather, he suggests an algorithm that greedily looks for correlations that might be evidence for variable equality
- Variables are exogenous to Abney's model

Other Possibilities

- Only assign value once (Stolke, 1995; Goodman, 1997; Schmid, 2002)
- "Remove the variable." Instead of:

$$\begin{bmatrix} \text{S} \\ \\ \end{bmatrix} \rightarrow \begin{bmatrix} \text{A} \\ \boxed{1} \end{bmatrix} \begin{bmatrix} \text{A} \\ \boxed{1} \end{bmatrix}$$

Use two rules:

$$\begin{bmatrix} \text{S} \\ \\ \end{bmatrix} \rightarrow \begin{bmatrix} \text{A} \\ a \end{bmatrix} \begin{bmatrix} \text{A} \\ a \end{bmatrix}, \quad \begin{bmatrix} \text{S} \\ \\ \end{bmatrix} \rightarrow \begin{bmatrix} \text{A} \\ b \end{bmatrix} \begin{bmatrix} \text{A} \\ b \end{bmatrix}$$

Future Work

- Improve pruning strategies
- Tune hand-written rules
- Some more experiments with both lexicalization and features

The Overall Argument

- Implicitly expressive
  - Efficient parsing
  - Accurate
- Explicitly expressive
  - Competitive accuracy
  - More information
    - e.g. for discourse analysis
  - Efficiency: yes, if smoothing is not used
  - Coverage
    - Yes, if grammar is simple
    - Yes, if grammar is complex & smoothing is used