The problem: discontinuity/movement/free word order

- *The American Bar Association's House of Delegates passed* ***a resolution in 1985 condemning the IRS reporting requirement****.*

- wh-movement
- relative-clause extraposition
- heavy NP shift
- right node raising
- free word order
- . . .

- phrase-structure *vs.* dependency

Existing approaches – Phrase structure

- traces – problematic, cf. below
- Charniak 1996–2001 ≈90% on PARSEVAL
- Collins 1997
  - ≈90% on PARSEVAL
  - 91% accuracy for recovering dependencies
- Bod 2000, Ratnaparkhi 1999, Plaehn 1999
- Collins *et al.* 1999 for Czech, 76–81% accuracy on dependencies

Existing approaches – Dependency and Co.:

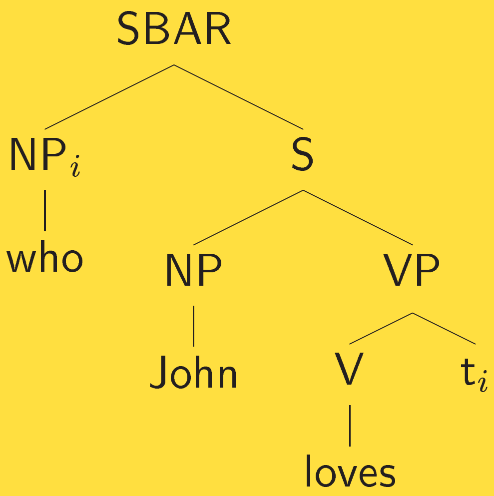
- ????
- Collins 1996(?) ≈85% on PARSEVAL
- Hockenmaier 1999, CCG-approach, feeble results

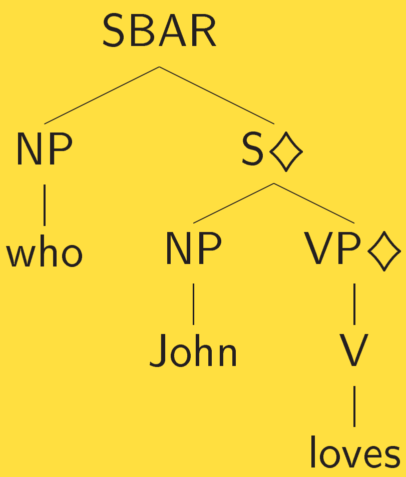
Why does dependency not work?

- more generative power $\rightarrow$ sparser data
  - seen/unseen distinction is less reliable
  - different behaviour of a lexeme is levelled out
- heavily lexicalised – sparse data
- relies on lexical-semantic information not present in the treebank
- slow
- not easy (and often costly) to make them robust

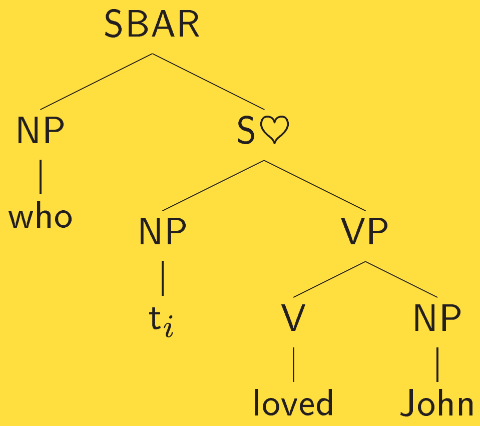
Why does phrase structure not work?

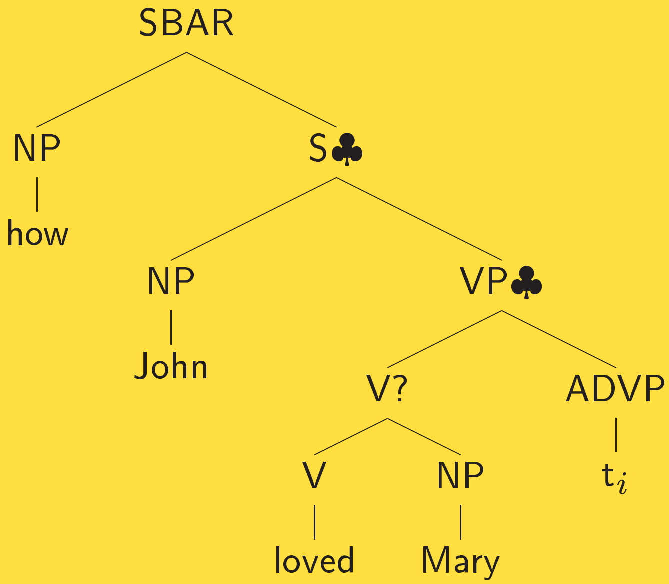
- it is not designed to capture discontinuity
- traces – they are not in good terms with bottom-up parsing





- *I wonder* ***who loved John***.
- *I wonder* ***how John loved Mary***.





- New rules – instead of $\texttt{VP} \rightarrow \texttt{V NP}$ and $\texttt{S} \rightarrow \texttt{NP VP}$, we have

$\texttt{VP} \rightarrow \texttt{V NP}$	$\texttt{S} \rightarrow \texttt{NP VP}$
$\texttt{VP}_{\diamondsuit} \rightarrow \texttt{V}$	$\texttt{S}_{\diamondsuit} \rightarrow \texttt{NP VP}_{\diamondsuit}$
$\texttt{VP}_{\clubsuit} \rightarrow \texttt{V? ADVP}$	$\texttt{S}_{\heartsuit} \rightarrow \texttt{NP VP}$
$\ldots$	$\texttt{S}_{\clubsuit} \rightarrow \texttt{NP VP}_{\clubsuit}$

- *Who$_i$ do you think John really loves t$_i$?*

- number of rules – an important factor in speed

- data sparsity

Further problems with phrase-structure

- full parse
- making decisions even if it does not have enough information – typically dependency decisions (verb argument structure, PP attachment)
- probability models are rather poor (CFG backbone)
- lexicalised but back-off usually to tags (though Amit . . .)

Why phrase structure/dependency?

- even in freer word order languages (eg. Hungarian, Czech), the internal structure of most of the phrases is more or less fixed
- phrase structure parsers are quite efficient – in principle
- in many tasks we are interested in both phrase structure and dependency information

A possible way out: let's have them both!

- reducing the search space for the second parser – speed-up
- combining bottom-up/chart and top-down/head-driven parsing
- parse selection problem: the dependency parser selects from the parses proposed by the PS parser
- PS parser does not have to produce full parse: higher precision and possible speed-up

Three main issues

- the PS parser
- the dependency parser
- the interaction between them

The dependency parser

- not necessarily stochastic
- Collins'96?
- Miles: Stochastic SetCCG
- stochastic Link Grammar?
- external source: induced or hand-crafted valency lexicon

The phrase structure parser

- high precision
- fast
- stop when uncertain in a decision (esp. at "traces" and argument structures)
- parses can contain "holes" – ie. some phrase-internal structures are "underspecified".
  - attachment ambiguity not resolved
  - subclause-internal verb-argument structure not filled in if not known for sure

Three sub-issues

- the input to the parser
- the linguistic model for the parser – rules and their probabilities
- partial parse selection

The input

- words or tags or words+tags
  - one tag per word
  - several tags per word
  - several tags per word with their probability distribution
- experiment: to compare the three approaches (Charniak *et al.* ms.)
- idea: guiding the parser, instead of deciding instead of it
- why not words: highly inflecting languages

Part-of-Speech (and morphological) tagging

- modularity
- possible tags (emission probabilities) and best tag sequence (transition probabilities)
- an external guesser: for each word the possible tags and their probability distribution
- providing tags is rule-based, providing the probabilities is corpus-based

Part-of-Speech (and morphological) tagging

- class-based approach

- when calculating the lexical probabilities
  - based on the word
  - and the ambiguity class (eg. *book* N-V)
  - the more frequent a word is in the training corpus, the less influence the class has on its behaviour

- results: ≈2% improve on accuracy (from ≈96% to ≈98%)

Part-of-Speech (and morphological) tagging

- using backoff
- using knowledge-bases with corpora
- variation on the same theme: communication between different modules (and probability models!) via probability distributions

Rule induction

high precision $\rightarrow$ frequent rules

$$\left.\begin{array}{l}
\mathsf{NP} \rightarrow \ \mathsf{NNP\ NNP} \\
\mathsf{NP} \rightarrow \ \mathsf{NNP\ NNP\ NNP\ NNP} \\
\mathsf{NP} \rightarrow \ \mathsf{NNP\ NNP\ NNP\ NNP\ NNP} \\
\mathsf{NP} \rightarrow \ \mathsf{NNP\ NNP\ NNP\ NNP\ NNP\ NNP}
\end{array}\right\} \begin{array}{rl}
\mathsf{NP} \rightarrow & \mathsf{NNP\ NNP}_{\heartsuit} \\
\mathsf{NP}_{\heartsuit} \rightarrow & \mathsf{NNP\ NP}_{\heartsuit} \\
\mathsf{NP}_{\heartsuit} \rightarrow & \mathsf{NNP\ NNP}
\end{array}$$

$$\mathsf{NP} \rightarrow \ \mathsf{CD\ ,\ CD\ ,\ CD\ ,\ CD\ ,\ CD\ ,\ CD\ CC\ CD}$$

Rule induction

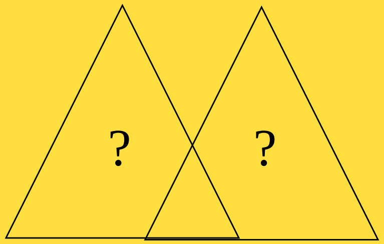
- introducing new non-terminals – pre- and postprocessing the treebank
- decreases the number of rules
- automatic
- inducing probabilities for rules tree-bank learning or EM-training on this special corpus
- a possible way of smoothing

Parse selection

- generally: the most probable full parse
- most probable partial parse
- danger: no parse at all
- the question: how to compare partial trees with different yield?

Parse selection

1.



2.

Parse selection

- Viterbi-search on the chart
- $\sim$finding the best sequence in a word-lattice
- $\sim$finding the best state-sequence in a HMM
- combined with emission probabilities – cf. Thorsten
- or recursively

The interaction between the parsers

- k-best partial parses or partial parses within the beam of the best partial parse
- 1-best: sequential processing
- all parses: merging two probability models
- somewhere in between: guiding the dependency parser as well as restricting the search space
- more interaction between the two parsers?

Summary

- hybrid system to handle discontinuity/free word order
- stochastic phrase structure parser producing high precision partial parses
- (stochastic?) dependency-based parser guided by the lower-level parser and possibly other modules
- communication between the modules via probability distributions

To do

- experiments with PoS tagging
  - tagger–parser interaction
  - class-based, modular PoS tagging
- rule induction from rules
- parse selection – “best paths” on the chart
- dependency-parser