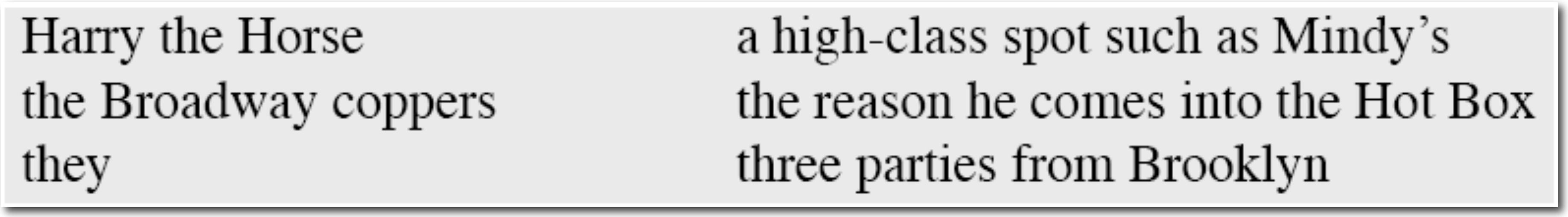
Natural Language Processing: Lab on Syntactic Parsing

Olga Uryupina
DISI, University of Trento
uryupina@gmail.com

Two views of linguistic structure:
1. Constituency (phrase structure)

- The basic idea here is that groups of words within utterances can be shown to act as single units
- For example, it makes sense to the say that the following are all ***noun phrases*** in English...

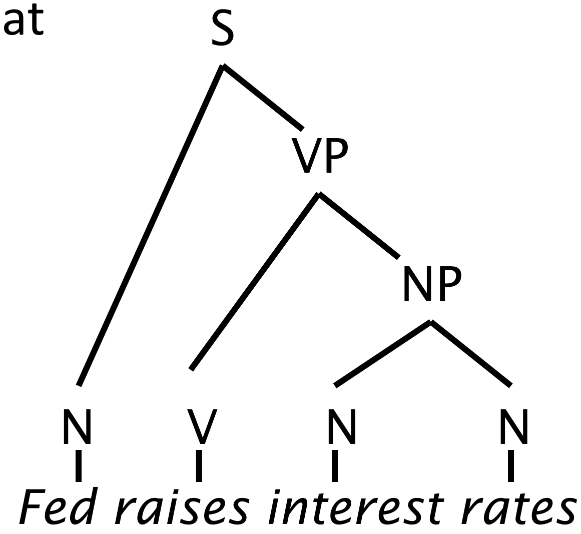


Harry the Horse
the Broadway coppers
they
a high-class spot such as Mindy's
the reason he comes into the Hot Box
three parties from Brooklyn

- Why? One piece of evidence is that they can all precede verbs.

Two views of linguistic structure:
1. Constituency (phrase structure)

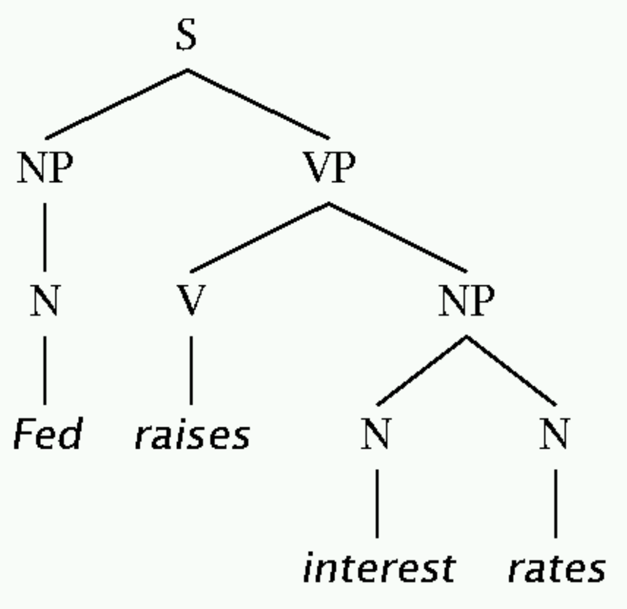
- Phrase structure organizes words into nested constituents.
- How do we know what is a constituent? (Not that linguists don't argue about some cases.)
  - Distribution: a constituent behaves as a unit that can appear in different places:
    - John talked [to the children] [about drugs].
    - John talked [about drugs] [to the children].
    - *John talked drugs to the children about
  - Substitution/expansion/pro-forms:
    - I sat [on the box/right of the box/there].



Headed phrase structure

To model constituency structure:

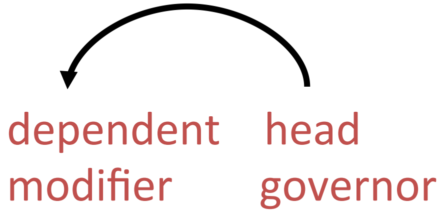
- VP → … VB* …
- NP → … NN* …
- ADJP → … JJ* …
- ADVP → … RB* …
- PP → … IN* …



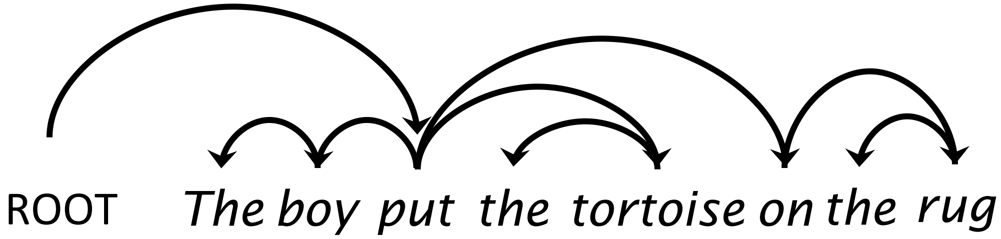
- Bracket notation of a tree (Lisp S-structure):
  (S (NP (N Fed)) (VP (V raises) (NP (N interest) (N rates)))

Two views of linguistic structure:
2. Dependency structure

- In CFG-style phrase-structure grammars the main focus is on *constituents.*
- But it turns out you can get a lot done with binary relations among the lexical items (words) in an utterance.
- In a dependency grammar framework, a parse is a tree where
  - the nodes stand for the words in an utterance
  - The links between the words represent dependency relations between pairs of words.
    - Relations may be typed (labeled), or not.

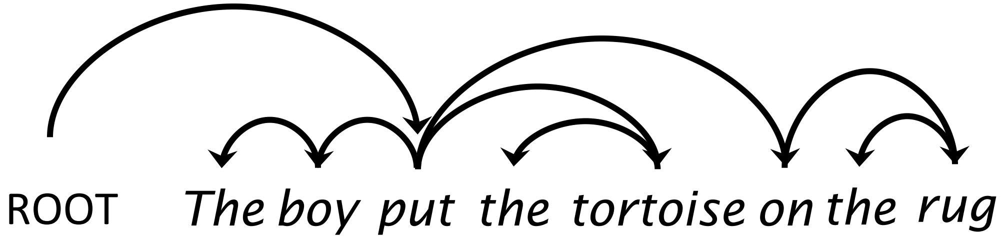

dependent head
modifier governor

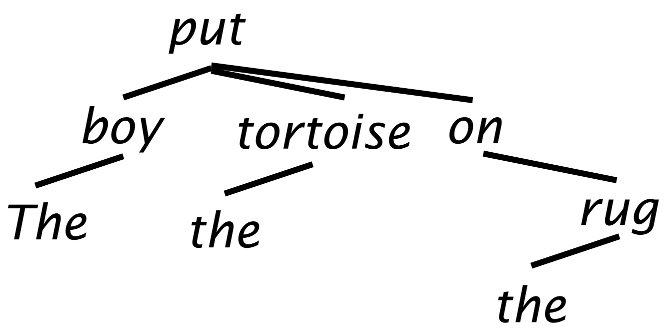
Sometimes arcs drawn
in opposite direction


ROOT *The boy put the tortoise on the rug*

Two views of linguistic structure: 2. Dependency structure

- Alternative notations (e.g. rooted tree):





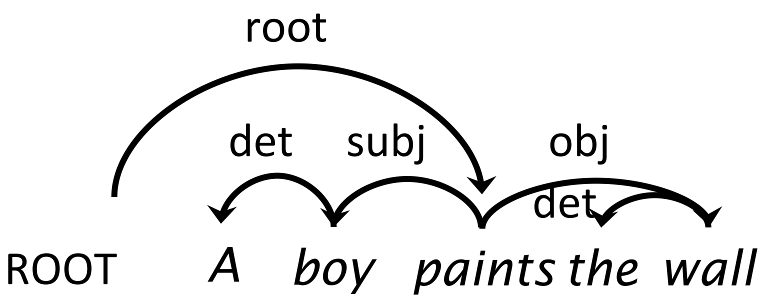
Dependency Labels

Argument dependencies:

- Subject (subj), object (obj), indirect object (iobj)…

Modifier dependencies:

- Determiner (det), noun modifier (nmod), verbal modifier (vmod), etc.



Tools

- Charniak Parser (constituent parser with discriminative reranker)
- Stanford Parser (provides constituent and dependency trees)
- Berkeley Parser (constituent parser with latent variables)
- MST parser (dependency parser, needs POS tagged input)
- Bohnet's parser (dependency parser, needs POS tagged input)
- Malt parser (dependency parser, needs POS tagged input)

Berkeley Parser

"Learning Accurate, Compact, and Interpretable Tree Annotation"

Slav Petrov, Leon Barrett, Romain Thibaux and Dan Klein

in COLING-ACL 2006

and

"Improved Inference for Unlexicalized Parsing"

Slav Petrov and Dan Klein

in HLT-NAACL 2007

Downloading files

Berkeley parser

http://code.google.com/p/berkeleyparser/

-> parser

-> English grammar

EVALB

http://nlp.cs.nyu.edu/evalb/

-> “make” to install

Data & slides

http://bart-coref.org/labs

Test runs

Running the parser on a toy bnews test set:

```
java  -Xmx2000m -jar BerkeleyParser-1.7.jar
-gr eng_sm6.gr <prs-lab/data/bn_raw.test
>bn_prs.out
```

Running EVALB to assess the performance:

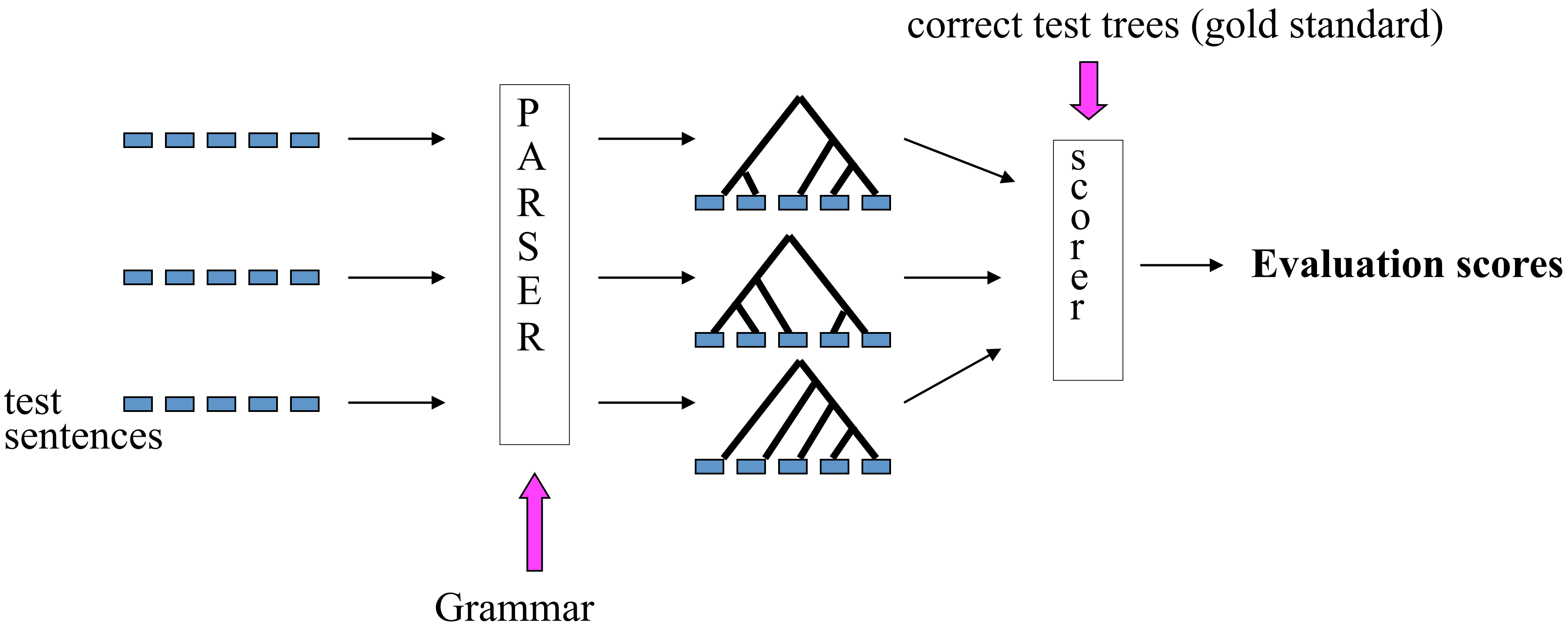
```
./evalb -p sample/sample.prm ../prs-lab/
data/bn_prs.test ../bn_prs.out
```

Does it make sense?

- Evaluation
  - EVALB, in a minute
- Grammar

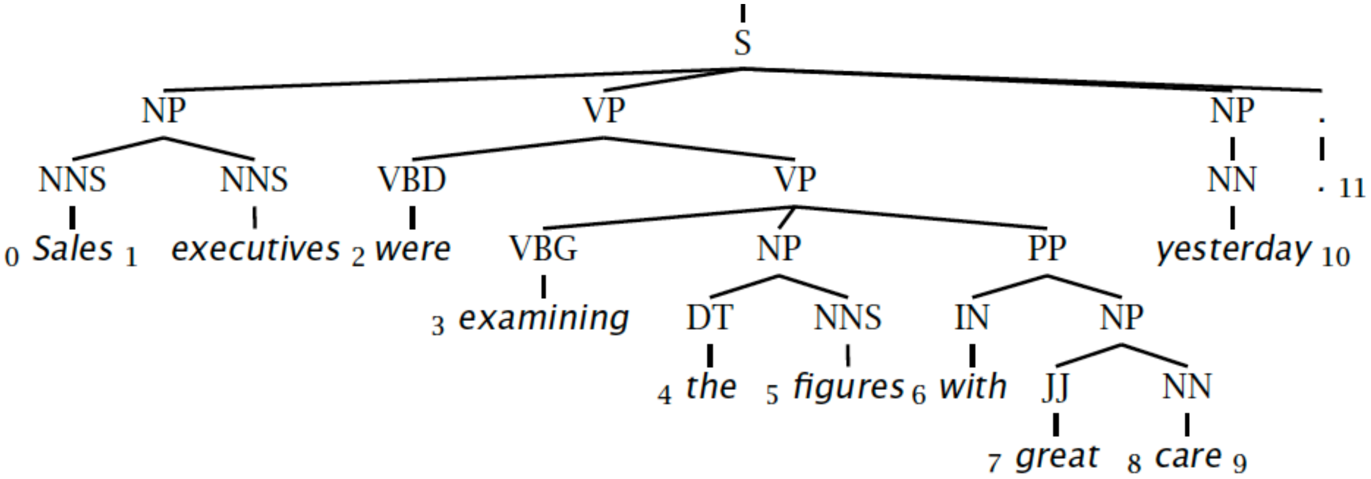
```
java  -Xmx2000m  -cp BerkeleyParser-1.7.jar
edu/berkeley/nlp/PCFGLA/
WriteGrammarToTextFile eng_sm6.gr grammartxt
```

Evaluating Parser Performance

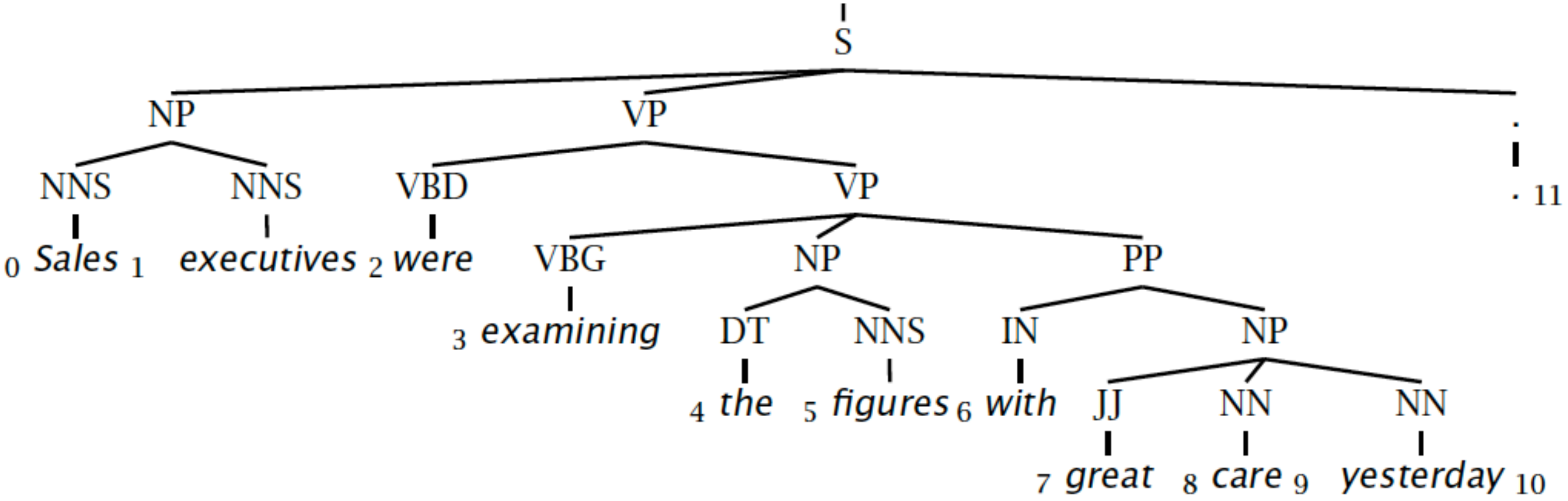


Evaluation of Constituency Parsing: bracketed P/R/F-score

Gold standard brackets: **S-(0:11), NP-(0:2)**, VP-(2:9), VP-(3:9), **NP-(4:6)**, PP-(6-9), NP-(7,9), NP-(9:10)



Candidate brackets: **S-(0:11), NP-(0:2)**, VP-(2:10), VP-(3:10), **NP-(4:6)**, PP-(6-10), NP-(7,10)



Evaluation of Constituency Parsing: bracketed P/R/F-score

Gold standard brackets:

S-(0:11), NP-(0:2), VP-(2:9), VP-(3:9), **NP-(4:6)**, PP-(6-9), NP-(7,9), NP-(9:10)

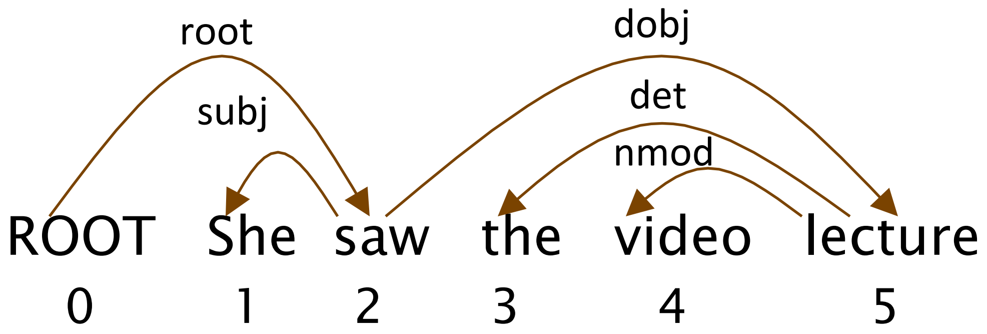
Candidate brackets:

S-(0:11), **NP-(0:2)**, VP-(2:10), VP-(3:10), **NP-(4:6)**, PP-(6-10), NP-(7,10)

Labeled Precision	3/7 = 42.9%
Labeled Recall	3/8 = 37.5%
F1	40.0%

(Parseval measures)

Evaluation of Dependency Parsing: (labeled) dependency accuracy



Unlabeled Attachment Score (UAS)
Labeled Attachment Score (LAS)
Label Accuracy (LA)

UAS = 4 / 5 = 80%
LAS = 2 / 5 = 40%
LA = 3 / 5 = 60%

Gold

1	She	2	subj
3	saw	0	root
4	the	5	det
5	video	5	nmod
6	lecture	2	dobj

Parsed

1	She	2	subj
3	saw	0	root
4	the	4	det
5	video	5	vmod
6	lecture	2	iobj

Learning a new grammar

```
java  -Xmx2000m  -cp BerkeleyParser-1.7.jar
edu.berkeley.nlp.PCFGLA.GrammarTrainer -path
prs-lab/data/bn_prs.train -out eng_bn.gr -
treebank SINGLEFILE
```

Learning a new grammar: tips

Need a lot of training data!

- WSJ: 1 million tokens, 40k sentences

Tagsets: data sparsity problem

- You might have to simplify your tagset