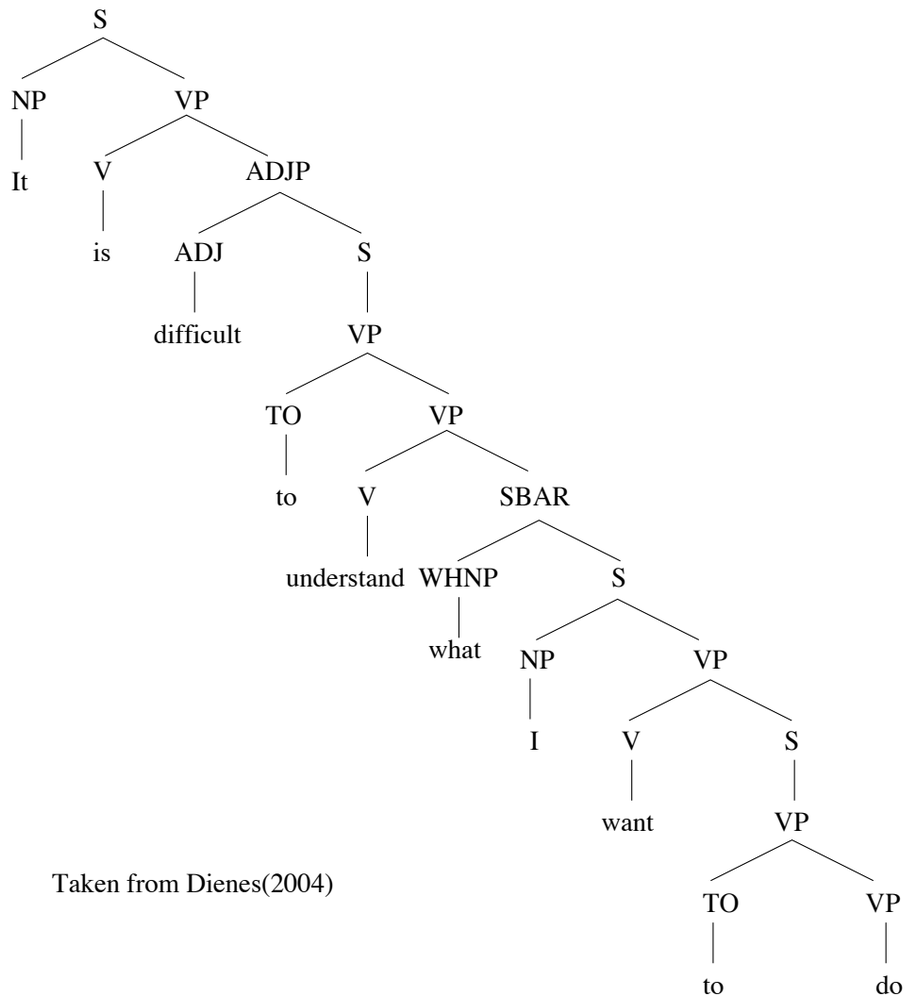
Introduction to Syntax and Parsing
ACS 2015/16
Stephen Clark

L2: Introduction to Statistical Parsing

UNIVERSITY OF
CAMBRIDGE

Automatic Parsing

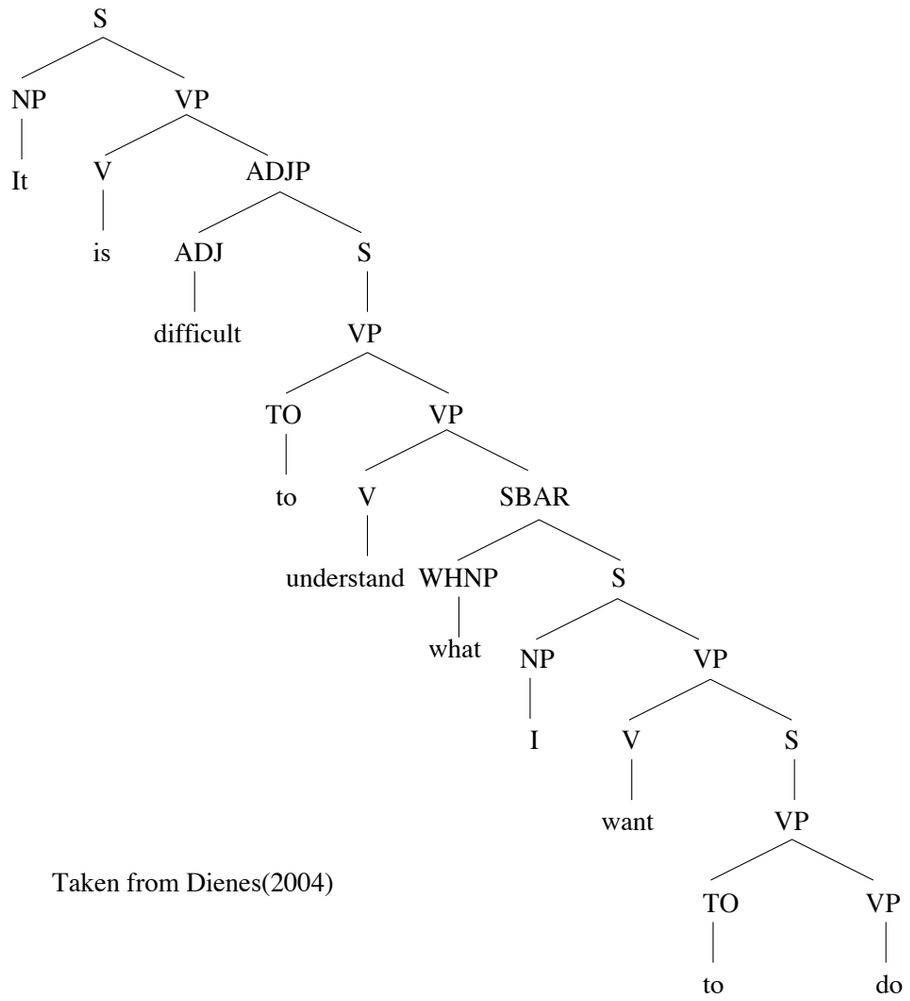


Taken from Dienes(2004)

- Where does the grammar come from?
- What's the algorithm for generating possible parses?
- How do we decide between all the parses?

UNIVERSITY OF CAMBRIDGE

The Penn Treebank



Taken from Dienes(2004)

- Provides the possible phrase structure rules
- Provides data for estimating parse selection models
- Provides test data for evaluation

UNIVERSITY OF CAMBRIDGE

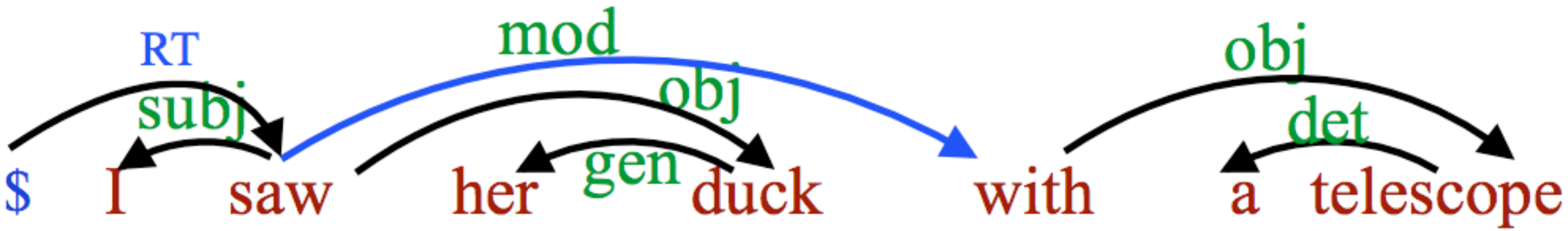
Problems with the PTB Parsing Task

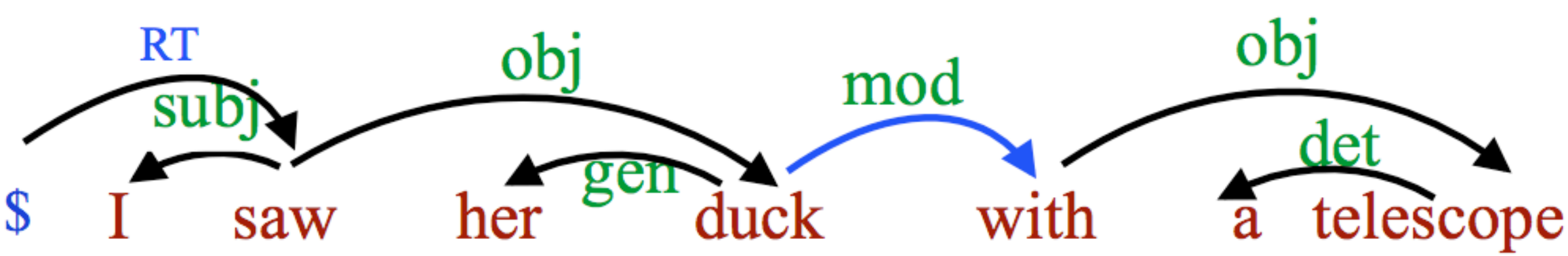
- It focuses on one language (English)
- It focuses on one domain (newswire)
- The test data hasn't changed for 20 years

Dependency Parsing

- Currently the dominant parsing paradigm:
  - there are treebanks for lots of languages
  - dependencies are useful for various tasks
  - performance is comparable to PS parsers
  - the “grammar” is easy to understand (!)
  - almost entirely data driven

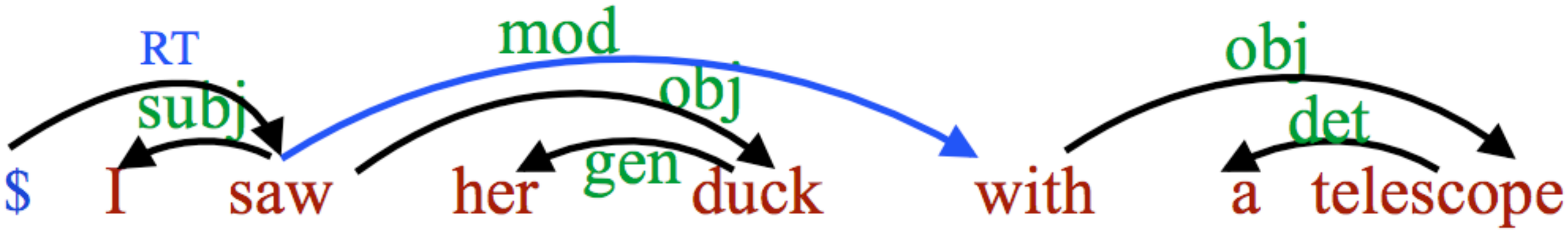
Dependency Trees





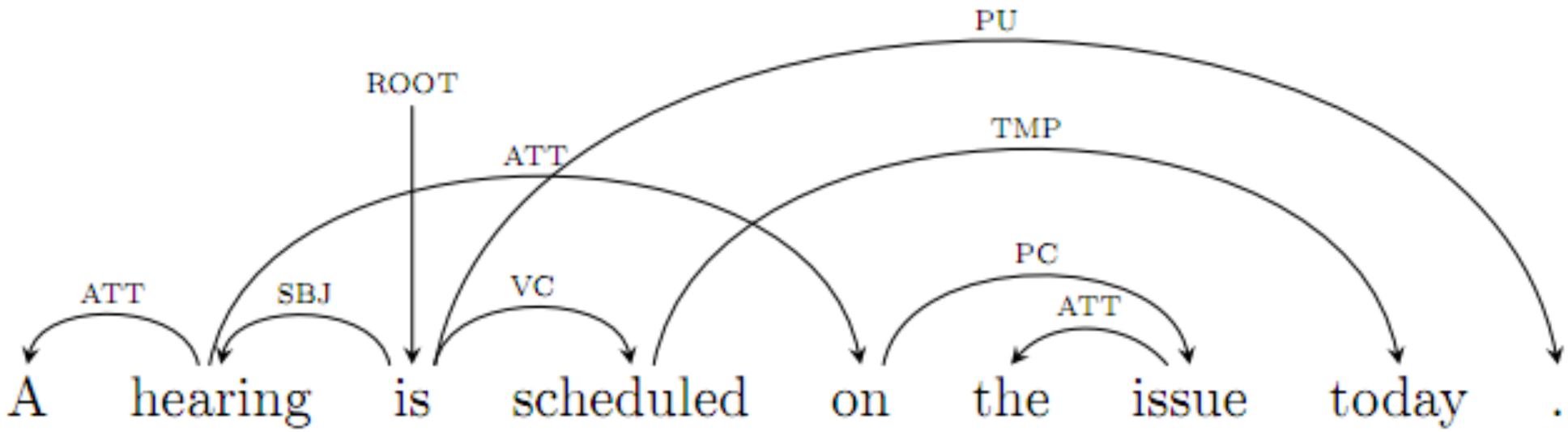
taken from Wang and Zhang, NAACL tutorial 2010

Dependency Trees more Formally



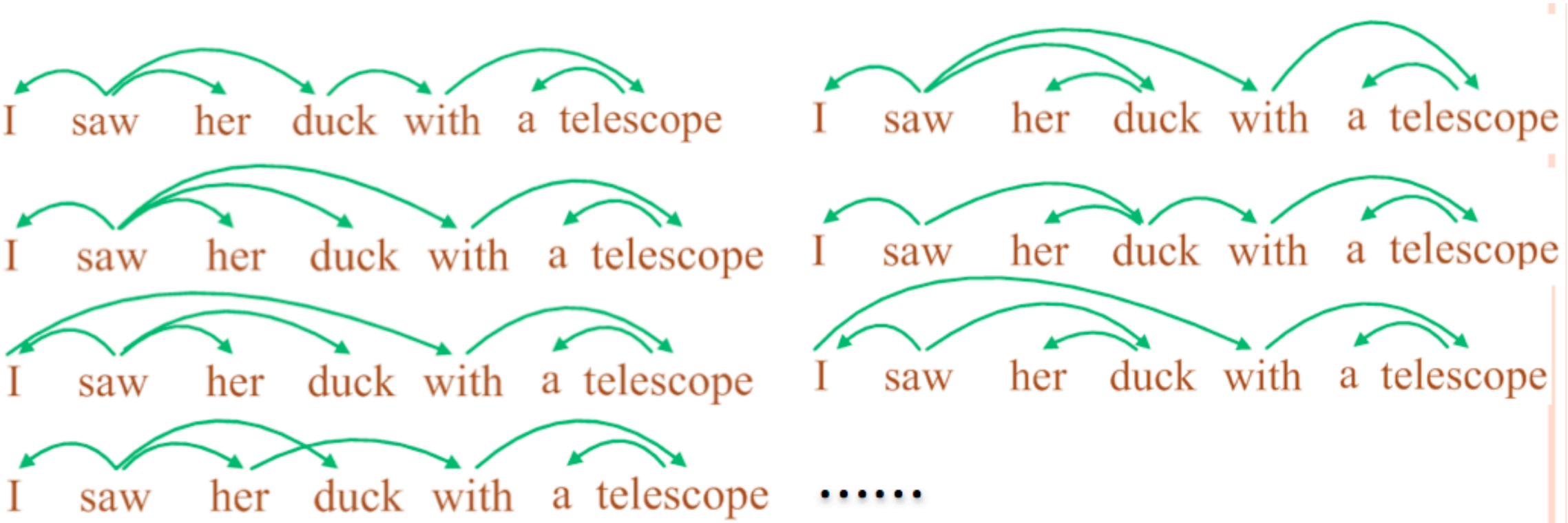
- A directed graph with the following constraints:
  - connected
  - acyclic
  - single-head
  - projective (no crossing links)

Crossing Dependencies



- Rare in English, but more common in other languages (Czech, German)
- Requires a different parsing algorithm

Graph-Based Models



taken from Wang and Zhang, NAACL tutorial 2010

- Score each possible tree (according to some model)
- Search for the tree with the highest score
- Dynamic Programming (DP) typically used to do the (optimal) search

Edge-Based Factorisation Model

$$
\begin{aligned}
Y^* &= \arg \max_{Y \in \Phi(X)} score(Y|X) \\
&= \arg \max_{Y \in \Phi(X)} \sum_{x_i \rightarrow x_j \in Y} score(x_i \rightarrow x_j)
\end{aligned}
$$

where X is the sentence,

$\Phi(X)$ is the set of possible dependency trees for X,

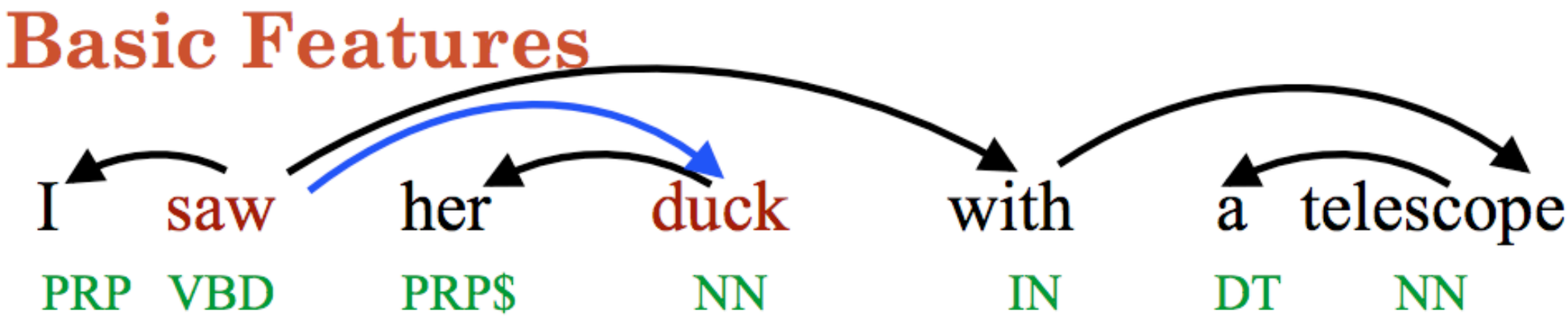
$x_i \rightarrow x_j$ is a dependency link between words x_i and x_j

Edge-Based Linear Model

$$score(x_i \rightarrow x_j) = \sum_k \lambda_k \, . \, f_k(x_i \rightarrow x_j)$$
$$= \overline{\lambda} . \overline{f}(x_i \rightarrow x_j)$$

- Features have to be local to an edge in the graph
  - but can span the whole sentence
- Various ways to estimate the weights, including structured perceptron
- Large numbers of binary features are needed for good performance
  - but recent work using neural networks obviates this need

Example Features



- Uni-gram features
- Bi-gram features
- In between POS features
- Surrounding word POS features

Saw_VBD, saw, VBD duck_NN, duck, NN
saw_VBD_duck_NN, VBD_duck_NN, saw_duck_NN, saw_VBD_NN, saw_VBD_duck, Saw_duck, VBD_NN
VBD_PRP$_NN
VBD_PRP$_PRP$_NN, PRP_VBD_PRP$_NN, VBD_PRP$_NN_IN, PRP_VBD_NN_IN

taken from Wang and Zhang, NAACL tutorial 2010