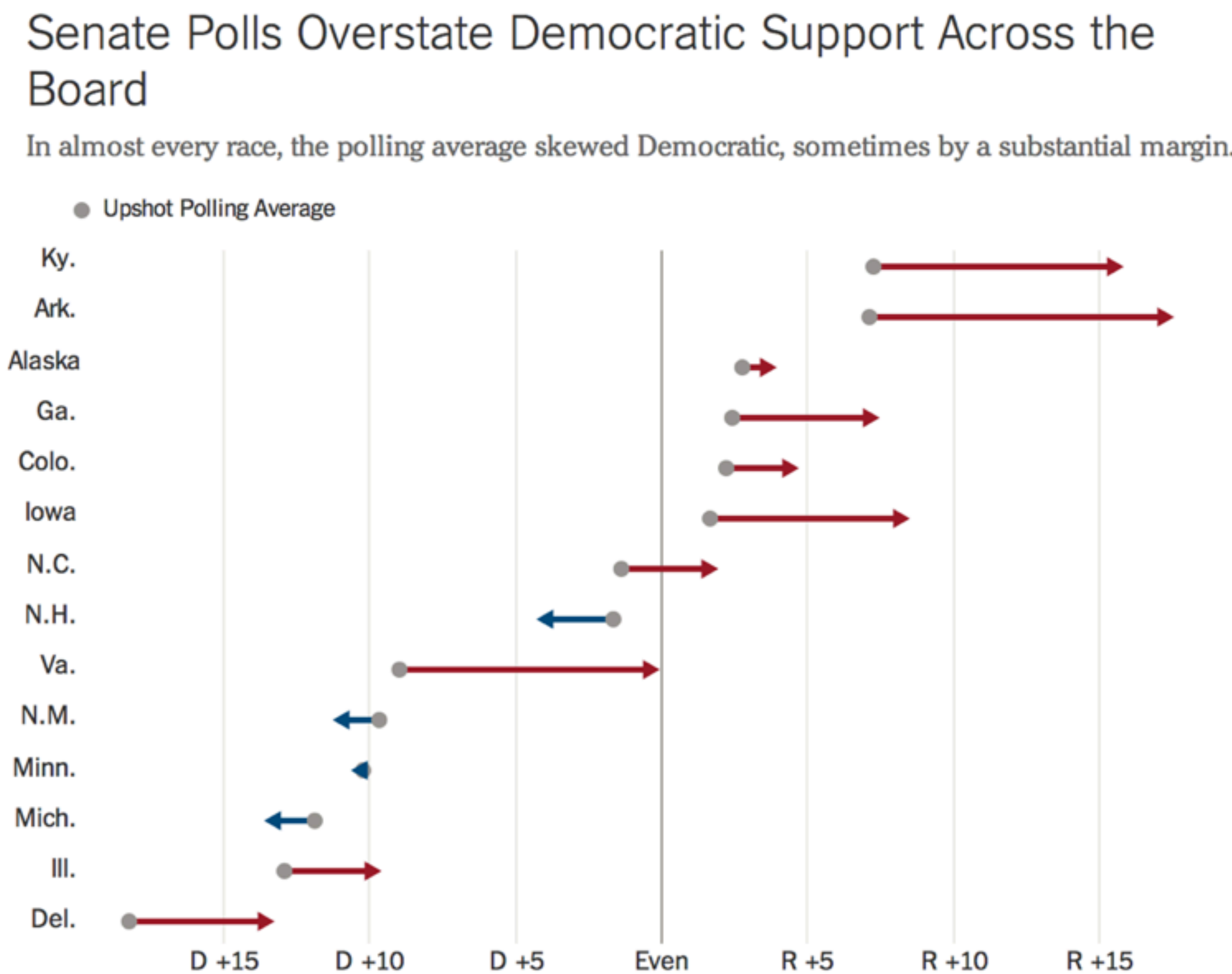
Lecture 18
Syntactic Dependencies

Intro to NLP, CS585, Fall 2014

http://people.cs.umass.edu/~brenocon/inlp2014/

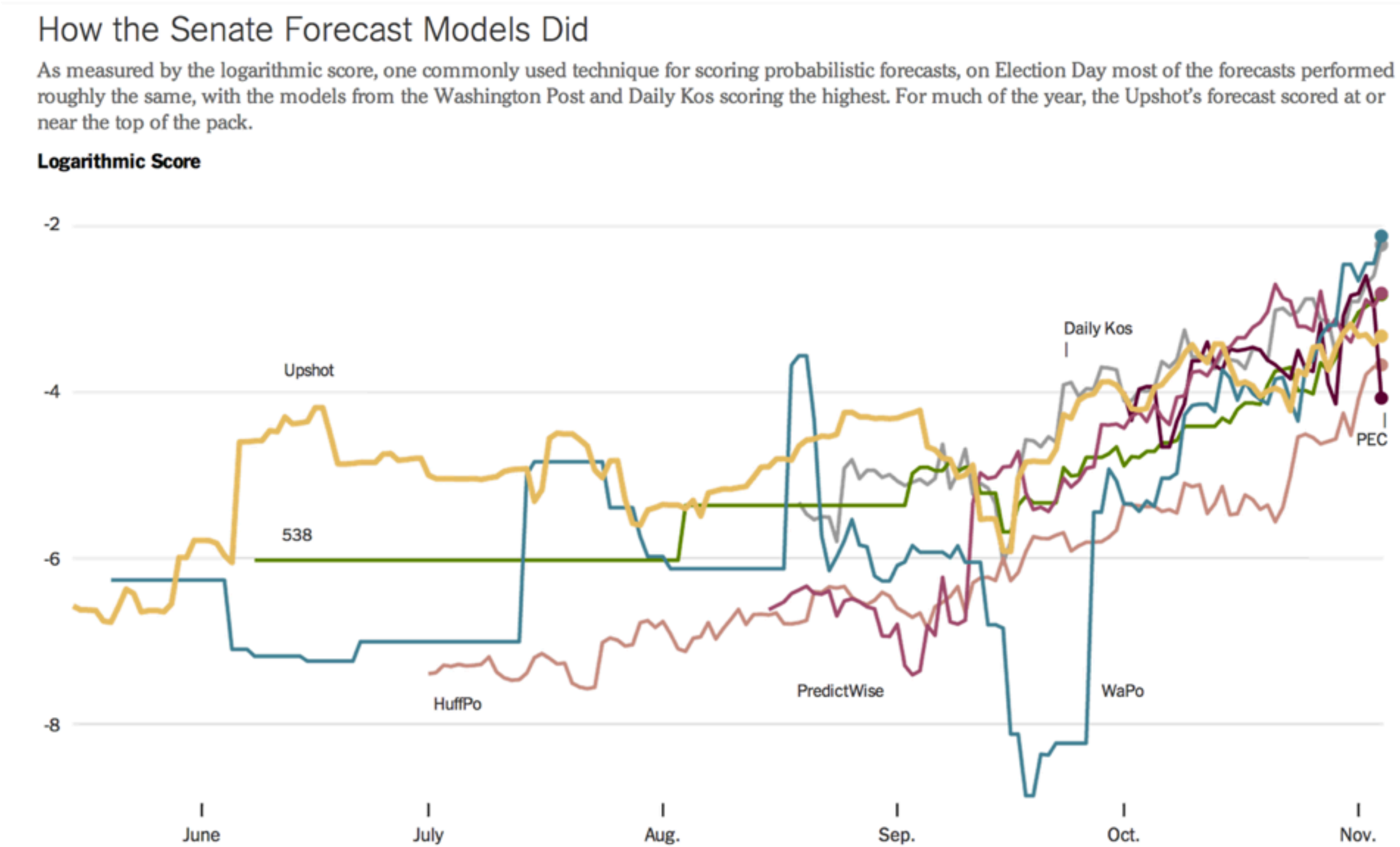
Brendan O'Connor

Error analysis: election forecasting



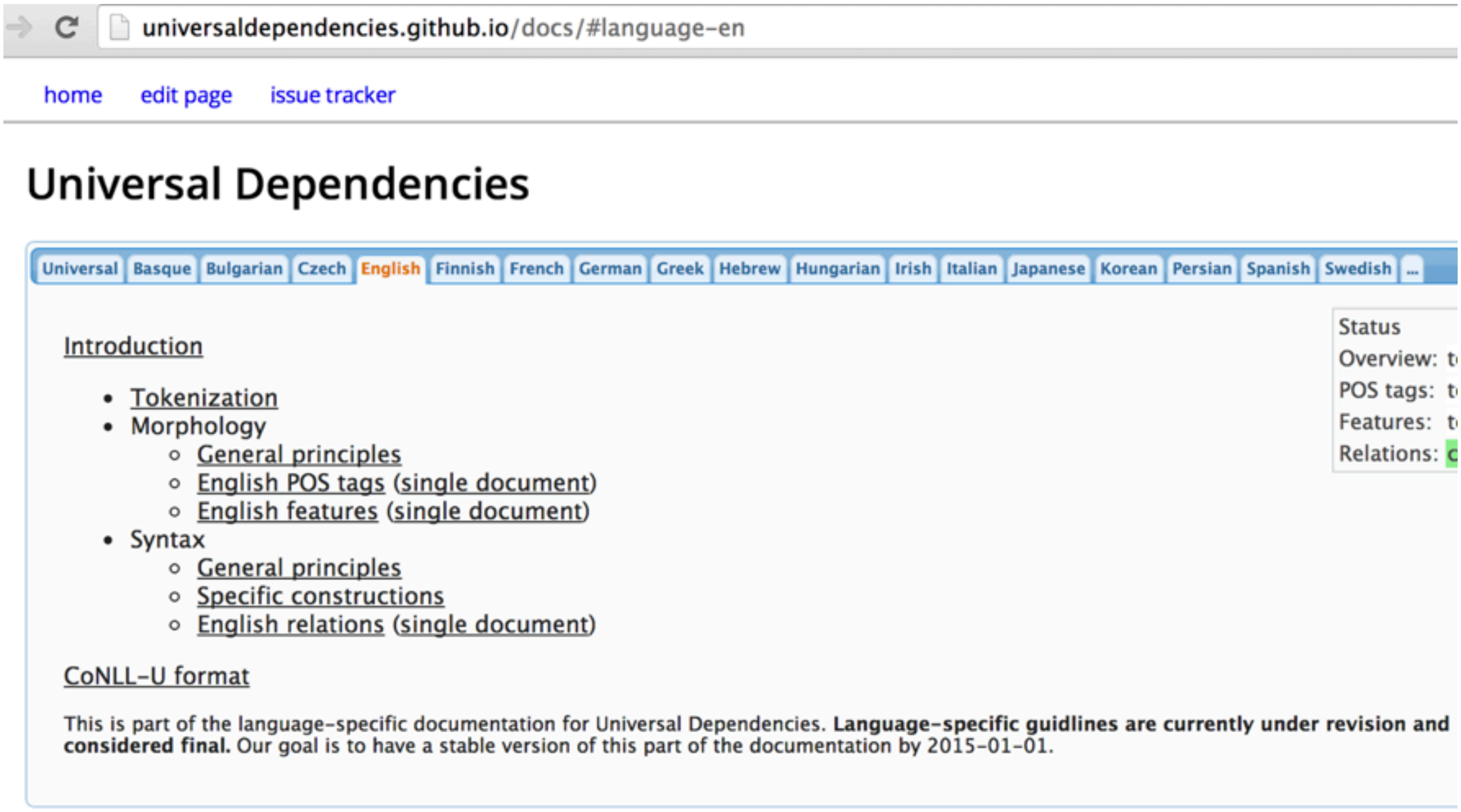
http://www.nytimes.com/2014/11/06/upshot/what-the-forecasts-got-right-and-wrong.html

Competitive evaluation



http://www.nytimes.com/2014/11/06/upshot/what-the-forecasts-got-right-and-wrong.html

Dependencies on their own



universaldependencies.github.io/docs/#language-en

home edit page issue tracker

Universal Dependencies

Universal | Basque | Bulgarian | Czech | English | Finnish | French | German | Greek | Hebrew | Hungarian | Irish | Italian | Japanese | Korean | Persian | Spanish | Swedish | ...

Introduction

- Tokenization
- Morphology
  - General principles
  - English POS tags (single document)
  - English features (single document)
- Syntax
  - General principles
  - Specific constructions
  - English relations (single document)

CoNLL-U format

This is part of the language-specific documentation for Universal Dependencies. **Language-specific guidlines are currently under revision and considered final.** Our goal is to have a stable version of this part of the documentation by 2015-01-01.

Status
Overview: t
POS tags: t
Features: t
Relations: c

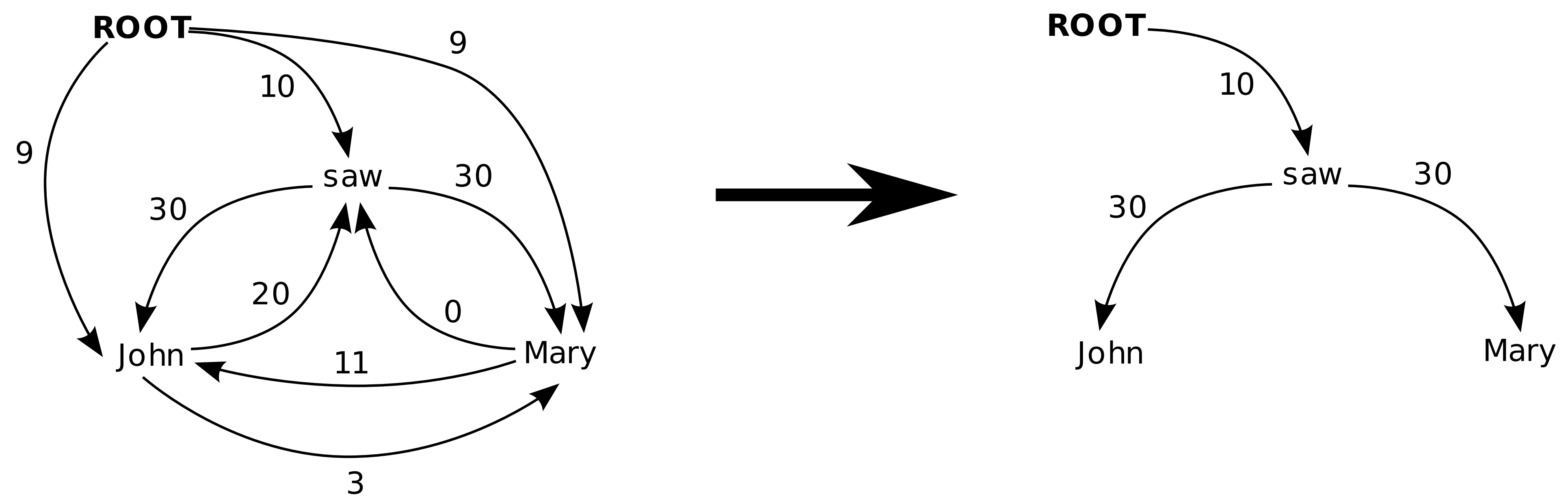
http://universaldependencies.github.io/docs/#language-en

http://nlp.stanford.edu/software/dependencies_manual.pdf

Parsing to dependencies

- One approach: parse to constituents, then convert.
  - Appears to be most accurate method, for English
- Alternative: direct dependency parsing
  - Advantages: training data availability, algorithms sometimes simpler, no need for converter
  - Disadvantages: may lose deeper syntax information encoded in constituency tree
- Methods for dependency parsing
  - Discriminative approaches are most popular
  - Graph-based: predict whole tree.
  - Transition-based (shift-reduce): incrementally predict left-to-right.

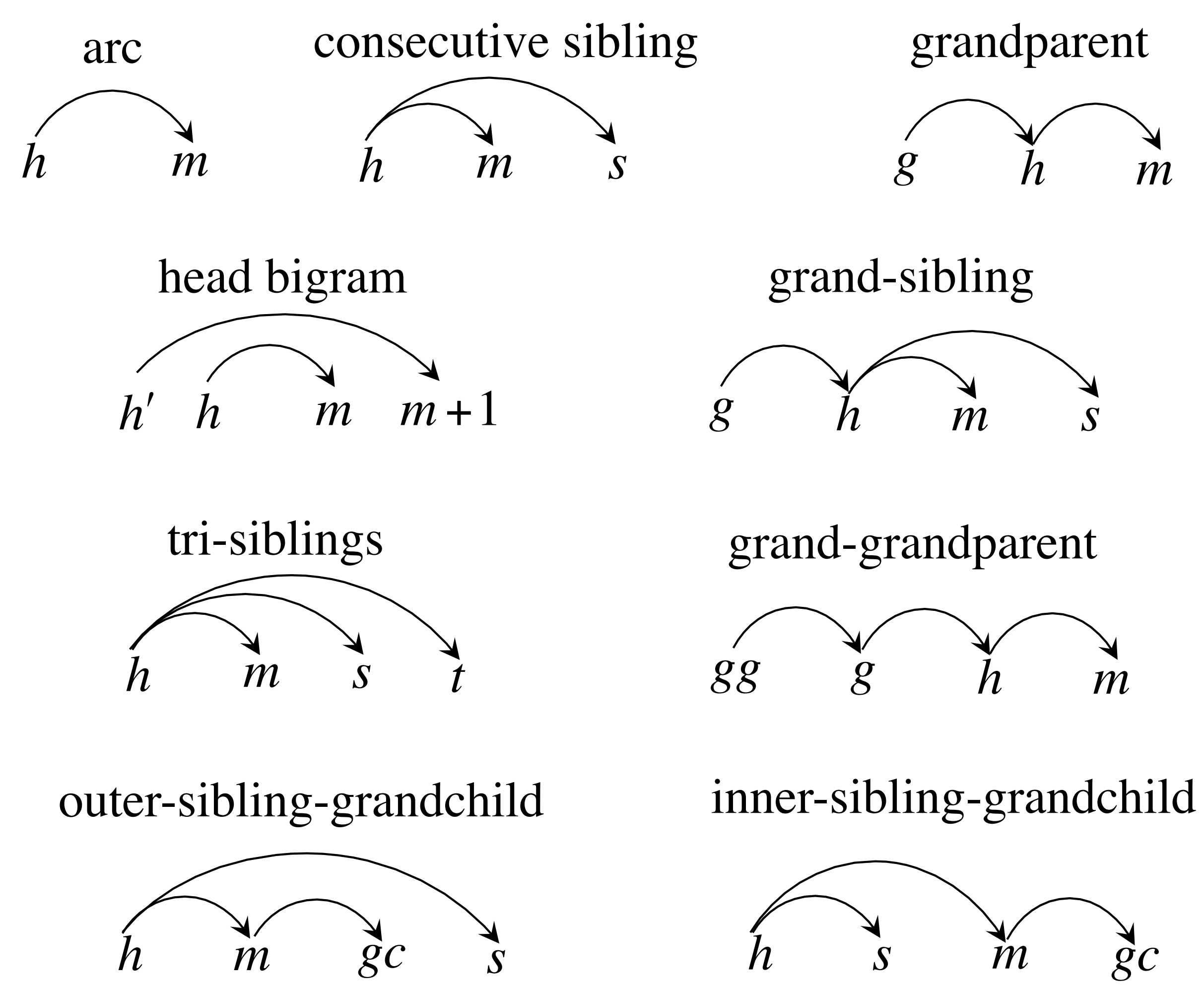
Graph-based parsing



Inference: dynamic programming, minimum spanning trees...
Learning: structured perceptron (or similar)

Graph-based parsing

Current research: how to use higher order features
Decoding is more difficult



<u>Inference</u>: dynamic programming, minimum spanning trees...
<u>Learning</u>: structured perceptron (or similar)

Arc-Eager Transition System [Nivre 2003]

Configuration: (S, B, A) $[S = \text{Stack}, B = \text{Buffer}, A = \text{Arcs}]$

Initial: $([\ ], [0, 1, \ldots, n], \{\ \})$

Terminal: $(S, [\ ], A)$

Shift:	$(S, i\|B, A)$	$\Rightarrow$	$(S\|i, B, A)$	
Reduce:	$(S\|i, B, A)$	$\Rightarrow$	(S, B, A)	$h(i, A)$
Right-Arc(k):	$(S\|i, j\|B, A)$	$\Rightarrow$	$(S\|i\|j, B, A \cup \{(i, j, k)\})$	
Left-Arc(k):	$(S\|i, j\|B, A)$	$\Rightarrow$	$(S, j\|B, A \cup \{(j, i, k)\})$	$\neg h(i, A) \wedge i \neq 0$

Notation: $S|i$ = stack with top i and remainder S
$j|B$ = buffer with head j and remainder B
$h(i, A)$ = i has a head in A

Example Transition Sequence

$[\text{ROOT}]_S \quad [\text{Economic, news, had, little, effect, on, financial, markets, .}]_B$

ROOT	Economic	news	had	little	effect	on	financial	markets	.
	adj	noun	verb	adj	noun	prep	adj	noun	.

Example Transition Sequence

$[\text{ROOT, Economic}]_S$ $[\text{news, had, little, effect, on, financial, markets, .}]_B$

ROOT	Economic	news	had	little	effect	on	financial	markets	.
	adj	noun	verb	adj	noun	prep	adj	noun	.

Example Transition Sequence

$[\text{ROOT}]_S$ $[\text{news, had, little, effect, on, financial, markets, .}]_B$

amod

ROOT	Economic	news	had	little	effect	on	financial	markets	.
	adj	noun	verb	adj	noun	prep	adj	noun	.

Example Transition Sequence

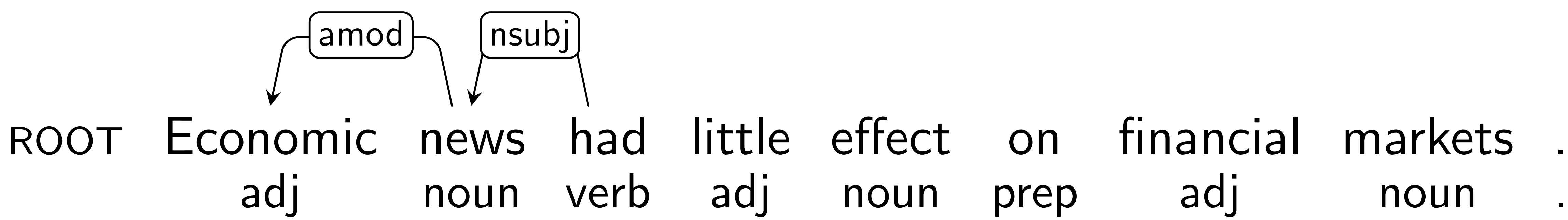
$[\text{ROOT, news}]_S$ $[\text{had, little, effect, on, financial, markets, .}]_B$

amod

ROOT	Economic	news	had	little	effect	on	financial	markets	.
	adj	noun	verb	adj	noun	prep	adj	noun	.

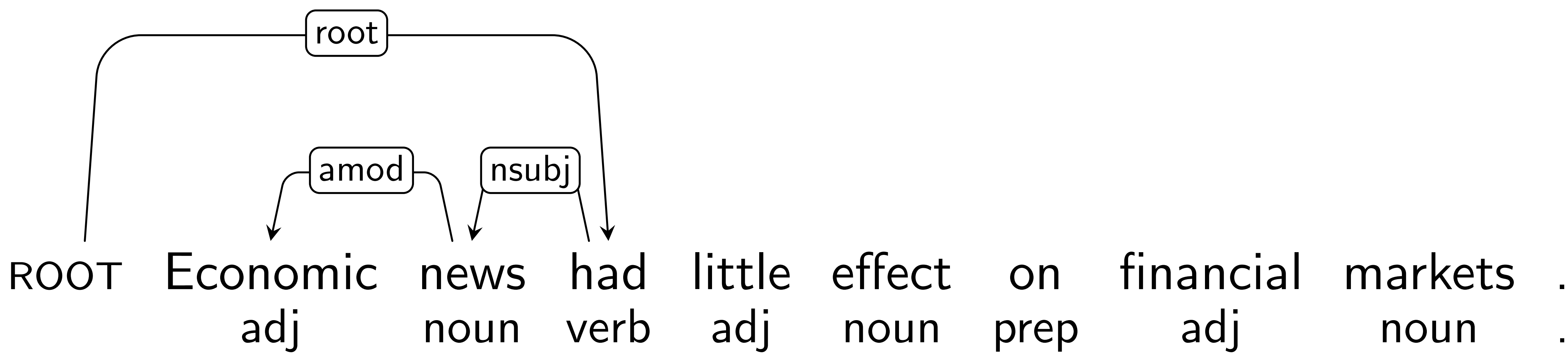
Example Transition Sequence

$[\text{ROOT}]_S$ $[\text{had, little, effect, on, financial, markets, .}]_B$



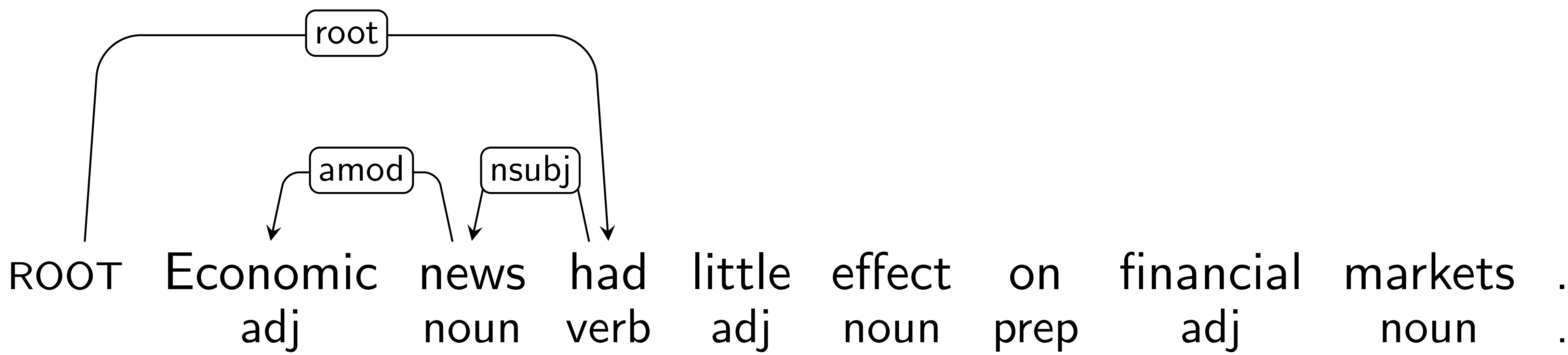
Example Transition Sequence

$[\text{ROOT, had}]_S$ $[\text{little, effect, on, financial, markets, .}]_B$



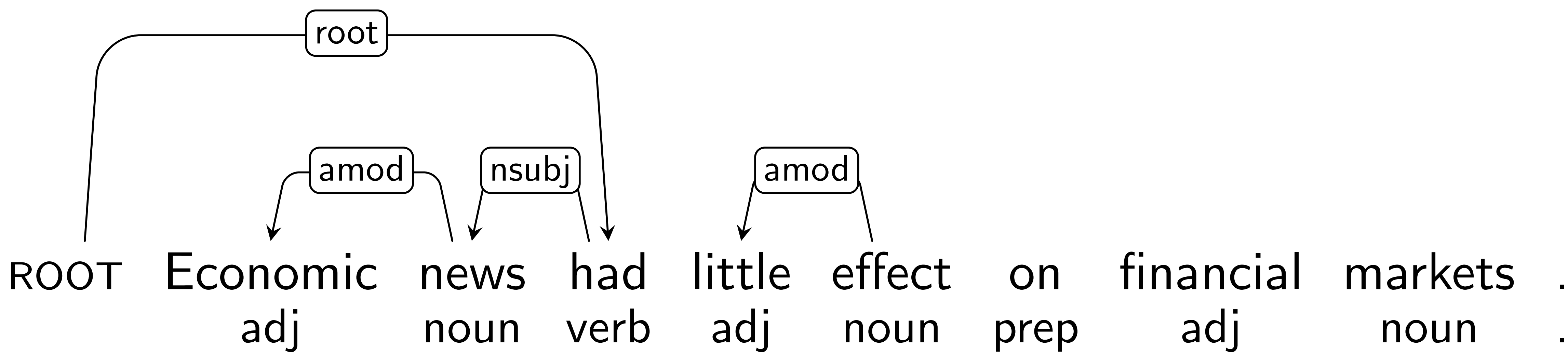
Example Transition Sequence

$[$ROOT, had, little$]_S$ $[$effect, on, financial, markets, .$]_B$



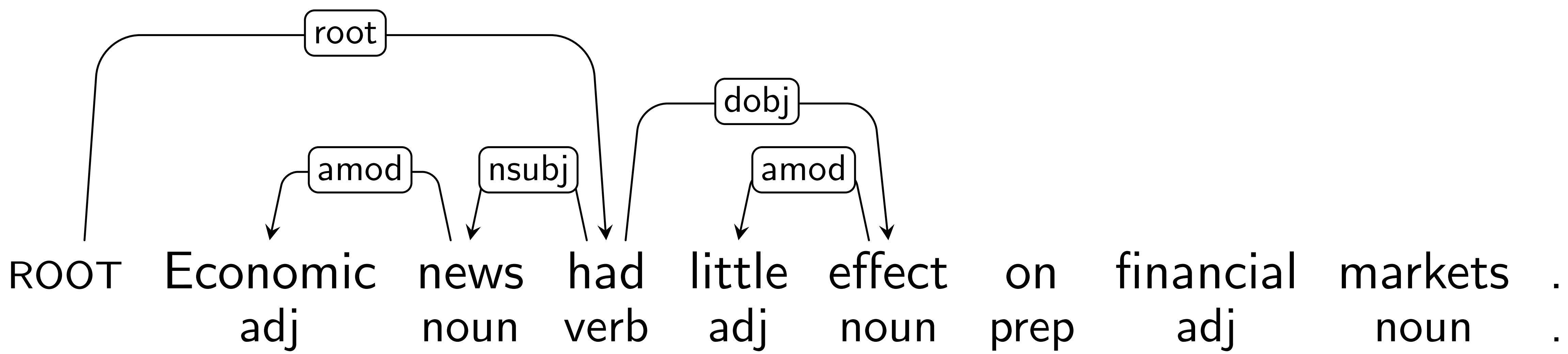
Example Transition Sequence

$[\text{ROOT, had}]_S \quad [\text{effect, on, financial, markets, .}]_B$



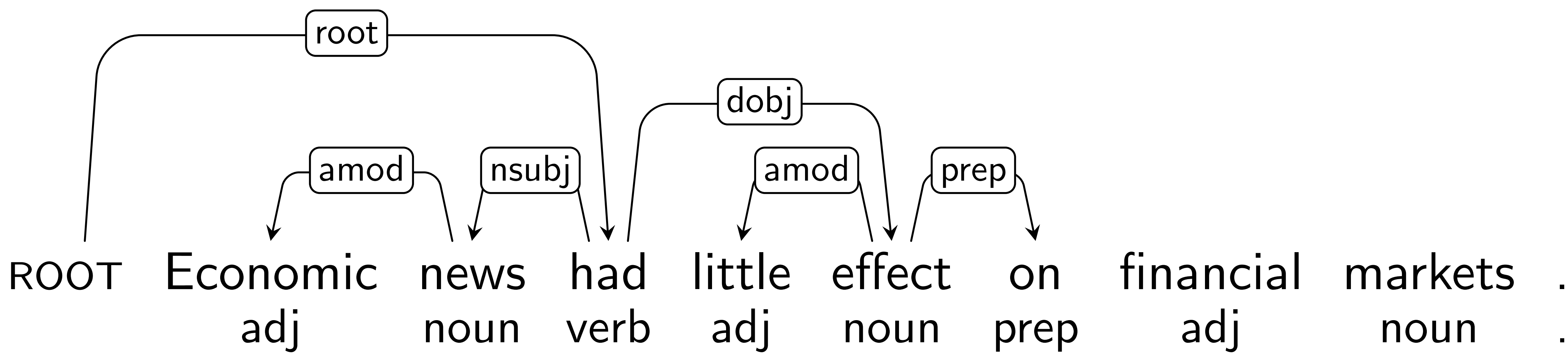
Example Transition Sequence

$[\text{ROOT, had, effect}]_S$ $[\text{on, financial, markets, .}]_B$



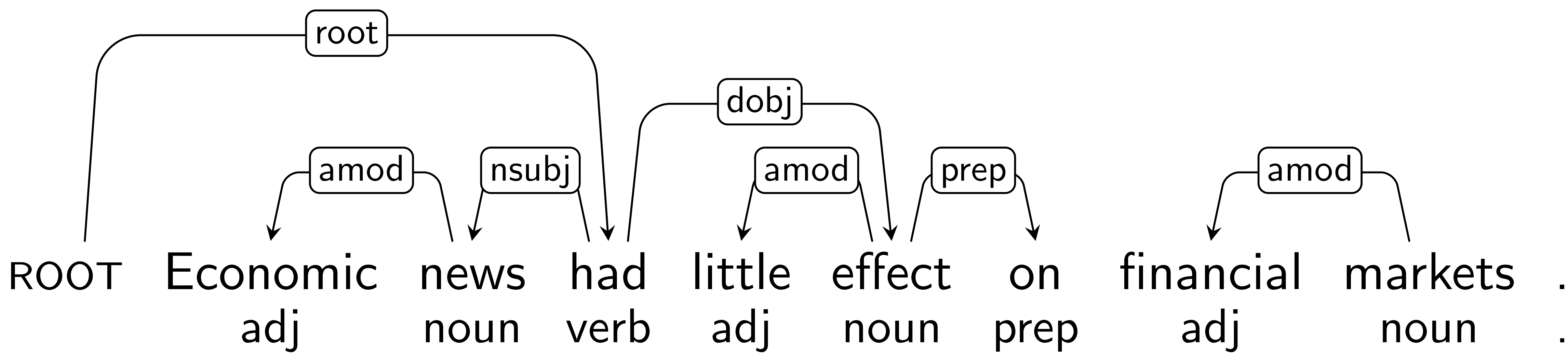
Example Transition Sequence

[ROOT, had, effect, on, financial]$_S$ [markets, .]$_B$



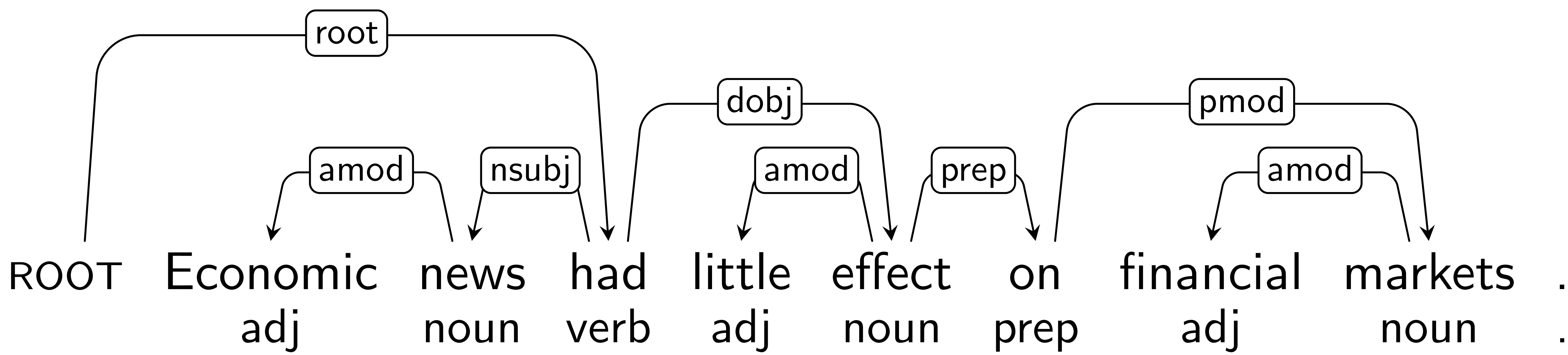
Example Transition Sequence

$[\text{ROOT, had, effect, on}]_S$ $[\text{markets, .}]_B$



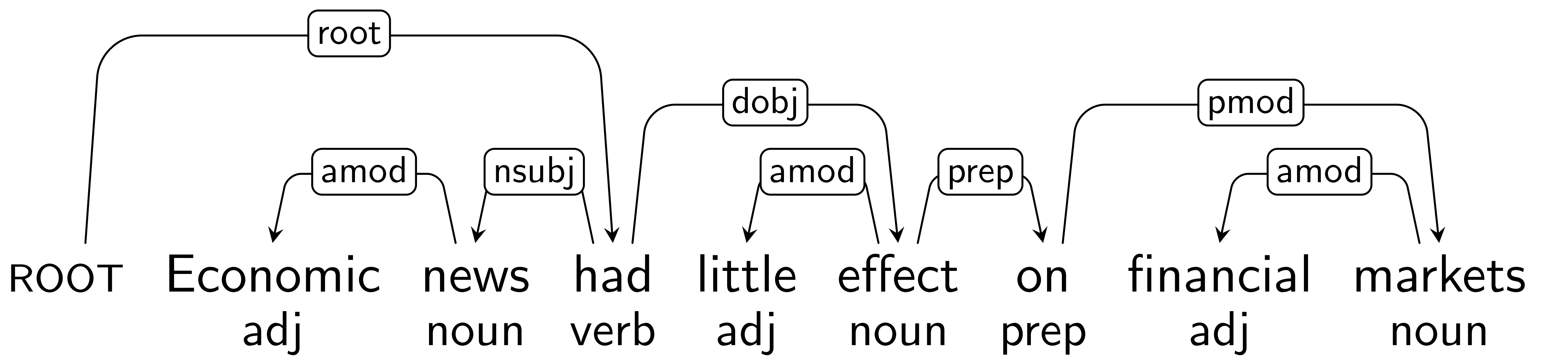
Example Transition Sequence

$[\text{ROOT, had, effect, on, markets}]_S \ \ [.]_B$



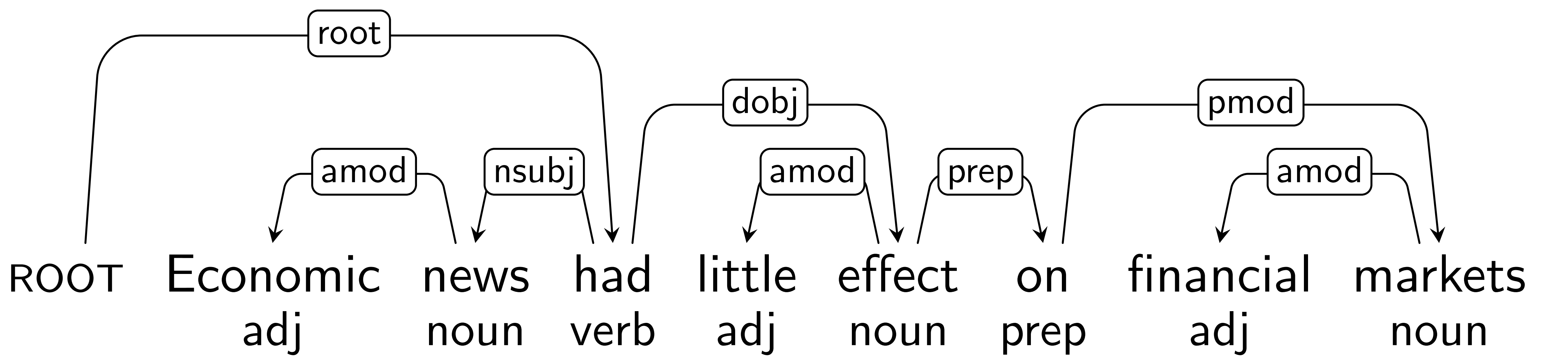
Example Transition Sequence

[ROOT, had, effect, on]$_S$ [.]$_B$



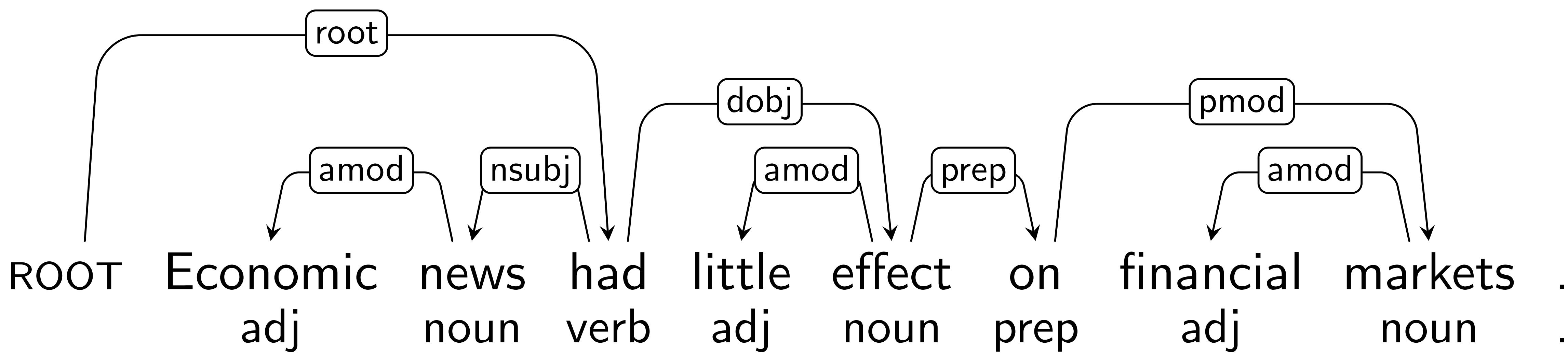
Example Transition Sequence

$[\text{ROOT, had, effect}]_S \quad [.]_B$



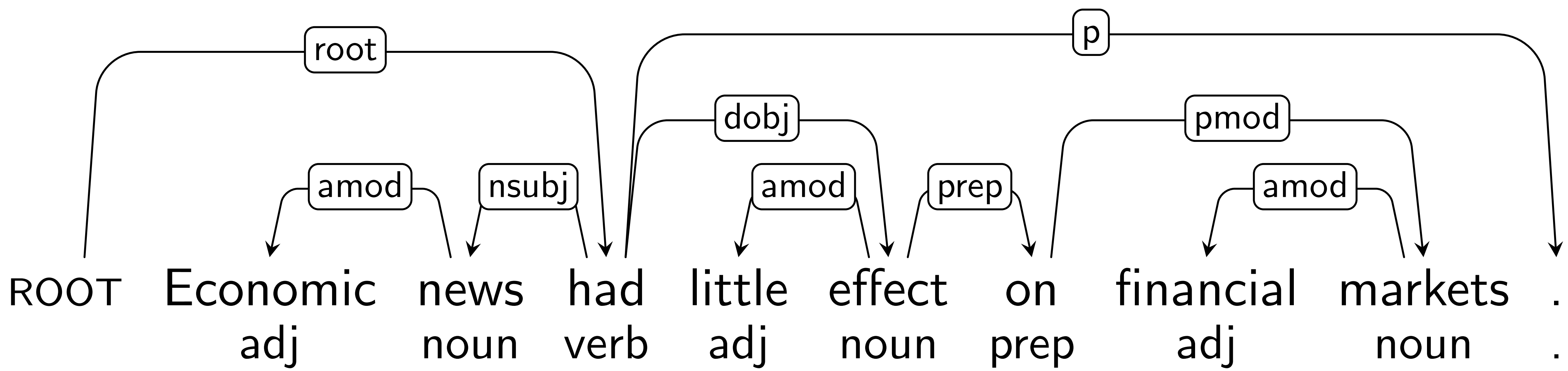
Example Transition Sequence

$[\text{ROOT, had}]_S \quad [.]_B$



Example Transition Sequence

$[\text{ROOT, had, .}]_S \quad [\;]_B$



Greedy Inference

- Given an oracle o that correctly predicts the next transition $o(c)$, parsing is deterministic:

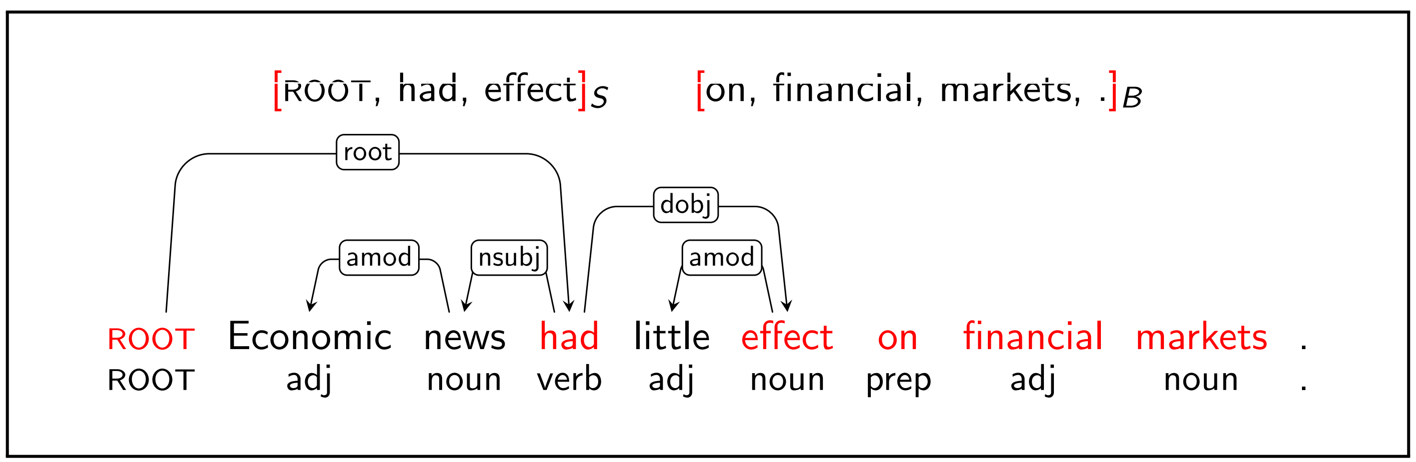
```
Parse(w_1, ..., w_n)
  c ← ([ ]_S, [0, 1, ..., n]_B, { })
  while B_c ≠ [ ]
      t ← o(c)
      c ← t(c)
  return G = ({0, 1, ..., n}, A_c)
```

- Complexity given by upper bound on number of transitions
- Parsing in $O(n)$ time for the arc-eager transition system

Feature Representation

- Features over input tokens relative to S and B

Configuration



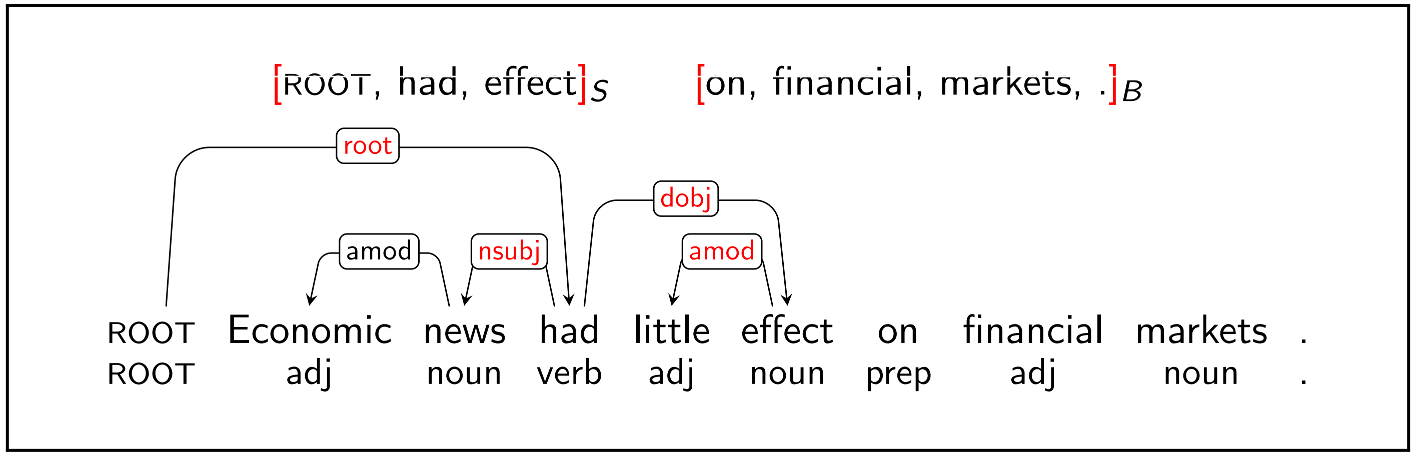
Features

$$
\begin{aligned}
\text{word}(S_2) &= \text{ROOT} \\
\text{word}(S_1) &= \text{had} \\
\text{word}(S_0) &= \text{effect} \\
\text{word}(B_0) &= \text{on} \\
\text{word}(B_1) &= \text{financial} \\
\text{word}(B_2) &= \text{markets}
\end{aligned}
$$

Feature Representation

- Features over input tokens relative to S and B
- Features over the (partial) dependency graph defined by A

Configuration



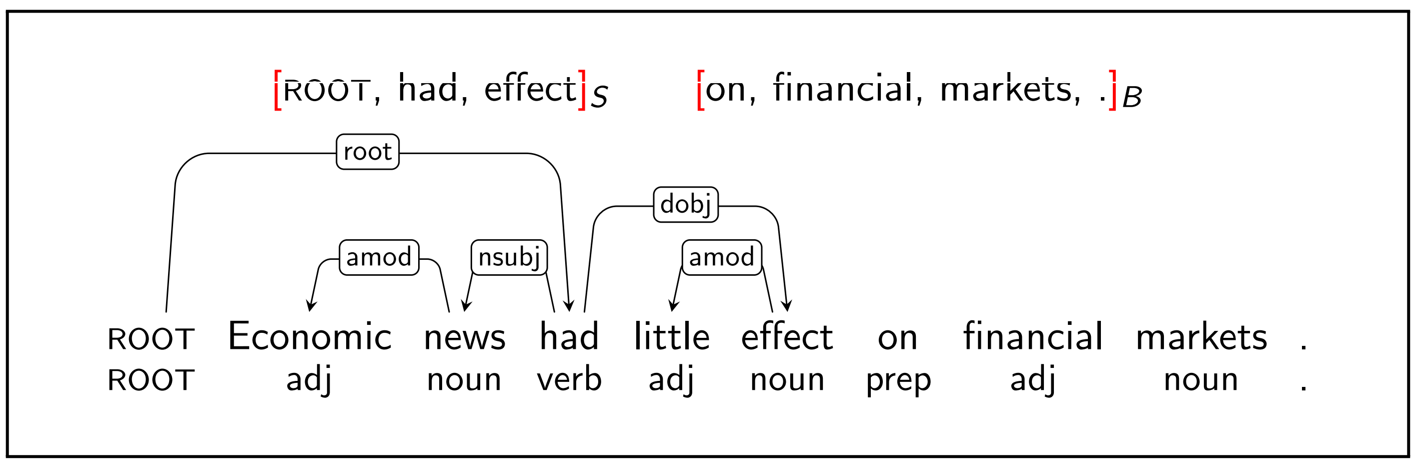
Features

$\text{dep}(S_1)$	$=$	root
$\text{dep}(\text{lc}(S_1))$	$=$	nsubj
$\text{dep}(\text{rc}(S_1))$	$=$	dobj
$\text{dep}(S_0)$	$=$	dobj
$\text{dep}(\text{lc}(S_0)$	$=$	amod
$\text{dep}(\text{rc}(S_0)$	$=$	NIL

Feature Representation

- Features over input tokens relative to S and B
- Features over the (partial) dependency graph defined by A
- Features over the (partial) transition sequence

Configuration



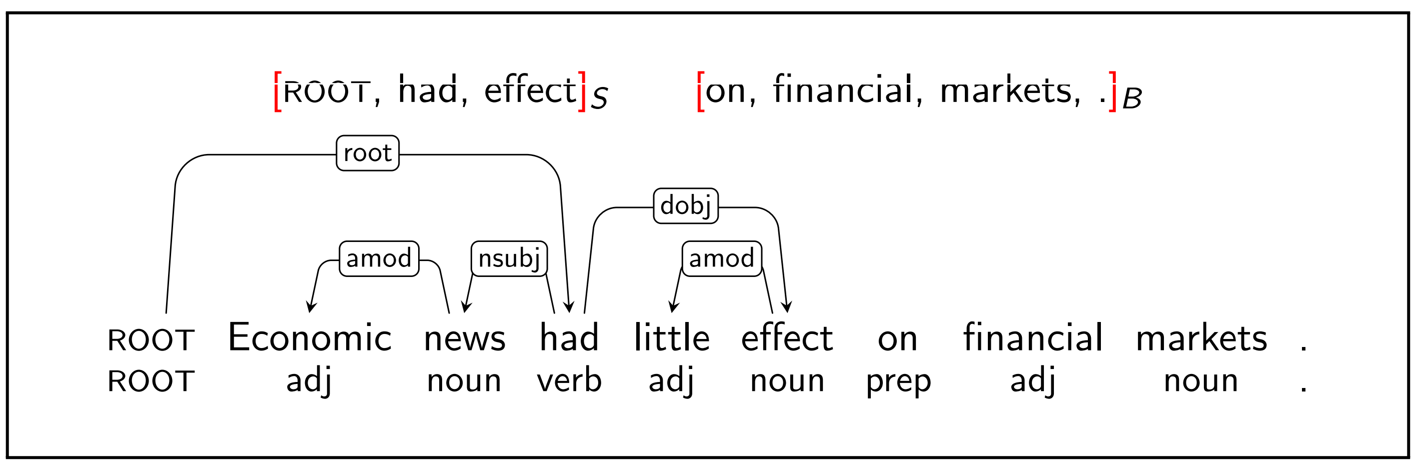
Features

$$
\begin{aligned}
t_{i-1} &= \text{Right-Arc(dobj)} \\
t_{i-2} &= \text{Left-Arc(amod)} \\
t_{i-3} &= \text{Shift} \\
t_{i-4} &= \text{Right-Arc(root)} \\
t_{i-5} &= \text{Left-Arc(nsubj)} \\
t_{i-6} &= \text{Shift}
\end{aligned}
$$

Feature Representation

- Features over input tokens relative to S and B
- Features over the (partial) dependency graph defined by A
- Features over the (partial) transition sequence

Configuration



Features

$$
\begin{aligned}
t_{i-1} &= \text{Right-Arc(dobj)}\\
t_{i-2} &= \text{Left-Arc(amod)}\\
t_{i-3} &= \text{Shift}\\
t_{i-4} &= \text{Right-Arc(root)}\\
t_{i-5} &= \text{Left-Arc(nsubj)}\\
t_{i-6} &= \text{Shift}
\end{aligned}
$$

- Feature representation unconstrained by parsing algorithm

Local Learning

- Given a treebank:
  - Reconstruct oracle transition sequence for each sentence
  - Construct training data set $D = \{(c, t) \mid o(c) = t\}$
  - Maximize accuracy of local predictions $o(c) = t$
- Any (unstructured) classifier will do (SVMs are popular)
- Training is local and restricted to oracle configurations

Greedy, Local, Transition-Based Parsing

- Advantages:
  - Highly efficient parsing – linear time complexity with constant time oracles and transitions
  - Rich history-based feature representations – no rigid constraints from inference algorithm
- Drawback:
  - Sensitive to search errors and error propagation due to greedy inference and local learning
- The major question in transition-based parsing has been how to improve learning and inference, while maintaining high efficiency and rich feature models

Beam Search

- Maintain the k best hypotheses [Johansson and Nugues 2006]:

  Parse$(w_1, \ldots, w_n)$

  Beam $\leftarrow \{([\ ]_S, [0, 1, \ldots, n]_B, \{\ \})\}$

  while $\exists c \in$ Beam $[B_c \neq [\ ]]$

    foreach $c \in$ Beam

      foreach t

        Add($t(c)$, NewBeam)

    Beam $\leftarrow$ Top(k, NewBeam)

  return $G = (\{0, 1, \ldots, n\}, A_{\text{Top}(1,\,\text{Beam})})$

- Note:
  - $\text{Score}(c_0, \ldots, c_m) = \sum_{i=1}^{m} \mathbf{w} \cdot \mathbf{f}(c_{i-1}, t_i)$
  - Simple combination of locally normalized classifier scores
  - Marginal gains in accuracy

From Oracles to Classifiers

- An oracle can be approximated by a (linear) classifier:

$$o(c) = \underset{t}{\operatorname{argmax}}\, \mathbf{w} \cdot \mathbf{f}(c, t)$$

- History-based feature representation $\mathbf{f}(c, t)$
- Weight vector $\mathbf{w}$ learned from treebank data

State of the art

- Unlabeled attachment scores as of 2014
  - Accuracy of choose-the-parent
  - Labeled scores a little lower
- Datasets vary in quality, so take with a grain of salt

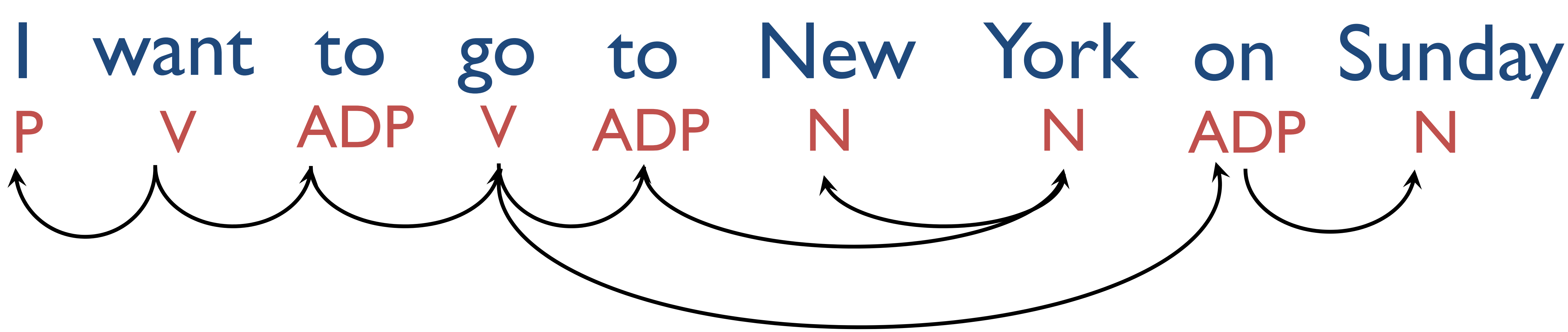
	Best Published
Arabic	81.12 (MS11)
Bulgarian	94.02 (ZH13)
Chinese	92.68 (LX14)
Czech	91.04 (ZL14)
Danish	92.00 (ZH13)
Dutch	86.47 (ZL14)
English	93.22 (MA13)
German	92.41 (MA13)
Japanese	93.74 (LX14)
Portuguese	93.03 (KR10)
Slovene	86.95 (MS11)
Spanish	88.24 (ZL14)
Swedish	91.62 (ZH13)
Turkish	77.55 (KR10)
Average	89.58

Results table from
http://people.csail.mit.edu/regina/my_papers/rand14.pdf

- Transition-based dependency parsers: extremely fast (e.g. MaltParser)

- Does syntax encode interesting semantics?

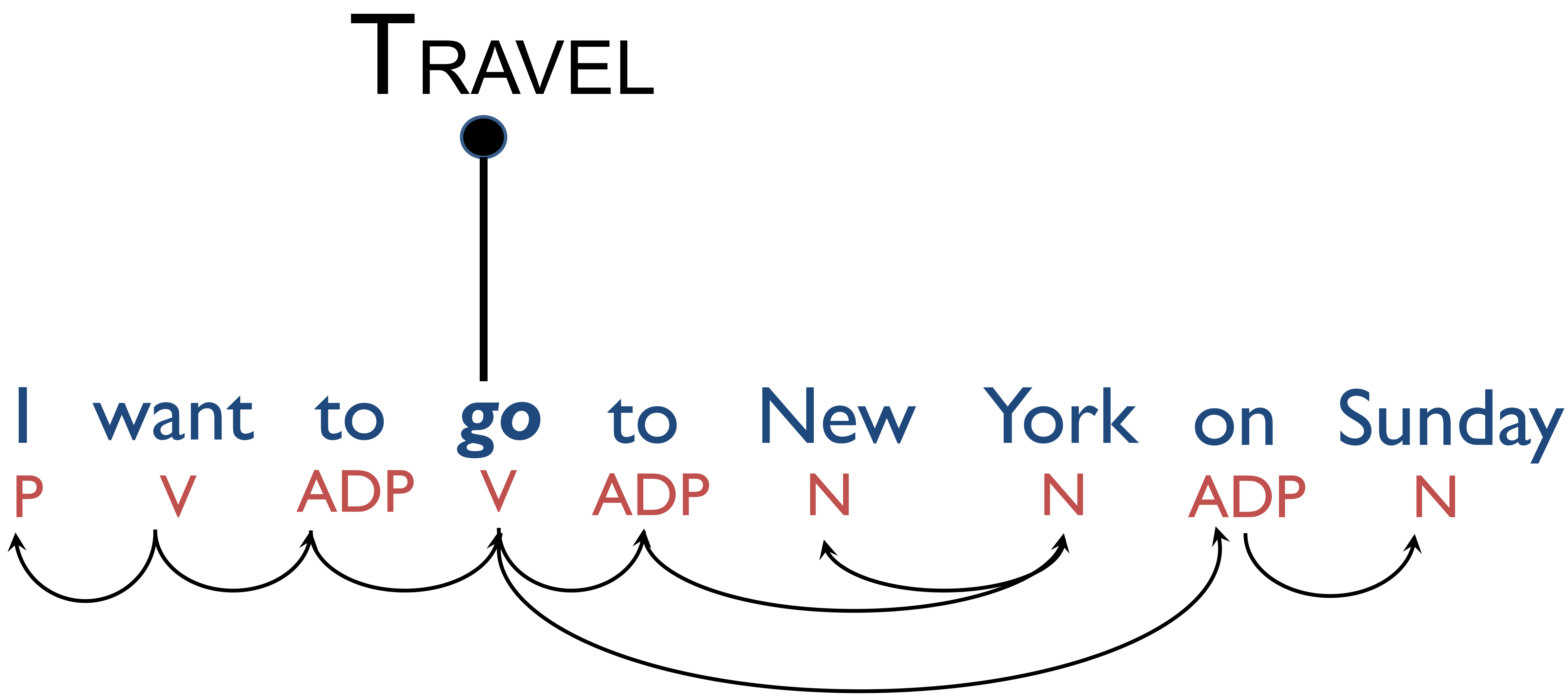
Natural Language Understanding



[Slides: Dipanjan Das]

Natural Language Understanding

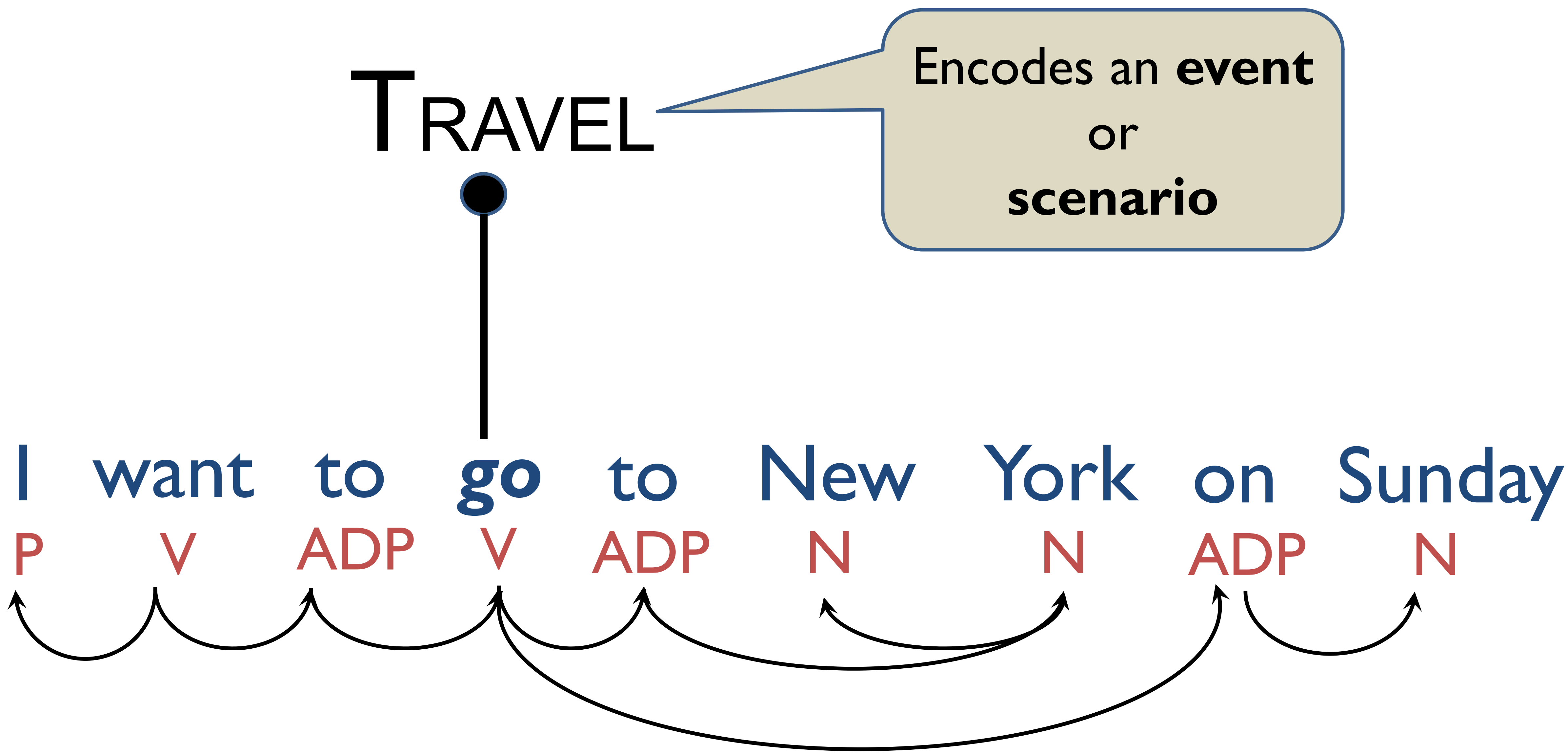
Shallow Semantics: ***Frames*** *and* ***Roles***



[Slides: Dipanjan Das]

Natural Language Understanding

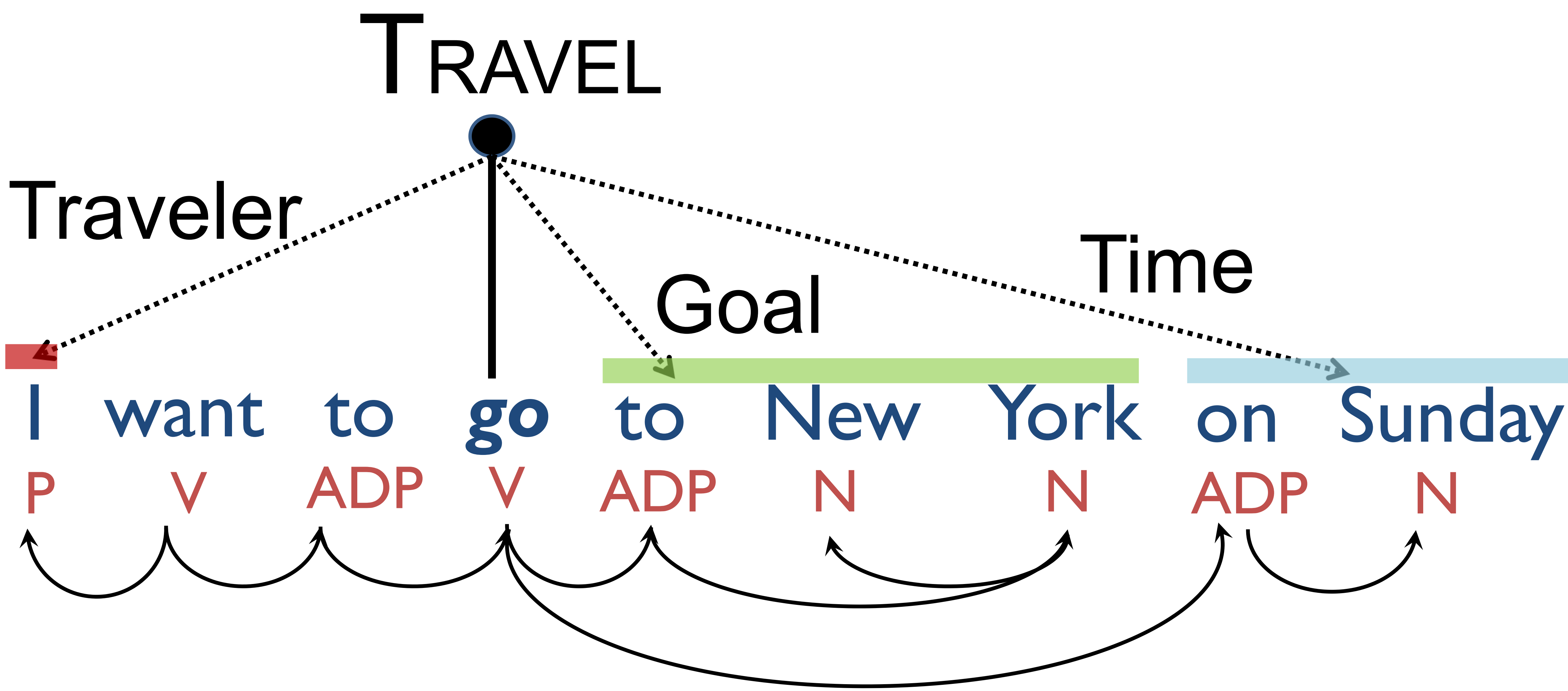
Shallow Semantics: ***Frames*** *and* ***Roles***



[Slides: Dipanjan Das]

Natural Language Understanding

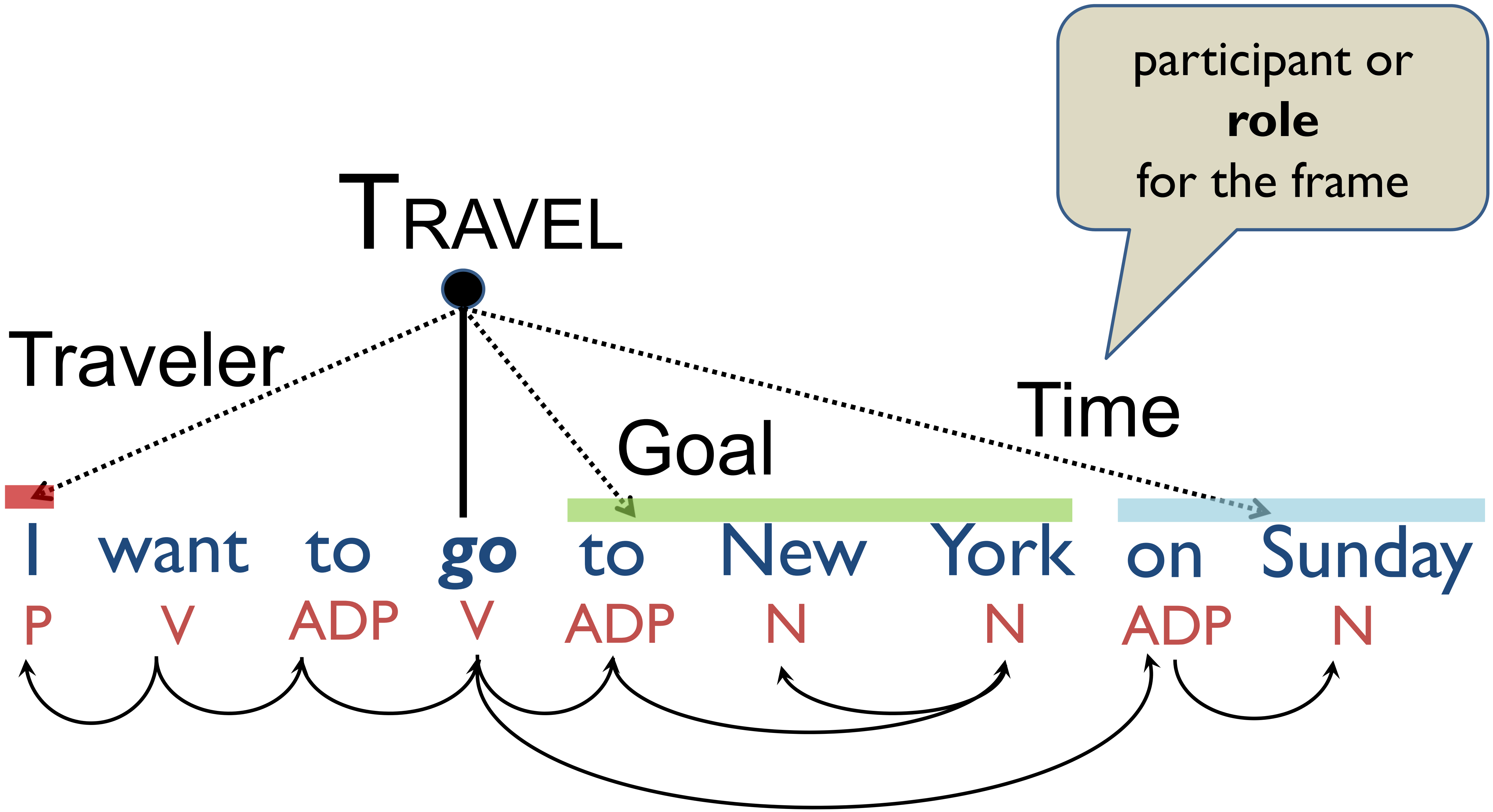
Shallow Semantics: ***Frames*** *and* ***Roles***



[Slides: Dipanjan Das]

Natural Language Understanding

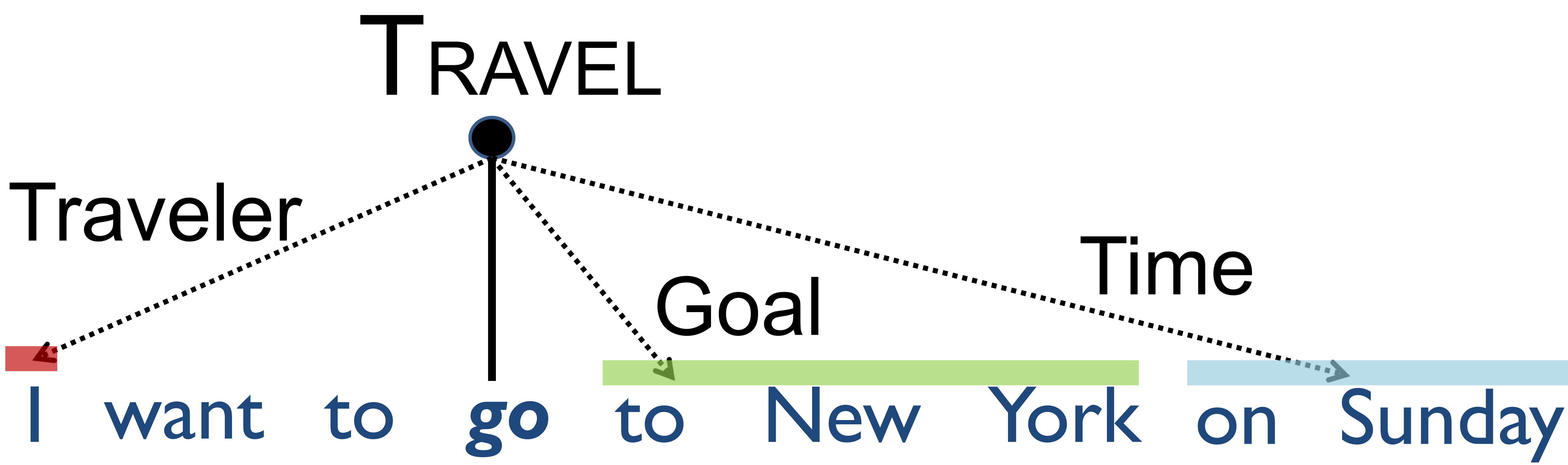
Shallow Semantics: ***Frames** and **Roles***



[Slides: Dipanjan Das]

Natural Language Understanding

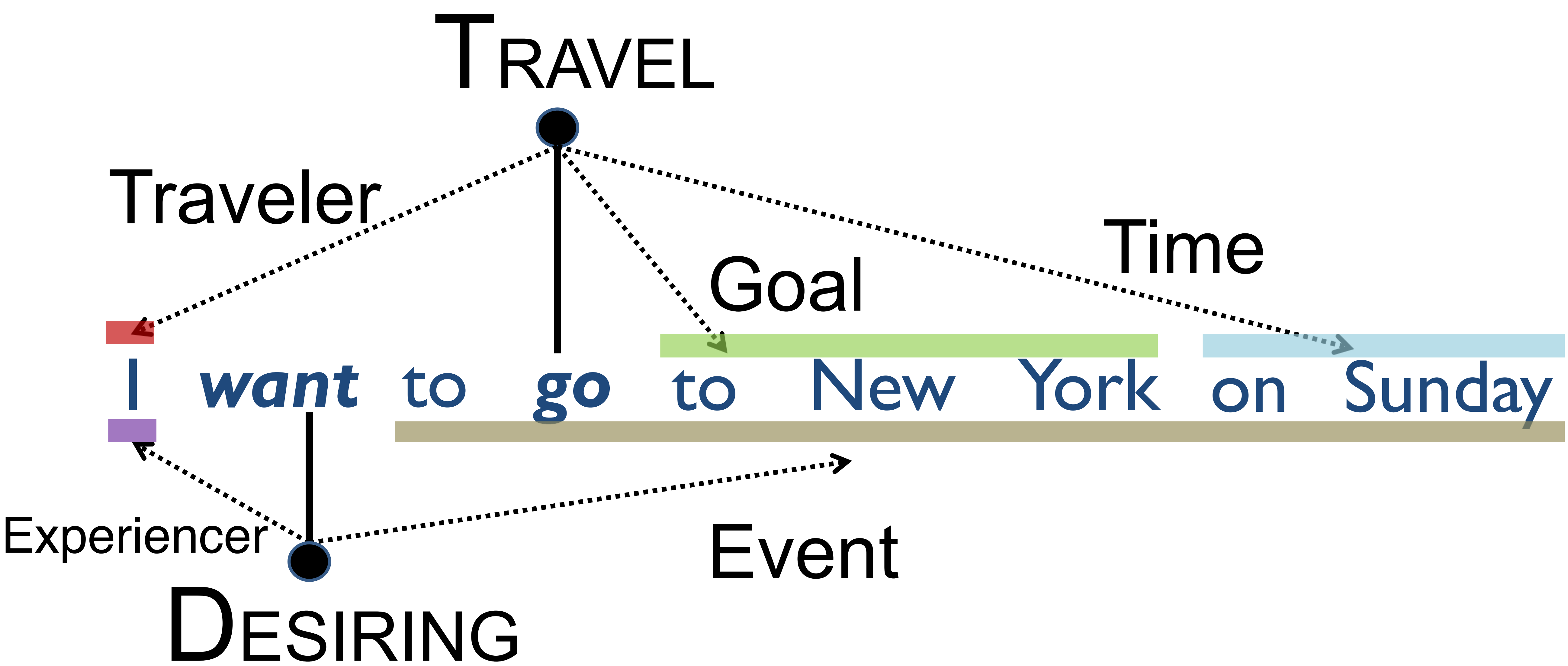
Shallow Semantics: ***Frames*** *and* ***Roles***



[Slides: Dipanjan Das]

Natural Language Understanding

Shallow Semantics: ***Frames*** *and* ***Roles***



[Slides: Dipanjan Das]

Semantics: MRs

- For question-answering, dialogue systems, story understanding, etc… one subproblem: want a relational meaning representation
  - (Why relational?)
- Predicate-Argument structures
  - e.g. V(S, O): verb has noun arguments
  - (~Verb) Actions/Events/Frames, *having*
  - (~Noun) Roles/Slots/Arguments

Example

Text	I saw a person
SVO syntactic structures	see(I, person) [*verb*=see, *subj*=I, *directobj*=person]
Semantic roles	[*event*=see, *agent*=I, *patient*=person]

Example

Text	I saw a person
Feature-structure (frame-style?) representation	[*type*= SeeingEvent *time*= Past *subj*= [*word*= I, *grampers*=1st, *num*= sg] ...]

(High-level syntax like LFG / HPSG?)
(Or is it low-level semantics?)

Example

Text	I believe I saw a person
Frame-style representation	TopCtx => *event*= believe *agent*= I *theme*= BeliefCtx BeliefCtx => *event*= see *agent*= I *patient*= person
	ctx(TopCtx) ctx(BeliefCtx) inctx(TopCtx, event(believe)) inctx(TopCtx, agent(believe, I)) inctx(TopCtx, theme(believe, BeliefCtx)) inctx(BeliefCtx, event(see)) inctx(BeliefCtx, agent(see, I)) inctx(BeliefCtx, patient(see, person))

(Factivity via Davidsonian semantics,
description/modal logic formalism: Bobrow et al 2005)