ESSLLI

Incremental Speech and Language Processing for Interactive Systems

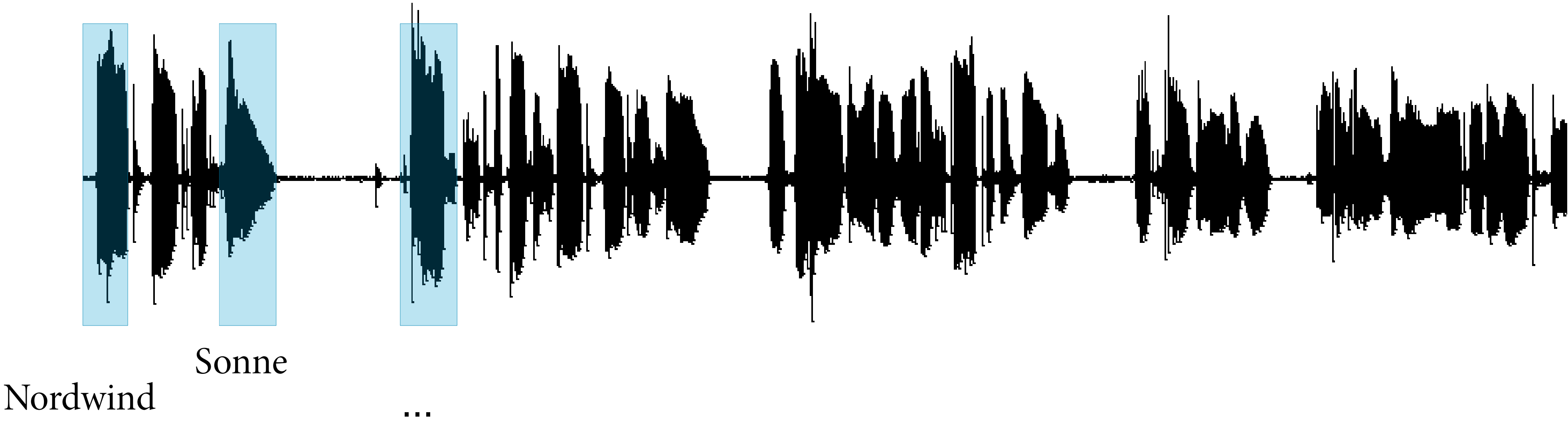
Timo Baumann, Arne Köhn,
Universität Hamburg, Informatics Department
Natural Language Systems Division
{baumann,koehn}@informatik.uni-hamburg.de

Contents of the Course

- Monday:
  - introduction, major features of incremental processing
- Tuesday:
  - incremental processing for sequence problems
- today:
  - incremental processing for structured problems
- Thursday:
  - generating output based on structured and partial input
- Friday:
  - wrap-up and outlook, also based on your questions and interests

Short Recap

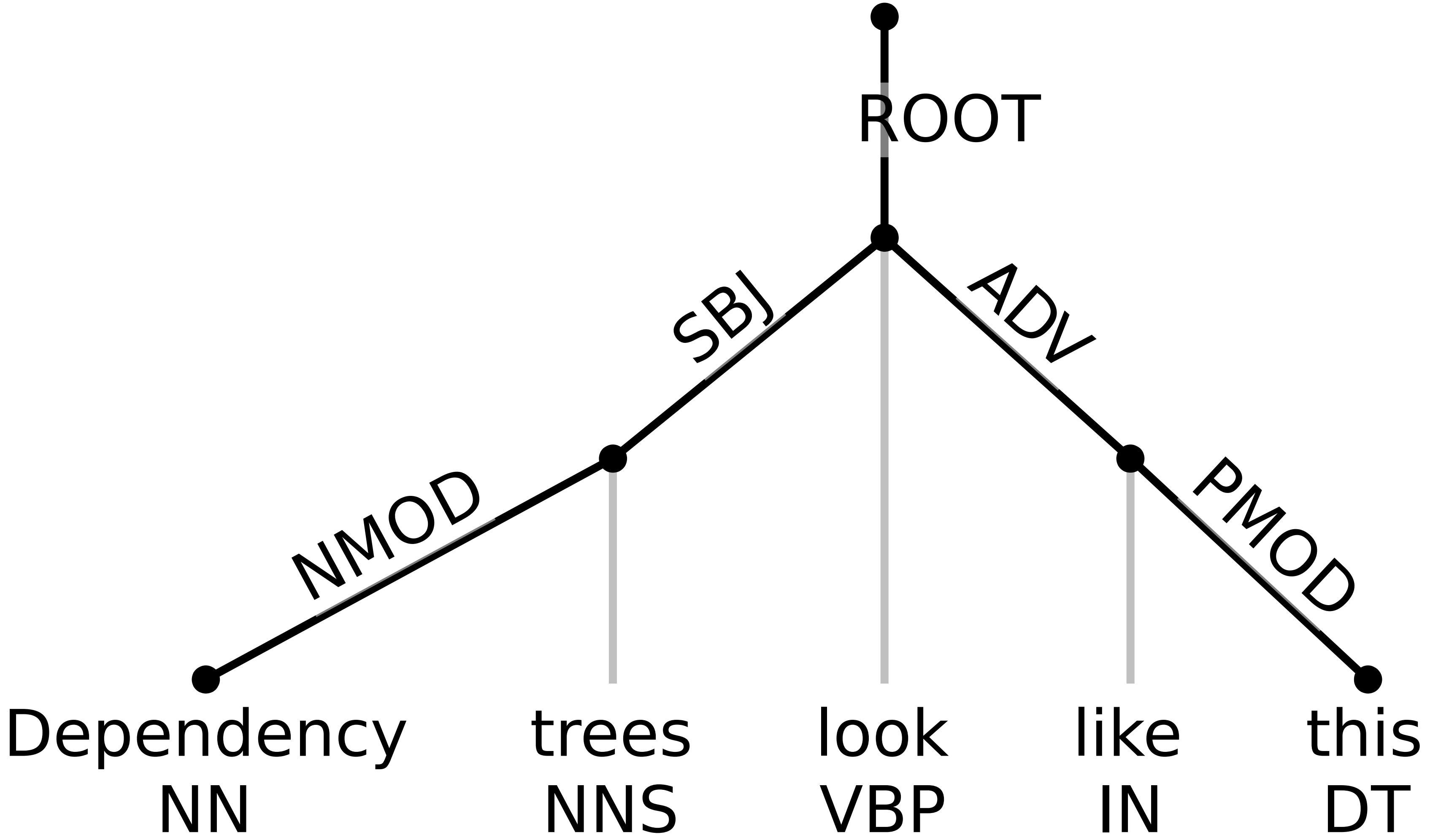
- Incrementalizing sequence problems
- Early decisions affect quality and stability
- Mostly **clear mapping** between input tokens and output tokens
  - Speech recognition: span of audio yields word



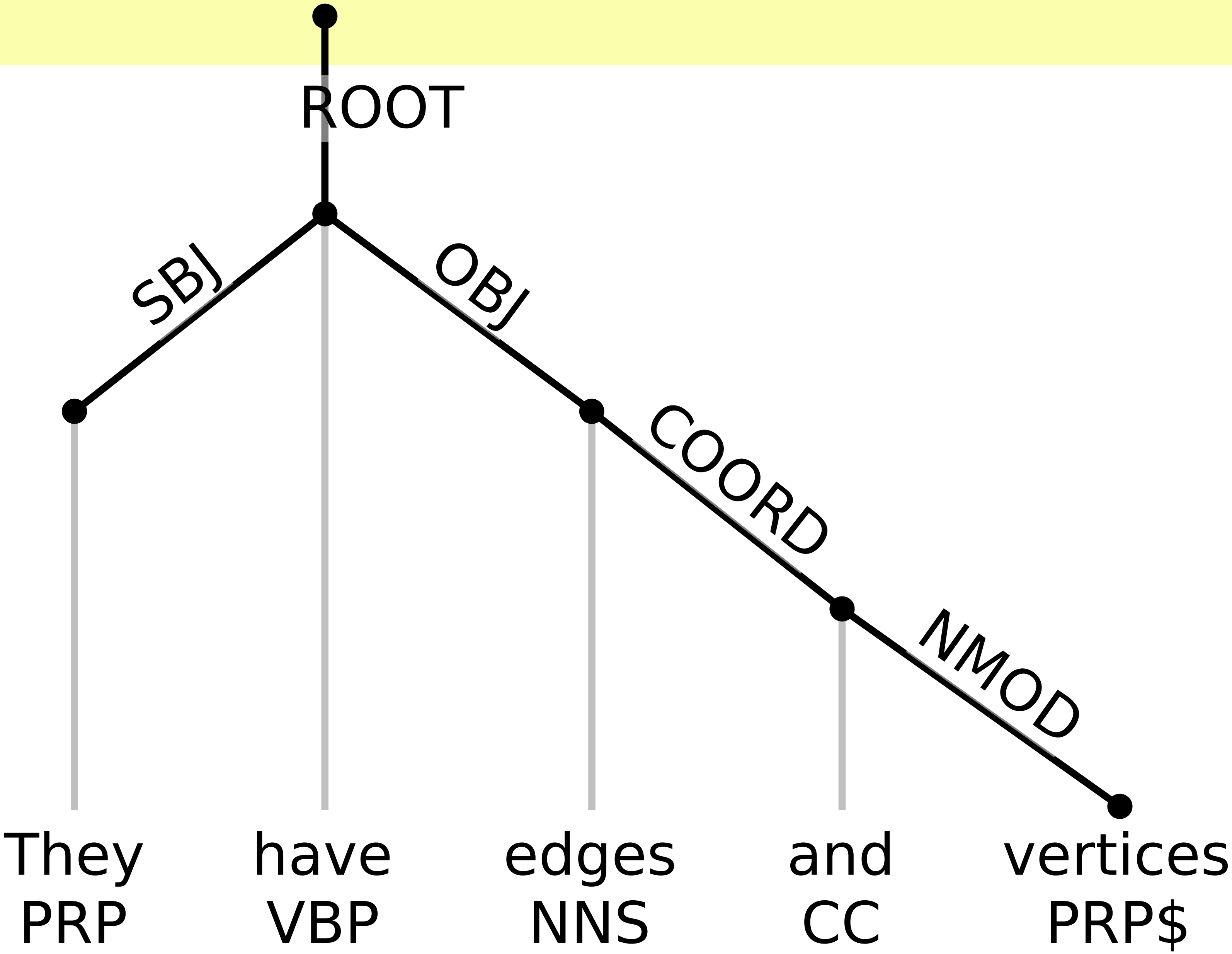
Overview for today

- Real Incremental Parsing
- Parsing with Prediction
- Restart-incremental predictive Parsing
- Creating Incremental Training Data

Dependency Trees

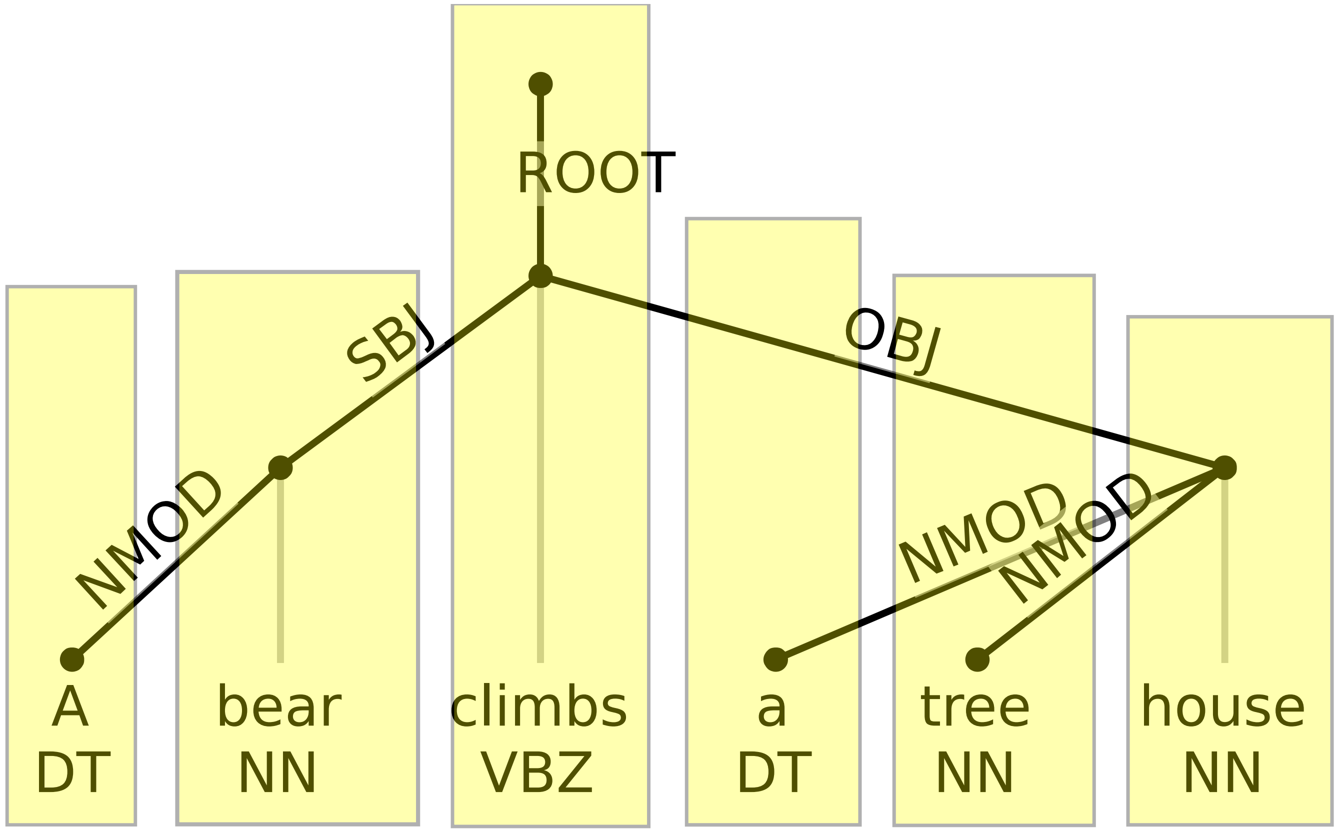


Dependency Trees



Incremental Structured Prediction

- An input sequence is mapped to a structure
- No clear correspondence between input and output



Output Guarantees

- Monotonic output
  - Definition of "monotonic" unclear
  - Severely restricts output (in contrast to sequence problems)
- No guarantees
  - Is the output useful if we don't know whether it lasts?

Stability

Let $a_1 .. a_i .. a_n$ be a series of outputs

- Full monotonicity ($a_i \subseteq a_n$) leads to low accuracy

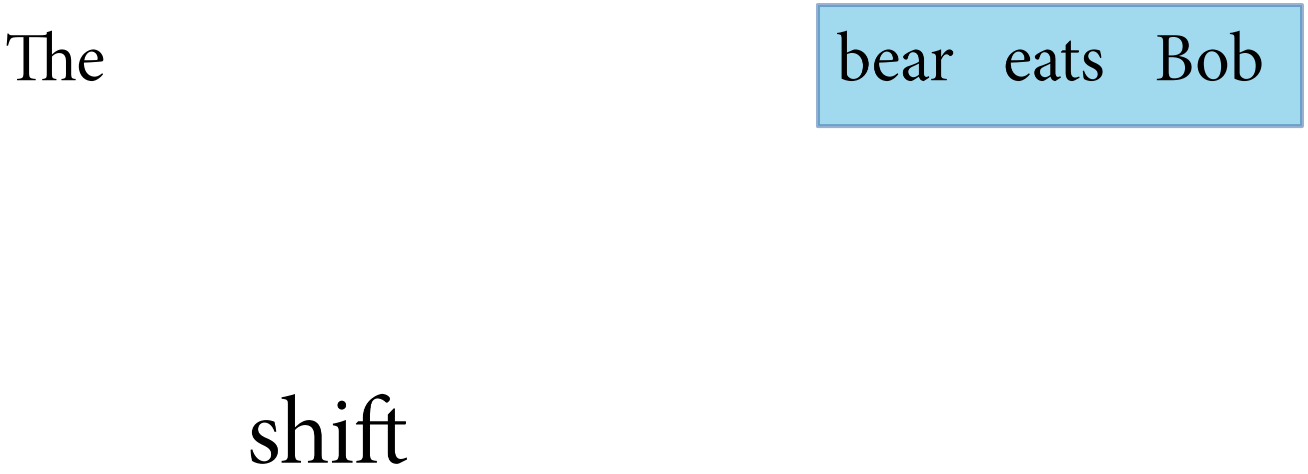
drops to ~50% with prediction (Hassan et al. 2009)

- Instead: Focus on high stability (maximize $a_i \cap a_n$)

Transition-based Parsing

The bear eats Bob

Transition-based Parsing



The

bear eats Bob

shift

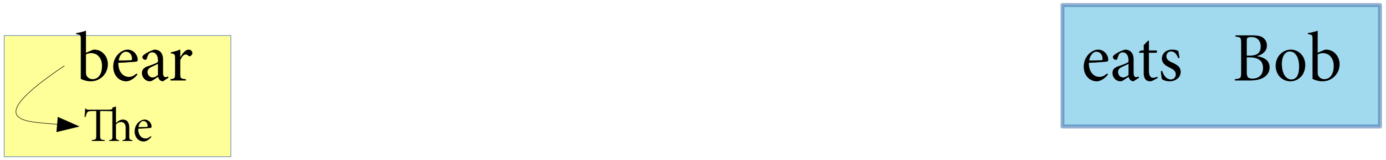
Transition-based Parsing

bear
The

eats Bob

shift

Transition-based Parsing



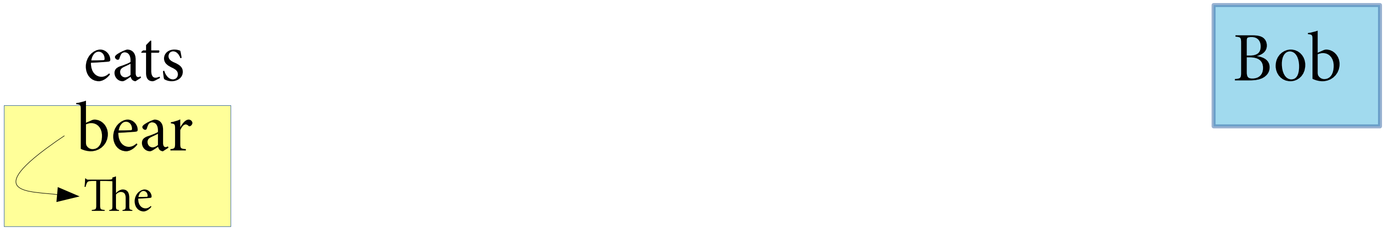
bear

The

eats Bob

Left-arc

Transition-based Parsing



eats

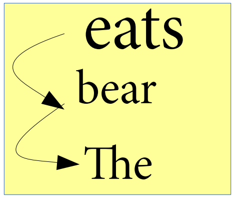
bear

The

Bob

Shift

Transition-based Parsing



eats

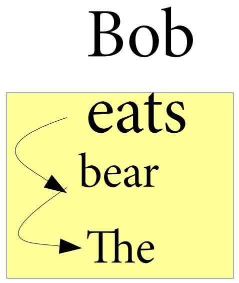
bear

The

Bob

Left-arc

Transition-based Parsing



Bob

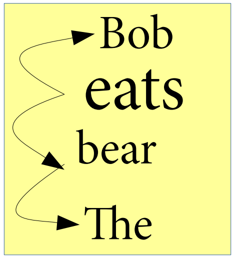
eats

bear

The

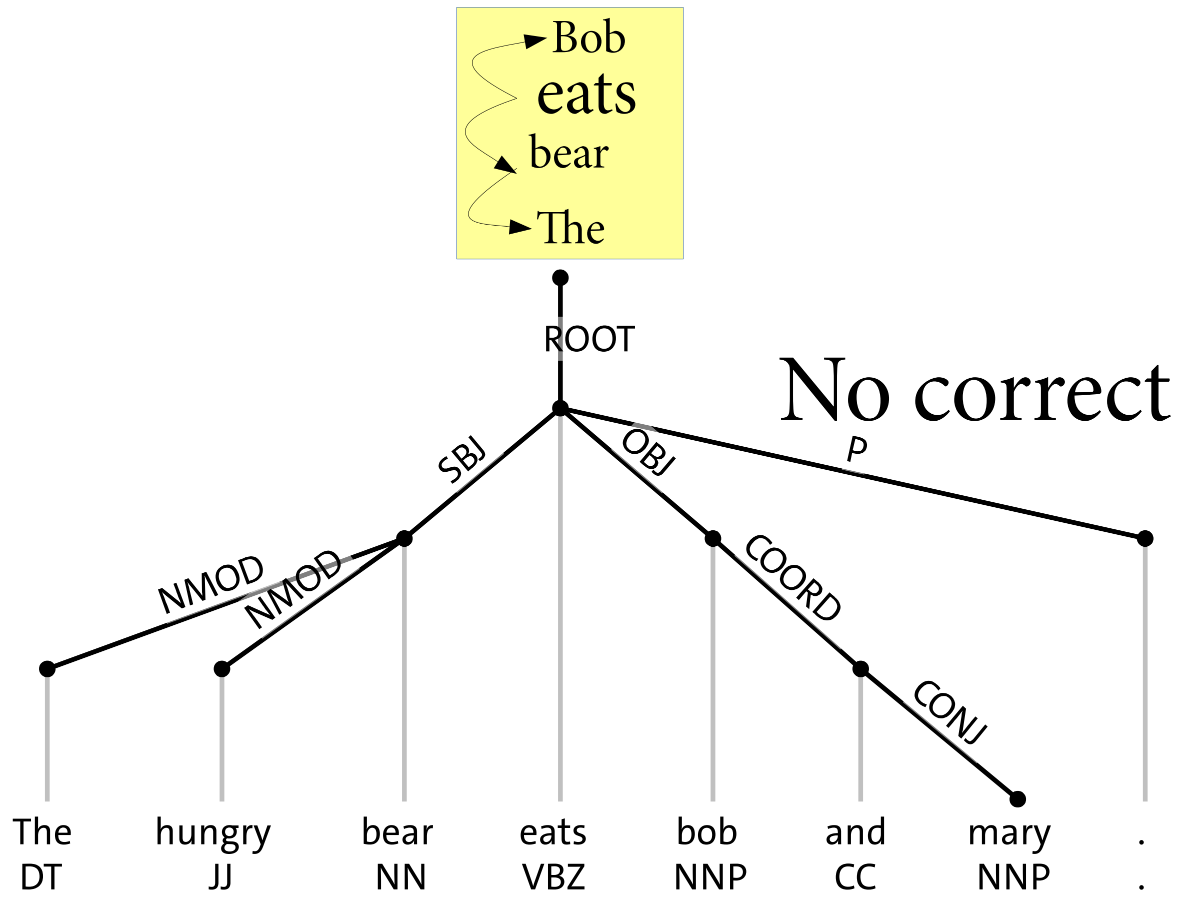
Shift

Transition-based Parsing



Right-arc

Transition-based Parsing



and Mary

No correct transition (?)

Beam Search to the Rescue

- Keep multiple hypotheses in a beam
- Output the most likely one at each step
- Works without lookahead! (Andor et al. 2016)
  - Greedy w/o lookahead: 72% acc.
  - Beam 32 w/o lookahead: 94% acc.

Problem solved?

Bottom-up Parsing

- Most transition systems build trees bottom-up
- How do the sub-trees integrate?
- Once a word is attached, no other can be attached to it
  - Especially problematic for right-branching constructs

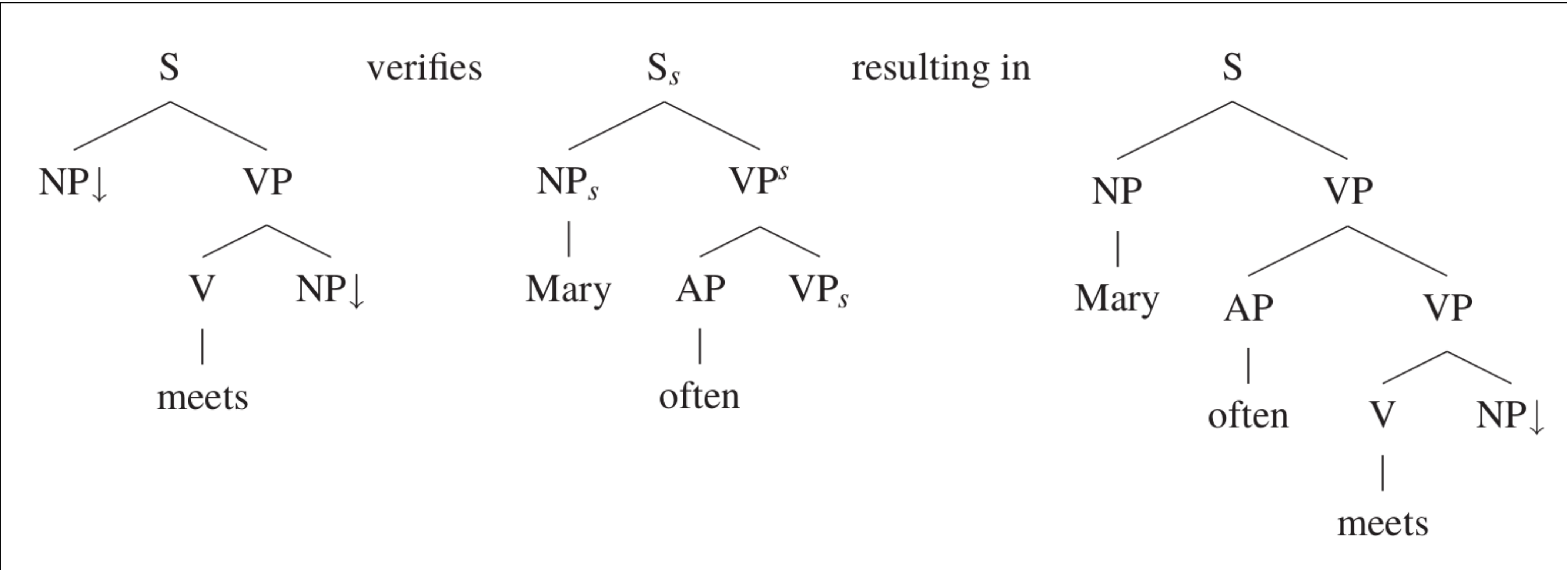
Predictive Parsing

PLTAG

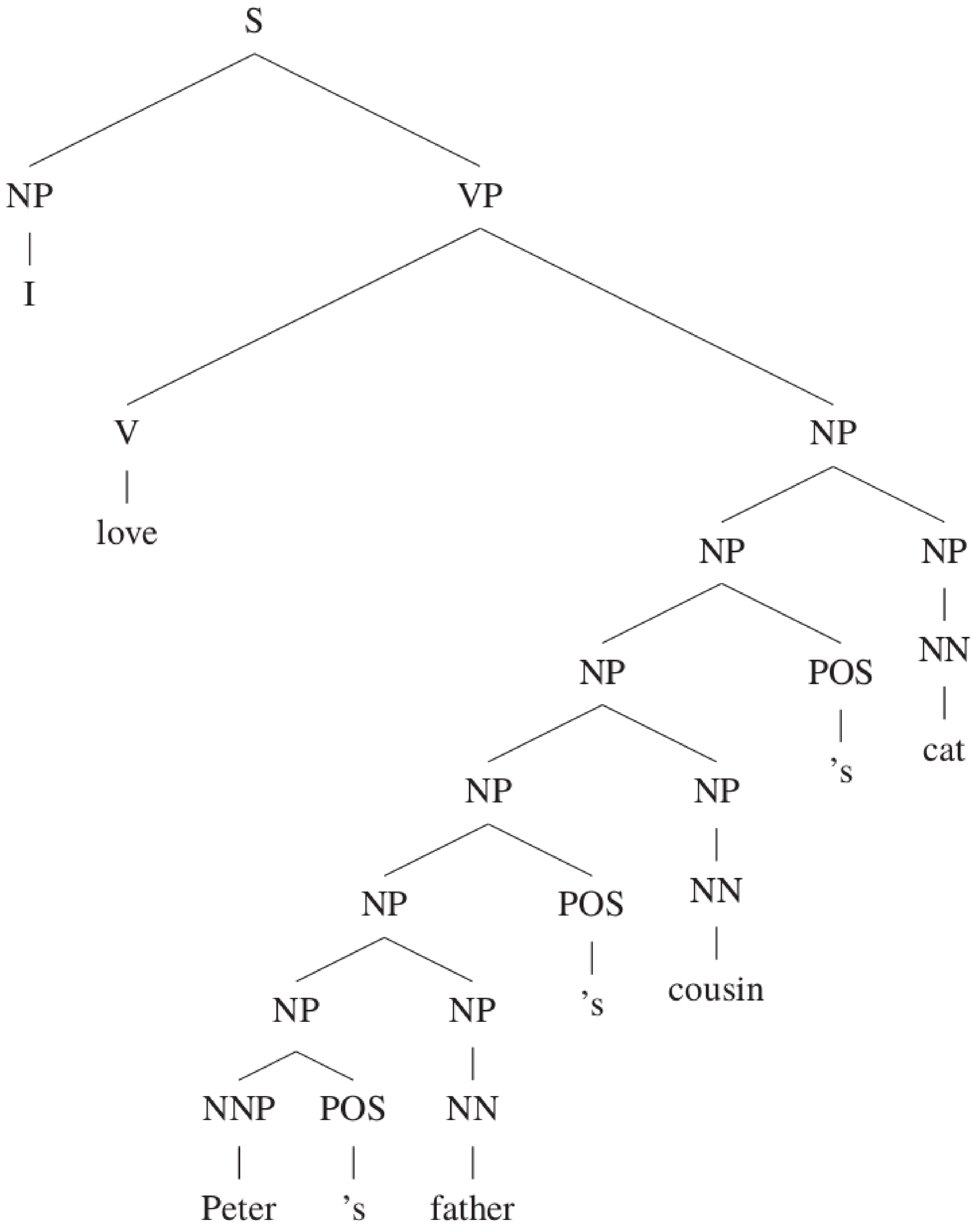
- Predictive Tree Adjoining Grammar
  - Lexicon with tree snippets
  - Operations to combine trees
- Open non-terminals denote upcoming input
- Psycholinguistically motivated
  - Operations correspond to reading times

Demberg (2010)

PLTAG



Demberg (2010)

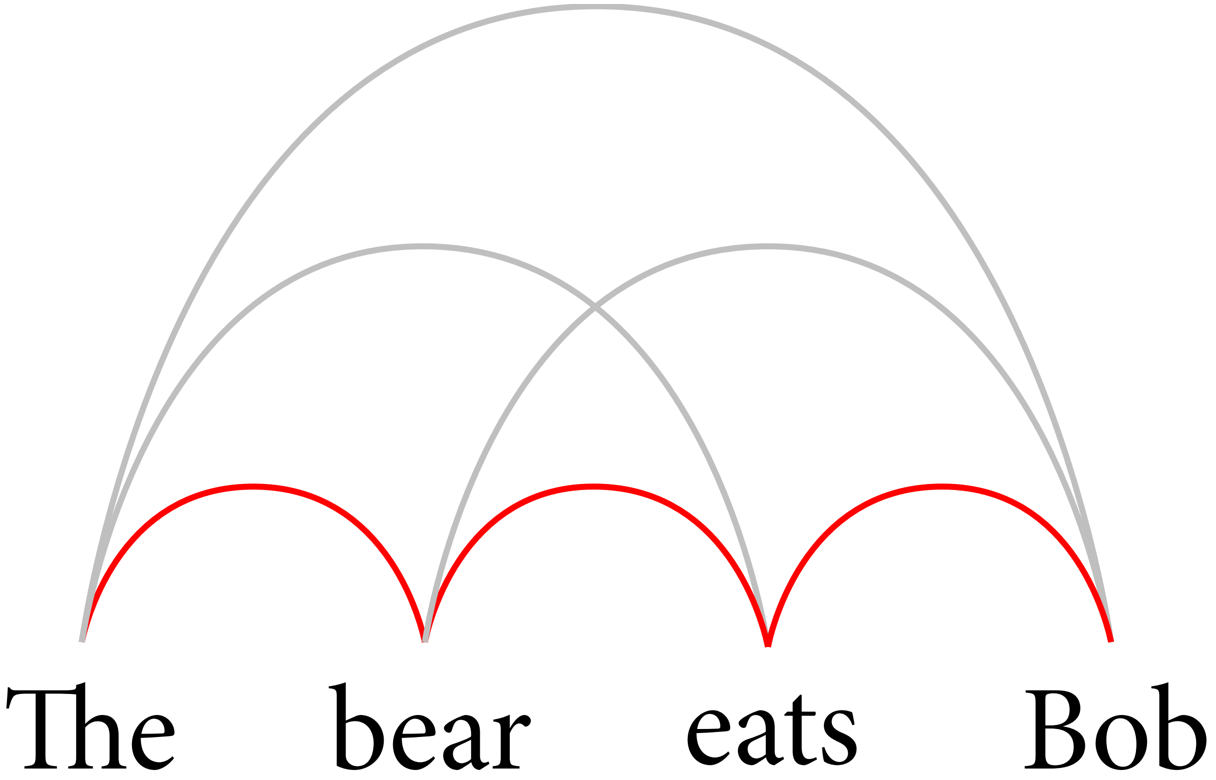

S
NP
VP
I
V
NP
love
NP
NP
NP
POS
NN
NP
NP
's
cat
NP
POS
NN
NP
NP
's
cousin
NNP
POS
NN
Peter
's
father

Limitations of PLTAG

- Monotonic output for each beam
- More prediction → more beams needed
  - Overcommitment
- Left attachments similar problem to right ones in transition-based parsing

Graph-based Parsing

- Nutshell: of all possible edges, select best ones
  - Maximum spanning tree
  - ILP (Constraint optimization)

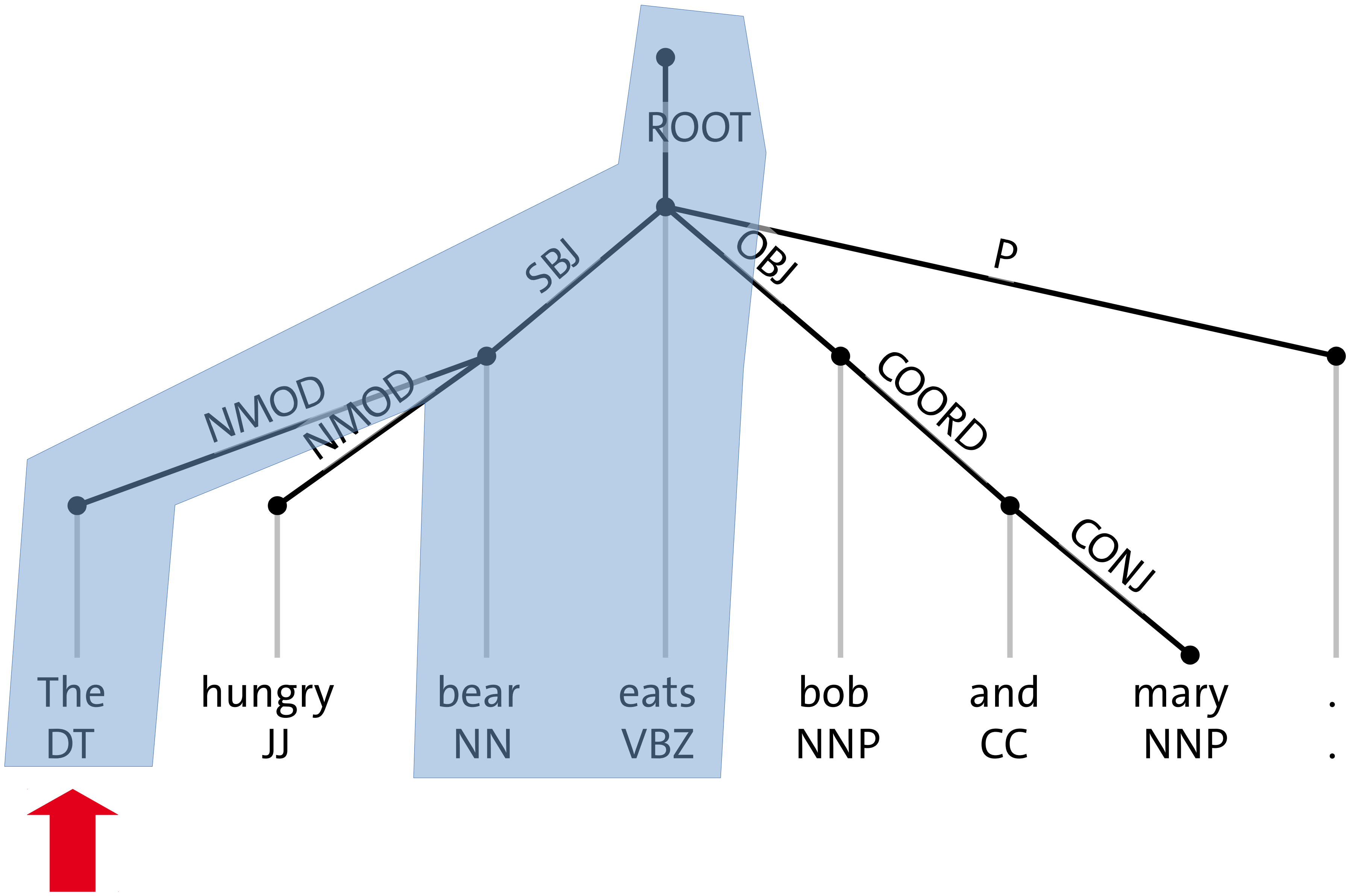


The bear eats Bob

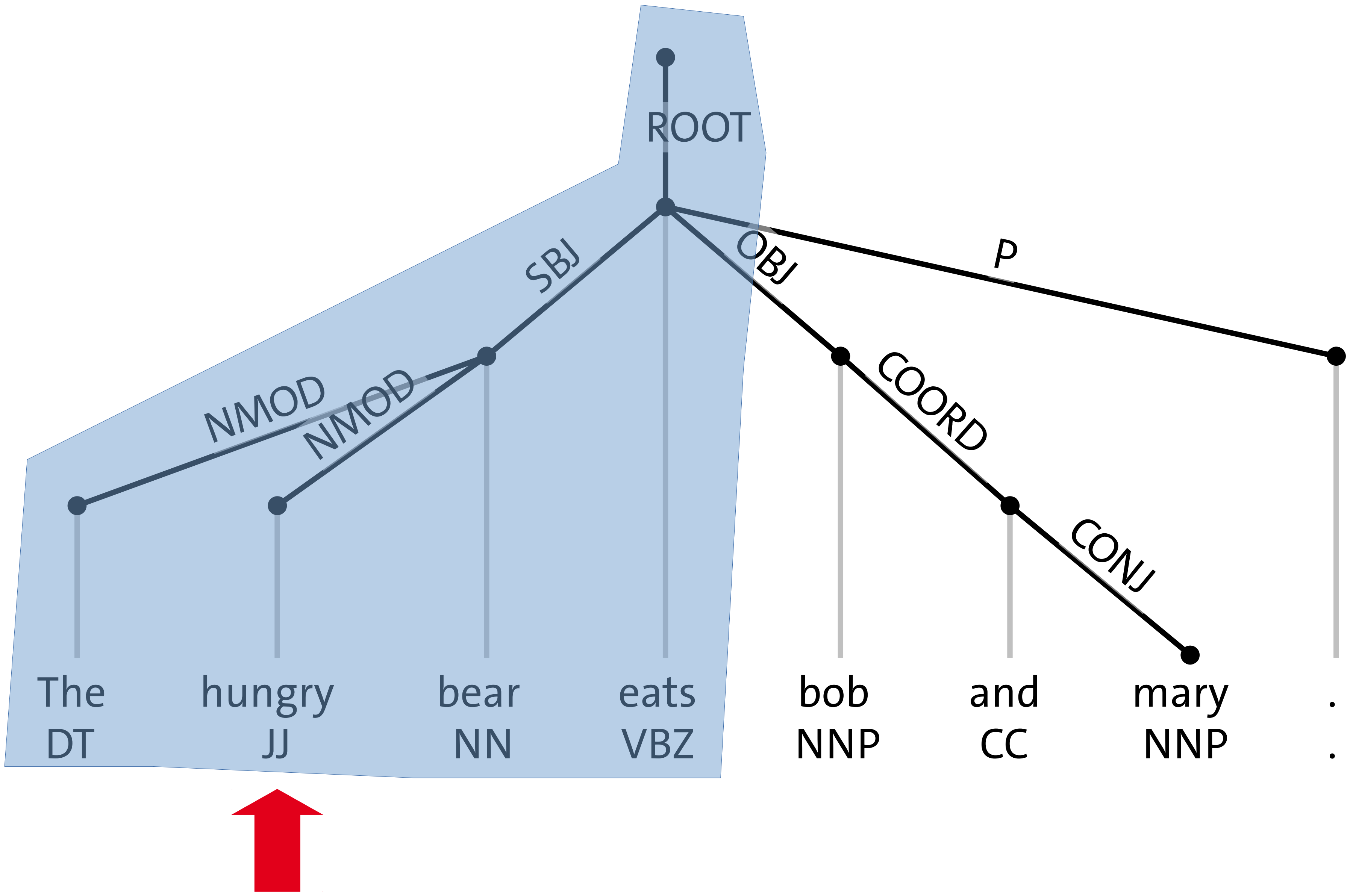
Incremental Gold Standard

- Important for training and evaluation
- Unlike Sequence problems, there is no 1:1 mapping from input to output
- Model reasonable expectations
  - Hand-written rules

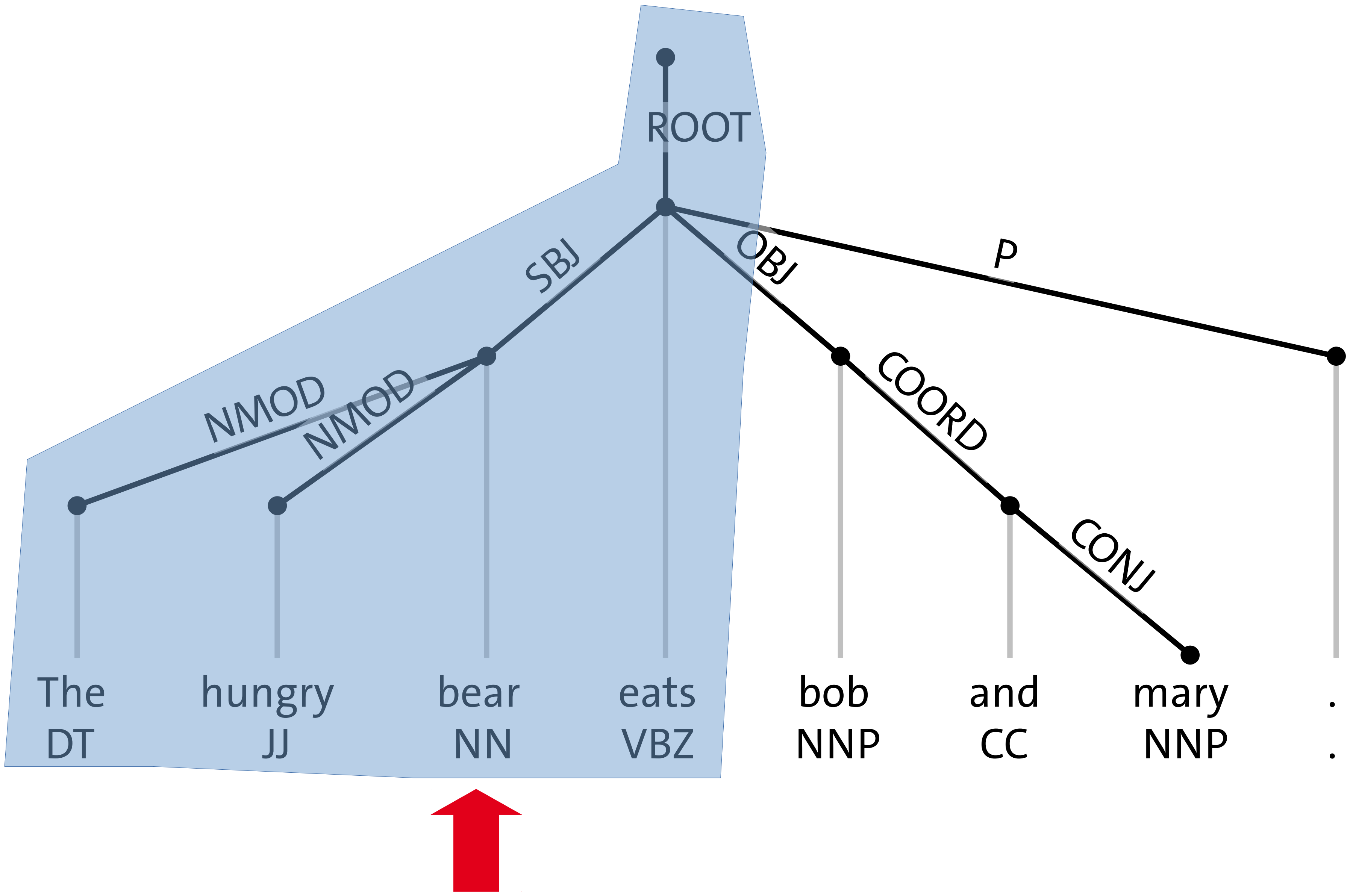
Incremental Gold Standard



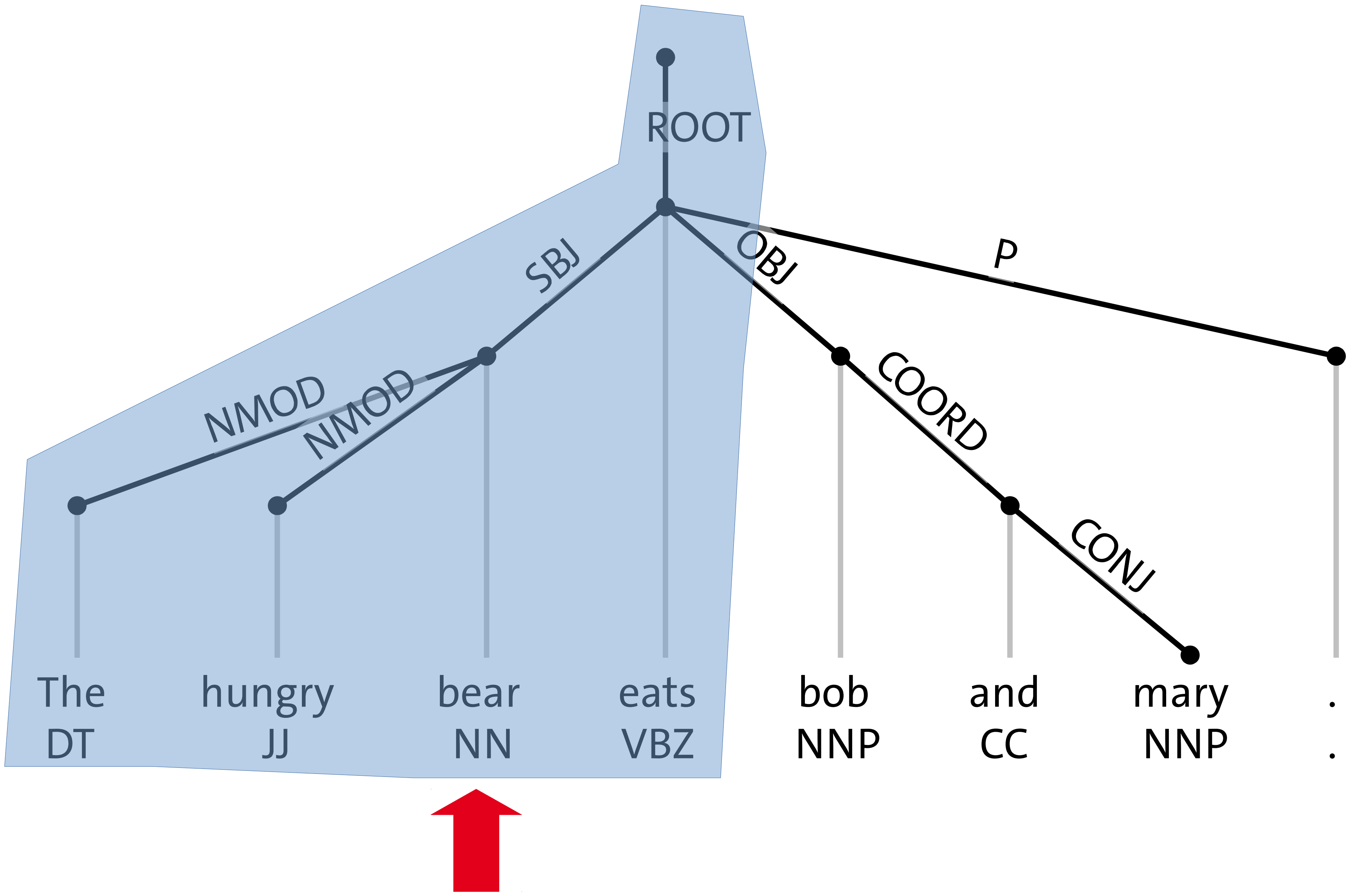
Incremental Gold Standard



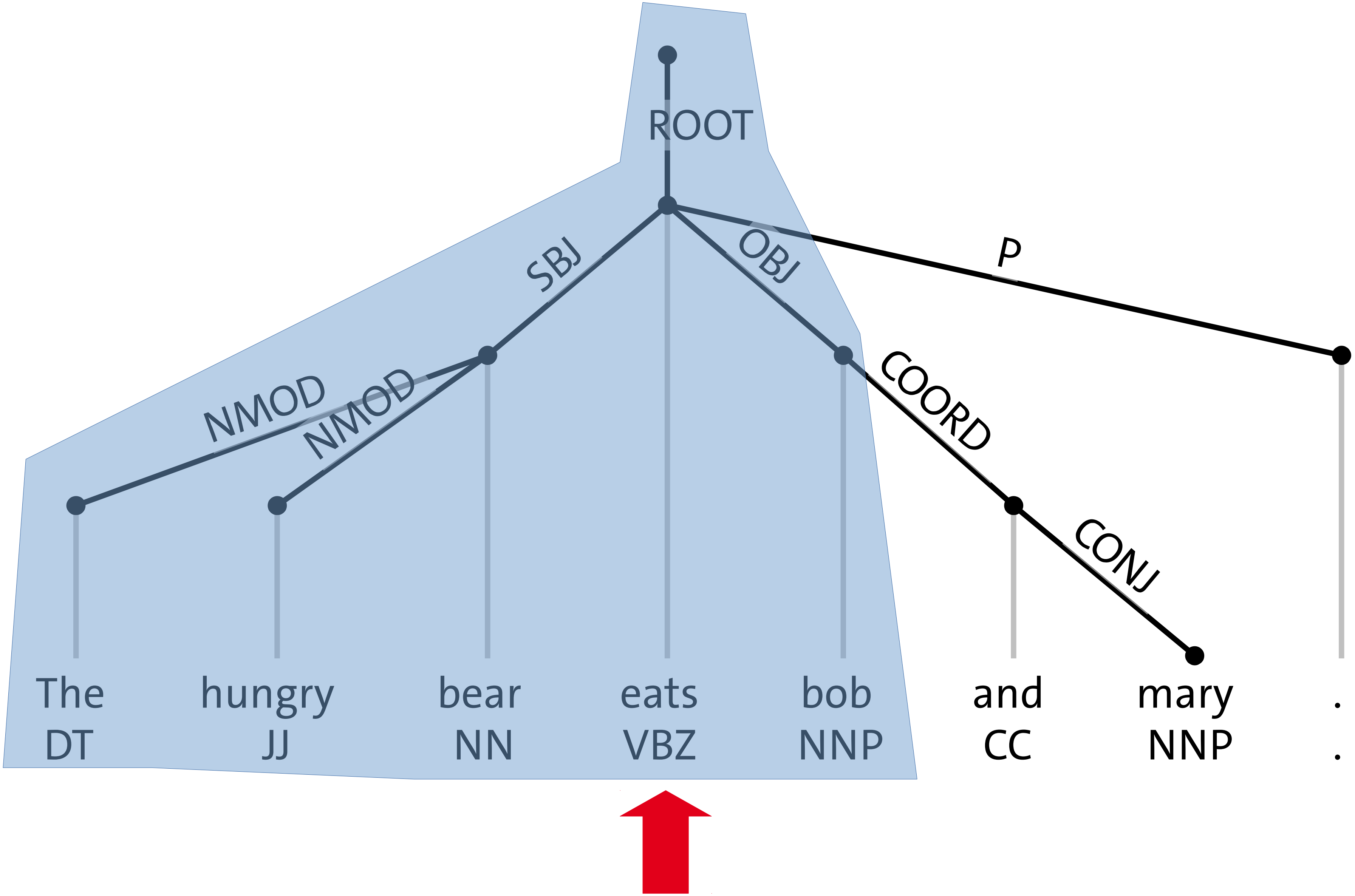
Incremental Gold Standard



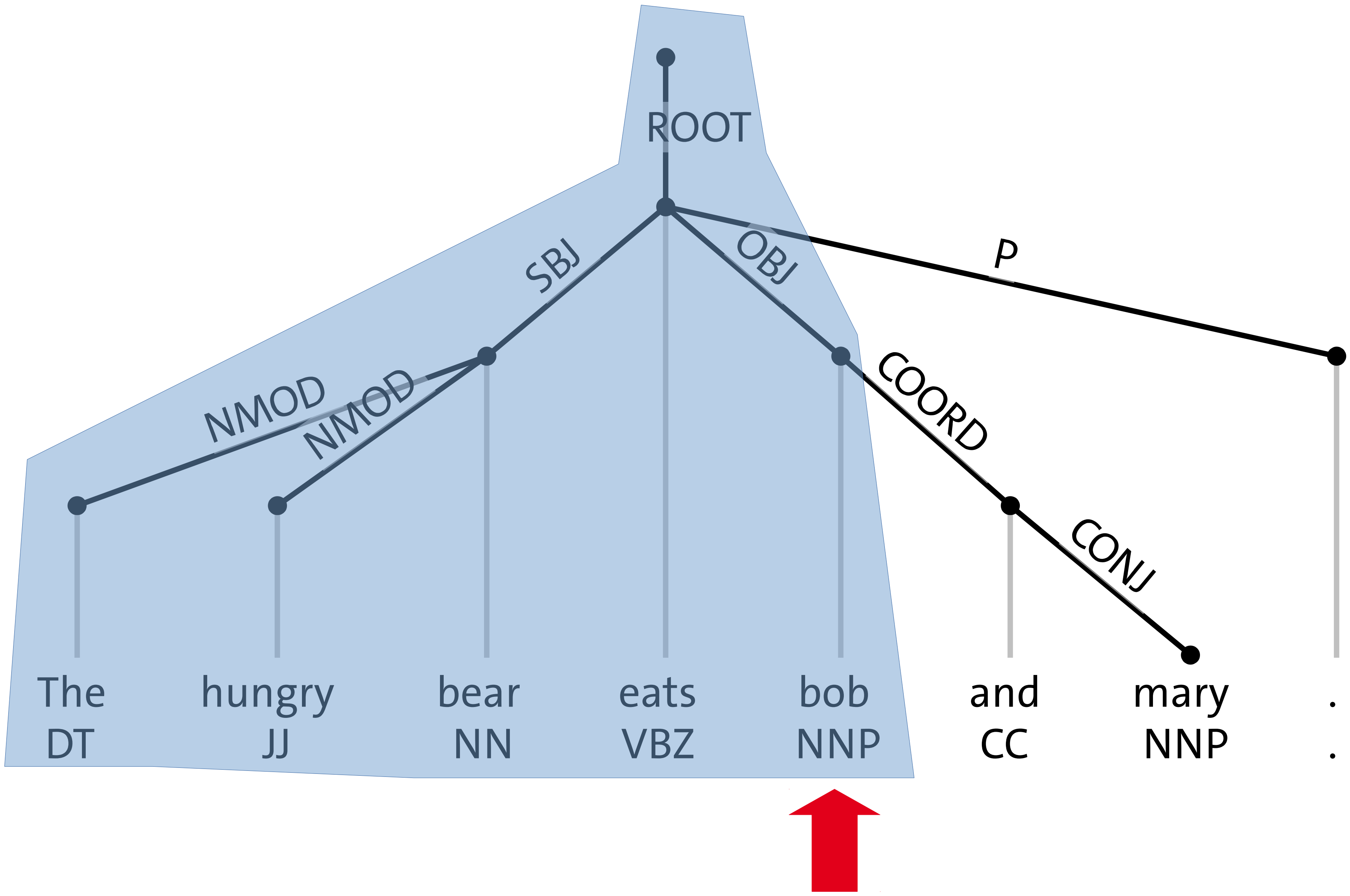
Incremental Gold Standard



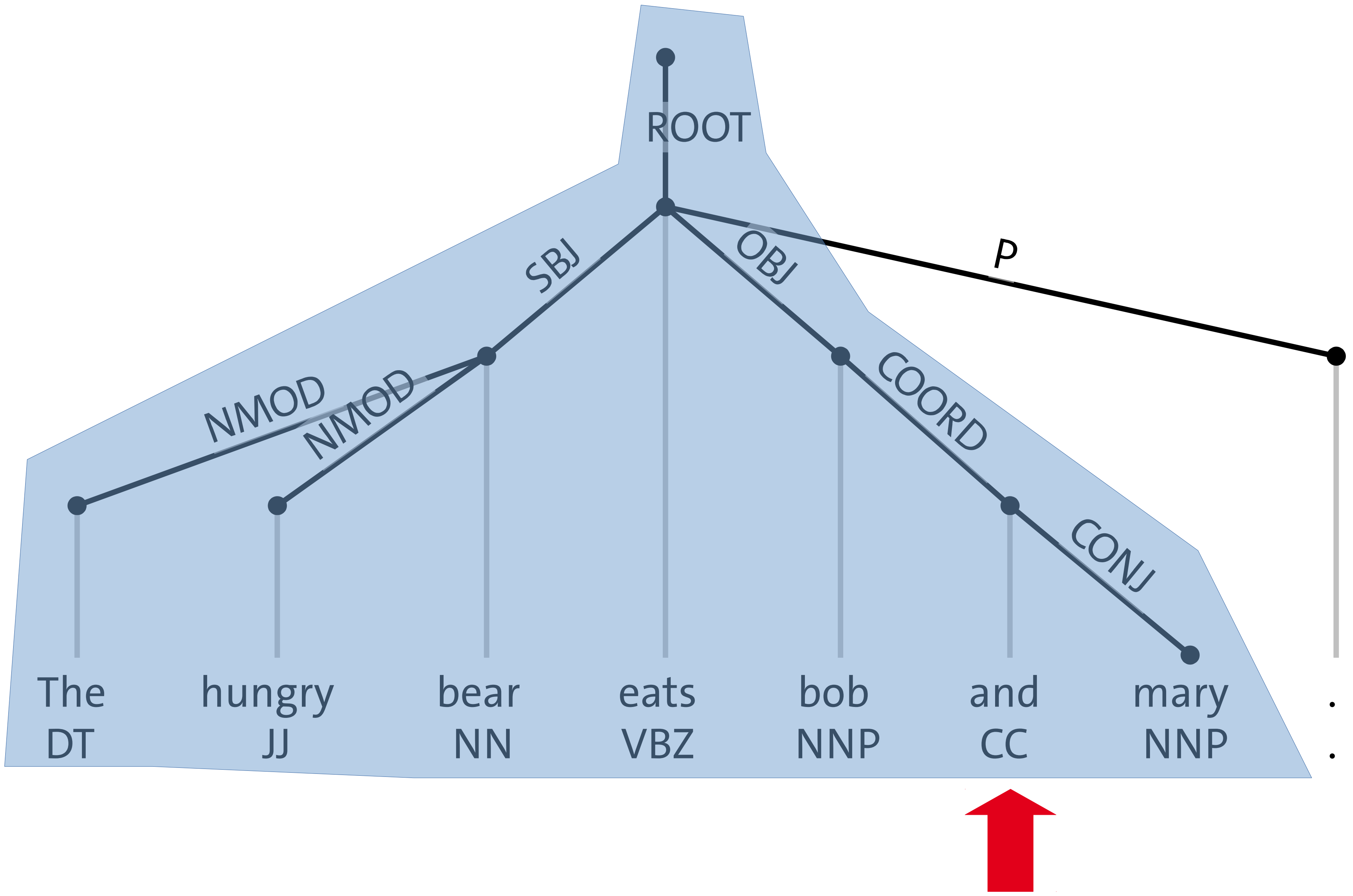
Incremental Gold Standard



Incremental Gold Standard

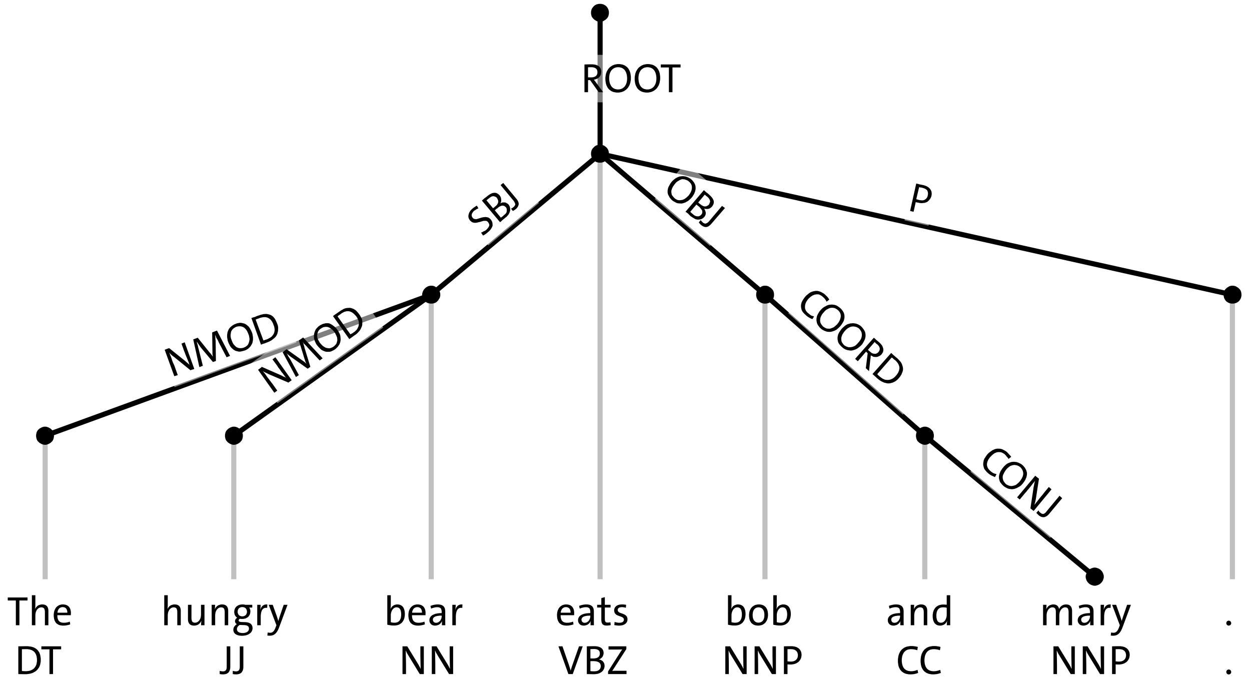


Incremental Gold Standard



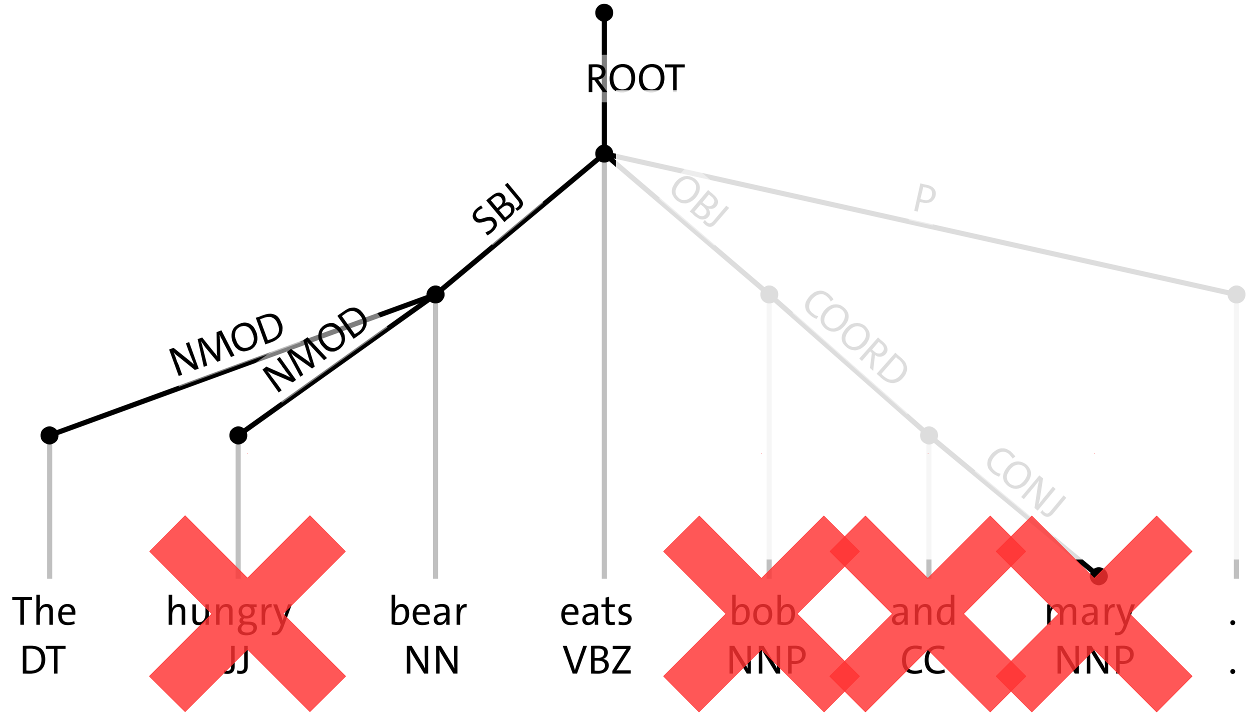
Incremental Gold Standard

- Determine what is predictable



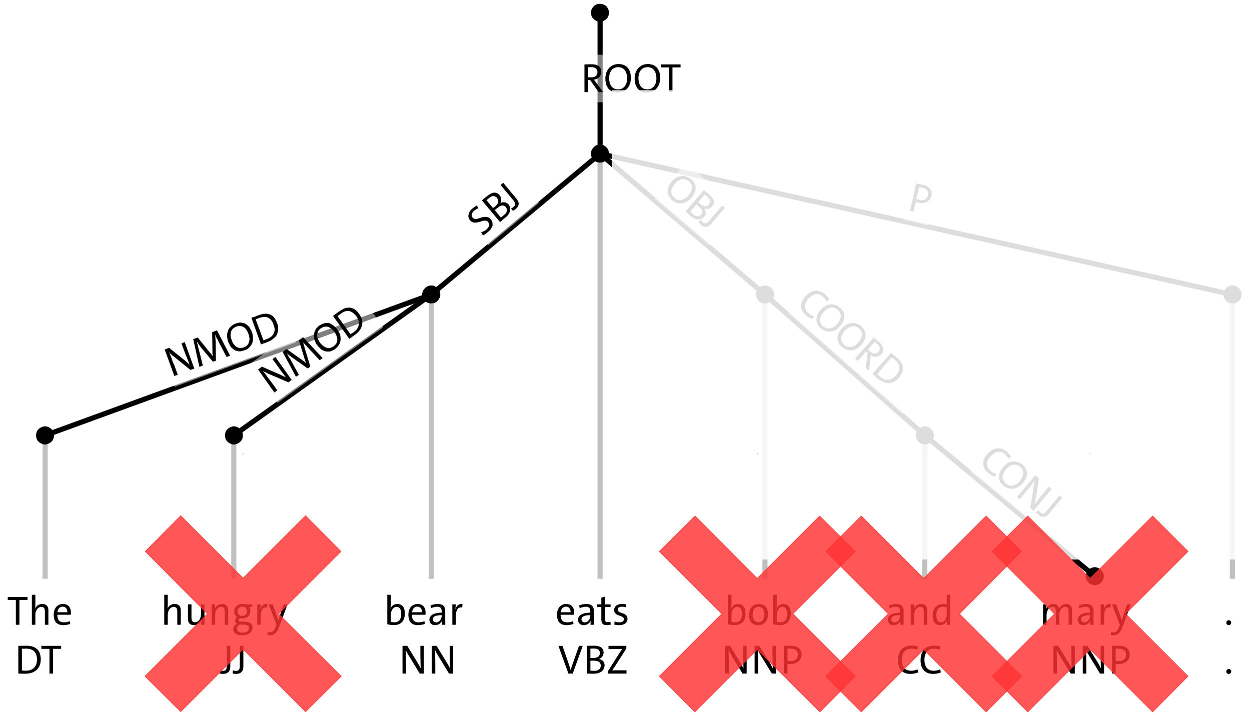
Incremental Gold Standard

- Determine what is predictable



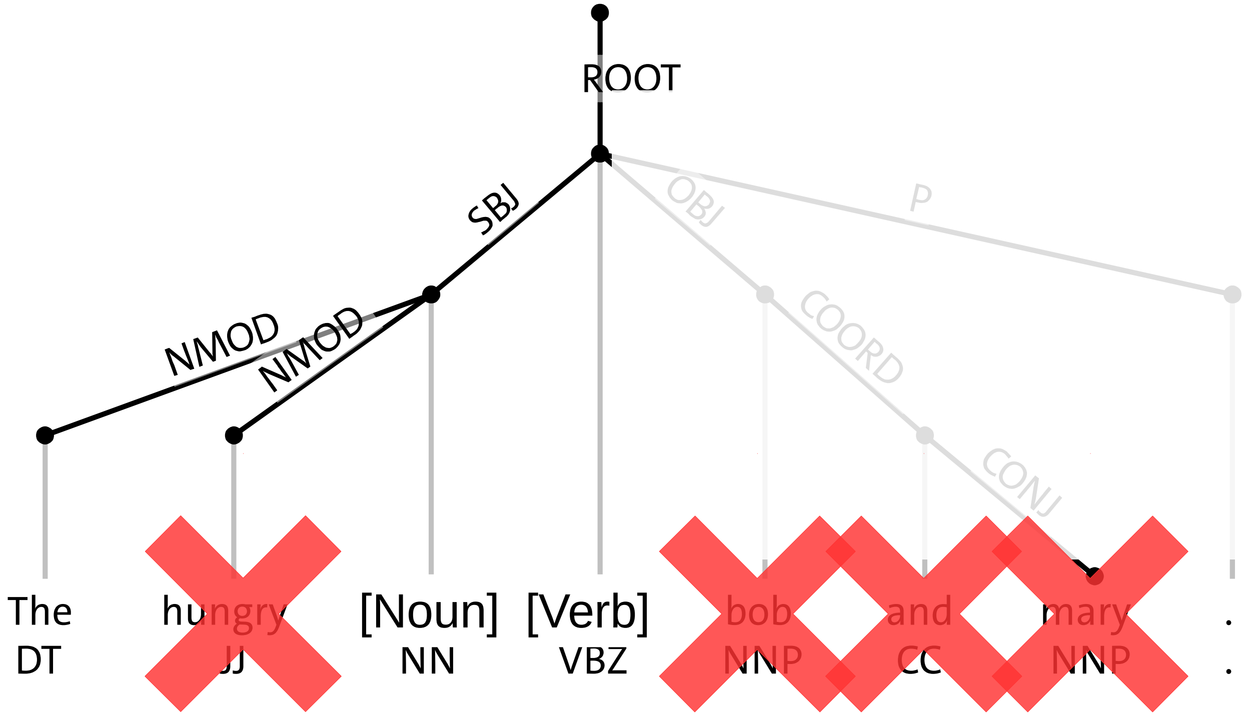
Incremental Gold Standard

- Determine what is predictable
- Delexicalize predicted words
  - Nounish → [Noun]
  - Verbish → [Verb]



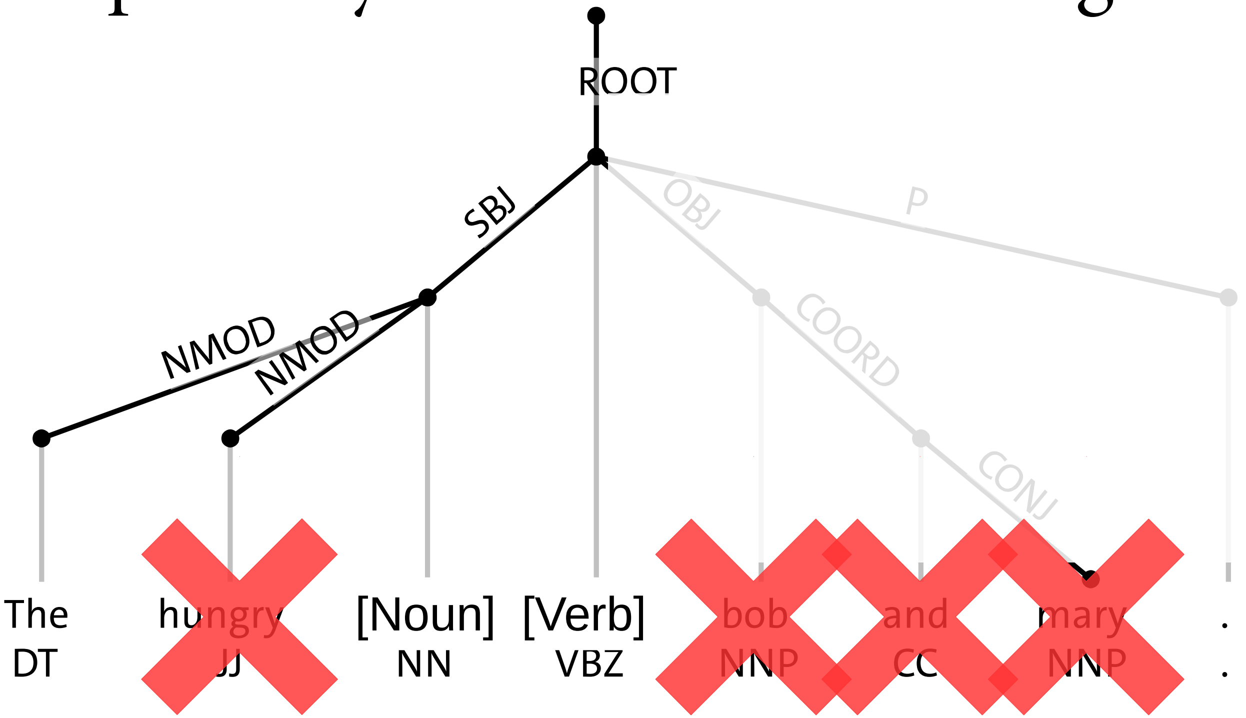
Incremental Gold Standard

- Determine what is predictable
- Delexicalize predicted words
  - Nounish → [Noun]
  - Verbish → [Verb]



Incremental Gold Standard

- Determine what is predictable
- Delexicalize predicted words
  - Nounish → [Noun]
  - Verbish → [Verb]
- Hope that your domain knowledge matches reality



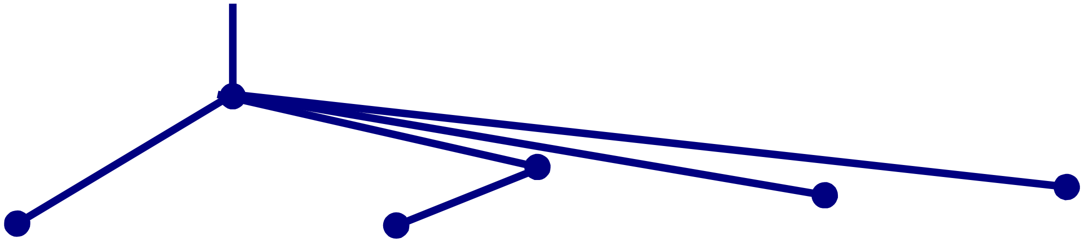
The Challenge of Virtual Nodes

- “Normal” dependency parsing:
  - **Where** should each word be attached to?
- Incremental dependency parsing:
  - Additionally: **Which** Virtual Nodes to include?
- Both problems depend on each other

A two-step process doesn't work!

The Need for a Fixed Token Set

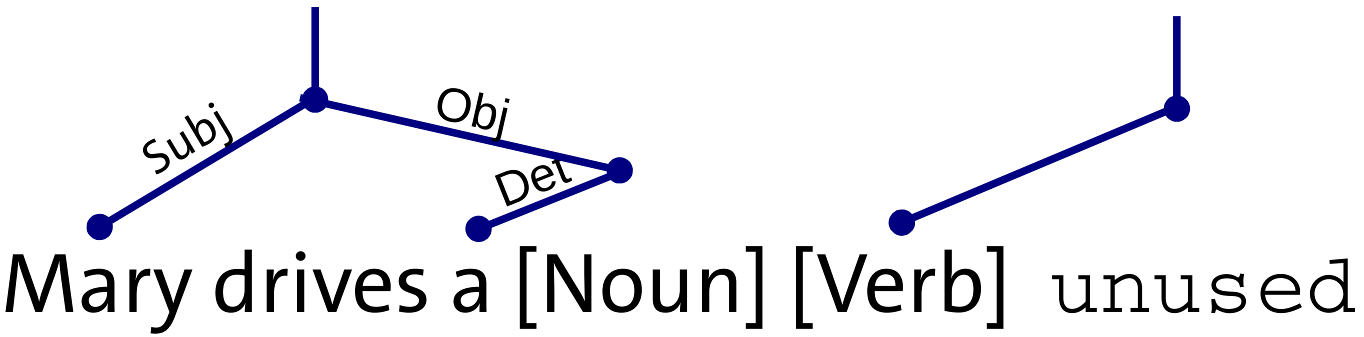
- Graph-based parsing: find MST
- Drives→[Noun] has positive score
- Which parse will be the best?
- If one prediction is good, two are better!



Mary drives a [Noun] [Noun] [Noun] ...

Solving “which” with “where”

- Use fixed set of VNs
  - Language-dependent
  - Obtained from generated prefixes
- Introduce `unused` node
- `unused` and VNs attached to it not part of analysis

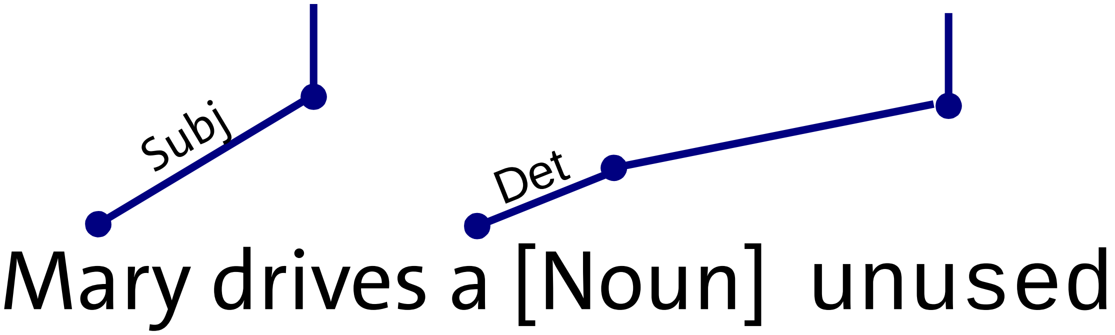


Mary drives a [Noun] [Verb] `unused`

Adapting TurboParser

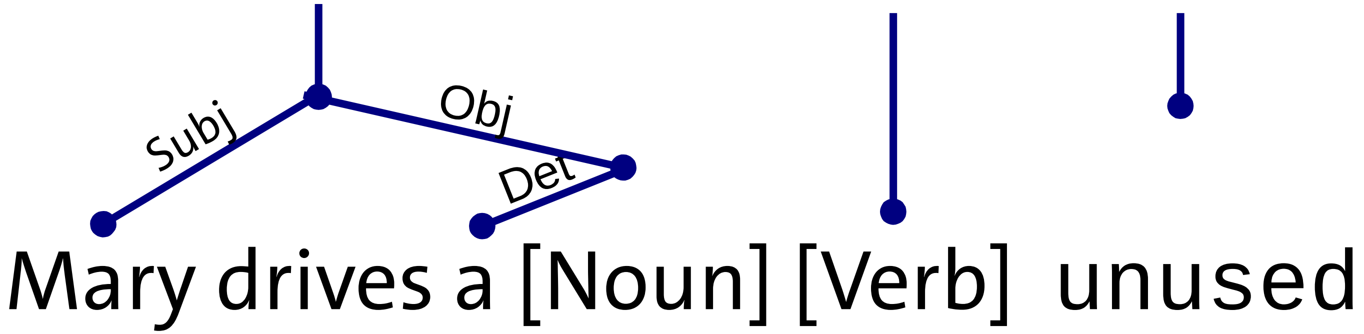
Adapting TurboParser

- TP uses ILP to formulate the parsing problem
- Tree structure of analyses ensured by factors
- Additional factors for VN and `unused`:
  - A VN attached to unused may not have any dependents



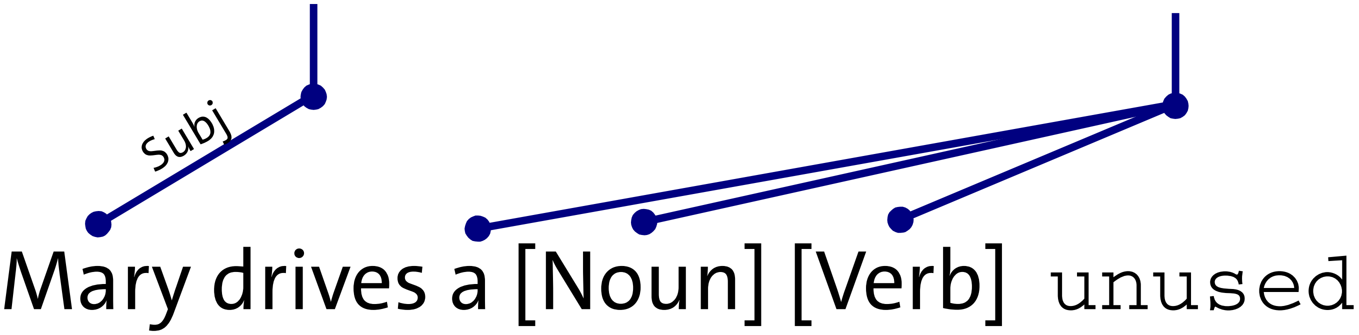
Adapting TurboParser

- TP uses ILP to formulate the parsing problem
- Tree structure of analyses ensured by factors
- Additional factors for VN and `unused`:
  - A VN attached to unused may not have any dependents
  - A VN may not be attached to 0 if it has no dependents.



Adapting TurboParser

- TP uses ILP to formulate the parsing problem
- Tree structure of analyses ensured by factors
- Additional factors for VN and `unused`:
  - A VN attached to unused may not have any dependents
  - A VN may not be attached to 0 if it has no dependents.
  - Only VNs may be attached to the unused node.



Mary drives a [Noun] [Verb] `unused`

Training TurboParser

- Training doesn't work on
  - whole sentences
  - Prefixes from gold standard

- Instead: use padded prefixes
  - Take gold standard
  - Add virtual nodes until set used for parsing is reached

Demonstration

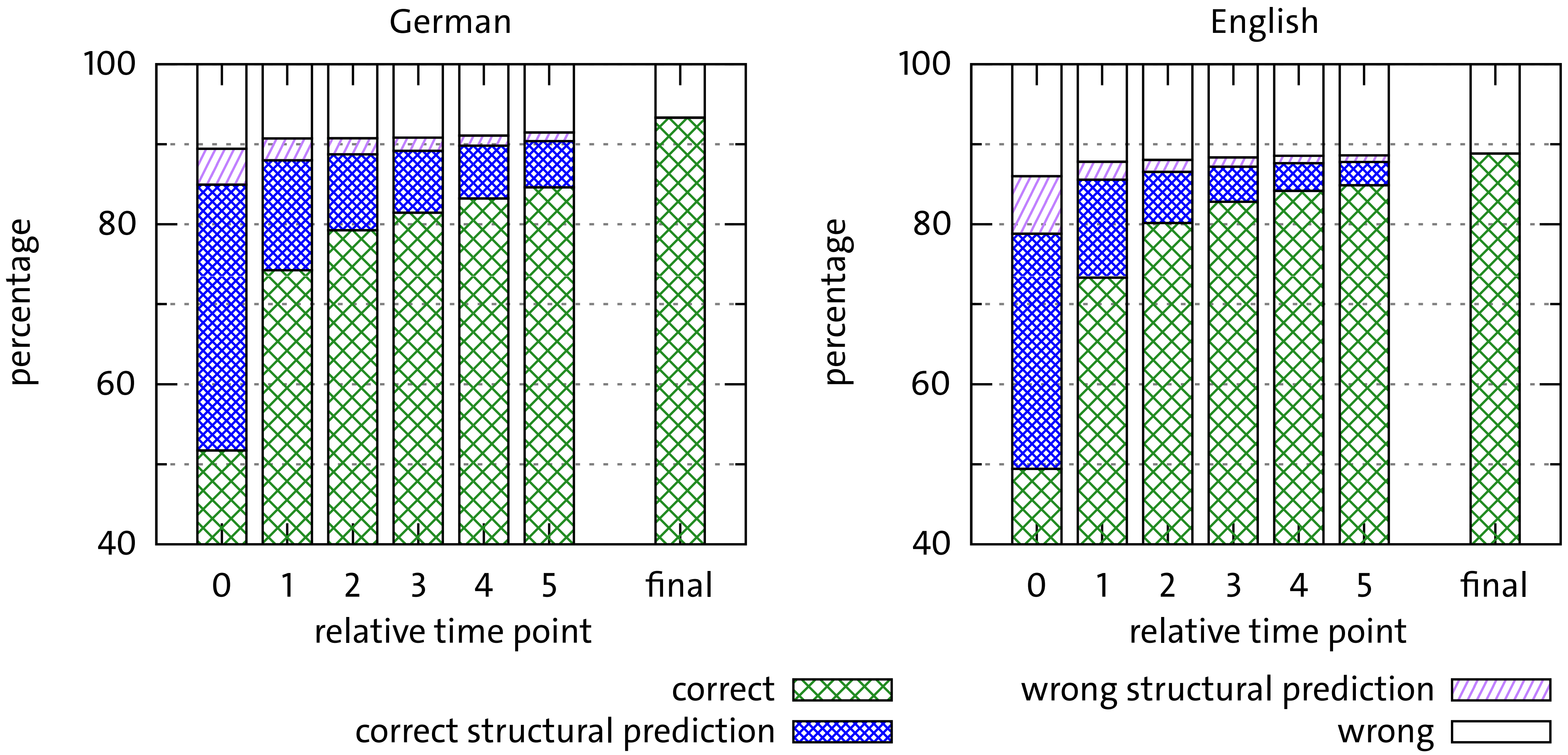
Evaluation

- Final accuracy: easy to measure, little use for incremental properties
- Attachment accuracy for newest words
- Prediction mapping
  - To complete sentence
  - To predictive gold standard

Evaluation (2)

- Capture dynamics of the incremental process
- Measure attachment accuracy of newest word
  - Wrt. Gold standard (“accuracy”)
  - Wrt. Parse of complete sentence (“stability”)
- Measure prediction precision and recall

incrTurboParser



Prediction

Quality of the predicted VNs

	English	German	German&Tagger	German (jwcdg)
precision	75%	67%	65%	33%
recall	58%	47%	46%	36%

- High prec. and rec. for TurboParser
- Clearly outperforms jwcdg (our previous parser)

Stability

How stable are attachments wrt. final analysis?

	Newest word		Sixth newest word	
	unlabeled	**labeled**	**unlabeled**	**labeled**
English	89.3%	84.9%	97.3%	97.11%
German	90.9%	88.9%	96.1%	95.7%

- High stability even for the newest word
- Note: We do **not** optimize for stability!

Wrap-up

- Every guarantee comes with a cost
- New training data needed for incremental processing
- Key concept for structured problems: Predictability
- If incremental is hard, try restart-incremental