Dependency Parsing

lecture 2

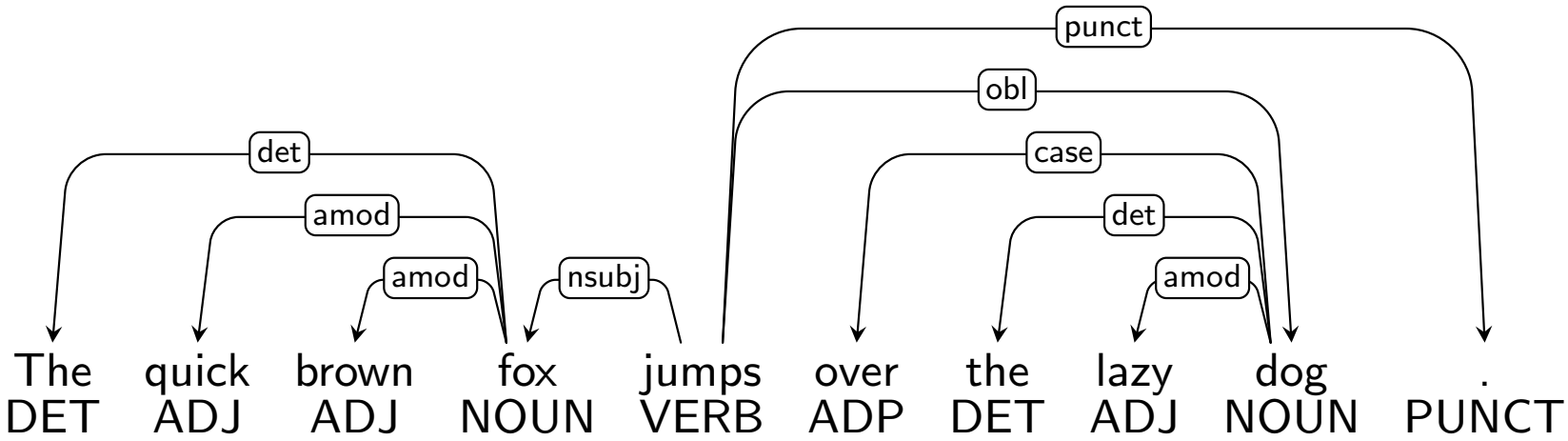
Jakub Waszczuk, Kilian Evang

Heinrich Heine Universität

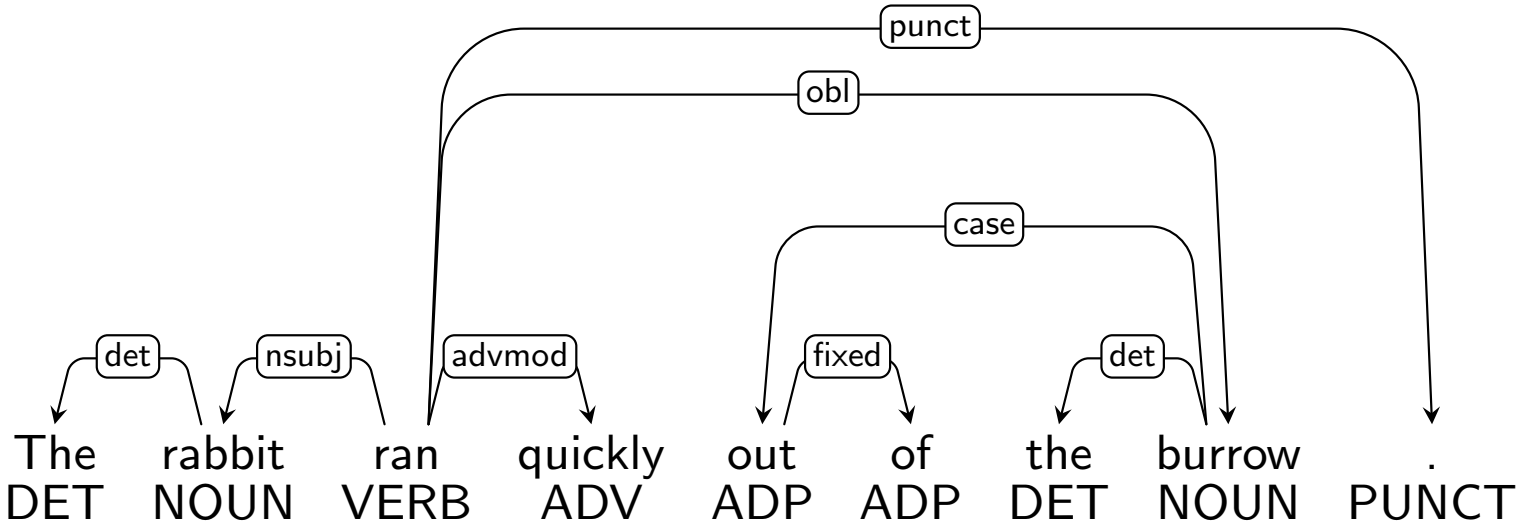
Summer semester 2021

Table of Contents

Annotation


punct
obl
det
case
amod
det
amod
nsubj
amod
The DET
quick ADJ
brown ADJ
fox NOUN
jumps VERB
over ADP
the DET
lazy ADJ
dog NOUN
. PUNCT

Annotation continued



Annotation continued

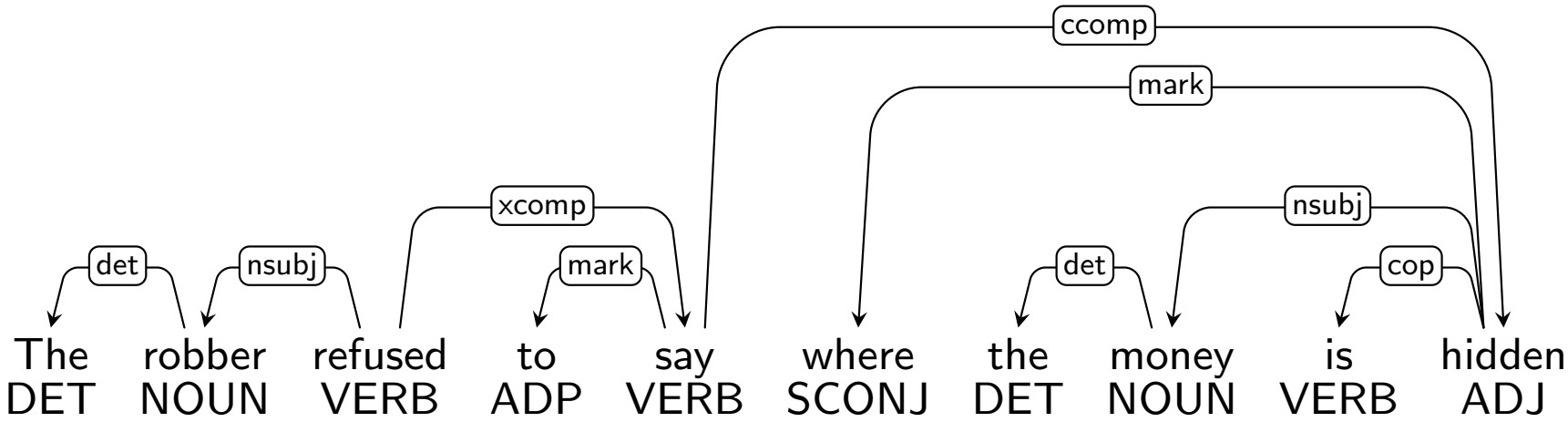


Figure: See also `https://universaldependencies.org/u/dep/all.html#aux-pass-passive-auxiliary`

Ambiguity

PP attachment:

- ??: I shot an elephant in my pyjamas
- ??: We should be discussing love on TV
- ??: John sees a girl with a telescope
- John likes a girl_1 with_1 a telescope

Ambiguity

PP attachment:

- ??: I shot an elephant in my pyjamas
- ??: We should be discussing love on TV
- ??: John sees a girl with a telescope
- John likes a girl_1 with_1 a telescope

Compounds:

- ??: He fed her cat food
- He $\text{fed}_{1,2}$ her cow_1 food_2

Ambiguity

PP attachment:

- ??: I shot an elephant in my pyjamas
- ??: We should be discussing love on TV
- ??: John sees a girl with a telescope
- John likes a girl_1 with_1 a telescope

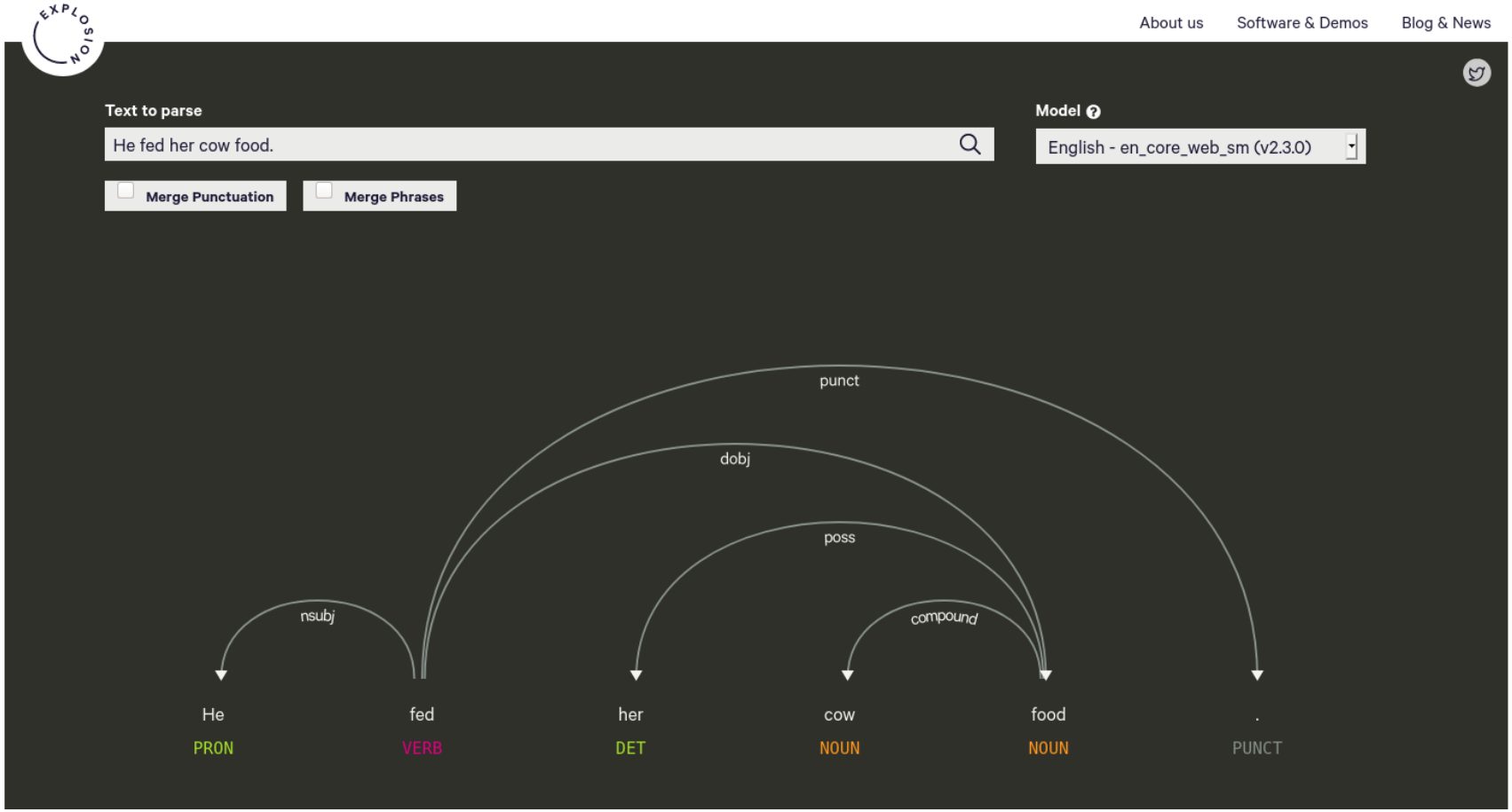
Compounds:

- ??: He fed her cat food
- He $\text{fed}_{1,2}$ her cow_1 food_2

Lexical/POS tag:

- ??: Each of us saw her duck

Ambiguity continued

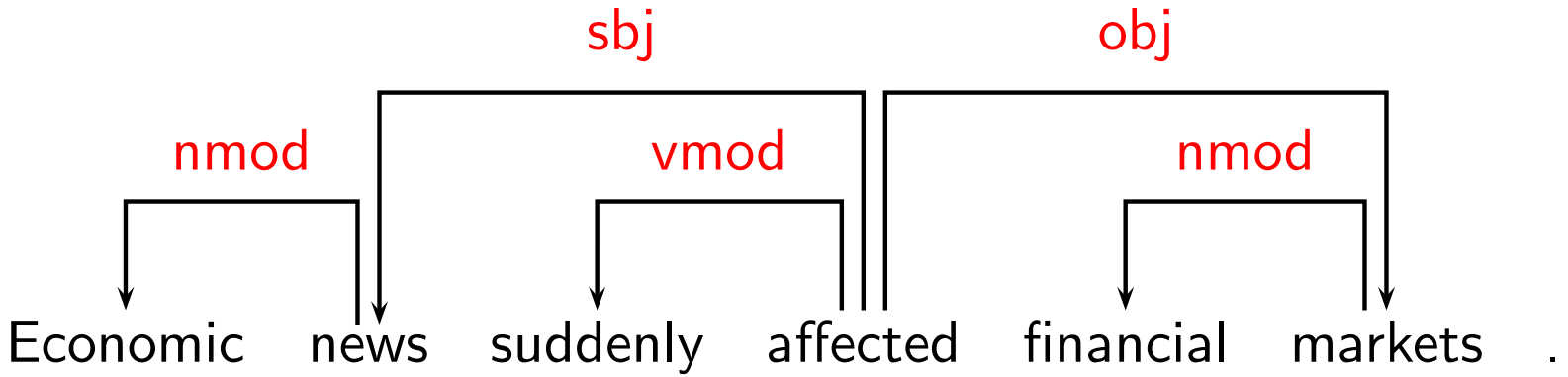


Criteria for Heads and Dependents

- Criteria for a syntactic relation between a head H and a dependent D in a construction C [Zwicky 1985, Hudson 1990]:
  1. H determines the syntactic category of C; H can replace C.
  2. H determines the semantic category of C; D specifies H.
  3. H is obligatory; D may be optional.
  4. H selects D and determines whether D is obligatory.
  5. The form of D depends on H (agreement or government).
  6. The linear position of D is specified with reference to H.
- Issues:
  - Syntactic (and morphological) versus semantic criteria
  - Exocentric versus endocentric constructions

Some Clear Cases

Construction	Head	Dependent
Exocentric	Verb	Subject (sbj)
	Verb	Object (obj)
Endocentric	Verb	Adverbial (vmod)
	Noun	Attribute (nmod)

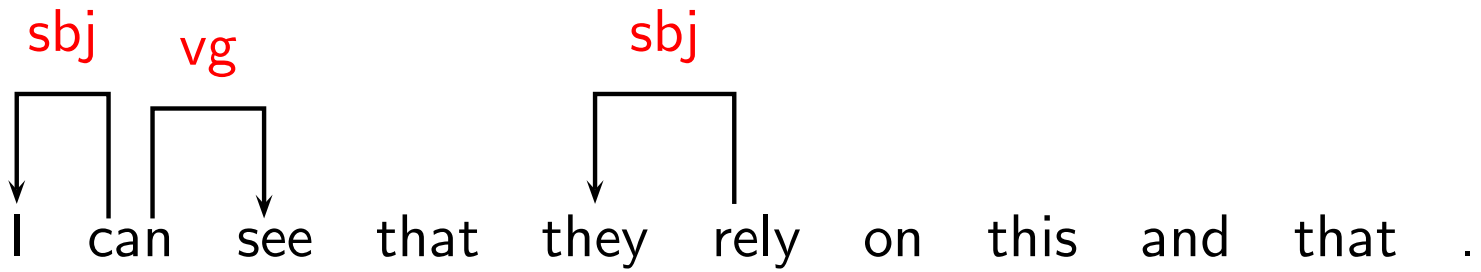


Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

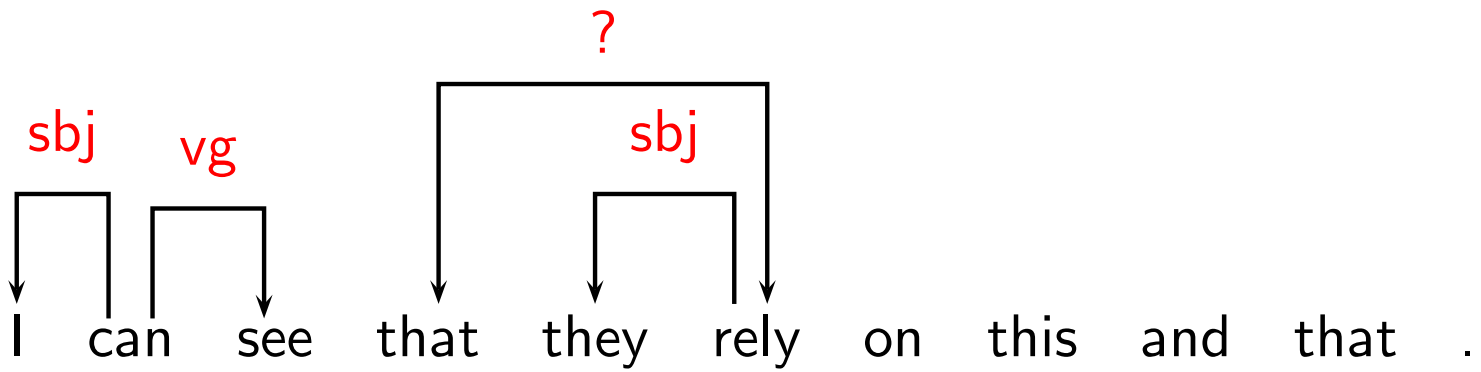
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



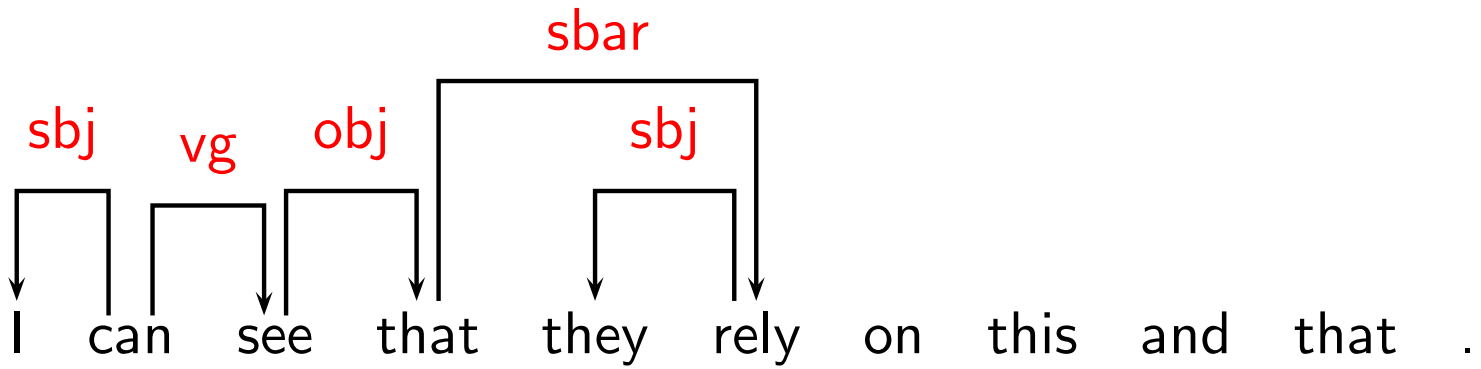
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



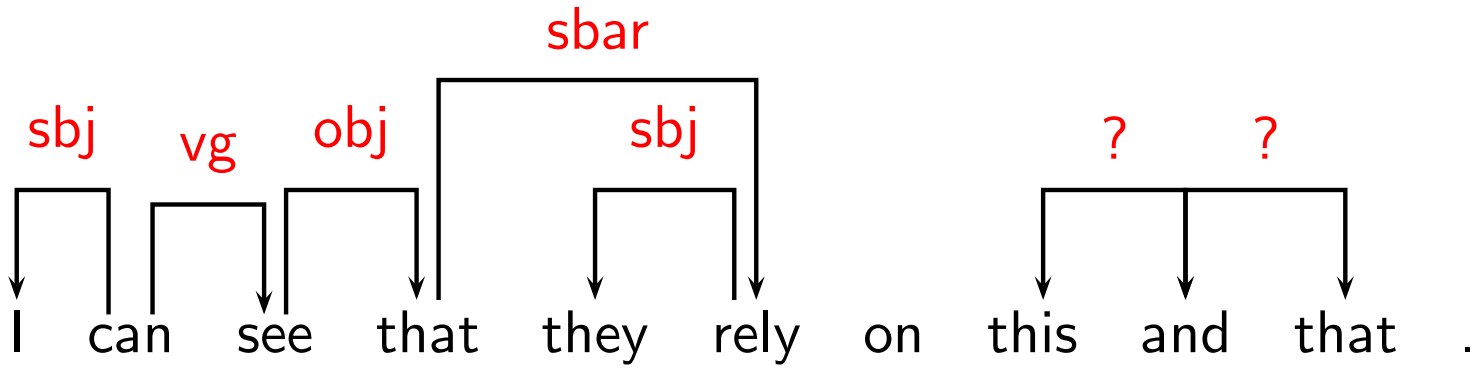
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



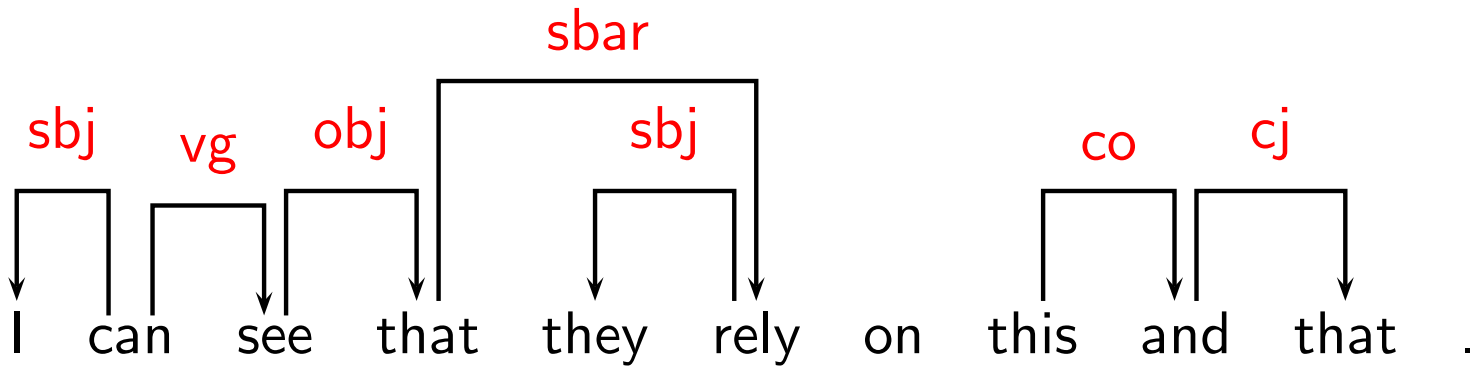
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



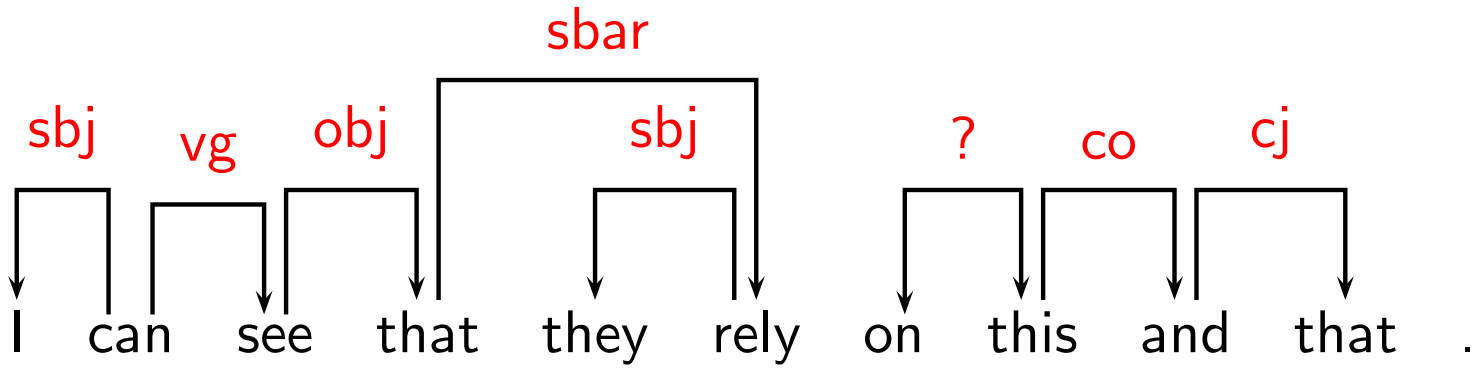
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



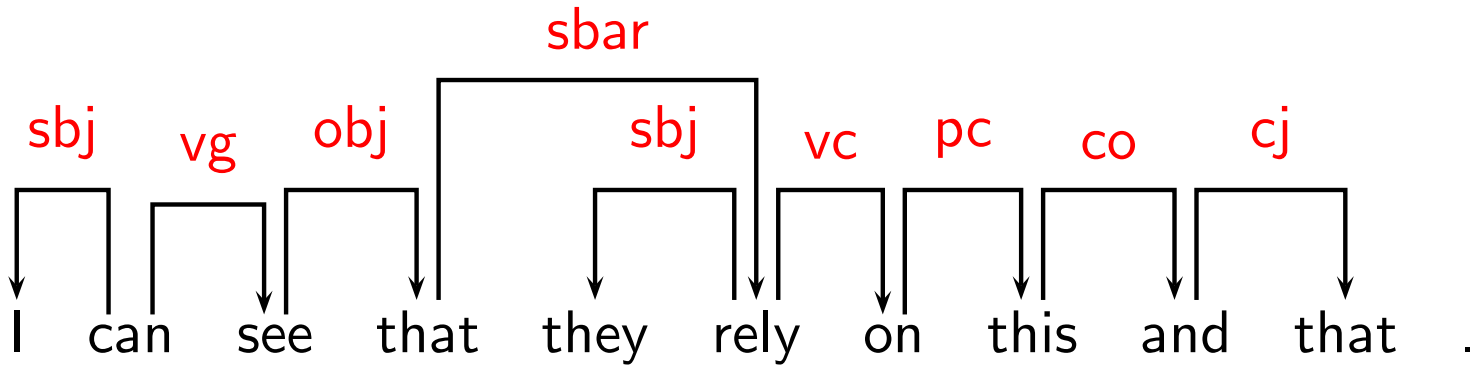
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



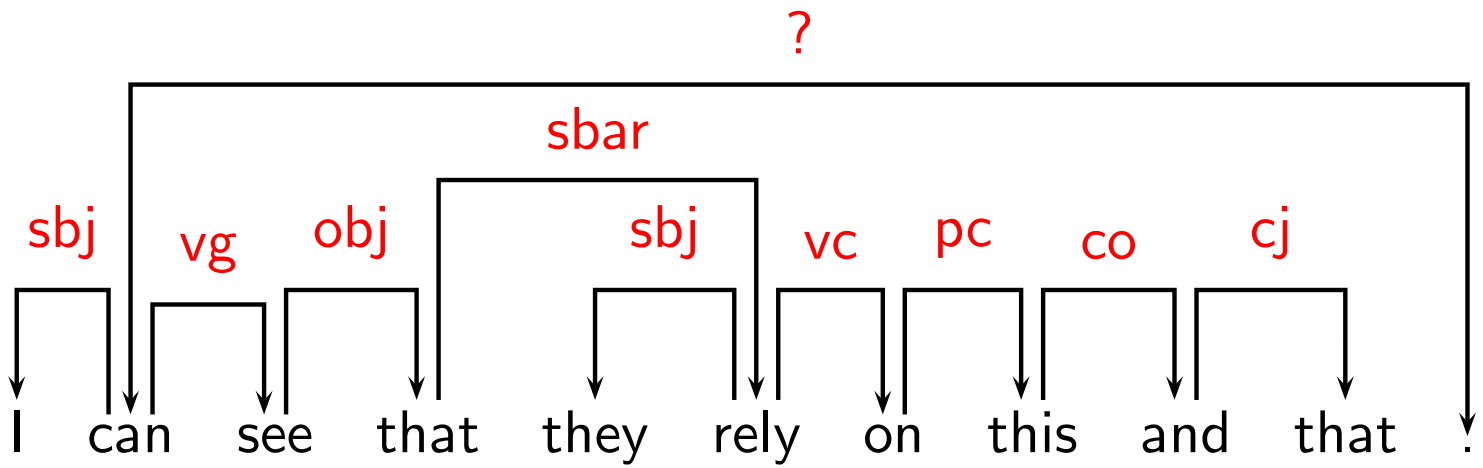
Some Tricky Cases

- Complex verb groups (auxiliary $\leftrightarrow$ main verb)
- Subordinate clauses (complementizer $\leftrightarrow$ verb)
- Coordination (coordinator $\leftrightarrow$ conjuncts)
- Prepositional phrases (preposition $\leftrightarrow$ nominal)
- Punctuation



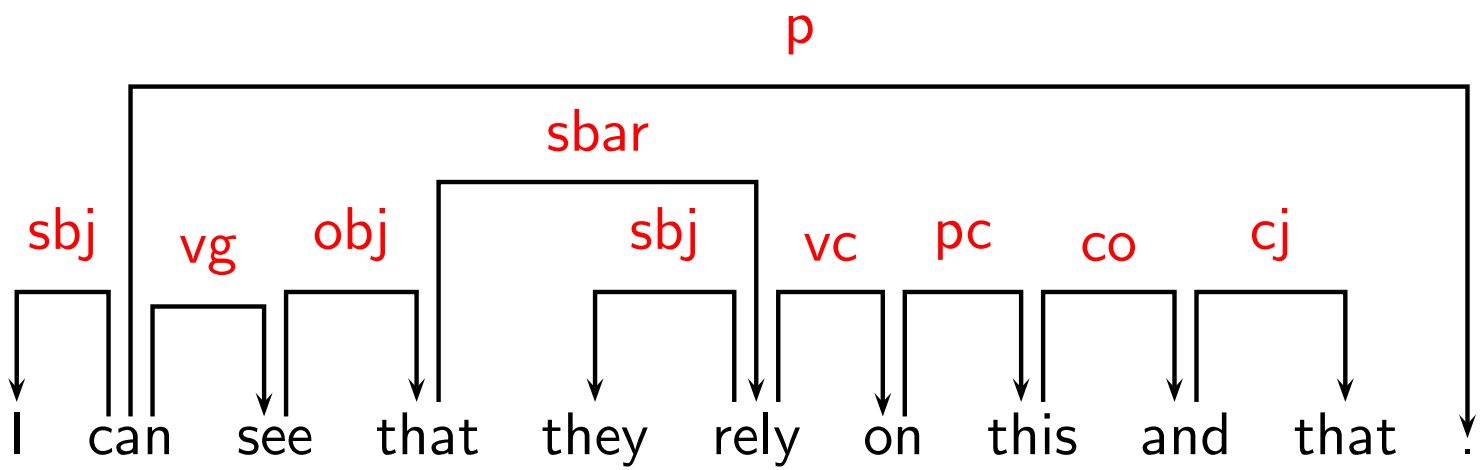
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

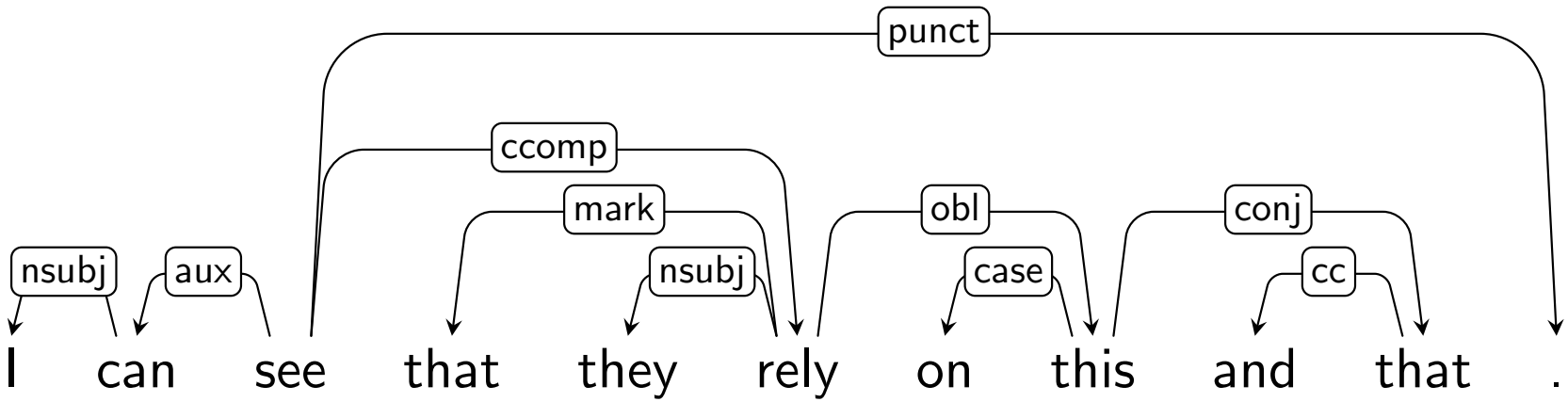


Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation



UD-based analysis



Is any of the two encoding formats better than the other?

- UD strives to be more universal, and thus propose parallel analyses for the same phenomena in different languages
- In practice, one or the other encoding format might be easier to handle for a particular data-driven parser
- They encode the same linguistic phenomena differently, but as long as we can convert between the two formats (without information loss), the choice doesn't really matter

Following [Rambow, 2010], we can distinguish:

- Representation type (dependency trees, constituency trees, hybrid)
- Syntactic theory (UD, PTB, LFG, TAG, ...)
- Linguistic content

Data-driven parsers are mostly concerned with the representation type

Table of Contents

Dependency Graphs

- A dependency structure can be defined as a directed graph G, consisting of
  - a set V of nodes (vertices),
  - a set A of arcs (directed edges),
  - a linear precedence order $<$ on V (word order).
- Labeled graphs:
  - Nodes in V are labeled with word forms (and annotation).
  - Arcs in A are labeled with dependency types:
    - $L = \{l_1, \ldots, l_{|L|}\}$ is the set of permissible arc labels.
    - Every arc in A is a triple (i, j, k), representing a dependency from w_i to w_j with label l_k.

Dependency Graph Notation

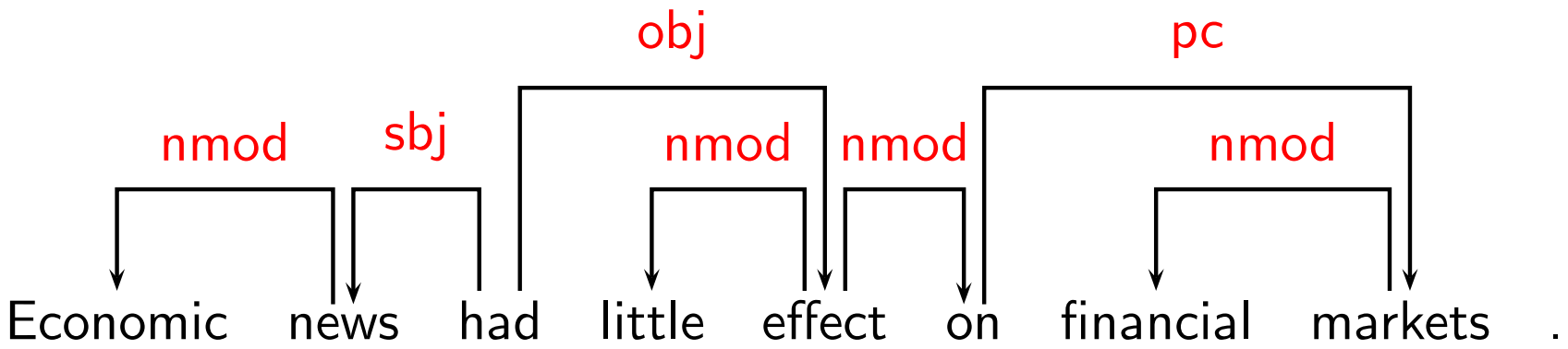
- For a dependency graph $G = (V, A)$
- With label set $L = \{l_1, \ldots, l_{|L|}\}$
  - $i \rightarrow j \equiv \exists k : (i, j, k) \in A$
  - $i \leftrightarrow j \equiv i \rightarrow j \vee j \rightarrow i$
  - $i \rightarrow^* j \equiv i = j \vee \exists i' : i \rightarrow i',\ i' \rightarrow^* j$
  - $i \leftrightarrow^* j \equiv i = j \vee \exists i' : i \leftrightarrow i',\ i' \leftrightarrow^* j$

Formal Conditions on Dependency Graphs

- G is (weakly) connected:
  - If $i, j \in V$, $i \leftrightarrow^* j$.
- G is acyclic:
  - If $i \rightarrow j$, then not $j \rightarrow^* i$.
- G obeys the single-head constraint:
  - If $i \rightarrow j$, then not $i' \rightarrow j$, for any $i' \neq i$.
- G is projective:
  - If $i \rightarrow j$, then $i \rightarrow^* i'$, for any i' such that $i < i' < j$ or $j < i' < i$.

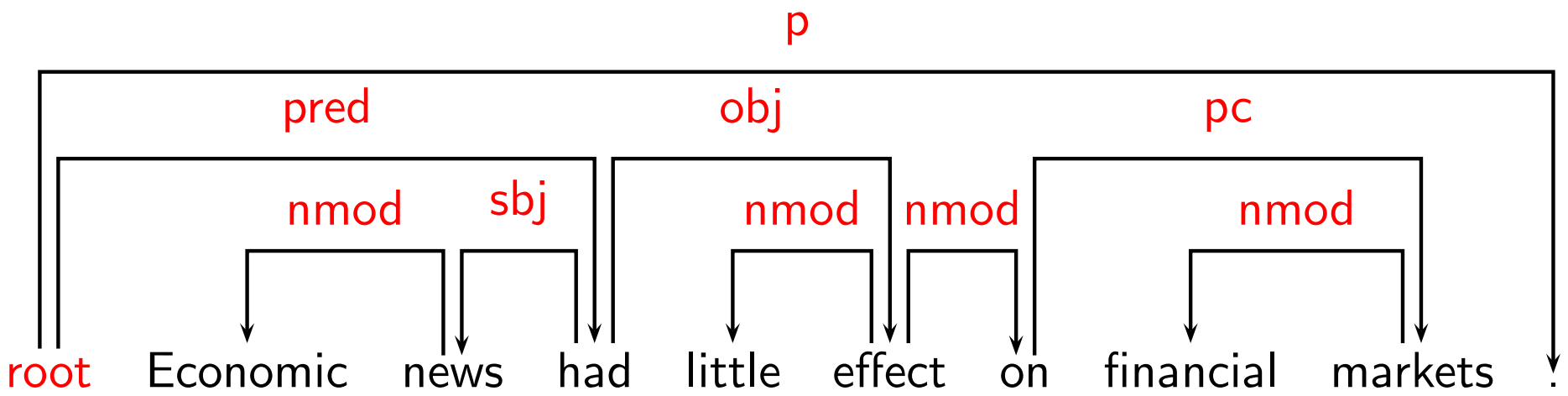
Connectedness, Acyclicity and Single-Head

- Intuitions:
  - Syntactic structure is complete (Connectedness).
  - Syntactic structure is hierarchical (Acyclicity).
  - Every word has at most one syntactic head (Single-Head).
- Connectedness can be enforced by adding a special root node.



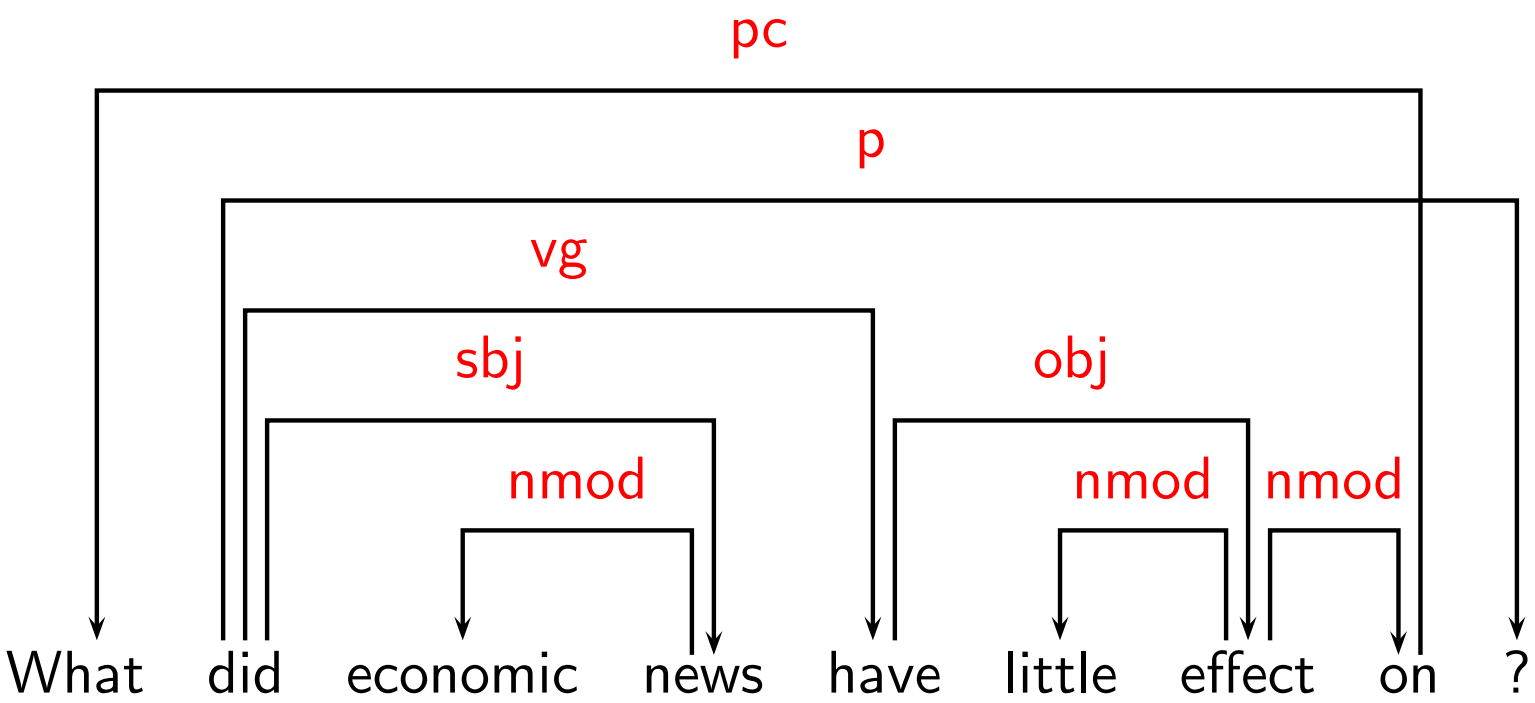
Connectedness, Acyclicity and Single-Head

- Intuitions:
  - Syntactic structure is complete (Connectedness).
  - Syntactic structure is hierarchical (Acyclicity).
  - Every word has at most one syntactic head (Single-Head).
- Connectedness can be enforced by adding a special root node.

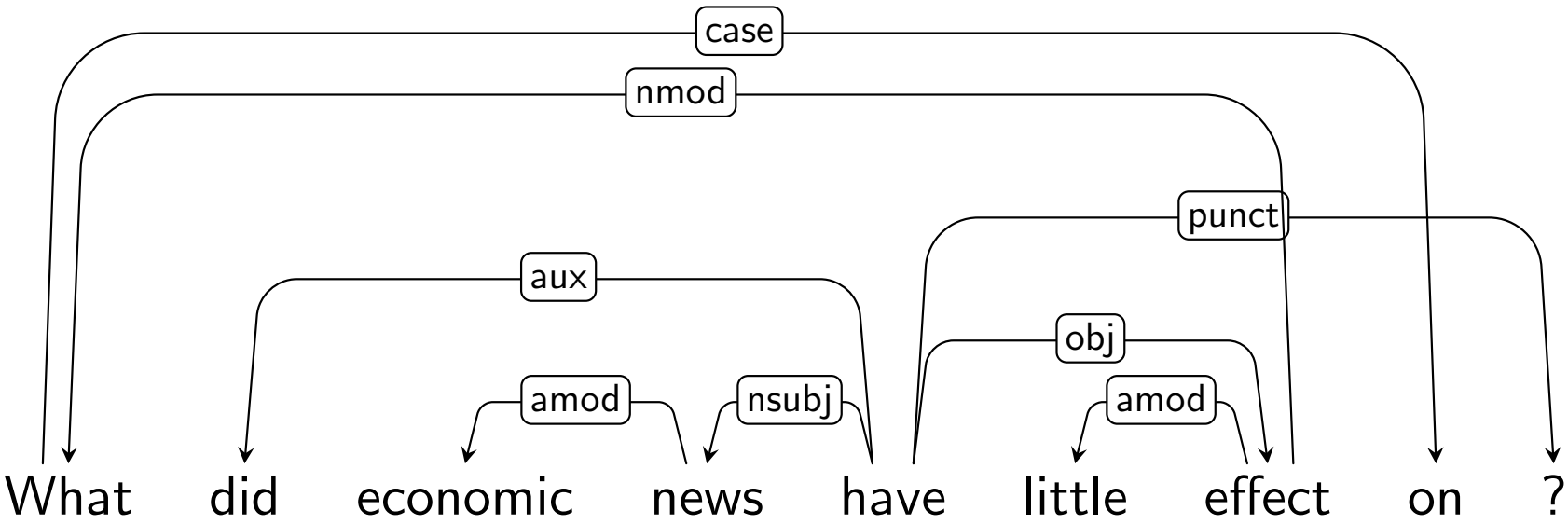


Projectivity

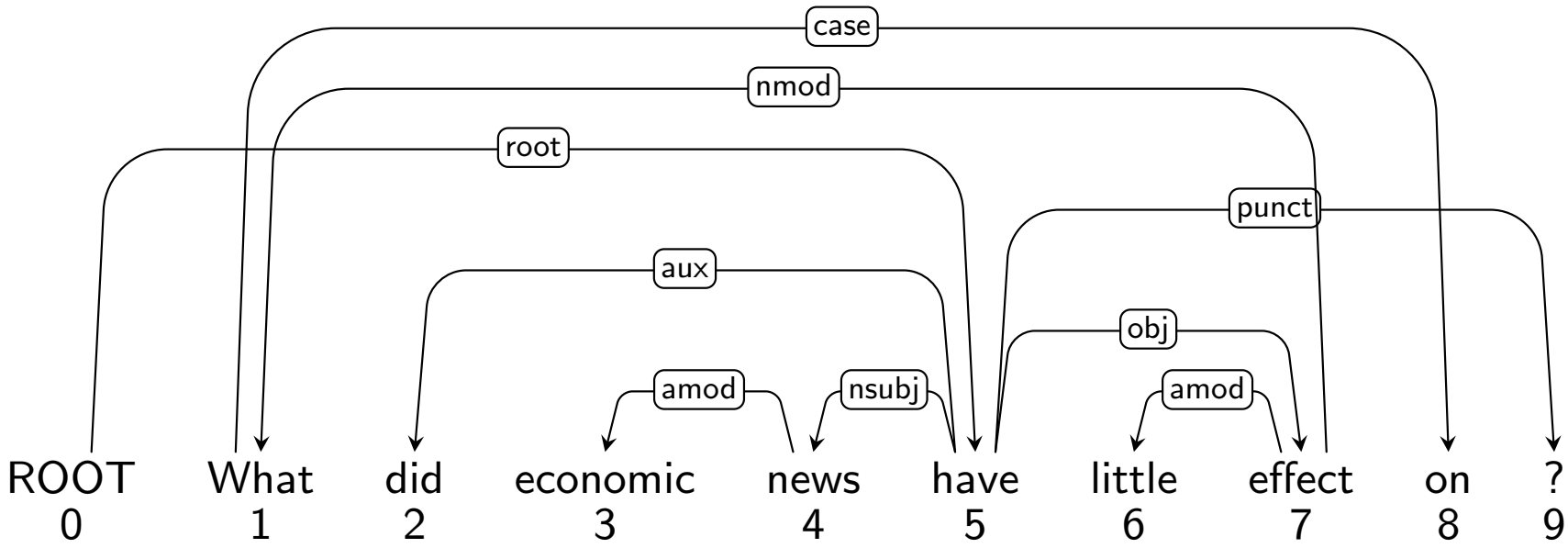
- Most theoretical frameworks do not assume projectivity.
- Non-projective structures are needed to account for
  - long-distance dependencies,
  - free word order.



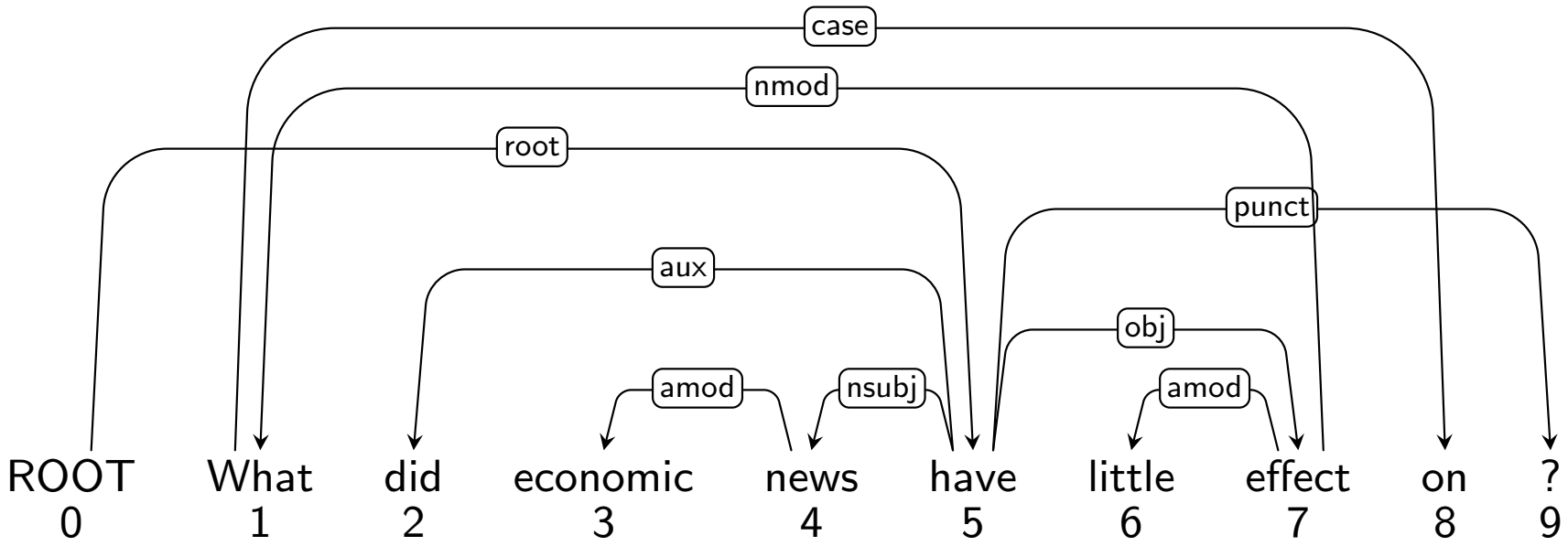
Projectivity: UD-based Analysis



Projectivity: UD-based Analysis



Projectivity: UD-based Analysis



$7 \rightarrow 1$ does not imply $7 \rightarrow^* 5$

Table of Contents

Dependency Parsing

- The problem:
  - Input: Sentence $x = w_0, w_1, \ldots, w_n$ with $w_0 = root$
  - Output: Dependency graph $G = (V, A)$ for x where:
    - $V = \{0, 1, \ldots, n\}$ is the vertex set,
    - A is the arc set, i.e., $(i, j, k) \in A$ represents a dependency from w_i to w_j with label $l_k \in L$
- Two main approaches:
  - Grammar-based parsing
    - Context-free dependency grammar
    - Constraint dependency grammar
  - Data-driven parsing
    - Transition-based models
    - Graph-based models

Transition-Based Models

- Basic idea:
  - Define a transition system (state machine) for mapping a sentence to its dependency graph.
  - Learning: Induce a model for predicting the next state transition, given the transition history.
  - Parsing: Construct the optimal transition sequence, given the induced model.
- Characteristics:
  - Local training of a model for optimal transitions
  - Greedy search/inference

Graph-Based Models

- Basic idea:
  - Define a space of candidate dependency graphs for a sentence.
  - Learning: Induce a model for scoring an entire dependency graph for a sentence.
  - Parsing: Find the highest-scoring dependency graph, given the induced model.
- Characteristics:
  - Global training of a model for optimal dependency graphs
  - Exhaustive search/inference

Table of Contents

Pros and Cons of Dependency Parsing

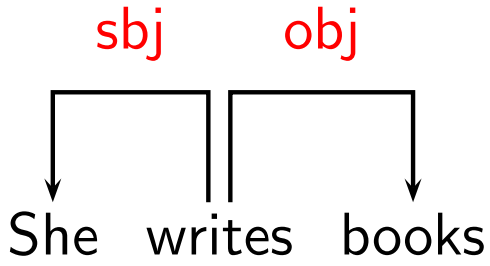
- What are the advantages of dependency-based methods?
- What are the disadvantages?
- Four types of considerations:
  - Complexity
  - Transparency
  - Word order
  - Expressivity

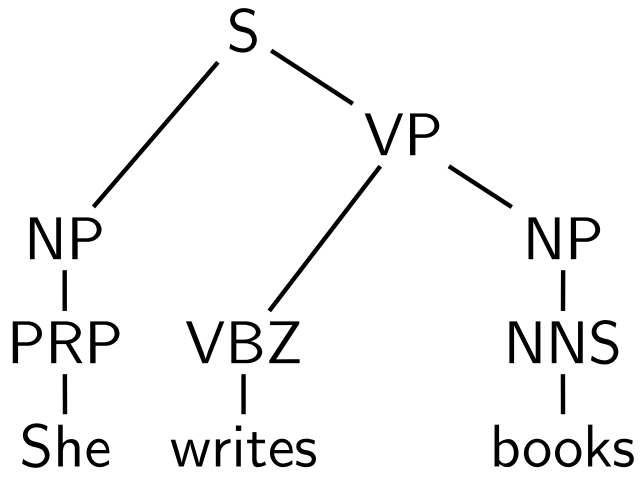
Complexity

- Practical complexity:
  - Given the Single-Head constraint, parsing a sentence $x = w_1, \ldots, w_n$ can be reduced to labeling each token w_i with:
    - a head word h_i,
    - a dependency type d_i.
- Theoretical complexity:
  - By exploiting the special properties of dependency graphs, it is sometimes possible to improve worst-case complexity compared to constituency-based parsing:
    - Lexicalized parsing in $O(n^3)$ time [Eisner 1996]

Transparency

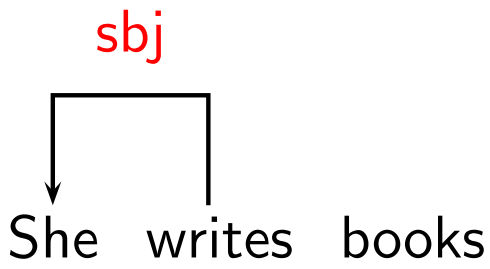
- Direct encoding of predicate-argument structure

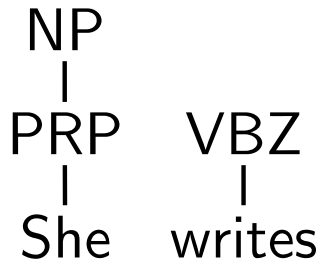


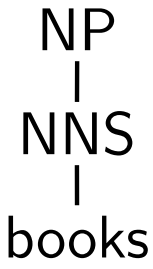


Transparency

- Direct encoding of predicate-argument structure
- Fragments directly interpretable

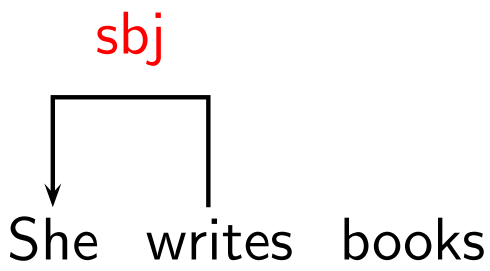






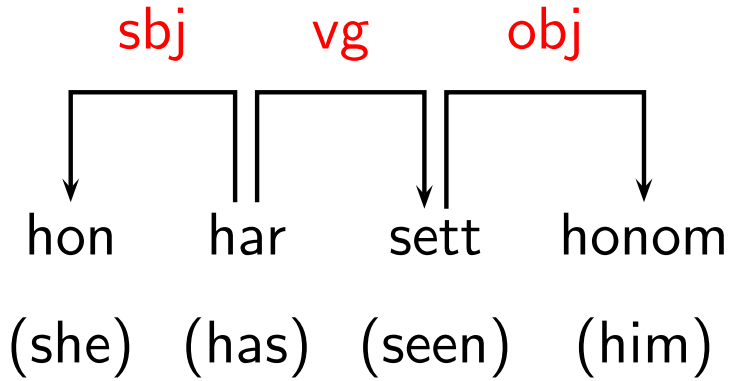
Transparency

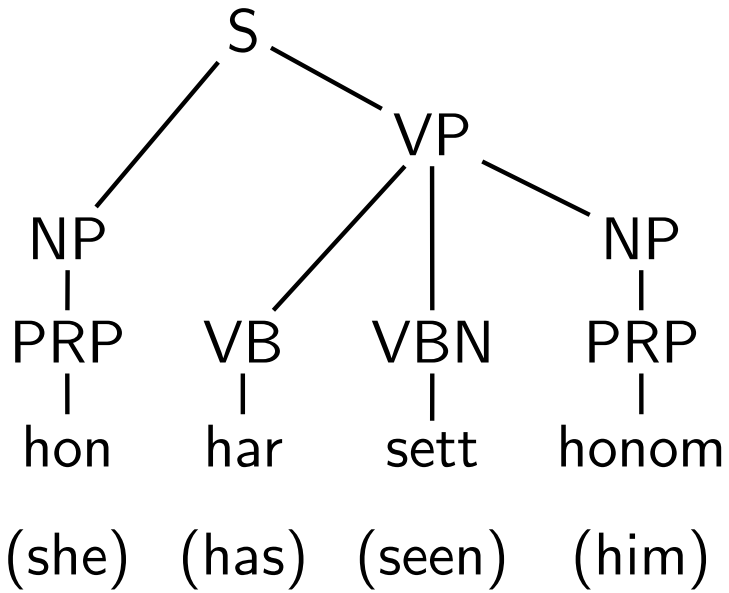
- Direct encoding of predicate-argument structure
- Fragments directly interpretable
- But only with labeled dependency graphs



Word Order

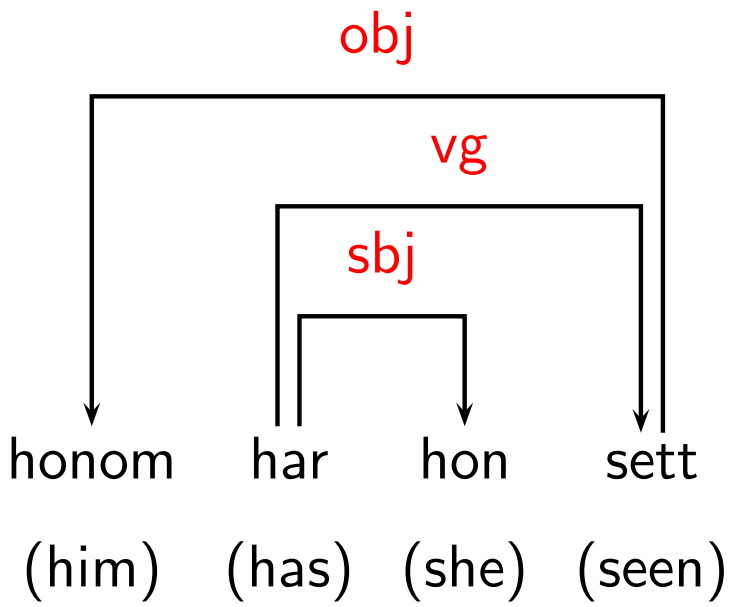
- Dependency structure independent of word order
- Suitable for free word order languages

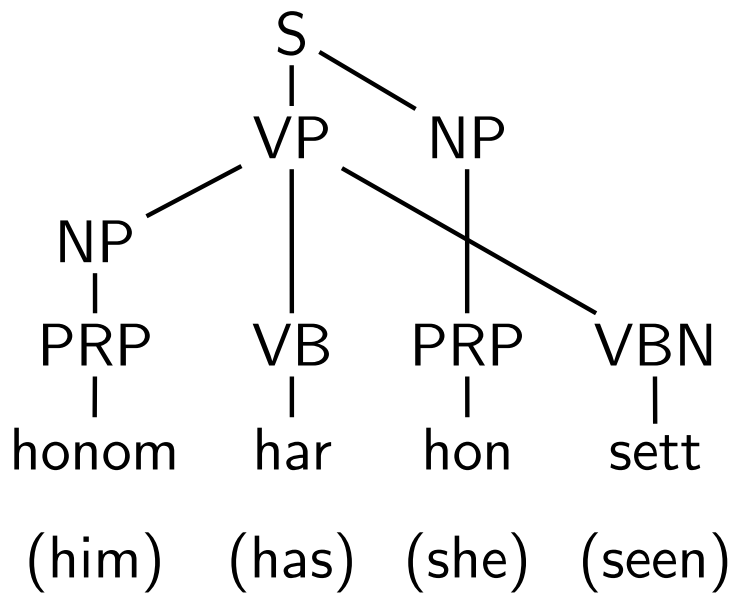




Word Order

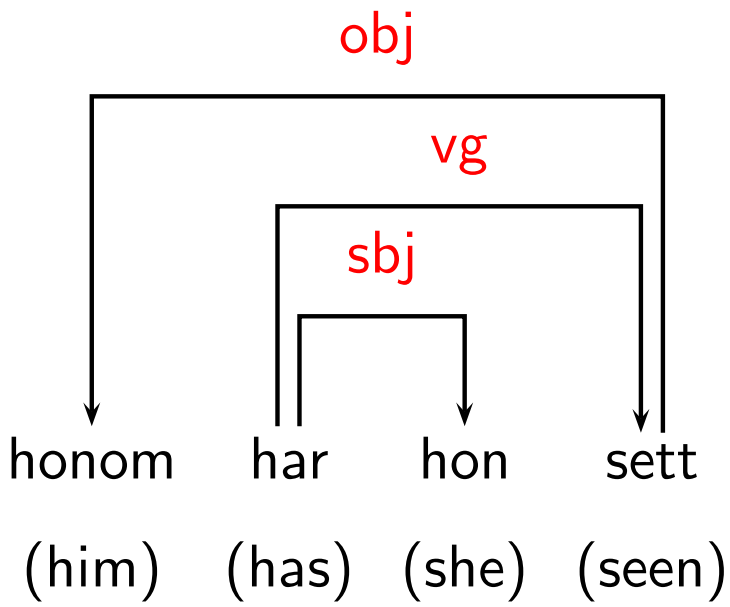
- Dependency structure independent of word order
- Suitable for free word order languages

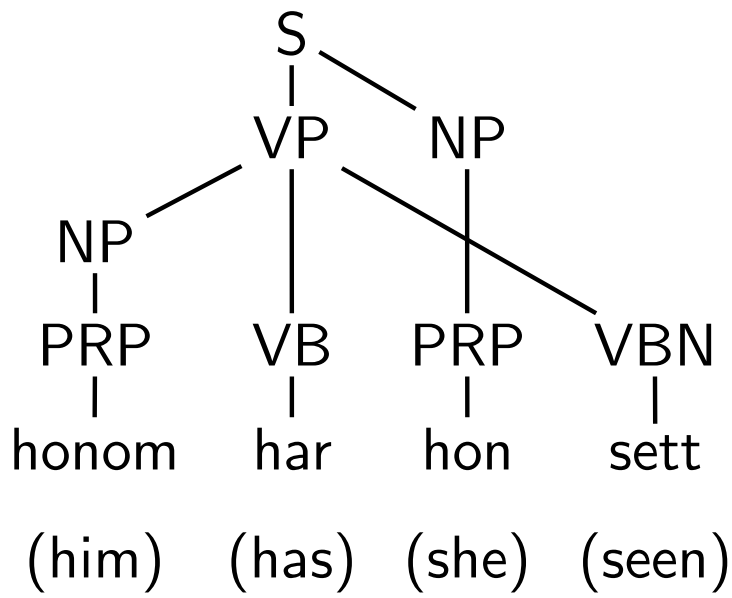




Word Order

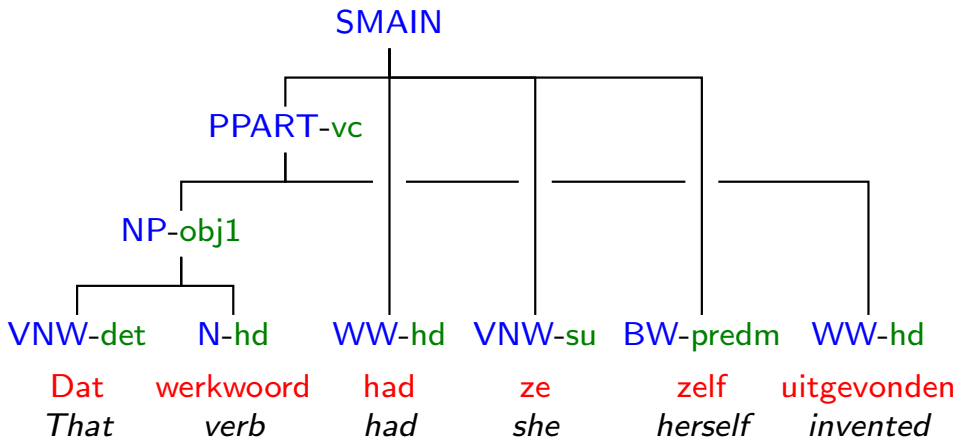
- Dependency structure independent of word order
- Suitable for free word order languages
- But only with non-projective dependency graphs

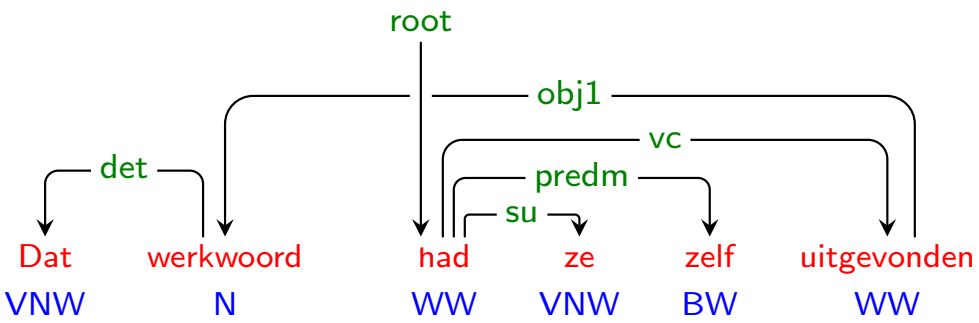




Discontinuous Constituents and Non-Projective Dependencies

Close connection between
Discontinuous Constituents and Non-Projective dependencies;
e.g., crossing branches.





Discontinuous Treebanks:

- German: Negra, Tiger
- Dutch: Spoken Dutch Corpus (CGN), Lassy
- English: Discontinuous Penn Treebank (conversion of PTB; Evang & Kallmeyer, IWPT 2011)

Expressivity

- Limited expressivity:
  - Every projective dependency grammar has a strongly equivalent context-free grammar, but not vice versa [Gaifman 1965].
  - Impossible to distinguish between phrase modification and head modification in unlabeled dependency structure [Mel'čuk 1988].

- What about labeled non-projective dependency structures?

I leave a fuller discussion aside for lack of space, but I claim that the syntactic content which is expressed in intermediate projections can also be expressed in a DT [dependency tree], through the use of features and arc labels.

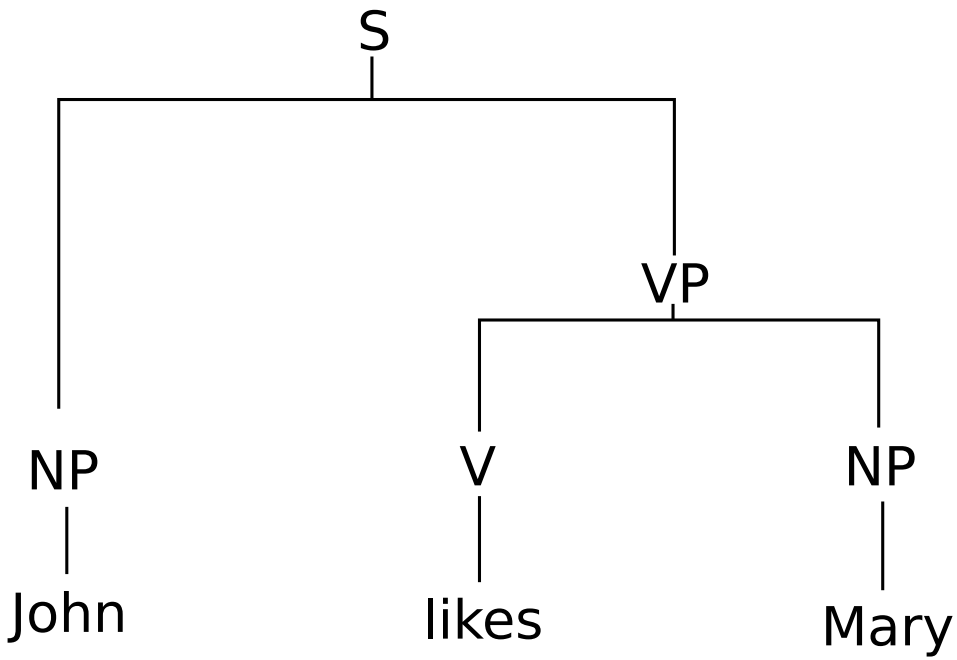
[Rambow, 2010]

Table of Contents

Constituency to dependency conversion

How to convert:

- What are the heads?
- What are the functions?

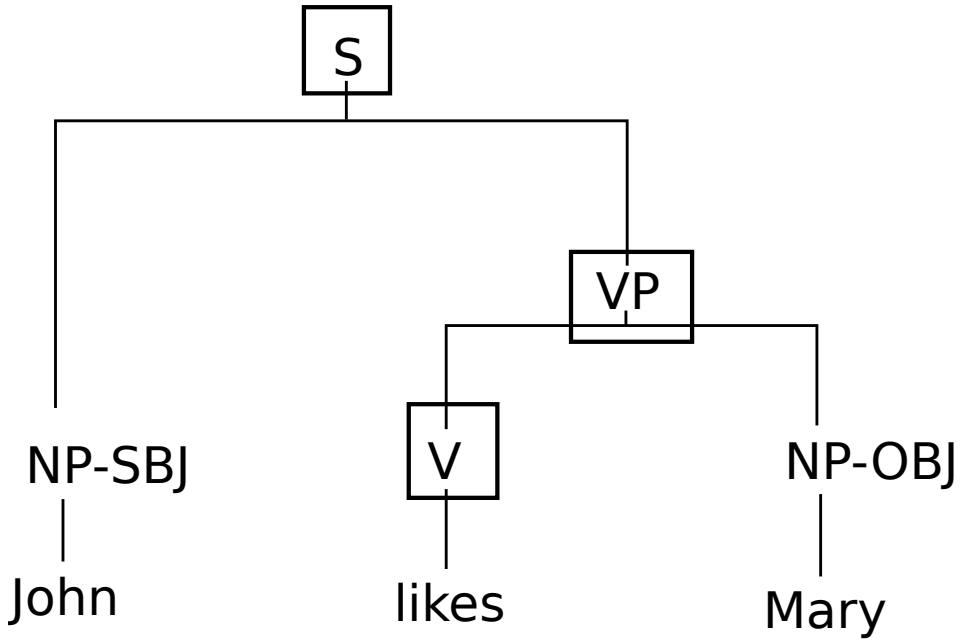


Lin (1995): A Dependency-based Method for Evaluating [. . .] Parsers.
http://ijcai.org/Proceedings/95-2/Papers/052.pdf

Constituency to dependency conversion

How to convert:

- What are the heads?
- What are the functions?
- What are the dependencies?

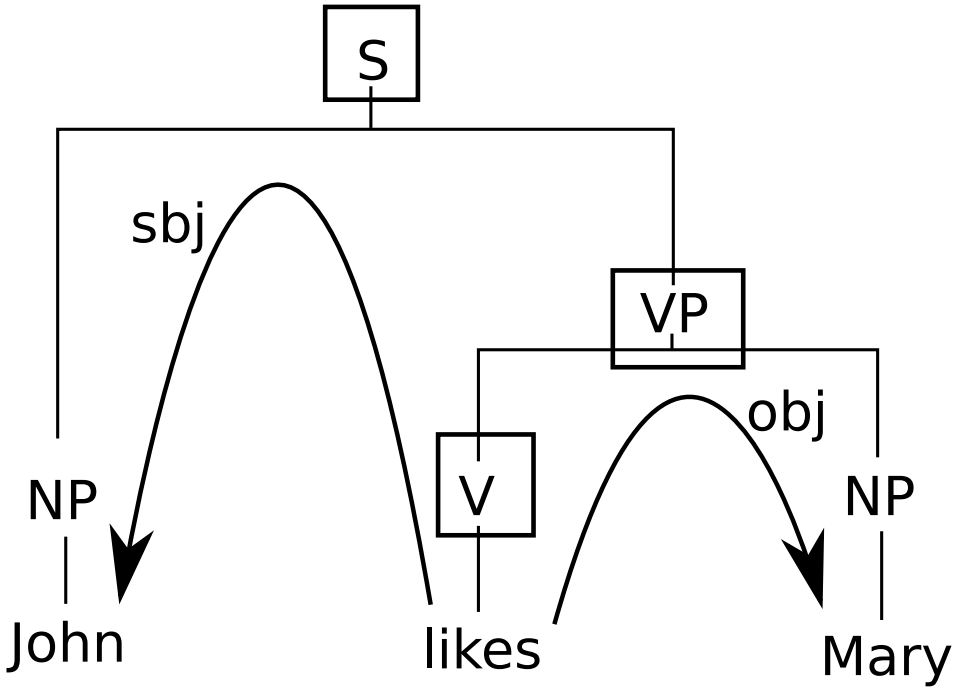


Lin (1995): A Dependency-based Method for Evaluating [...] Parsers.
http://ijcai.org/Proceedings/95-2/Papers/052.pdf

Constituency to dependency conversion

How to convert:

- What are the heads?
- What are the functions?
- What are the dependencies?



Lin (1995): A Dependency-based Method for Evaluating [. . .] Parsers.
http://ijcai.org/Proceedings/95-2/Papers/052.pdf

THE

END

Rambow, O. (2010).
The simple truth about dependency and phrase structure representations: An opinion piece.
In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 337–340, Los Angeles, California. Association for Computational Linguistics.

References and Further Reading

- Ralph Debusmann, Denys Duchier, and Geert-Jan M. Kruijff. 2004. Extensible dependency grammar: A new methodology. In *Proceedings of the Workshop on Recent Advances in Dependency Grammar*, pages 78–85.
- Denys Duchier and Ralph Debusmann. 2001. Topological dependency trees: A constraint-based account of linear precedence. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 180–187.
- Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 340–345.
- Jason M. Eisner. 2000. Bilexical grammars and their cubic-time parsing algorithms. In Harry Bunt and Anton Nijholt, editors, *Advances in Probabilistic and Other Parsing Technologies*, pages 29–62. Kluwer.
- Kilian Foth, Michael Daum, and Wolfgang Menzel. 2004. A broad-coverage parser for German based on defeasible constraints. In *Proceedings of KONVENS 2004*, pages 45–52.
- Haim Gaifman. 1965.

Dependency systems and phrase-structure systems. *Information and Control*, 8:304–337.

- Mary P. Harper and R. A. Helzerman. 1995. Extensions to constraint dependency parsing for spoken language processing. *Computer Speech and Language*, 9:187–234.
- David G. Hays. 1964. Dependency theory: A formalism and some observations. *Language*, 40:511–525.
- Peter Hellwig. 1986. Dependency unification grammar. In *Proceedings of the 11th International Conference on Computational Linguistics (COLING)*, pages 195–198.
- Peter Hellwig. 2003. Dependency unification grammar. In Vilmos Agel, Ludwig M. Eichinger, Hans-Werner Eroms, Peter Hellwig, Hans Jürgen Heringer, and Hening Lobin, editors, *Dependency and Valency*, pages 593–635. Walter de Gruyter.
- Richard A. Hudson. 1984. *Word Grammar*. Blackwell.
- Richard A. Hudson. 1990. *English Word Grammar*. Blackwell.
- Timo Järvinen and Pasi Tapanainen. 1998.

Towards an implementable dependency grammar. In Sylvain Kahane and Alain Polguère, editors, *Proceedings of the Workshop on Processing of Dependency-Based Grammars*, pages 1–10.

- Vincenzio Lombardo and Leonardo Lesmo. 1996.
  An Earley-type recognizer for dependency grammar. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 723–728.
- Hiroshi Maruyama. 1990.
  Structural disambiguation with constraint propagation. In *Proceedings of the 28th Meeting of the Association for Computational Linguistics (ACL)*, pages 31–38.
- Igor Mel'čuk. 1988.
  Dependency Syntax: Theory and Practice. State University of New York Press.
- Wolfgang Menzel and Ingo Schröder. 1998.
  Decision procedures for dependency parsing using graded constraints. In Sylvain Kahane and Alain Polguère, editors, *Proceedings of the Workshop on Processing of Dependency-Based Grammars*, pages 78–87.
- Ingo Schröder. 2002.
  Natural Language Parsing with Graded Constraints. Ph.D. thesis, Hamburg University.
- Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986.
  The Meaning of the Sentence in Its Pragmatic Aspects. Reidel.

- Daniel Sleator and Davy Temperley. 1991.
  Parsing English with a link grammar. Technical Report CMU-CS-91-196, Carnegie Mellon University, Computer Science.
- Pasi Tapanainen and Timo Järvinen. 1997.
  A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71.
- Lucien Tesnière. 1959.
  Éléments de syntaxe structurale. Editions Klincksieck.
- Wen Wang and Mary P. Harper. 2004.
  A statistical constraint dependency grammar (CDG) parser. In Frank Keller, Stephen Clark, Matthew Crocker, and Mark Steedman, editors, *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together (ACL)*, pages 42–29.
- A. M. Zwicky. 1985.
  Heads. *Journal of Linguistics*, 21:1–29.