Ling 566
Dec 4, 2025

Dependency Parsing, Final Preview

Overview

- Dependency grammar: High level overview
- Variation in dependency annotations
- Redwoods treebank
- Exporting dependencies from HPSG analyses
- MRP shared tasks
- Reading questions
- Final preview

Grammatical Dependencies

- Relate words in the sentence to each other
- A labeled with the type of dependency
- Are typically represented as graphs (sometimes trees)
  - Where each node is a word in the sentence
  - Where word in the sentence is (usually) a node

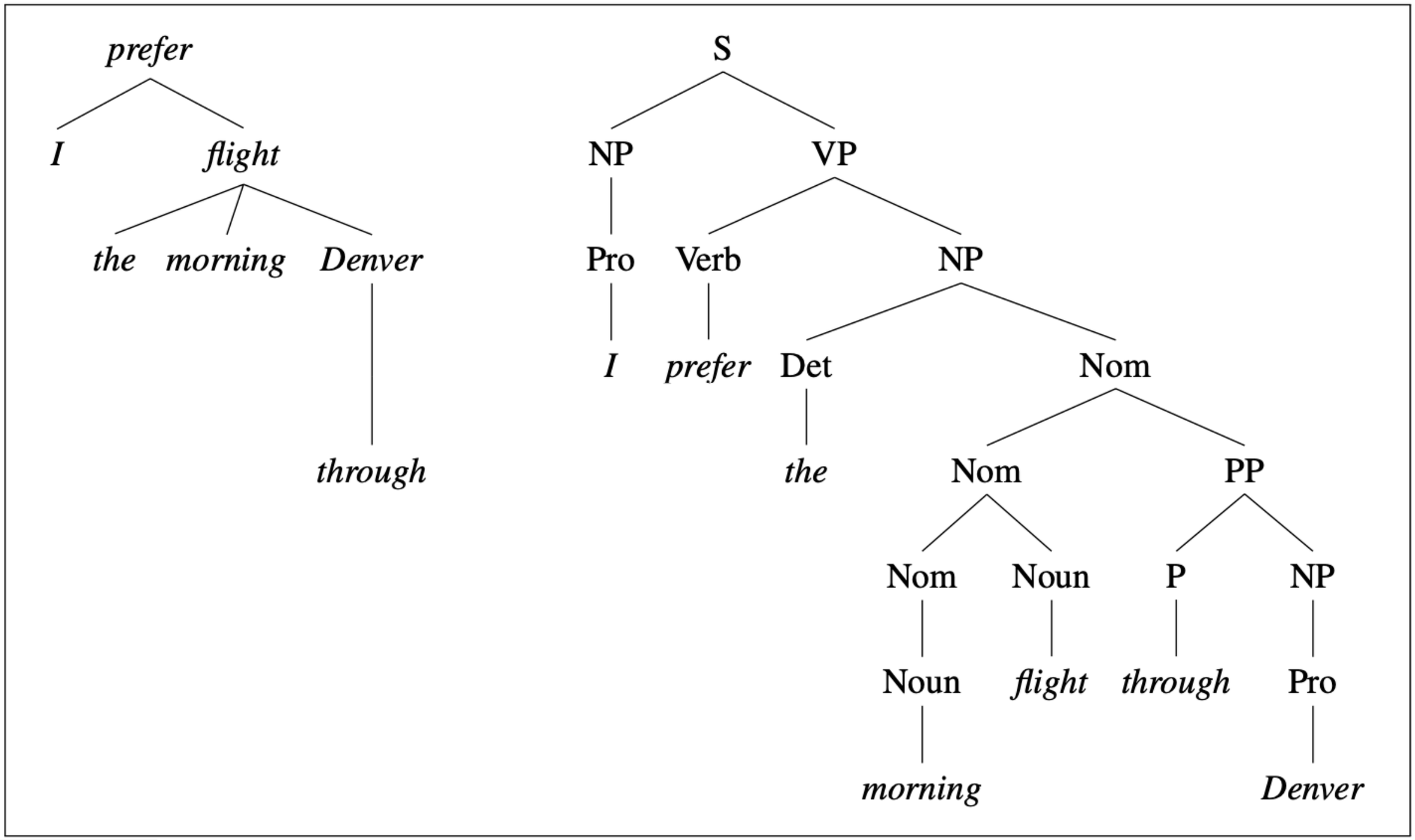
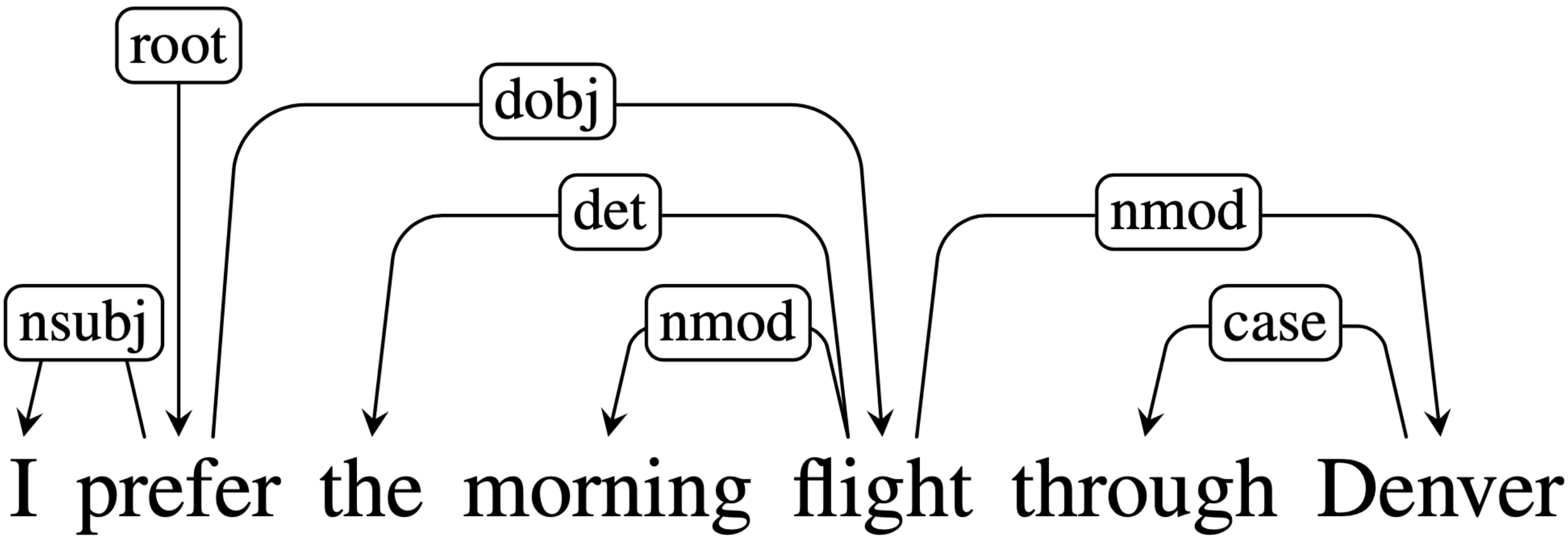


Figure 15.1 A dependency-style parse alongside the corresponding constituent-based analysis for *I prefer the morning flight through Denver.*


root
dobj
det
nmod
nsubj
nmod
case
I prefer the morning flight through Denver

Dependency Grammar

- Theoretical foundations: Tesnière 1959, Mel’čuk 1988, Hudson 1984, Sgall et al 1986

- Focus not on grammaticality (“What’s a possible sentence?”) but on grammatical structure, given a string

Dependency parsing, popularity of

- Very fast algorithms
- Dependency trees tend to be closer to semantics and therefore more useful in applications than phrase structure trees
- UD project produces really appealing datasets

Dependency Treebanks: Universal Dependencies

- https://universaldependencies.org/
- Builds on:
  - Stanford dependencies (LFG-inspired transformation of CFG representations for English from the Stanford parser)
  - Theoretical work on dependency grammar
  - “Universal” POS tagset developed initially for cross-linguistic error analysis (McDonald and Nivre 2007)

What is needed for UD to be successful? (from universaldependencies.org/introduction.html)

- The secret to understanding the design and current success of UD is to realize that the design is a very subtle compromise between approximately 6 things:
  - UD needs to be satisfactory on linguistic analysis grounds for individual languages.
  - UD needs to be good for linguistic typology, i.e., providing a suitable basis for bringing out cross-linguistic parallelism across languages and language families.
  - UD must be suitable for rapid, consistent annotation by a human annotator.
  - UD must be suitable for computer parsing with high accuracy.
  - UD must be easily comprehended and used by a non-linguist, whether a language learner or an engineer with prosaic needs for language processing. We refer to this as seeking a habitable design, and it leads us to favor traditional grammar notions and terminology.
  - UD must support well downstream language understanding tasks (relation extraction, reading comprehension, machine translation, …).
- It's easy to come up with a proposal that improves UD on one of these dimensions. The interesting and difficult part is to improve UD while remaining sensitive to all these dimensions.

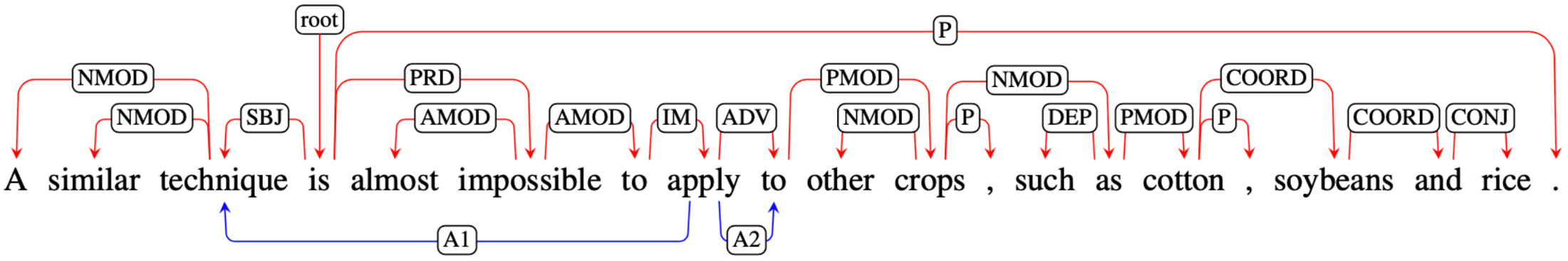
“Typical” dependency structures (quoted from Ivanova et al 2012, p.3)

- Dependency structures are directed trees: labeled, directed graphs, where the word tokens in a sentence constitute the nodes, and
  - (i) every token in the sentence is a node in the graph (combined with a designated root node, conventionally numbered as 0),
  - (ii) the graph is (weakly) connected,
  - (iii) every node in the graph has at most one head, and
  - (iv) the graph is acyclic (Nivre et al., 2007).

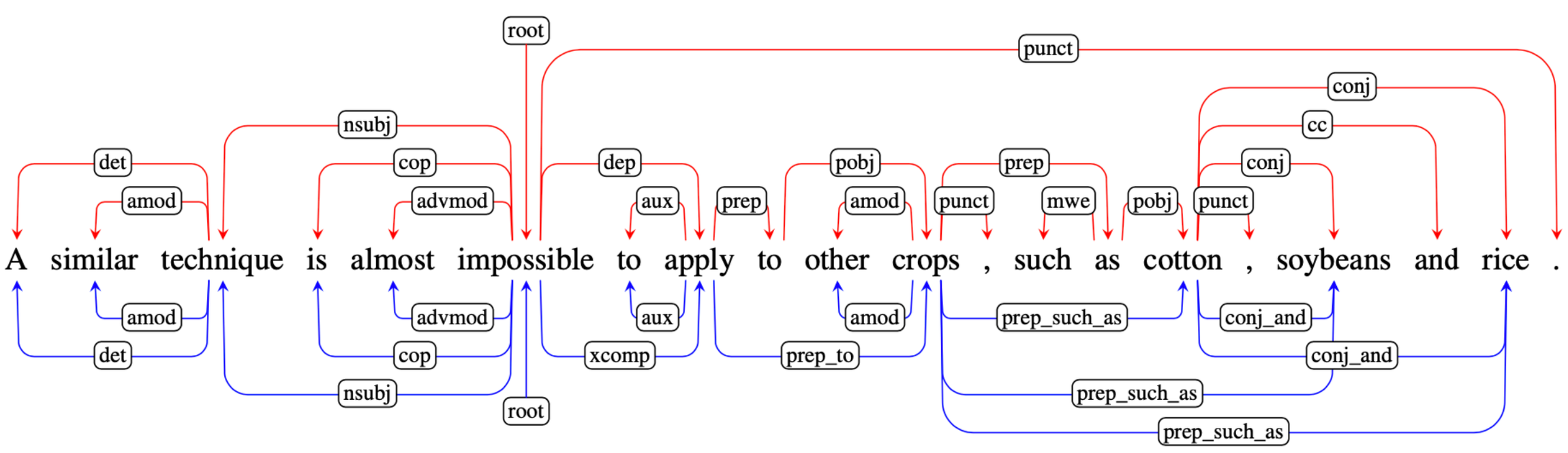
Variation in dependency structure standards

- Head selection (functional v. substantive)
- Dependency inventory
- Formal constraints on admissible graphs
- Bilexical dependencies v. allowing abstract nodes
- Are all words in the string accounted for?

Variation in dependency structure standards (Ivanova et al 2012, Fig 3)

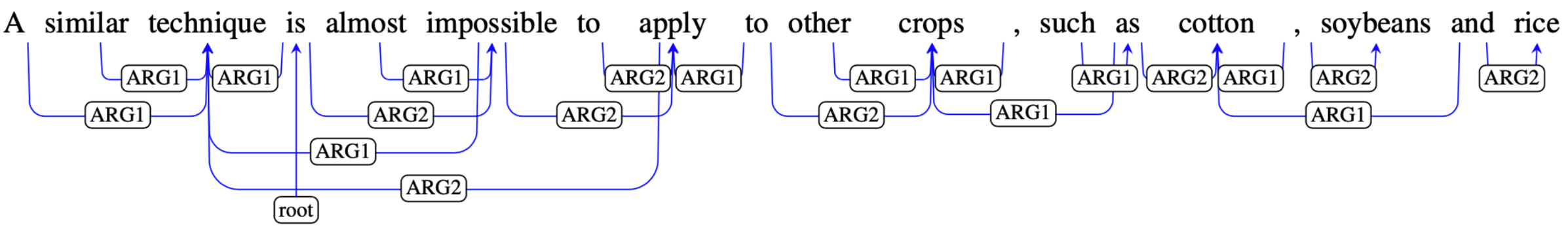


(a) CoNLL 2008 *syntactic dependencies* (CD; top) and *propositional semantics* (CP; bottom).

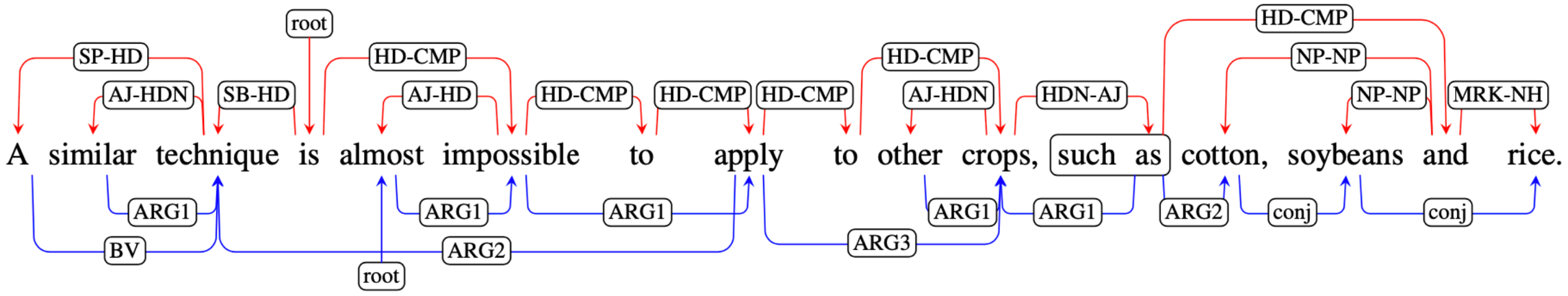


(b) Stanford Dependencies, in the so-called *basic* (SB; top) and *collapsed & propagated* (SD; bottom) variants.

Variation in dependency structure standards (Ivanova et al 2012, Fig 3)



(c) Enju *predicate-argument structures* (EP).



(d) DELPH-IN *syntactic derivation tree* (DT; top) and *Minimal Recursion Semantics* (DM; bottom).

Figure 3: Dependency representations in (a) CoNLL, (b) Stanford, (c) Enju, and (d) DELPH-IN formats.

RQ: Why do so many schemes put the root at *impossible*? Or *almost*?

Overview

- Dependency grammar: High level overview
- Variation in dependency annotations
- Redwoods treebank
- Exporting dependencies from HPSG analyses
- MRP shared tasks
- Reading questions
- Final preview

Flickinger et al 2017: Central claims

- Developing complex linguistic annotations calls for an approach which allows for the incremental improvement of existing annotations by encoding all manual effort in such a way that its value is preserved and enhanced even as the resource is improved over time

- Manual effort:
  - Annotation design => Encode in a grammar
  - Disambiguation => Store disambiguation decisions in a treebank

Redwoods Treebank (Oepen et al 2004)

- Under development since 2001
- As of ‘ninth growth’, 1.5 million tokens
- Initial motivation: train parse ranking models
- Also quite useful for grammar maintenance and development

Redwoods: Contents

- Rich syntactico-semantic structures, from which different ‘views’ can be projected.

- As illustrated in the following figures from Flickinger et al 2017

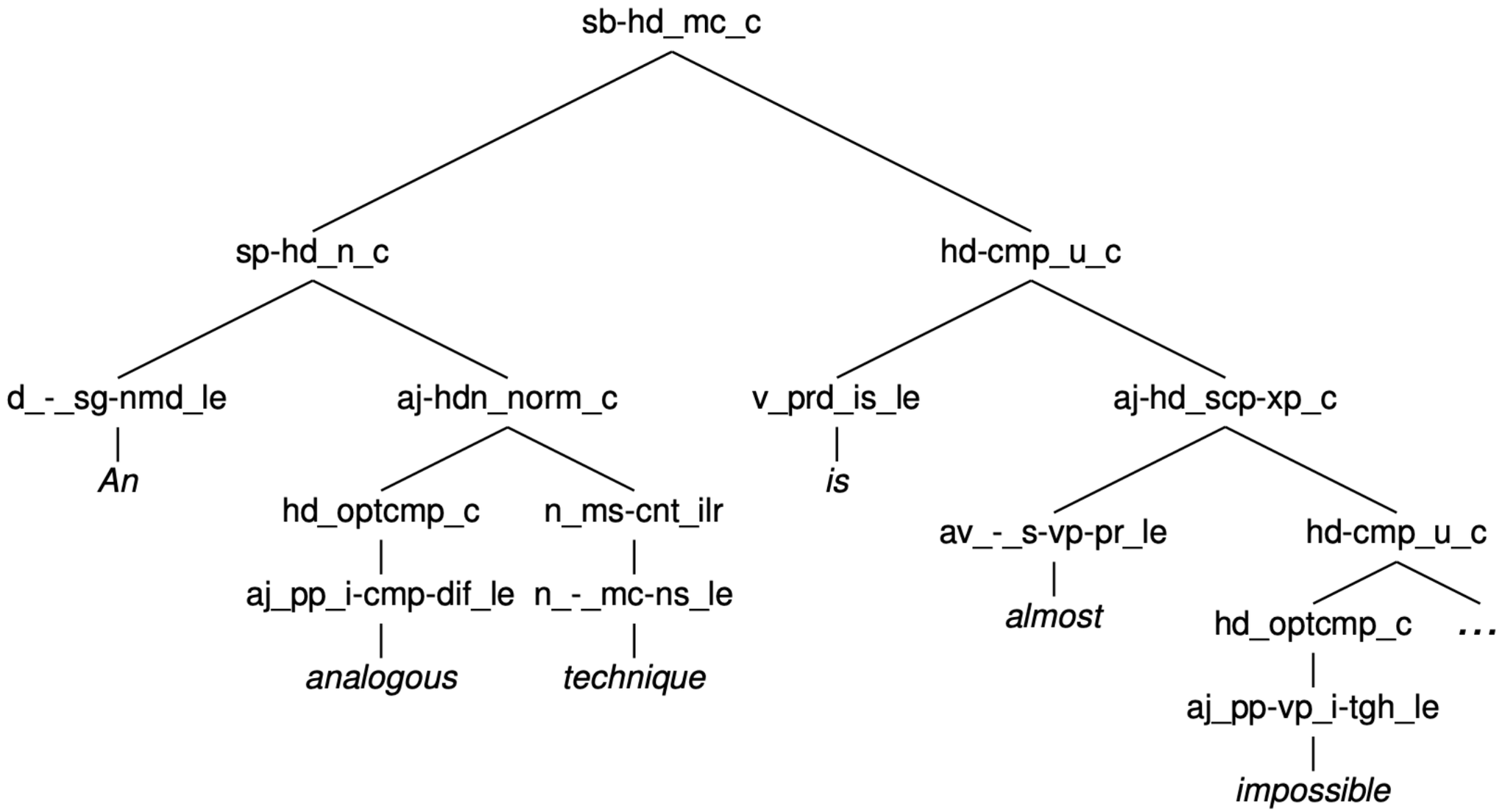
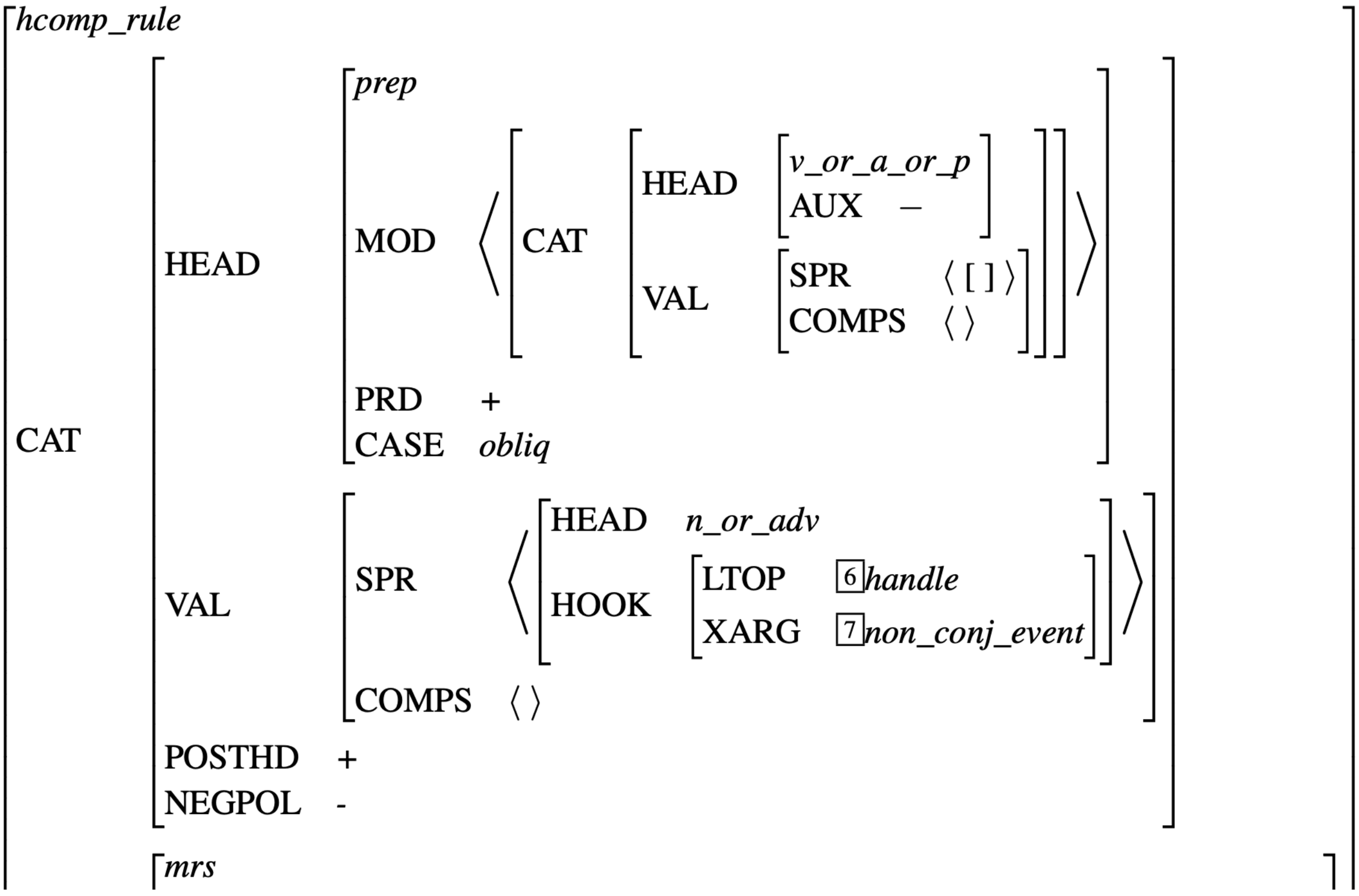


Fig. 1 ERG derivation tree for example (1).

This snippet from the ERG has non-branching nodes. Aren't we trying to avoid this?


hcomp_rule
CAT
HEAD
prep
MOD
CAT
HEAD
v_or_a_or_p
AUX –
VAL
SPR ⟨ [ ] ⟩
COMPS ⟨ ⟩
PRD +
CASE obliq
VAL
SPR
HEAD n_or_adv
HOOK
LTOP 6 handle
XARG 7 non_conj_event
COMPS ⟨ ⟩
POSTHD +
NEGPOL -
mrs

mrs

HOOK
- LTOP 6
- INDEX 7
- XARG 1

CONT
- RELS ⟨
  - *prep_relation*
    - PRED *_to_p_rel*
    - LBL 6
    - ARG0 7
    - ARG1 1
    - ARG2 13 [*nonconj_ref-ind*, PNG.PN *3p*]
  - *adj_relation*
    - PRED *_other_a_1_rel*
    - LBL 18 *handle*
    - ARG0 [*non_conj_event*, TENSE *no_tense*]
    - ARG1 13
  - *quant_or_wh_relation*
    - PRED *udef_q_rel*
    - LBL *handle*
    - BV 13
    - RSTR 16 *handle*
    - BODY *handle*
  - *reg_nom_relation*
    - PRED *_crop_n_1_rel*
    - LBL 18
    - ARG0 13

⟩

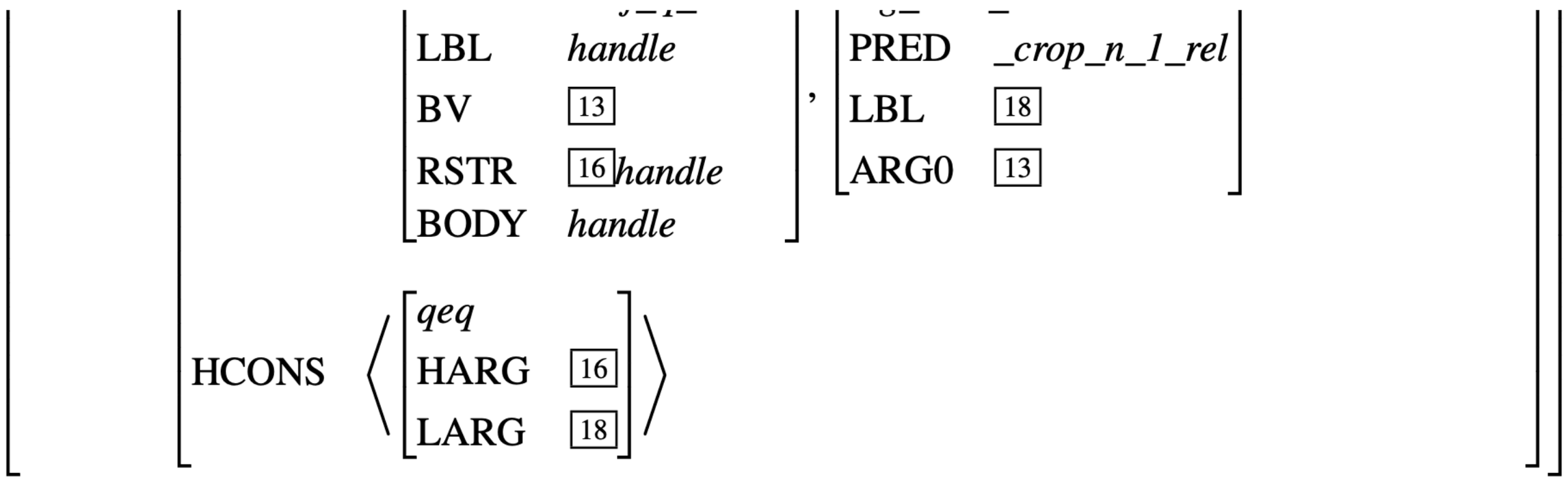


Fig. 2 Partial feature structure for PP *to other crops*

$$
\langle h_1,
\left|\begin{array}{l}
h_4{:}\_a\_q(\text{BV } x_6, \text{RSTR } h_7, \text{BODY } h_5),\\
h_8{:}\_analogous\_a\_to(\text{ARG0 } e_9, \text{ARG1 } x_6), h_8{:}comp(\text{ARG0 } e_{11}, \text{ARG1 } e_9, \text{ARG2 } \_),\\
h_8{:}\_technique\_n\_1(\text{ARG0 } x_6),\\
h_2{:}\_almost\_a\_1(\text{ARG0 } e_{12}, \text{ARG1 } h_{13}), h_{14}{:}\_impossible\_a\_for(\text{ARG0 } e_3, \text{ARG1 } h_{15}, \text{ARG2 } \_),\\
h_{17}{:}\_apply\_v\_to(\text{ARG0 } e_{18}, \text{ARG1 } \_, \text{ARG2 } x_6, \text{ARG3 } x_{20}),\\
h_{21}{:}udef\_q(\text{BV } x_{20}, \text{RSTR } h_{22}, \text{BODY } h_{23}), h_{24}{:}\_other\_a\_1(\text{ARG0 } e_{25}, \text{ARG1 } x_{20}),\\
h_{24}{:}\_crop\_n\_1(\text{ARG0 } x_{20})
\end{array}\right|\\
\{h_1 =_q h_2, h_7 =_q h_8, h_{13} =_q h_{14}, h_{15} =_q h_{17}, h_{22} =_q h_{24}\}\rangle
$$

Fig. 3 Minimal Recursion Semantics for example (1).

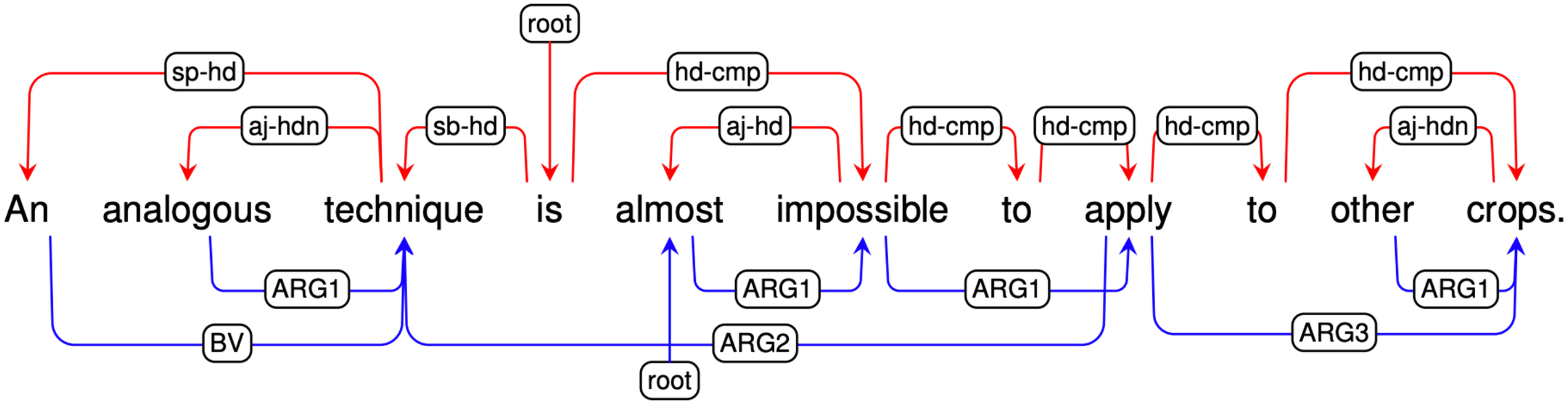


Fig. 4 Bi-lexical syntactic and semantic dependencies for (1).

Redwoods: Methodology

- Parse input corpus
- Calculate ‘discriminants’: properties shared by only a subset of the trees in parse forest (Carter 1997)
    - Picking one tree from among thousands or millions would be infeasible
    - Drawing trees with that level of detail would be infeasible
    - Picking discriminants is quite doable!
- Store both resulting tree & discriminants chosen (and inferred)
    - Maximum value out of all human annotator time

Very high inter-annotator agreement
=> very consistent annotation

- From Bender et al 2015, over 150 sentences from *The Little Prince*

	Annotator Comparison			
Metric	**A vs. B**	**A vs. C**	**B vs. C**	**Average**
Exact Match	0.73	0.65	0.70	0.70
EDM_a	0.93	0.92	0.94	0.93
EDM_{na}	0.94	0.94	0.95	0.94

Table 1: Exact match ERS and Elementary Dependency Match across three annotators.

- Comparable metric for AMR over the same data is 0.71 “SMATCH” (comparable to EDM) (Banarescu et al 2013)

Dynamic treebanking

- Dynamic *refinement* of the treebank
    - Parse corpus with new grammar (better coverage, improved representations)
    - Rerun discriminants chosen in previous annotation rounds
    - Address remaining added ambiguity / newly parsed sentences
- Dynamic *extension* of the treebank
    - Linguistic analysis encoded as a grammar (as opposed to annotation guidelines) can be automatically deployed to new text

Automatic conversion: Syntactic dependencies (Ivanova et al 2012)

- Remove (collapse) non-branching rules
- Head daughter is marked in (most?) ERG rules
- Dependencies labels come from valence features

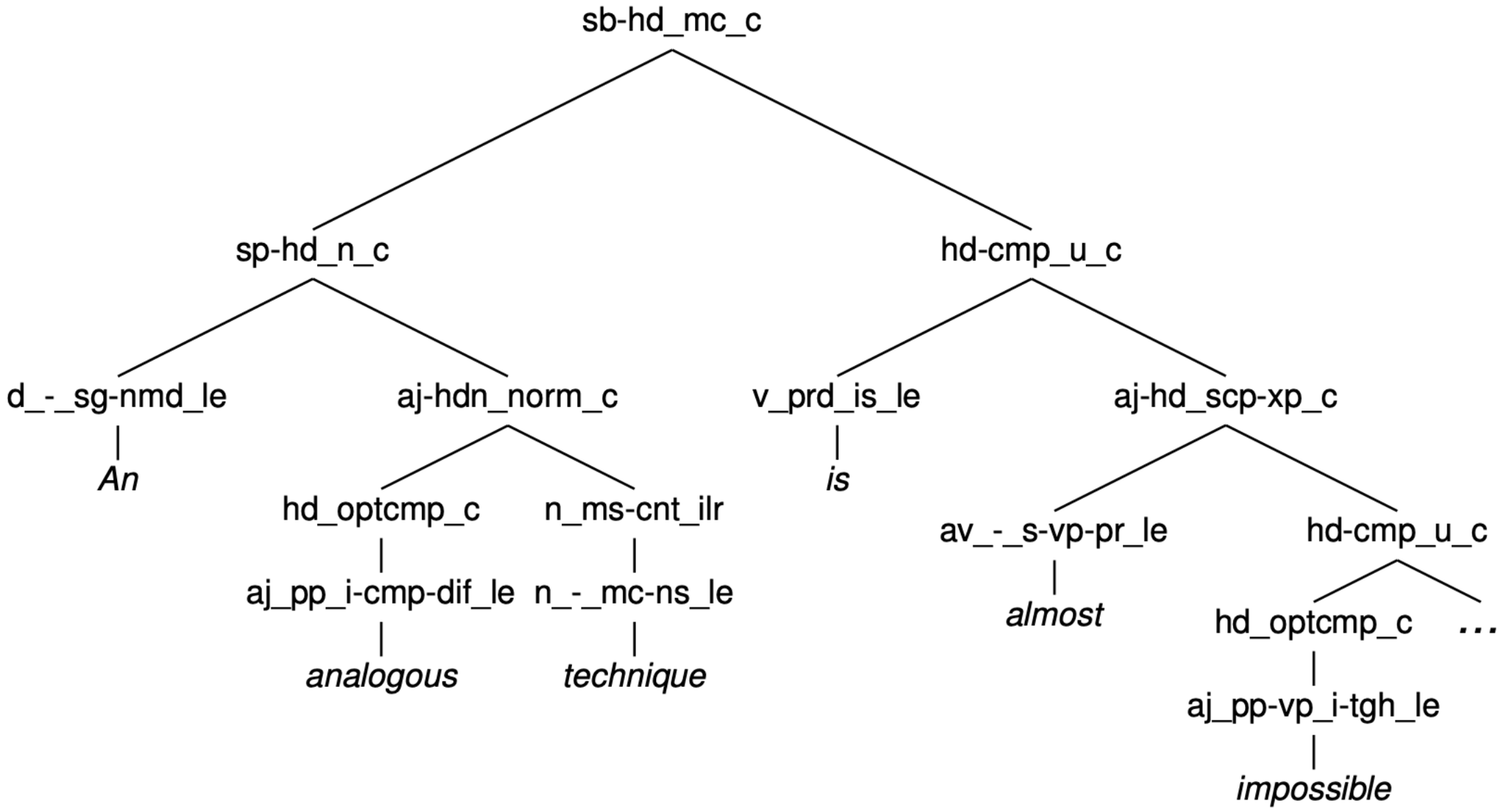


Fig. 1 ERG derivation tree for example (1).

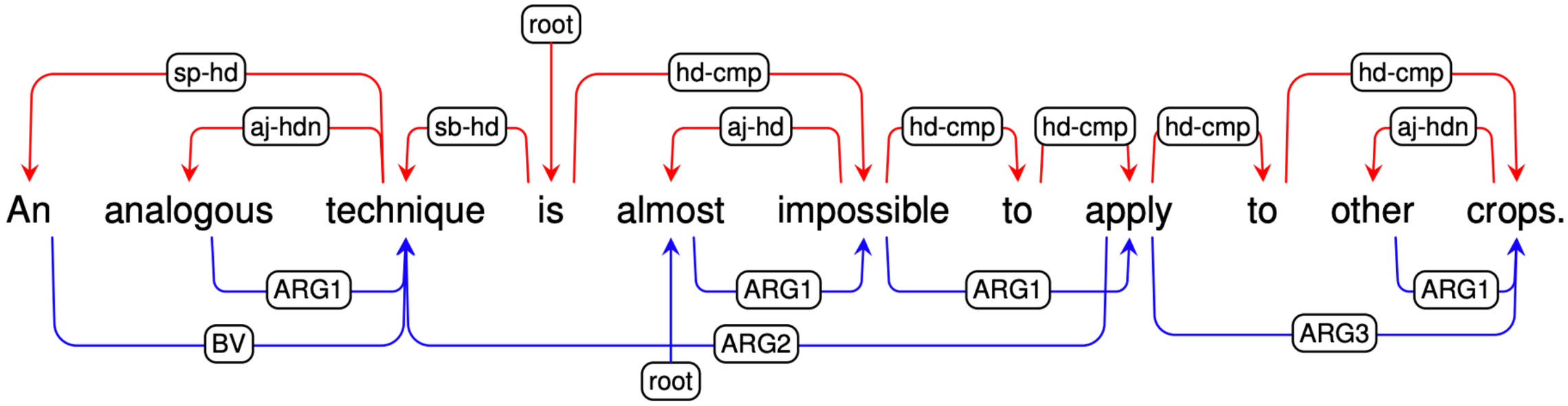


Fig. 4 Bi-lexical syntactic and semantic dependencies for (1).

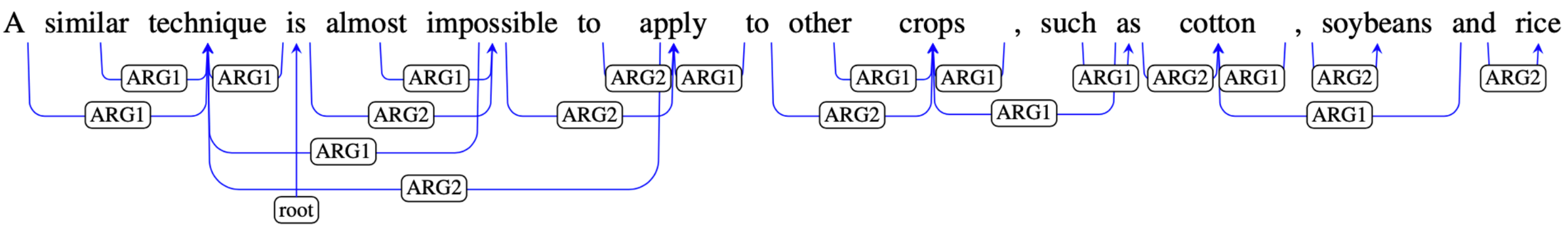
Automatic conversion: Semantic dependencies (Ivanova et al 2012)

- Lossy conversion of MRS to EDS (elementary dependency structures, Oepen & Lønning 2006)
- Regular predicates: ARGn dependency between tokens
- Transparent and redundant predicates: remove
- Relational predicates: predicate is dependency label

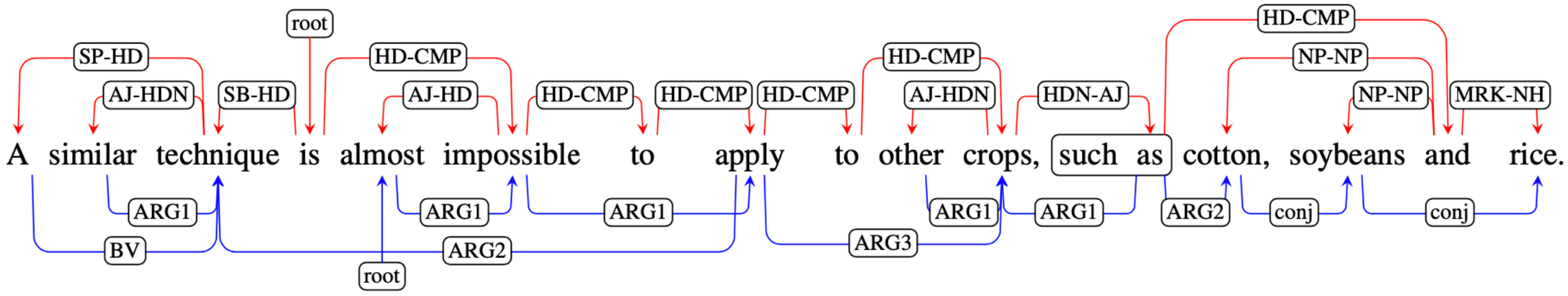
{ e_{12}
  $\_1$:_a_q(BV x_6)
  e_9:_similar_a_to(ARG1 x_6)
  x_6:_technique_n_1
  e_{12}:_almost_a_1(ARG1 e_3)
  e_3:_impossible_a_for(ARG1 e_{18})
  e_{18}:_apply_v_to(ARG2 x_6, ARG3 x_{19})
  $\_2$:udef_q(BV x_{19})
  e_{25}:_other_a_1(ARG1 x_{19})
  x_{19}:_crop_n_1
  e_{26}:_such+as_p(ARG1 x_{19}, ARG2 x_{27})
  $\_3$:udef_q(BV x_{27})
  $\_4$:udef_q(BV x_{33})
  x_{33}:_cotton_n_1
  $\_5$:udef_q(BV i_{38})
  x_{27}:implicit_conj(L-INDEX x_{33}, R-INDEX i_{38})
  $\_6$:udef_q(BV x_{43})
  x_{43}:_soybeans/nns_u_unknown
  i_{38}:_and_c(L-INDEX x_{43}, R-INDEX x_{47})
  $\_7$:udef_q(BV x_{47})
  x_{47}:_rice_n_1
}

Figure 2: ERG Elementary Dependency Structure.

Variation in dependency structure standards (Ivanova et al 2012, Fig 3)



(c) Enju *predicate-argument structures* (EP).



(d) DELPH-IN *syntactic derivation tree* (DT; top) and *Minimal Recursion Semantics* (DM; bottom).

Figure 3: Dependency representations in (a) CoNLL, (b) Stanford, (c) Enju, and (d) DELPH-IN formats.

MRP (meaning representation parsing) shared tasks

- http://mrp.nlpl.eu/2020/index.php

RQs: Bilexical dependencies

- This paper mentions that its focus is on bilexical dependencies. What is an example of a bilexical dependency? What is an example of a non-bilexical dependency? Why are bilexical dependency representations desirable?

RQs: Dependency treebank construction

- I was curious if there is a reason that most banks of dependency grammars are made by converting tree banks into dependency form and not any original effort to make a dependency bank.

- Also is there another reason that most dependencies graphs are acyclic other than the fact that it makes them much easier to compute over and we already have fast preexisting algorithms to deal with acyclic graphs.

RQs: Use cases

- Are syntactically-based dependency parses stronger in some use cases than semantically motivated ones, and vice versa?

RQs: Non-projective trees

- I wonder if the converter from HPSG to syntactic dependencies allows for any non-projective parses (e.g. for gapped elements).

RQs: Special relation types

- Could you expand upon the 'special' classes of relations (transparent, relational, redundant) and what their purpose is?

Besides this basic mechanism, our converter supports three 'special' classes of relations, which we call (a) *transparent*, (b) *relational*, and (c) *redundant*. The latter class is of a more technical nature and avoids duplicates in cases where the EDS gave rise to multiple dependencies that only differ in their label (and where labels are considered equivalent), as can at times be the case in coordinate structures.[9]

Our class of so-called *transparent* relations includes the semantic relation associated with, for example, nominalization, where in the underlying logic a referential instance variable is explicitly derived from an event. In terms of bilexical dependencies, however, we want to conceptually equate the two EDS nodes involved. In our running example, in fact, coordination provides an example of transparency: in the EDS, there are two binary conjunction relations (implicit_conj and _and_c), which conceptually correspond to group formation; node i_{38} (corresponding to *and*) is the second argument of the implicit conjunction. For our semantic bilexical dependencies, however, we opt for the analysis of Mel'čuk (see Section 3 above), which we achieve by making interchangeable conjunction nodes with their left arguments, i.e. nodes i_{38} and x_{43}, as well as x_{27} and x_{33}, in Figure 2.

Finally, somewhat similar to the 'collapsing' available in Stanford Dependencies, our class of so-called *relational* predicates allows the creation of dependency labels transcending EDS role indices, which we apply for, among others, possession, subordination, apposition, and conjunction. The two conj dependencies in Figure 3d, for example, hold between left and right arguments of the two conjunctions, as per the excerpt from the ERG-specific conversion specification shown in Figure 4.[10].

```
[transparent]
implicit_conj L-INDEX
/_c$/ L-INDEX
[relational]
/_c$/ conj L-INDEX R-INDEX
implicit_conj conj L-INDEX R-INDEX
```

Figure 4: Excerpt from the ERG configuration file.

(Ivanova et al 2012:9)

Overview

- Dependency grammar: High level overview
- Variation in dependency annotations
- Redwoods treebank
- Exporting dependencies from HPSG analyses
- MRP shared tasks
- Reading questions
- Final preview