Dependency Parsing exercises: Dependency Structures Part II

Deadline: 26.04.2021

Please send completed solutions to waszczuk@hhu.de and evang@hhu.de with subject "dependency homework" and attachment "ex2_lastname(s).pdf".

1. Consider the following unlabeled[1] dependency graphs, all defined on top of $V = \{1, 2, 3, 4, 5\}$:

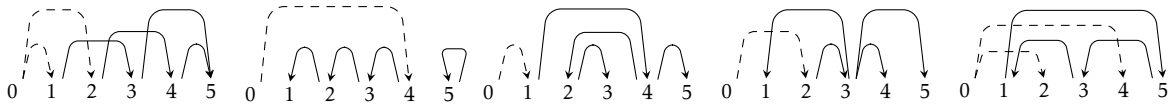
   (a) $A = \{(1, 3), (2, 4), (3, 5), (4, 5)\}$
   (b) $A = \{(5, 5), (4, 3), (3, 2), (2, 1)\}$
   (c) $A = \{(1, 4), (2, 3), (4, 2), (4, 5)\}$
   (d) $A = \{(3, 1), (2, 3), (3, 4), (3, 5)\}$
   (e) $A = \{(5, 3), (3, 1), (5, 3), (1, 5)\}$

   Assume there is an implicit *special root node* 0 and an implicit arc $(0, i)$ for every node i that doesn't have its head specified in A.

   For each graph (a)-(e), determine if it is (i) *connected*, (ii) *acyclic*, (iii) *projective*, (iv) and whether it obeys the *single-head* constraint.

   Hint: draw the graphs when in doubt.

   Graphs:

   (a) (b) (c) (d) (e)

   Solution:

Graph	*connected*	*acyclic*	*projective*	*single-head*
(a)	✓	✓	x	x
(b)	x	x	✓	✓
(c)	✓	✓	✓	✓
(d)	✓	✓	x	✓
(e)	x	x	x	✓

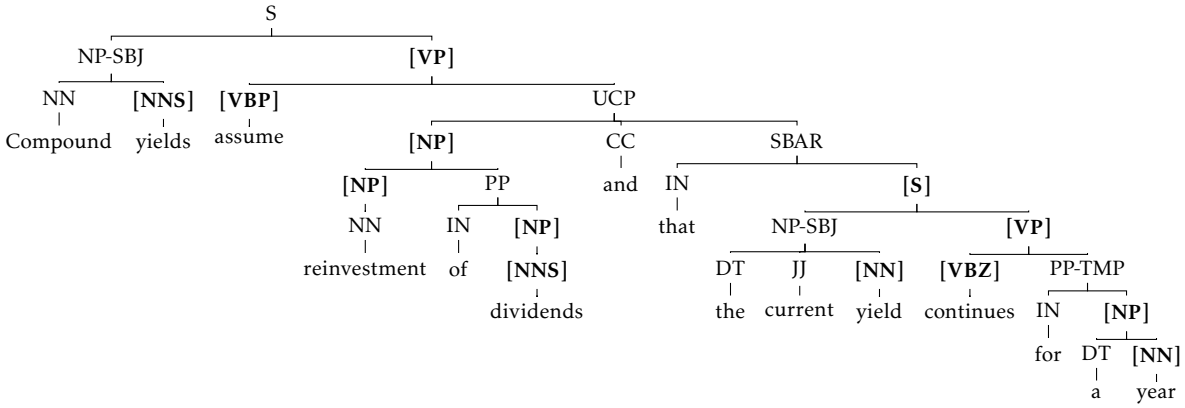
2. Convert the constituency analysis (following the PTB annotation scheme) into a UD-compliant dependency analysis:

   - Mark one child as the head in each constituent. Assume the following set of heuristic head-percolation rules (a slightly clarified version of the rules used in the example):

[1] In a labeled dependency graph, each element of A is a triple (i, j, k) where k determines the label; in an unlabeled graph, each arc can be simply represented as a pair $(i, j) \in A$.

- The head of S or VP is its rightmost (main verb or another VP)
- The head of an NP is its rightmost (noun or another NP)
- The head of PP is its rightmost daughter

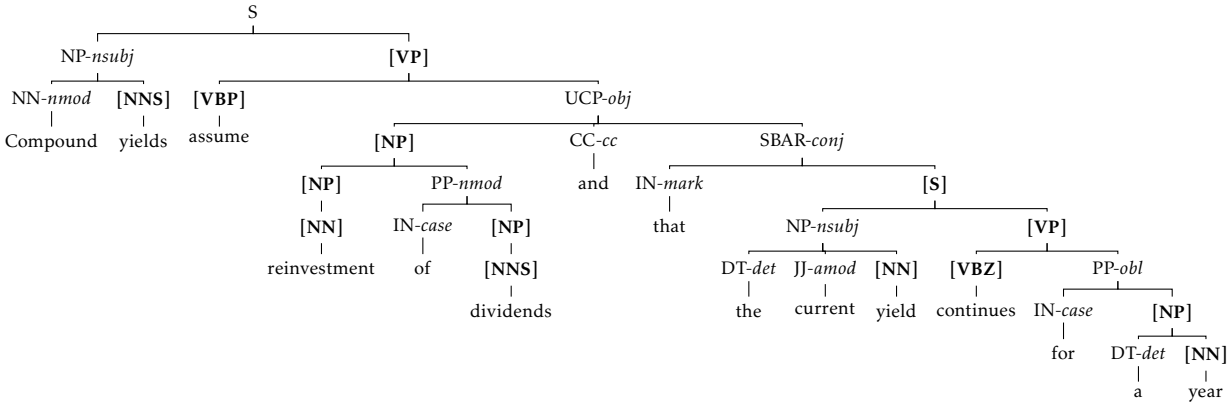
Are these rules sufficient to obtain a UD-compliant tree? If not, propose possible improvements.



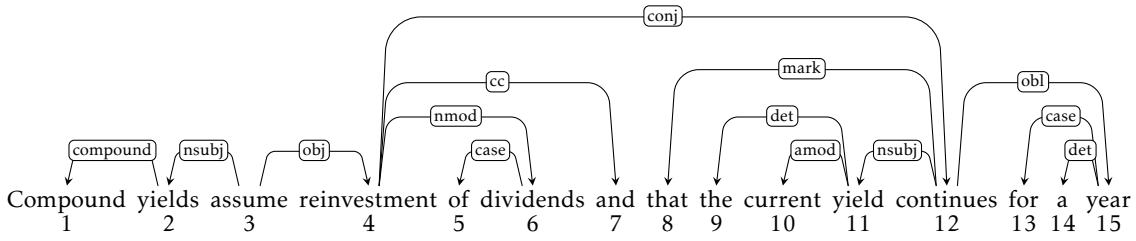
Additional head-percolation rules used:

- The head of SBAR is its rightmost S daughter
- The head of UCP is its leftmost daughter
- If there is one daughter only, it is the head

- Determine the implied function labels on case-by-case basis using the inventory of UD relations.



- Perform the last, fully automatic step of the conversion.



Note: it is not possible to connect the second conjunct (*continuous*) to the preceding coordinating conjunction (*and*), as required in UD, by simply adding or modifying the head-percolation rules. That would require some additional post-processing.
Update 27.04: changed the relation between *compound* and *yields* from *nmod* to *compound*; also see the relevant part of the guidelines.

3. Linguistic phenomena and properties of dependency graphs.

   (a) Give an example of a sentence with non-projective (universal) dependencies. What is the linguistic phenomenon that gives rise to the non-projectivity?

   - See here for examples. If the link doesn't work, go to the PML Tree Query website, pick the treebank/language of your choice, and use the following query to find dependency trees with non-projective dependencies of length ≤ 15:

```
a-root [
  15-x descendant a-node [],
  descendant a-node $p :=
    [
      child a-node $c := [],
      same-tree-as a-node $x :=
        [!ancestor $p
        , (order-follows $p and order-precedes $c) or
          (order-follows $c and order-precedes $p)
        ]
    ]
]
```

   (b) Give a dependency analysis of the same sentence in another language (again with universal dependencies), and see if the non-projectivity remains. Add glosses to the non-English words.

   (c) Can you think of linguistic phenomena that would benefit from relaxing any of the other properties: connectedness, acyclicity, or single-head constraint?

   - **Connectedness**: interjections (*ah*, *er*, *um*), punctuation
   - **Single-head constraint**: see *enhanced dependencies*