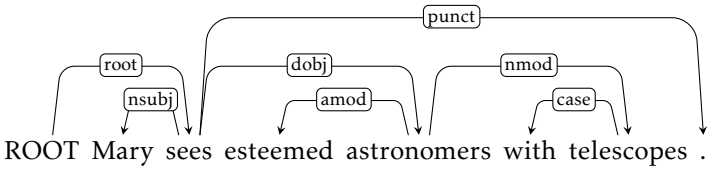
Dependency Parsing exercises: Transition-based arc-eager parsing, non-projective parsing

Deadline: 24.05.2021

Please send completed solutions to waszczuk@hhu.de and evang@hhu.de with subject "dependency homework" and attachment "ex6_lastname(s).pdf".

1. Arc-eager parsing.

   (a) Enumerate the configurations an arc-eager transition-based parser goes through when parsing the sentence:



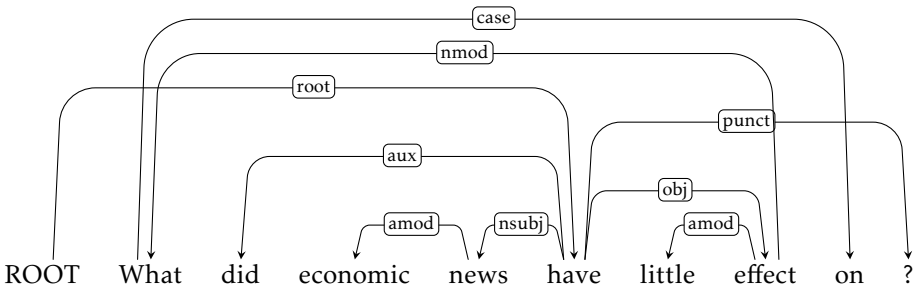
   A transition is a left-arc, right-arc, shift, or reduce operation (LA, RA, SH, RE). At each step, indicate the operation, the contents of the stack, the input buffer, and which dependency is added, if any:[1]

TRANSITION	STACK	BUFFER	ARCS
	[ROOT]	[Mary sees esteemed . . .]	$\emptyset$
SH	[ROOT Mary]	[sees esteemed astronomers . . .]	
LA_{NSUBJ}	[ROOT]	[sees esteemed astronomers . . .]	+(Mary $\overset{\text{NSUBJ}}{\leftarrow}$ sees)
. . .	. . .	. . .	. . .

   (b) How does this compare to arc-standard parsing? In particular, do you think there could be disadvantages (or advantages) for PP-attachments when using the arc-eager parser? In general, can you think of typological properties of languages that could make arc-eager perform better than arc-standard, or vice versa? Consider default word order (SVO, SOV, etc), free word order, head-final vs head-first, degree of inflection, etc.

[1] A configuration is *terminal* if its buffer is empty, just as in the arc-standard system.

2. Non-projective parsing. Consider the following dependency tree:



(a) Determine a *projective order* for the tree above. It can be obtained by traversing the sentence with an in-order traversal of the tree.[2] Then re-order the words in the sentence accordingly (i.e. place each word on its position according to the projective order) and draw the resulting dependency tree to verify that there are no crossing arcs with this word order.

(b) Propose a transition sequence that allows to reconstruct the non-projective dependency tree above using the online reordering parser. A transition is left-arc, right-arc, shift, or swap (LA, RA, SH, SW). At each step, indicate the operation, the contents of the stack, the input buffer, and which dependency is added, if any:

TRANSITION	STACK	BUFFER	ARCS
	[ROOT]	[What did economic news . . .]	∅
SH	[ROOT What]	[did economic news . . .]	
. . .	. . .	. . .	. . .

[2] See https://en.wikipedia.org/wiki/Tree_traversal#In-order.