Detaillierte Testergebnisse für Testdurchlauf 1

1. Single choice #1 [ID: 685974]

A parser that starts in an initial state and uses a machine learning model to repeatedly decide into which state to go next until the parse is complete is called…

- (●) transition-based
- ( ) graph-based
- ( ) static
- ( ) dynamic

2. Single choice #2 [ID: 685975]

A dependency tree is called *projective* if for every arc i → j and every word k between i and j it holds that

- ( ) there is an arc i → k
- (●) i dominates k
- ( ) there is an arc k → i
- ( ) k dominates i

3. Single choice #3 [ID: 685976]

A key difference between the Chu-Liu-Edmonds algorithm and the Eisner algorithm is:

- (●) Chu-Liu-Edmonds finds projective and non-projective trees; Eisner finds only projective trees.
- ( ) Chu-Liu-Edmonds is graph-based; Eisner is transition-based.
- ( ) Eisner finds graphs with and without cycles; Chu-Liu-Edmonds only finds graphs without cycles.
- ( ) Eisner is graph-based; Chu-Liu-Edmonds is transition-based.

4. Single choice #4 [ID: 685977]

The arc-eager transition system has a REDUCE transition, which the arc-standard transition system does not. Why does it need this additional transition?

- (●) The RIGHT-ARC transition does not remove the child of the new arc from the parser's data structures.
- ( ) The LEFT-ARC transition does not remove the child of the new arc from the parser's data structures.
- ( ) Neither the LEFT-ARC nor the RIGHT-ARC transition remove the child of the new arc from the parser's data structures.
- ( ) REDUCE allows the parser to remove certain words from the stack earlier, resulting in faster parsing speeds.

5. Single choice #5 [ID: 685978]

Consider the following two example sentences:

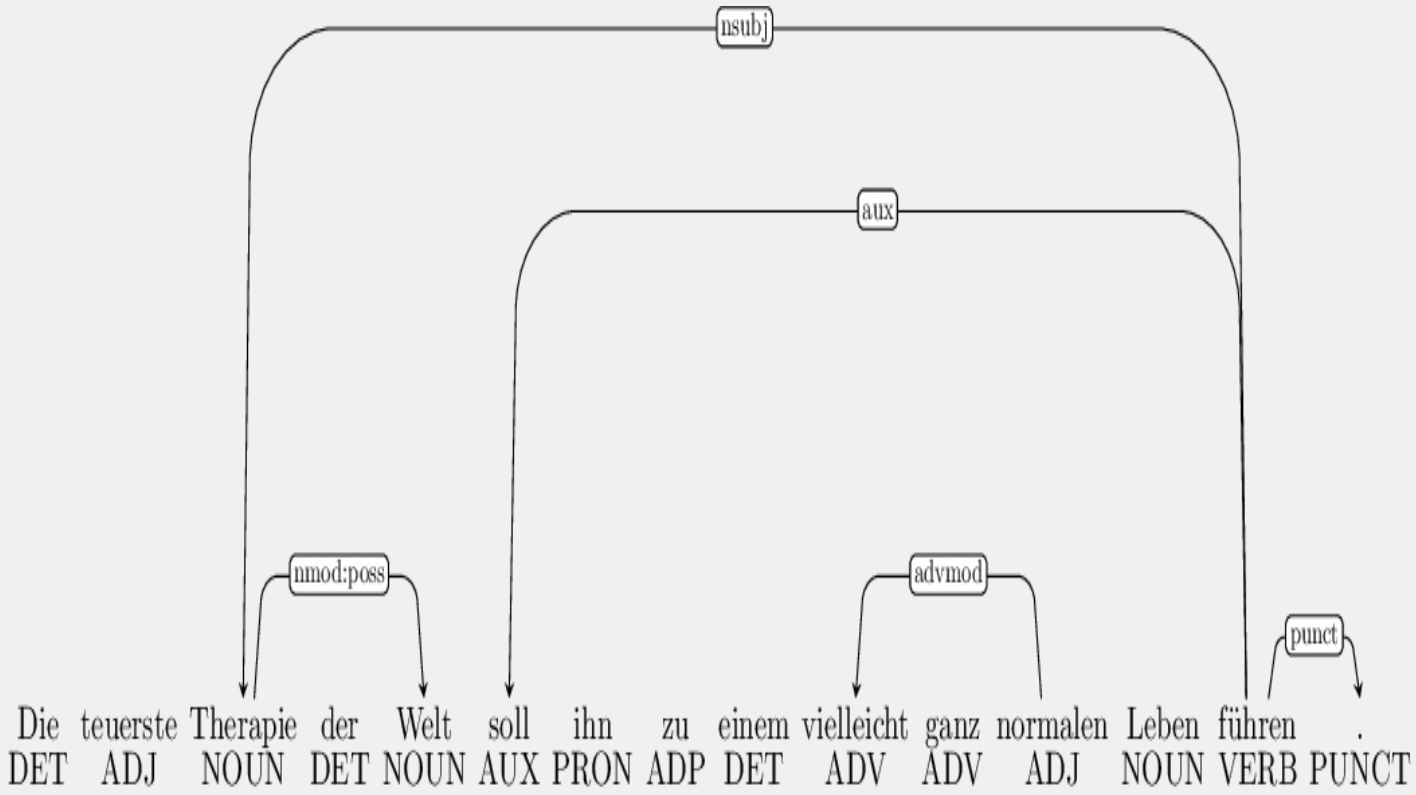
(1) Das Bild hängt **hinter dem Schrank**.
(2) The phone is lying **on the table**.

The phrases highlighted in bold are examples of *prepositional objects*. In Universal Dependencies v2, the dependency to connect them to the verb is labeled

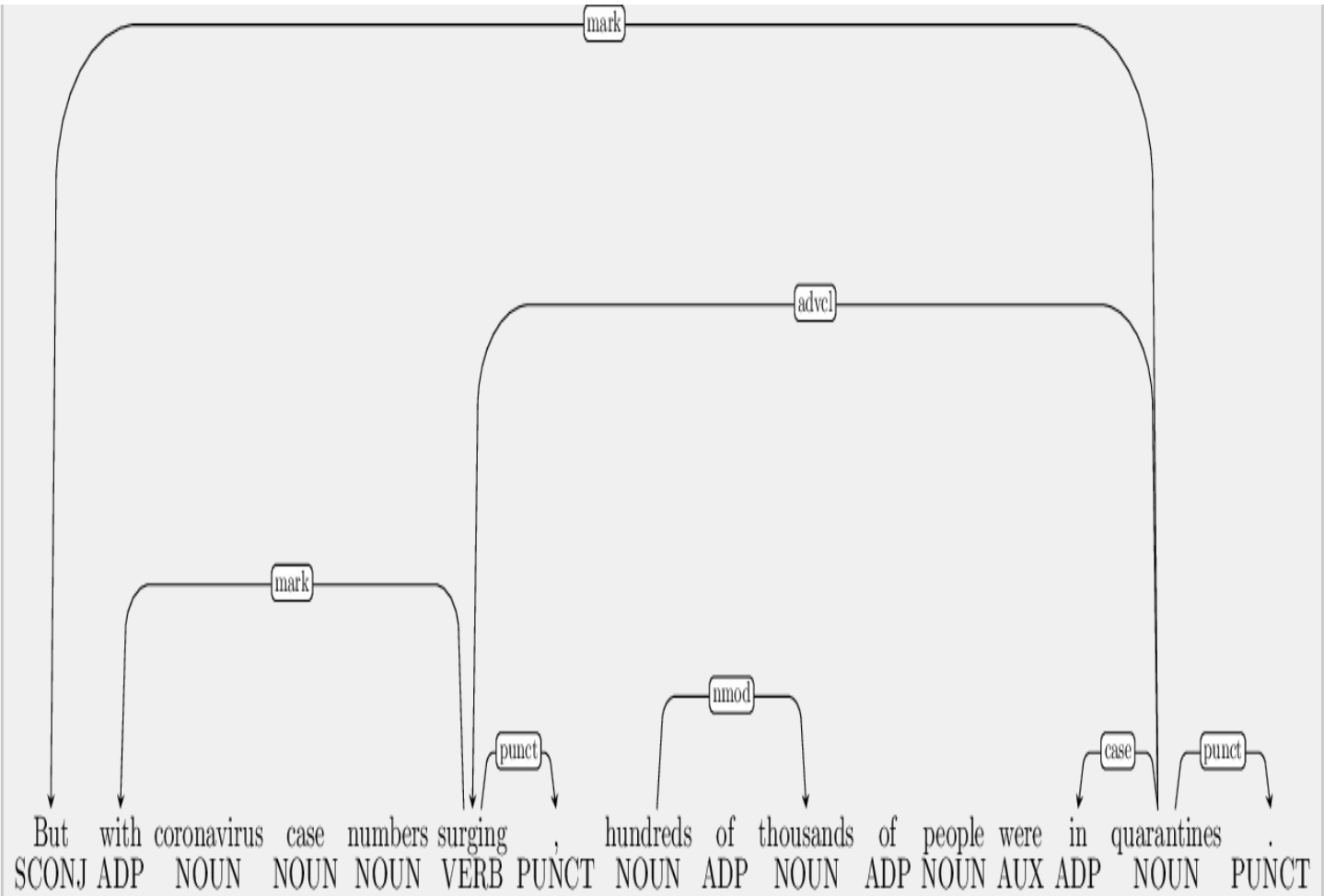
- nsubj
- dobj
- iobj
- obl

6. Complete a dependency tree [ID: 685972]

Below, you are given a German and an English **incomplete** dependency tree. They are shown graphically and in a textual form. Choose **either** the German **or** the English sentence and complete its dependency tree, using the same textual form.



führen --nsubj--> Therapie
führen --aux--> soll
Therapie --nmod:poss--> Welt
normalen --advmod--> vielleicht
führen --punct--> .



quarantines --mark--> But
quarantines --advcl--> surging
surging --mark--> with
hundreds --nmod--> thousands
surging --punct--> ,
quarantines --case--> in
quarantines --punct-->

Therapie --det--> Die
Therapie --amod--> teuerste
Welt --det--> der
führen --obj--> ihn
führen --obl--> Leben
Leben --case--> zu
Leben --det--> einem
Leben --amod--> normalen
normalen --advmod--> ganz

numbers --compound--> case
case --compound--> coronavirus
surging --nsubj--> numbers
thousands --case--> of
people --case--> of
thousands --nmod--> people
quarantines --nsubj--> hundreds
quarantines --cop--> were

Unbegrenzt Zeichen zugelassen, Anzahl der eingegebenen Zeichen: **397**

7. Complete a transition sequence [ID: 685973]

Remember the definition of the arc-eager transition system:

$$\text{LA}_k: \quad \frac{(\sigma|i,j|\beta,A)}{(\sigma,j|\beta,A\cup\{(j,i,k')\})} \quad i\neq 0 \wedge \neg\exists_{i',k'}(i',i,k')\in A$$

$$\text{RA}_k: \quad \frac{(\sigma|i,j|\beta,A)}{(\sigma|i|j,\beta,A\cup\{(i,j,k')\})} \quad \neg\exists_{i',k'}(i',j,k')\in A$$

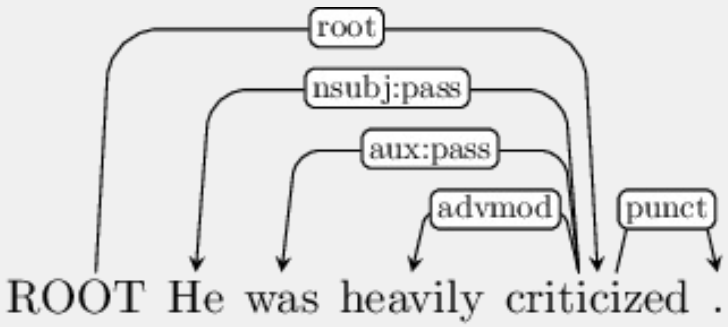
$$\text{RE}: \quad \frac{(\sigma|i,\beta,A)}{(\sigma,\beta,A)} \quad \exists_{i',k'}(i',i,k')\in A$$

$$\text{SH}: \quad \frac{(\sigma,i|\beta,A)}{(\sigma|i,\beta,A)}$$

Also remember the static oracle we defined for it:

$$o(c,T)=\begin{cases}\text{LA}_k & i\leftarrow j \text{ in } T\\ \text{RA}_k & i\rightarrow j \text{ in } T\\ \text{RE} & \exists_{v<i}\ v\leftrightarrow j \text{ in } T\wedge\exists_{i',k'}(i',i,k')\in A\\ \text{SH} & \text{otherwise}\end{cases}$$

Here's a gold dependency tree:



What is the transition sequence the static oracle generates for this tree? The first four steps are given below. Please write down the remaining steps. For each step, indicate the action, the new state of the stack and of the buffer, and the new dependency arc, if any.

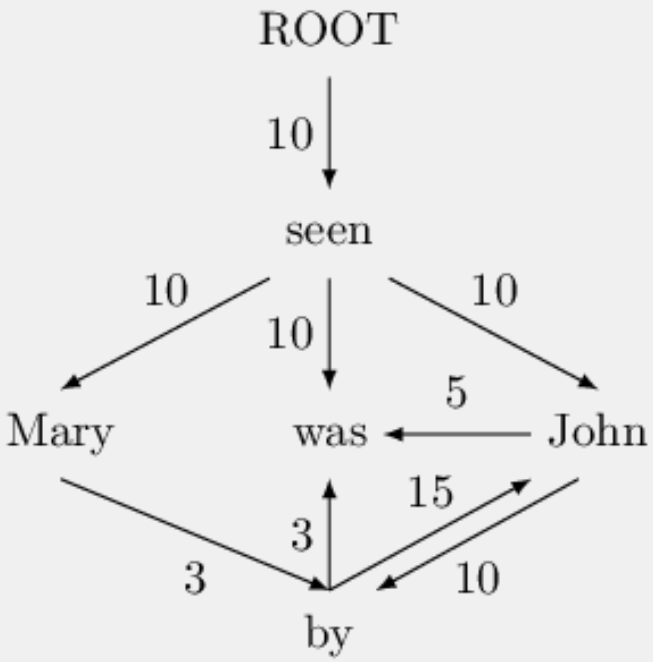
0) Stack: [ROOT]; Buffer: [He was heavily criticized .]
1) Transition: SH; Stack: [ROOT He]; Buffer: [was heavily criticized .]
2) Transition: SH; Stack: [ROOT He was]; Buffer: [heavily criticized .]
3) Transition: SH; Stack: [ROOT He was heavily]; Buffer: [criticized .]
4) Transition: LA_advmod; Stack: [ROOT He was]; Buffer: [criticized .]; New arc: criticized --advmod--> heavily
...

5) Transition: LA_aux:pass; Stack: [ROOT He]; Buffer: [criticized .]; New arc: criticized --aux:pass--> was
6) Transition: LA_nsubj:pass; Stack: [ROOT]; Buffer: [criticized .]; New arc: criticized --nsubj:pass--> He
7) Transition: RA_root; Stack: [ROOT criticized]; Buffer: [.]; New arc: ROOT --root--> criticized
8) Transition: RA_punct; Stack: [ROOT criticized .]; Buffer: []; New arc: criticized --punct--> .

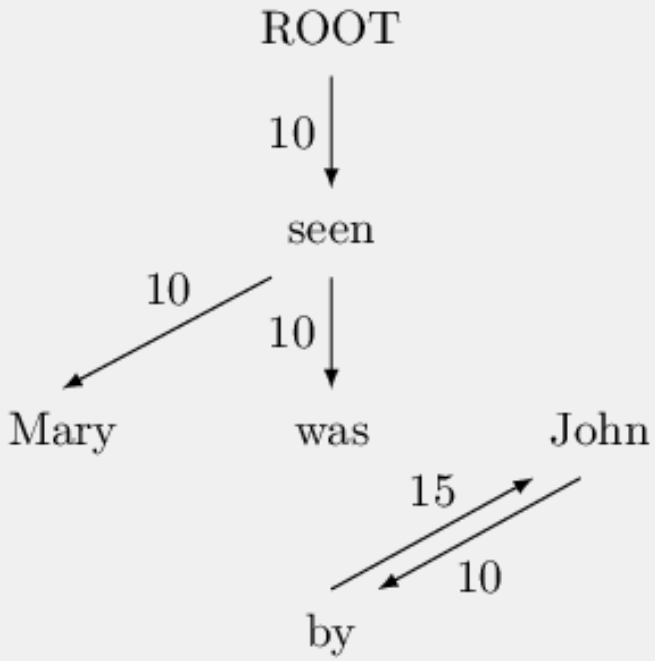
Unbegrenzt Zeichen zugelassen, Anzahl der eingegebenen Zeichen: **408**

8. Chu-Liu-Edmonds [ID: 685971]

Assume the arc weights for the sentence *Mary was seen by John* are given by the following complete digraph. (Arcs weighted 0 are not shown. Assume that weights are **not** in log scale, so need to be **multiplied** and not added.)

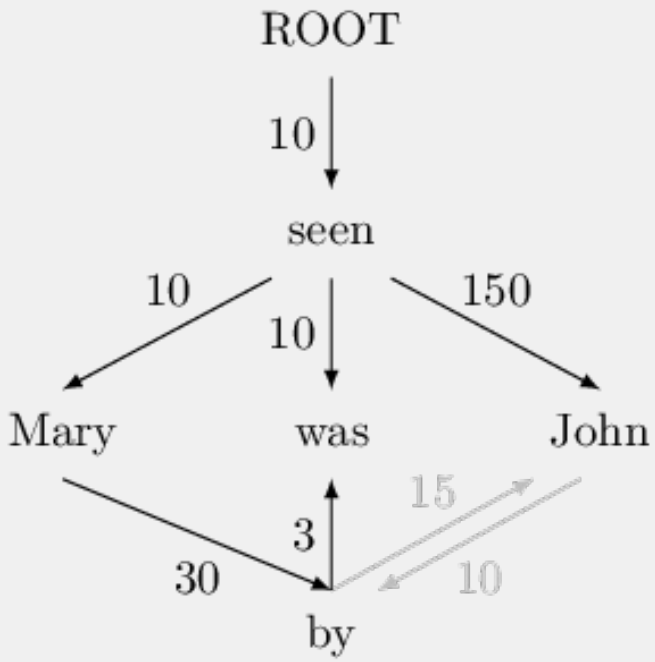


The first step of the Chu-Liu-Edmonds algorithm is to keep only the incoming arc with the heighest weight for each node:

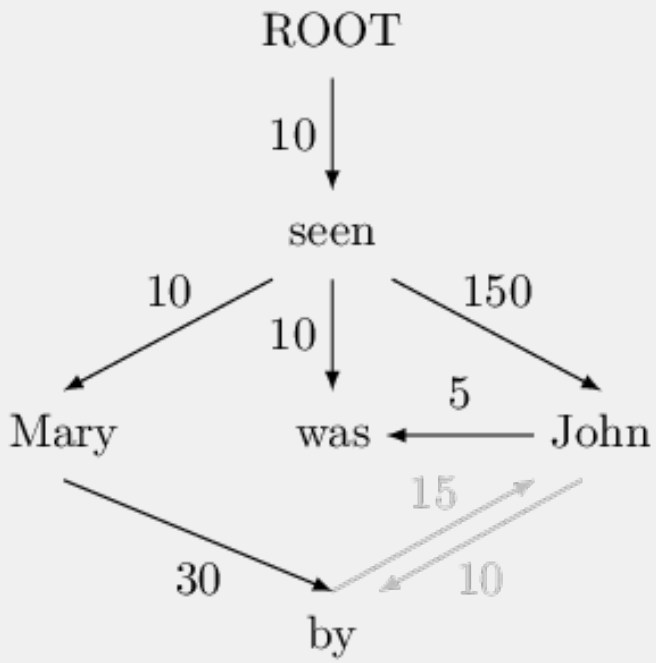


Since this graph has a cycle between "John" and "by", these nodes are contracted (we show this by making the arcs between them light gray) and the arc weights are recalculated. The result is for this step is... which of the following graphs? Explain why.

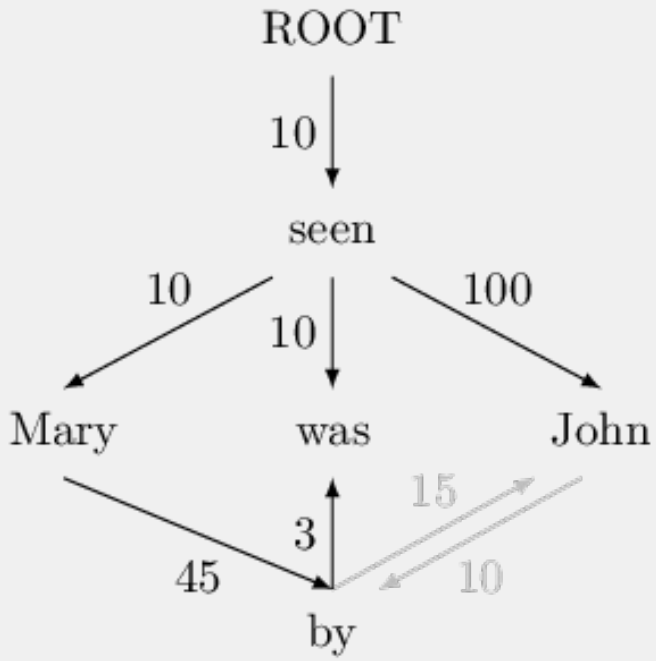
A)



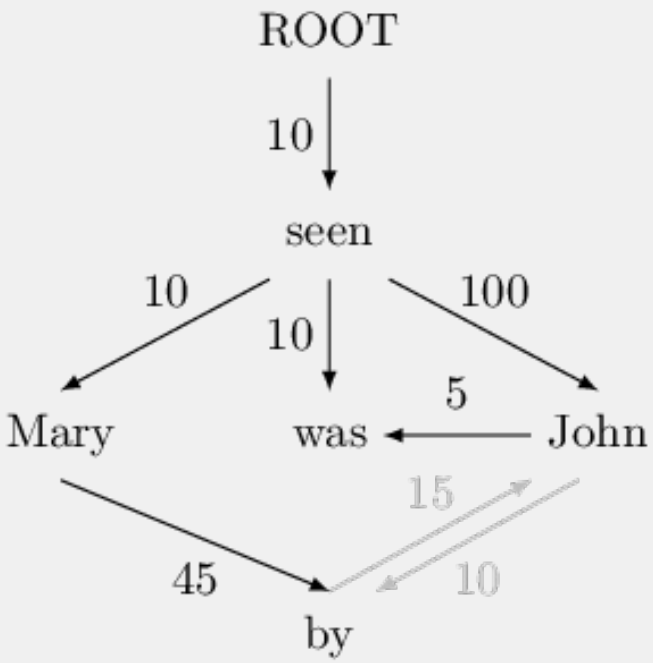
B)


ROOT
10
seen
10
10
150
5
Mary
was
John
15
30
10
by

C)


ROOT
10
seen
10
10
100
Mary
was
John
15
3
45
10
by

D)



D) is correct.

Concerning outgoing arcs from the cycle, we have to consider those to "was". There is one with weight 5 coming from "John" and one with weight 3 coming from "by". We choose the former because it has the higher weight.

Concerning incoming arcs into the cycle, we have one with weight 10 from "seen" to "john" and one with weight 3 from "Mary" to "by". If we choose the former, we need to drop the arc from "by" to "John" to break the cycle. If we choose the latter, we need to drop the arc from "John" to "by" to break the cycle. Now we have to multiply the weights of the incoming arcs with all arc weights in the cycle, except the one we drop. Thus, we multiply the former with 10, giving 100, and the latter with 15, giving 45.

Unbegrenzt Zeichen zugelassen, Anzahl der eingegebenen Zeichen: **742**