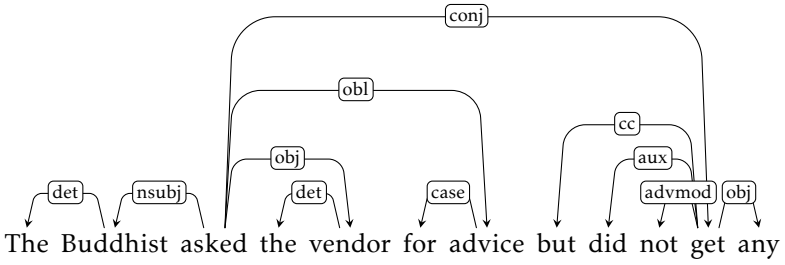
Dependency Parsing exercises: Dynamic oracles

June 10, 2021

1. Consider the following dependency tree:



Assume the arc-eager parser and the following partial transition sequence predicted by a classifier:[1]

TRANSITION	STACK	BUFFER	ARCS
	[ROOT]	[The Buddhist asked the vendor . . .]	∅
SH	[ROOT The]	[Buddhist asked the vendor . . .]	
LA	[ROOT]	[Buddhist asked the vendor . . .]	+(The ← Buddhist)
SH	[ROOT Buddhist]	[asked the vendor . . .]	
LA	[ROOT]	[asked the vendor . . .]	+(Buddhist ← asked)
RA	[ROOT asked]	[the vendor for advice . . .]	+(ROOT → asked)
SH	[ROOT asked the]	[vendor for advice . . .]	
LA	[ROOT asked]	[vendor for advice . . .]	+(the ← vendor)
SH	[ROOT asked vendor]	[for advice . . .]	

As you can see, the parser made a mistake in the last transition. Answer the following two questions:

- How will the parser recover from the mistake if it follows the static oracle, i.e., if the remaining predictions are consistent with the static oracle? Show the remaining part of the transition sequence.
- What if the remaining predictions of the parser are consistent with the dynamic oracle instead? What will be the set of reconstructed arcs then? Show one possible transition sequence and in each step (row) indicate alternative transitions licensed by the dynamic oracle (if any).

Solution: The static oracle fails to retrieve two additional arcs:

[1] The labels are ignored for simplicity.

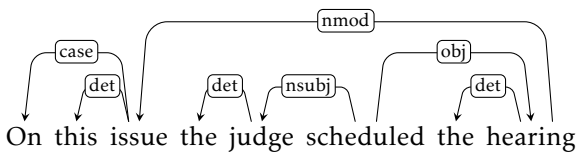
TRANSITION	STACK	BUFFER	ARCS
...	...	...	...
SH	[ROOT asked vendor]	[for advice but did not ...]	
SH	[ROOT asked vendor for]	[advice but did not get any]	
LA	[ROOT asked vendor]	[advice but did not get any]	+(for ← advice)
SH	[ROOT asked vendor advice]	[but did not get any]	
SH	[... vendor advice but]	[did not get any]	
SH	[... advice but did]	[not get any]	
SH	[... advice but did not]	[get any]	
LA	[... advice but did]	[get any]	+(not ← get)
LA	[... advice but]	[get any]	+(did ← get)
LA	[... advice]	[get any]	+(but ← get)
SH	[... advice get]	[any]	
RA	[... advice get any]	[]	+(get → any)

The dynamic oracle on the other hand is able to retrieve all but one arc, at the price of attaching *vendor* to *for* or *advice*...:

TRANSITION	STACK	BUFFER	ARCS
...	...	...	...
SH	[ROOT asked vendor]	[for advice but did not ...]	
LA (SH)	[ROOT asked]	[for advice but did not ...]	+(vendor ← for)
SH	[ROOT asked for]	[advice but did not ...]	
LA	[ROOT asked]	[advice but did not ...]	+(for ← advice)
RA	[ROOT asked advice]	[but did not get any]	+(asked → advice)
RE (SH)	[ROOT asked]	[but did not get any]	
SH	[ROOT asked but]	[did not get any]	
SH	[ROOT asked but did]	[not get any]	
SH	[ROOT asked but did not]	[get any]	
LA	[ROOT asked but did]	[get any]	+(not ← get)
LA	[ROOT asked but]	[get any]	+(did ← get)
LA	[ROOT asked]	[get any]	+(but ← get)
RA	[ROOT asked get]	[any]	+(asked → get)
RA	[ROOT asked get any]	[]	+(get → any)

The alternative transitions, licensed by the oracle but not selected by the parser, are marked in gray. In particular, in the second row of the table above, LA is optimal because (i) there's no right-arc arc from *vendor* to a word in the buffer, (ii) nor there is a left-arc from a word in the (tail[2] of the) buffer to *vendor*. In the same context, there is no arc between *for* and any word in the stack, hence SH is optimal as well.[3] See the notes from lecture 7 for more details about the conditions required for a transition to be optimal according to the dynamic oracle for the arc-eager system.

2. Consider the following dependency tree:



Assume the arg-eager parsing strategy, where:

- The initial configuration is ([ROOT], [On, this, issue, ..., the, hearing], $\emptyset$)

[2] I.e., the buffer with the exception of its head.

[3] SH is also used in the transition sequence of the static oracle at this point, but the dynamic oracle allows to later attach *vendor* to *advice*, thus removing it from the stack.

- A terminal configuration is any configuration of the form $(\sigma, [], A)$

What will be the set of arcs if the *dynamic oracle* is used to reconstruct this tree? Will it differ from the set of arcs in the tree shown above? If so, why is that the case?

Solution: Using the conventional static oracle:

TRANSITION	STACK	BUFFER	ARCS
	[ROOT]	[On this issue ...]	$\emptyset$
SH	[ROOT On]	[this issue ...]	
SH	[ROOT On this]	[issue the judge ...]	
LA_{DET}	[ROOT On]	[issue the judge ...]	+(this $\overset{\text{DET}}{\leftarrow}$ issue)
LA_{CASE}	[ROOT]	[issue the judge ...]	+(on $\overset{\text{CASE}}{\leftarrow}$ issue)
SH	[ROOT issue]	[the judge scheduled ...]	
SH	[ROOT issue the]	[judge scheduled ...]	
LA_{DET}	[ROOT issue]	[judge scheduled ...]	+(the $\overset{\text{DET}}{\leftarrow}$ judge)
SH	[ROOT issue judge]	[scheduled the hearing]	
LA_{NSUBJ}	[ROOT issue]	[scheduled the hearing]	+(judge $\overset{\text{NSUBJ}}{\leftarrow}$ scheduled)
SH	[ROOT issue scheduled]	[the hearing]	
SH	[ROOT issue scheduled the]	[hearing]	
LA_{DET}	[ROOT issue scheduled]	[hearing]	+(the $\overset{\text{DET}}{\leftarrow}$ hearing)
RA_{OBJ}	[ROOT issue scheduled hearing]	[]	+(scheduled $\overset{\text{OBJ}}{\rightarrow}$ hearing)

Thus the static oracle is able to retrieve 6 arcs out of 7 (or rather 6 out of 8, if we count the implicit *ROOT* → *scheduled* arc). In particular, it is not able to retrieve the NMOD arc – this is because the tree is non-projective.

If we look at the dynamic oracle, it's even worse:

TRANSITION	STACK	BUFFER	ARCS
	[ROOT]	[On this issue ...]	$\emptyset$
SH	[ROOT On]	[this issue ...]	
SH	[ROOT On this]	[issue the judge ...]	
LA_{DET}	[ROOT On]	[issue the judge ...]	+(this $\overset{\text{DET}}{\leftarrow}$ issue)
LA_{CASE}	[ROOT]	[issue the judge ...]	+(on $\overset{\text{CASE}}{\leftarrow}$ issue)
SH	[ROOT issue]	[the judge scheduled ...]	
SH	[ROOT issue the]	[judge scheduled ...]	
LA_{DET}	[ROOT issue]	[judge scheduled ...]	+(the $\overset{\text{DET}}{\leftarrow}$ judge)
SH	[ROOT issue judge]	[scheduled the hearing]	
LA_{NSUBJ}	[ROOT issue]	[scheduled the hearing]	+(judge $\overset{\text{NSUBJ}}{\leftarrow}$ scheduled)

At this point, SH (proposed by the static oracle) is not allowed, since there's an (implicit) arc *ROOT* → *scheduled* (see Eq. 5 in the lecture notes).[4]

A transition sequence actualy exists which allows to retrieve 7 arcs out of 8 arcs (at the price of creating a false-positive arc: *issue* ← *scheduled*):

[4]However, even if we removed this arc from consideration, the dynamic oracle would not be able to restore more arcs than the static oracle.

TRANSITION	STACK	BUFFER	ARCS
	[ROOT]	[On this issue . . .]	∅
SH	[ROOT On]	[this issue . . .]	
SH	[ROOT On this]	[issue the judge . . .]	
LA_{DET}	[ROOT On]	[issue the judge . . .]	+(this $\overset{\text{DET}}{\leftarrow}$ issue)
LA_{CASE}	[ROOT]	[issue the judge . . .]	+(on $\overset{\text{CASE}}{\leftarrow}$ issue)
SH	[ROOT issue]	[the judge scheduled . . .]	
SH	[ROOT issue the]	[judge scheduled . . .]	
LA_{DET}	[ROOT issue]	[judge scheduled . . .]	+(the $\overset{\text{DET}}{\leftarrow}$ judge)
SH	[ROOT issue judge]	[scheduled the hearing]	
LA_{NSUBJ}	[ROOT issue]	[scheduled the hearing]	+(judge $\overset{\text{NSUBJ}}{\leftarrow}$ scheduled)
$\text{LA}_{?}$	[ROOT]	[scheduled the hearing]	+(issue $\overset{?}{\leftarrow}$ scheduled)
RA_{ROOT}	[ROOT scheduled]	[the hearing]	+(ROOT $\overset{\text{ROOT}}{\rightarrow}$ scheduled)
SH	[ROOT scheduled the]	[hearing]	
LA_{DET}	[ROOT scheduled]	[hearing]	+(the $\overset{\text{DET}}{\leftarrow}$ hearing)
RA_{OBJ}	[ROOT scheduled hearing]	[]	+(scheduled $\overset{\text{OBJ}}{\rightarrow}$ hearing)

This somewhat surprising result stems from the fact that the arc-eager parser is only arc-decomposable w.r.t. projective structures. Otherwise, we would expect the dynamic oracle to capture this transition sequence (and restore 7 out of 8 arcs).