Prof. Dr. Andreas Hotho,
M.Sc. Janna Omeliyanenko
Lecture Chair X for Data Science, Universität Würzburg

7. Exercise for "Sprachverarbeitung und Text Mining"

17.12.2021

1 Knowledge Questions

1. State the advantages and disadvantages of the CKY and the Earley parser.

   Earley Parser: Top-down
   The Earley Parser generates many redundant candidates that may never contain the observed words.

   CKY: Bottom-up
   The CKY algorithm constructs many subtrees that may never form a valid sentence.

2. Briefly describe the characteristic procedures of the Earley Parser.

   - `scan`: Scan grammar and create new EarlyItems in a top-down strategy, which can be inferred from the current intermediate result of the parser.
   - `predict`: Read the current input word and add to the stateset of the next word a rule generating the current word from a given part of speech.
   - `complete`: For each completed EarlyItem, go back offset statesets and try to continue the derivation of this stateset's EarlyItems with the completed EarlyItem.

3. Name and describe the data structure used in the lecture to represent the dependencies of a sentence.

The dependencies of a sentence can be represented as a graph, with:

- V: Set of nodes (words)
- A: Set of directed edges (dependencies).
- Linear order of the set of nodes (word sequence)

What are the three important properties of this structure?

A dependency graph is

- connected
- acyclic (one tree)
- Each node has at most one incoming edge

4. What is the role of an oracle in Shift-Reduce dependency parsing?

An oracle predicts the best operation, according to its knowledge, to be executed next in the current system state.

How is an oracle provided for this task?

As an oracle, a classifier (such as SVM, MaxEnt, neural network) can be learned from training data.

2 CKY

Use the given grammar to parse the following sentence with the CYK Algorithm: `Much to learn you still have.`
You don't have to calculate probabilities in this exercise, since we focus on how to apply the rules here.

$$
\begin{aligned}
\text{ADV} &\rightarrow \textit{Much} \\
\text{APPR} &\rightarrow \textit{to} \\
\text{V} &\rightarrow \textit{learn} \\
\text{PRO} &\rightarrow \textit{you} \\
\text{ADV} &\rightarrow \textit{still} \\
\text{V} &\rightarrow \textit{have} \\
\text{S} &\rightarrow \text{X0 VP} \\
\text{X0} &\rightarrow \text{VP PRO} \\
\text{VP} &\rightarrow \text{APPR V} \\
\text{VP} &\rightarrow \text{ADV V} \\
\text{VP} &\rightarrow \text{ADV VP}
\end{aligned}
$$

CKY

[Much]	[ADV]	[]	[VP]	[X0]	[]	[S]
	[to]	[APPR]	[VP]	[X0]	[]	[S]
		[learn]	[V]	[]	[]	[]
			[you]	[PRO]	[]	[]
				[still]	[ADV]	[VP]
					[have]	[V]

3 Early Parsing

Check if the sentence below can be parsed with the given grammar by doing the "forward parse" of the Earley Algorithm. In other words: write down all state sets as seen in the lecture and check if the last one contains an item that satisfies the conditions for a successful parse.

`Teacher and students philosophize`

The following grammar is used:

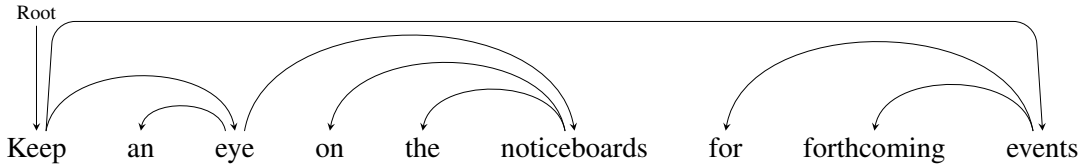
$$
\begin{aligned}
\text{S} &\rightarrow \textit{NP VP} \\
\text{NP} &\rightarrow \textit{NN KONJ NN} \\
\text{VP} &\rightarrow \textit{V} \\
\text{NN} &\rightarrow \textit{students} \\
\text{KONJ} &\rightarrow \textit{and} \\
\text{NN} &\rightarrow \textit{Teacher} \\
\text{V} &\rightarrow \text{philosophize}
\end{aligned}
$$

Earley

Stateset	Rule	Offset
Stateset[0]	S → • NP VP	[0]
	NP → • NN KONJ NN	[0]
	NN → • students	[0]
	NN → • Teacher	[0]
Stateset[1]	NN → Teacher •	[0]
	NP → NN • KONJ NN	[0]
	KONJ → • and	[1]
Stateset[2]	KONJ → and •	[1]
	NP → NN KONJ • NN	[0]
	NN → • students	[2]
	NN → • Teacher	[2]
Stateset[3]	NN → students •	[2]
	NP → NN KONJ NN •	[0]
	S → NP • VP	[0]
	VP → • V	[3]
	V → • philosophize	[3]
Stateset[4]	V → philosophize •	[3]
	VP → V •	[3]
	S → NP VP •	[0]

4 Shift-Reduce Parsing

Given is the following dependency tree:



In order to learn the state transitions of Shift-Reduce parsing, data labelled through dependency trees must be converted into Shift-Reduce state transitions.

Perform this conversion by decomposing the graph into a sequence of states during Shift-Reduce parsing. Initially, root is on the stack and the input buffer contains the entire set.

For each state, specify the contents of the stack, the input buffer and the selected operation in tabular form (as shown below). The column *Operation* contains the action that will be performed to move from the current line to the next - so start by adding the first operation.

Stack	Buffer	Operation
root	Keep, an, eye, on, the, noticeboards, for, forthcoming, events	????

Stack	Buffer	Operation
root	Keep, an, eye, on, the, noticeboards, for, forthcoming, events	SHIFT
root, Keep	an, eye, on, the, noticeboards, for, forthcoming, events	SHIFT
root, Keep, an	eye, on, the, noticeboards, for, forthcoming, events	SHIFT
root, Keep, an, eye	on, the, noticeboards, for, forthcoming, events	LEFTARC
root, Keep, eye	on, the, noticeboards, for, forthcoming, events	SHIFT
root, Keep, eye, on	the, noticeboards, for, forthcoming, events	SHIFT
root, Keep, eye, on, the	noticeboards, for, forthcoming, events	SHIFT
root, Keep, eye, on, the, noticeboards	for, forthcoming, events	LEFTARC
root, Keep, eye, on, noticeboards	for, forthcoming, events	LEFTARC
root, Keep, eye, noticeboards	for, forthcoming, events	RIGHTARC
root, Keep, eye	for, forthcoming, events	RIGHTARC
root, Keep	for, forthcoming, events	SHIFT
root, Keep, for	forthcoming, events	SHIFT
root, Keep, for, forthcoming	events	SHIFT
root, Keep, for, forthcoming, events		LEFTARC
root, Keep, for, events		LEFTARC
root, Keep, events		RIGHTARC
root, Keep		RIGHTARC
root		DONE