# NLP 2023-2024, Homework 4: Context Free Grammars and Parsing (corrected version of 13 Oct 2023)

**Deadline:** 15 October (23:59).
**Questions?** Ask them in the discussion on Canvas, on Discord or send them to nlp-course@utwente.nl.

*Please adhere to the Guidelines for using AI during your studies at UT.*

## Exercise 1: Context Free Grammar (3 pt)

In this exercise, we create a small context free grammar that can deal with a number of (possibly ambiguous) English sentences. We implement and test the grammar using PC-PATR. This is really old software with a command line interface, but it is very practical for creating and testing context free grammars that are meant for parsing natural language. PC-PATR was made for Windows (run it using the .exe file), but we also provide a version for Mac or Linux.

You can find the PC-PATR executables and other files for this exercise in `NLP-HW4-files.zip` on Canvas. *See the PC-PATR instructions at the end of this document for information on how to use the program.*

1a (2 pt) Create a context free grammar in PC-PATR that can parse all the sentences in the file `sentences.txt` (provided in the zip file). Make sure your grammar respects constituent structure, and uses similar part of speech tags and constituent symbols as in Ch. 17 and Appendix D.3 of the book. Your grammar should assign multiple parses to syntactically ambiguous sentences, but not to non-ambiguous ones.

Use the files `exercise1.grm` (grammar rules) and `exercise1.lex` (lexicon) that are provided with the assignment files on Canvas as a starting point for your grammar. Use the file `exercise1.tak` to load your grammar and lexicon and test it on `sentences.txt` (see the instructions at the end of this document). When you run the program, make sure all files are in the same folder, together with `pc-patr.exe`.

You need to HAND IN your PC-PATR grammar and lexicon files (plain text format, named `exercise1.grm` and `exercise1.lex`.

1b (1 pt)

For which sentences, if any, does your grammar produce more than one parse? Explain whether this is as it should be, or not.
If there are sentences with multiple parses, indicate in your explanation which of the parses corresponds to the most likely interpretation of the sentence, and why.

## Exercise 2: CKY parsing (3.5 pt)

Consider the following simple grammar:

**Grammar rules**
S → NP VP
VP → V NP
VP → V NP PP
NP → Nominal
NP → Det Nominal
NP → Nominal PP
NP → Det Nominal PP
Nominal → Noun
PP → P NP

**Lexicon**
Noun → ladies, cake, forks
P → with, in
V → eat, bake
Det → a, the

2a (1 pt) Convert the grammar to Chomsky Normal Form.

2b (2 pt) Using the CNF version of the grammar, draw a CKY parse table for the sentence *ladies eat cake with forks.* For the last column, show step-by-step how it is filled, in the same way as is done in Figure 17.14 of J&M Ch. 17.

2c (0.5 pt) Draw the parse tree or parse trees (with root S) that can be derived from the filled CKY parse table.

## Exercise 3: dependency parsing (3.5 pt)

Several dependency parsers are available as part of NLP toolkits, such as NLTK[1] or the Stanford CoreNLP package[2]. In this exercise we try out the dependency parser of SpaCy, another open-source NLP software library. We use the web demo of its dependency visualizer:

```
https://explosion.ai/demos/displacy
```

Detailed explanations of most of the dependency labels SpaCy uses are provided here.

3a (0.5 pt) Use the web demo to parse the ambiguous sentence *I saw a man in boxer shorts.* You have to **uncheck** the "Merge phrases" checkbox.

What do you think about the way this sentence was parsed? Is it a good parse or not? Why (not)?

3b (1 pt) Now, use the web demo to parse the following variations of the previous sentence. Keep the "merge phrases" checkbox **unchecked**.

1. I like a man in boxer shorts
2. I shot a man in boxer shorts
3. I shot a man in my boxer shorts
4. men seduce women in boxer shorts
5. women seduce men in boxer shorts

Discuss the differences between the dependency parses SpaCy assigns to these sentences. Do you think they are good parses? Why (not)? Can you think of reasons for the different parses?

[1] https://www.nltk.org/

[2] https://nlp.stanford.edu/software/lex-parser.shtml

3c (2 pt) Take the dependency parse produced by SpaCy for*I saw a man in boxer shorts*. Provide the trace of this dependency parse as it would be produced by a transition-based, arc-standard dependency parser.Use the notation that is also used in Figure 18.6 of J&M Chapter 18.

## Handing in

Submit the following items via Canvas. Please mention your NAMES in all files and documents, *including* your grammar and lexicon files!

- For exercise 1, submit your PC-PATR grammar and lexicon files (plain text format, named `exercise1.grm` and `exercise1.lex`).
- For all other questions: submit your answers in a Word or pdf document.

## PC-PATR instructions

PC-PATR[3] is a program in which you can define context-free grammars and use them to parse sentences. Use PC-PATR in combination with the other files provided in the zip package on Canvas:

- The grammar file (`exercise1.grm`) provided on Canvas provides you with a starting point for specifying the grammar rules for exercise 1. You can add your own rules following the same format. Your grammar rules don't have to be CNF.
- If you want to include multiple rules for the same constituent (say, multiple VP rules) you have to list each rule separately, preceded by the keyword 'Rule'.
- IMPORTANT: if you want to use the same symbol multiple times within the same rule, you should distinguish the repeated instances of this symbol (after the arrow) by giving them an index number. Index numbers are preceded by the underscore (_) character. For example, $A \rightarrow A\_1\ B$ or $A \rightarrow A\_1\ A\_2$. You only need to do this inside the rule with the repeated symbols, and not across the entire grammar.

[3] `https://software.sil.org/pc-patr/`

- The lexicon file (`exercise1.lex`) lists the words and their parts of speech. After \w you specify the word string, after \c you specify the part-of-speech category.

- The sentences your grammar must be able to parse are listed in the file `sentences.txt` (one sentence per line).

- The take file (`exercise1.tak`) (re)loads your rules and lexicon and uses them to parse the correct and the incorrect sentences. You execute the take file with the command `take exercise1`. The parses for the sentences in `sentences.txt` are written to the file `out.txt`. The result should be at least one parse for each sentence.

Use `pcpatr.exe` (`pcpatr`) to run the program. At the `PC-PATR` command prompt, enter `take exercise1` to load the rules and lexicon. You can also use other commands to test your grammar:

- Enter `parse` followed by a sentence to parse one sentence at a time.

- Enter `parse` followed by <ENTER> and a prompt will appear where you can enter a sentence to be parsed. Exit by typing a single <ENTER>.

## Running PC-PATR for Mac users

How to run pcpatr for novice terminal users (with thanks to Lorenzo Gatti):

1. Put the PC-PATR executable and all the other files in the same folder

2. Launch the terminal

3. Type "cd " (without quotes, but notice the space after cd)

4. Drag on the terminal the folder where the pcpatr file is (e.g. "HW-4-NLP"). The terminal will automagically insert the full path of the folder. Press Enter.

5. You're now in the folder where the pcpatr executable is. To run it, type "./pcpatr" (again, no quotes) and press Enter.

If you get the message "Permission denied" when trying to run PC-PATR, go to System preferences, Security and Privacy, and under the General tab make sure you can run apps that come from the App Store and identified developers. If there's a confirmation request about opening the app, click “Open Anyway”. If that does not solve it, get to the correct folder with the terminal and type “chmod +x pcpatr” before launching it. This marks the file as “executable”, indicating to the shell that the file can indeed be launched as a computer program. Once it’s been marked, there is no need to run this command again for the same program.

Side note: all Mac and Linux users are recommended to read this introduction to the terminal, specifically targeted to NLP practitioners: Bash for NLP.[4] Knowing the basic tools (cat, grep, wc, sed, ...) will save you a lot of time in the long run!

[4] `https://nlp.stanford.edu/~johnhew/bash-for-nlp-tutorial-basic.html`