Katherine Wu
113171943
CSE 505: Course Project, Phases I, II, and V

Part I. Problem and plan

My project will be on parsing probabilistic context-free grammars (PCFG), more specifically implementing several well-known parsing algorithms in Prolog. Parsing essentially means how to assign a structure to a sequence of text. I would like to leverage Prolog's logical inference capabilities and declarative nature to handle probabilistic context-free grammars effectively. Additionally, I will include a GUI that helps visualize the parsing steps for each algorithm.

A PCFG is a formal grammar used in natural language processing and computational linguistics. Unlike traditional context-free grammars, which assign equal probabilities to all possible parse trees for a given sentence, PCFGs incorporate probabilities into the production rules of the grammar. These probabilities reflect the likelihood of a particular rule being used in generating or deriving a sentence. PCFGs are essential for capturing the ambiguity of natural language, and are particularly useful in tasks such as syntactic parsing, which uses dynamic programming algorithms to compute the most likely parse of a sentence given a statistical model of the syntactic structure of the language [1,2].

Input: The input is a sentence in natural language, along with a grammar consisting of probabilities assigned to each of its rules.
Output: The output is a syntactic parse tree representing the grammatical structure of the input sentence. I hope to include a GUI component as well, which will illustrate each step of the parsing algorithm.
Other Requirements: The implementation will be done using Prolog. Performance evaluation will include metrics such as parsing time and accuracy compared to manually annotated parse trees.
Example Uses: The outputted parse tree will have applications in various NLP tasks such as information extraction and question answering systems. For an example of a question answering system, see here: https://github.com/cbaziotis/prolog-cfg-parser?tab=readme-ov-file. This project extracts knowledge (facts) from text so that when a question is asked by a user, the system is able to generate a relevant answer, and this system is implemented using a toy CFG parser.

More specific example of input and output:
The input sentence is "I saw in the dictionary". The grammar is shown below, where S stands for sentence, NP stands for noun phrase, VP stands for verb phrase, PP stands for prepositional phrase, Det stands for determiner, N stands for noun, V stands for verb, and P stands for preposition.

```
S -> NP VP [1.0]
NP -> Det N [0.6] | Pronoun [0.4]
VP -> V [0.7] | VP PP [0.3]
PP -> P NP [1.0]
Det -> "the" [0.7] | "a" [0.3]
N -> "dog" [0.4] | "cat" [0.3] | "ball" [0.3] | “dictionary” [0.1]
Pronoun -> "I" [0.5] | "he" [0.3] | "she" [0.2]
V -> "saw" [0.6] | "ate" [0.4]
P -> "with" [0.6] | "in" [0.4]
```

We can use probabilistic parsing to find the most likely parse tree for this sentence according to the given PCFG. The output is shown below.

```
         S
  ┌──────┴──────┐
 NP             VP
 |           ┌──┴──┐
Pronoun      VP    PP
 |           |     |
 "I"         V    P    NP
             |    |
            "saw" “in”  |
                        |
                       Det N
                       |   |
                      "the" "dictionary"
```

State of the art:
The CYK and Earley parsers are classic algorithms for parsing context-free grammars [4,5,6].

- The CYK algorithm is a bottom-up parsing algorithm based on dynamic programming. It only operates on context-free grammars in Chomsky Normal Form, where each production rule has either two non-terminals or one terminal on the right-hand side. The runtime complexity of the algorithm is O(n^3) and the space complexity is O(n^2).
- The Earley algorithm uses a top-down approach, as it starts with the initial symbol of the grammar and recursively expands it to match the input sentence. The runtime complexity of the algorithm is O(n^2) and the space complexity is O(n^2).
- Both algorithms may be adapted to work on probabilistic context-free grammars.

Some other parsing algorithms include:

- The inside-outside algorithm is a way of re-estimating production probabilities in a PCFG, and it iteratively improves the parameters of the PCFG based on observed data [7]. You start with a PCFG with arbitrary initial probabilities for the production rules, and update the probabilities of the rules based on the expected counts derived from the inside and outside probabilities.
- The Viterbi algorithm can be used to find the most probable parse tree for a given input sentence by exploring the space of possible parse trees and selecting the one with the highest probability [8].

More modern techniques involve the use of neural networks and deep learning [9,10]. However, for the sake of this project, we will not incorporate machine learning techniques since the goal of the project is to use powerful languages, methods, and systems for logic reasoning and programming at large.

In terms of open-sources projects that are currently available:

1. https://github.com/rdorado/pcyk/tree/master is an implementation of a probabilistic version of the CYK algorithm, in Python.
2. https://github.com/deborausujono/pcfgparser is another Python implementation of the CYK algorithm for PCFG parsing.
3. https://github.com/KarthikMAM/Earley-Algorithm is a Python implementation of the Earley algorithm for CFGs
4. https://github.com/jgontrum/PCFG-EM is a basic implementation in C++ of the inside-outside algorithm for PCFGs

For my project, I will be implementing the CYK, Earley, inside-outside, and Viterbi algorithms in Prolog. The plan is to evaluate my program with the above programs (not implemented in a logic programming language), to highlight the clarity and efficiency that logic programming brings in handling PCFG parsing. For each algorithm, the plan is to include a GUI or visualization to better illustrate how the parsing algorithm works step-by-step for the user. Note: the GUI may have to be implemented using either Python or another web development framework, as Prolog does not have good features for supporting GUIs.

Project plan:
Each task should be completed in about a week, with the due date set for April 19.

- Task 1: Research and understand the CYK, Earley, inside-outside, and Viterbi algorithms
- Task 2: Design and implement CYK and Earley parsers in Prolog (along with their GUIs)
- Task 3: Design and implement the inside-outside and Viterbi algorithms in Prolog (along with their GUIs)

- Task 4: Develop test cases to ensure correctness, evaluate the parsers' performance by comparing the performance and accuracy of the implemented parsers against each other and against existing implementations
- Task 5: Document the implementation, including explanations of algorithms and usage instructions, fix any bugs in the project

References:

[1] https://gawron.sdsu.edu/compling/course_core/lectures/pcfg/prob_parse.htm, accessed 14 March 2024.
[2]https://nlp.stanford.edu/projects/stat-parsing.shtml#:~:text=Probabilistic%20parsing%20is%20using%20dynamic,as%20well%20as%20the%20algorithms., accessed 14 March 2024.
[3] Mehryar Mohri, Brian Roark (2006). Probabilistic Context-Free Grammar Induction Based on Structural Zeros, https://aclanthology.org/N06-1040.pdf.
[4] David Rodriguez-Velazquez (2009). The CYK Algorithm, https://www.cs.ucdavis.edu/~rogaway/classes/120/winter12/CYK.pdf.
[5] CYK Algorithm Made Easy (Parsing), https://www.youtube.com/watch?v=VTH1k-xiswM, accessed 14 March 2024.
[6] Scott Farrar (2010). Earley Algorithm, https://courses.washington.edu/ling571/ling571_fall_2010/slides/parsing_earley.pdf.
[7] Michael Collins, The Inside-Outside Algorithm, https://www.cs.columbia.edu/~mcollins/io.pdf, accessed 15 March 2024.
[8] Julia Hockenmaier, Statistical Parsing with PCFGs, https://courses.engr.illinois.edu/cs447/fa2018/Slides/Lecture10.pdf, accessed 15 March 2024.
[9] Songlin Yang, Yanpeng Zhao, Kewei Tu (2021). PCFGs Can Do Better: Inducing Probabilistic Context-Free Grammars with Many Symbols, https://aclanthology.org/2021.naacl-main.117.pdf.
[10] Yoon Kim, Chris Dyer, Alexander Rush (2020). Compound Probabilistic Context-Free Grammars for Grammar Induction, https://arxiv.org/pdf/1906.10225.pdf.