Language Parsers and Translators

CS142 - Spring 2020: Computability and Complexity Mid Semester Project.

This assignment focuses on the design of language parsers and it integrates your LBA for the semester. In fact, what better way to gain an understanding of the concepts of the theory of languages than to apply them to natural languages. We will use Spanish for M21s in Buenos Aires, Telugu for M22s in Hyderabad. Lastly, anyone in San Francisco should use Spanish.

The learning goals of this assignment are closely related to the CS142 learning outcomes: **#languages** and ***#computability.*** Thus focus on using the theory of languages and automata rather than the linguistic accuracy of the English-Spanish, English-Telugu translations. What this means in practice is that you get to define your own grammar rules for the limited English languages in this project.

1. An English Language Limited Parser [#languages, #ComputerSimulations].

In this section, you will design a grammar and automata to be able to parse a very limited subset of the English language strings. Therefore, you need to design a parser that recognizes the input string based on your subset of the English language. The following are a minimum set of requirements for your grammar:

[a] Your language includes at least 10 English nouns (e.g., morning, food, park, home, Tom)
[b] Your language includes at least 5 English verbs (e.g. have, want, eat, go, etc.)
[c] Your language includes at least 5 English adjectives (e.g, good, nice, etc.)
[d] Your language includes at least the subjective pronouns (e.g., I, you, she, he, they, and it)
[e] Use the modern Latin alphabet (includes the 26 characters from modern English, and additional special accented characters for Spanish) as your base alphabet.

1.a. Provide a formal definition of your grammar.
1.b. Is your grammar ambiguous? Either way, justify your answer.
1.c. Provide a formal definition of an equivalent machine of your choice that can recognize the strings in the language generated by your grammar. We call this Machine ***Parser1.***
1.d. Write a simulator in Python for your ***Parser1.*** You can define your own machine or use any of the available machines in the Python automata-lib.

2. Word by Word Translator [#languages, #computability].

In this section, you will design a word by word translator for all the subset English language that you defined in your in Section 1. You can use Google translate for this part of the assignment.

2.1 Design and give a formal definition of your automaton that carries the word by word translations into either English-Spanish, or English-Telugu depending on your rotation.
2.2 What are the main computational drawbacks and limitations of this word-by-word translator?

3. Auto-Code Function Generator [#languages, #computability, #ComputerSimulations].
In this section, you will design an application that takes a single line of text as an input specifying a mathematical function, if the input line is in your mathematical language, then you proceed to generate the equivalent Python code. Thus, you are designing an auto-code generator for Python functions. Table 1 illustrates three examples of input strings to the generator and with their corresponding Python functions.

Table 1. Examples of input strings to the Python Function Generator application.

Input String	Auto-generated Python Function

f(x) = x*0.5	def f(x):
	  y = x*0.5
	  return y

f(x,t) = sin(x*t)	def f(x,t):
	  y = sin(x*t)
	  return y

f(a,b) = (a+b)/2	def f(a, b):
	  y = (a+b)/2
	  return y

Note that you are free to define your own syntax for the input string (i.e., it doesn't have to match the one in the examples above,) In addition, you get to define the functions and operations that your auto-function generator understands. These are the **minimum** requirements for your generator:

[a] Your language includes at least five mathematical functions (e.g., sin, cos, sqrt, etc.)
[b] Your language includes the four basic arithmetic operators: +, -, / (division), and * (multiplication)
[c] Your language includes integers, and reals, both positive and negative values.
[d] Your language should include at least two variables to be used as input arguments for the functions (e.g. x and y in f(x,y) = x*sin[y])
[e] You can add other special characters to your input string to help you parse the strings and define your functions.

3.a. Provide a formal definition of your language and grammar.
3.b. Provide a formal definition of an equivalent machine of your choice that can recognize the language described by your grammar. We call this Machine ***ParseFunc.***
3.c. Write a simulator in Python for your ***ParseFunc.***

3.d. Write Python code that calls ***ParseFunc*** to determine if a given input is the language of your Python function generator. Then, if the input string is accepted, your program outputs back the equivalent Python function.

4. Concluding Remarks [#languages, #computability].

Natural language translators are part of the wide field of Natural Language Processing (NLP) [a]. In this assignment, we have explored a very simple example of rule-based NLP, which was used in the early decades of NLP, including AI applications. Nowadays, natural language translators incorporate machine learning techniques to provide more accurate language translations (these are examples of statistical-based NPL).

Briefly, write a short summary of how language translators have evolved from ruled-based translations to statistical-based ones (see references below). Focus on the syntax part only, as the field of NLP has a vast number of ramifications. Your summary should address the following points:

- Give an example of a statistical-based language translator (e.g. Google translate [c] or your favorite language translator) and briefly explain how it works.
- Highlight some of the benefits and drawbacks for statistical NLP vs rule-based NLP
- In your opinion, what kind of language generators and recognizers (machines) are needed for rule-based and statistical-based translators.

5. [#Professionalism and #Responsability] Provide at least 3 applications of HCs in this assignment (Problem 1-4).

6. Assignment Submission and Grading.

Submit a PDF file as the first resource with your answers to the above problems. Then submit a Python Notebook as your second resource.

The CS142 LO's to be graded in this assignment are:

- **#languages:** Explain, prove, or apply concepts related to automata or formal languages.
- **#computability:** Explain, prove, or apply concepts related to computability.
- **#ComputerSimulations:** Write a computer program that simulates the behavior of automata or language generators.

7. References.

[a] Wikipedia. Natural Language Processing, https://en.wikipedia.org/wiki/Natural_language_processing

[b] Hutchins, J., The history of machine translation in a nutshell. http://www.hutchinsweb.me.uk/Nutshell-2014.pdf

[c] Wikipedia. Google Translate. https://en.wikipedia.org/wiki/Google_Translate

[d] A simple introduction to Natural Language Processing,

[e] Wikipedia page on ***Telugu grammar***. https://en.wikipedia.org/wiki/Telugu_grammar. Sections of interest 1.1,

[f] **Gutman, A., & Avanzati, B**. (2013). *The Language Gulper: Telugu*. http://www.languagesgulper.com/eng/Telugu.html