# NLP Experiment 5

**Aim:** To implement shallow and deep Parsing

**Theory:**

Shallow Parsing (also known as chunking) focuses on grouping words or phrases based on their syntactic structures.

- We first tokenize each word.
- We perform pos tagging on the text.
- We perform chunking using RegExParser.
  “GP: {<JJ.*|VBG><NN.*>+}” is our RegexPattern, where:
  GP stands for genre phrase which gives data points that help get information
  regarding the genre of the movie

- JJ stands for adjective, JJ.* could parse superlative or comparative adjective used in the synopsis
- and NN.* stands for the nouns used in the movie synopsis
- VBG stands for gerund where any verbs ending with "ing" would be parsed
  It will either parse and adjective-noun combination or a gerund noun combination to gain information on the genre of the movie by its synopsis

- We can make a chunker object entering the above RegEx pattern.
- The RegEx pattern enables the chunker to parse objects in the pos-tagged entities.
- The parsed entities are then used to visualize a parsing tree.
- The libraries used in this:
  The NLTK (Natural Language Toolkit) downloads you've listed include various resources and models used for natural language processing (NLP) tasks, including shallow parsing and other related tasks. Let's briefly describe each of them and their relevance to shallow parsing:

  **Maxent_treebank_pos_tagger:**
  - This is a part-of-speech tagger model trained on the Treebank corpus. It assigns part-of-speech tags to words in a text, which is a fundamental step in shallow parsing. Part-of-speech tagging helps identify the grammatical roles of words in a sentence, which is essential for chunking and other syntactic analysis.

  **Treebank:**
  - The Treebank corpus is a large collection of parsed and annotated English sentences. It serves as a valuable resource for training and evaluating syntactic parsers and taggers. Shallow parsers and chunkers can benefit from using this corpus for training and testing.

**Punkt:**

- The Punkt tokenizer is a pre-trained sentence tokenizer that can segment text into sentences. While not directly related to shallow parsing, sentence segmentation is often a preliminary step before any parsing or tagging operation.

**Words:**

- The 'words' resource contains a list of words in English. It can be useful for various linguistic operations, including vocabulary analysis and text processing. Shallow parsing may involve working with words, so having access to a comprehensive list can be beneficial.

**Maxent_ne_chunker:**

- Named Entity Recognition (NER) is a task often associated with shallow parsing. While the 'maxent_ne_chunker' resource is primarily for NER, it shares some components with POS tagging and syntactic analysis, which are relevant to shallow parsing.

**Averaged_perceptron_tagger:**

- Similar to 'maxent_treebank_pos_tagger', this is another part-of-speech tagger model. It's trained using the averaged perceptron algorithm, and it can be used for assigning part-of-speech tags to words. Accurate POS tagging is crucial for shallow parsing tasks.

Deep parsing aims to provide a more comprehensive and detailed analysis of a sentence's grammatical structure.

- We import spacy for deep parsing.
- We load the model named “spacy_en_core_sm” to deep-parse sentences.
- “spacy_en_core_sm” is a model, where the sm indicates small, where the smaller lightweight models are downloaded.
- The spacy_en_core_sm model is designed for various NLP tasks, including tokenization, part-of-speech tagging, named entity recognition, and dependency parsing.
- Put through each sentence through the nlp() function.
- We parse the following from the words:
    - word: The original word.
    - lemma: The root of the original word.
    - pos: The part of speech tag of the word.
    - dep: Dependency, refers to the syntactic dependency relationship between the token and its parent in the parse tree or dependency tree.
    - head: represents the head of the token to which the current token is syntactically related in the parse tree.

Libraries and tools used:

1. nltk (for shallow parsing)
2. spacy (for deep parsing)
3. Pandas (for loading CSV files)