**WORDNET**

**What is WordNet?What is this project about?**

WordNet is a semantic lexicon for the English language that computational linguists and cognitive scientists use extensively. For example, WordNet was a key component in IBM’s Jeopardy-playing Watson computer system. WordNet groups words into sets of synonyms called synsets

Here our task is to build a wordnet digraph:each vertex *v* is an integer that represents a synset, and each directed edge *v→w* represents that *w* is a hypernym of *v*. The WordNet digraph is a *rooted DAG*: it is acyclic and has one vertex—the *root*—that is an ancestor of every other vertex. However, it is not necessarily a tree because a synset can have more than one hypernym.

**STEPS:**

The following are the steps that I have done to complete the project

WordNet.java:Complexity-O(n)

1.In the beginning, we have files which contain raw synsets and hypernyms.we have divided and parsed those raw synsest and stored it on a data structure.

2.For storing I have used three hashmaps

1.For storing synsets Id as a key and nouns as values

2.For storing nouns as a key and Id as values

3.For storing hypernyms

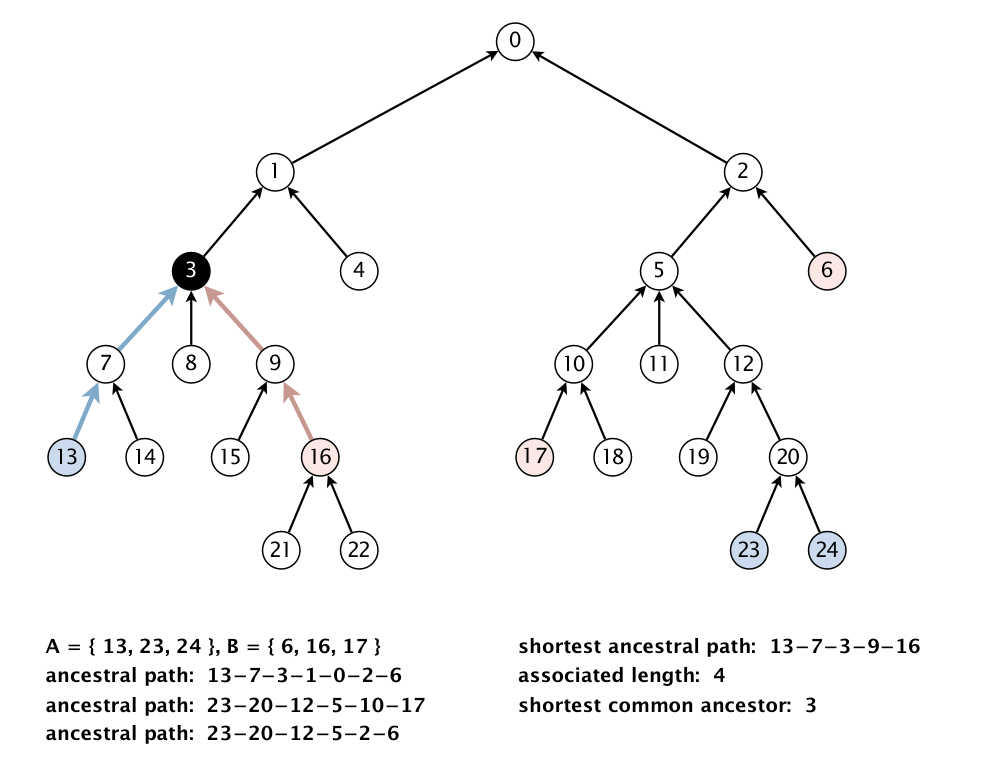
3.After parsing, we have added connections between hypernyms

4.For the other two distance and ancestor methods first we check whether the given nouns are part of synsets by using our hashmap then call the required function in sap

SAP.java:Complexity-O(n)

1.In SAP.java we have calculated length between two nouns by finding the common ancestor between two nouns

2.Ancestor method checks whether two nouns contains common path.If it contains, then it will retrieve the shortest common path otherwise it returns -1.



Outcast.java:Complexity-O(n)

In outcast.java we find the word which is least related to others.

% cat outcast8.txt

water soda bed orange\_juice milk apple\_juice tea coffee

O/P:bed

Implementation Code:

Link:

<https://github.com/Manasa81/ADS2_2019501081/tree/master/Day5>

Score:

