Homework 1

Questions

1. For each search algorithm:
    - Is the first path that it finds guaranteed to be the shortest path?
    - How efficient (time and space) is this algorithm for finding paths between words that are fairly similar (>50% characters in common)?
    - How efficient (time and space) is this algorithm for finding paths between words that are not similar (<50% characters in common)?
    - How efficient (time and space) is this algorithm for determining that there is no path between the two words?

Algorithm	Shortest Guaranteed?	Similar Time	Similar Space	Unsimilar Time	Unsimilar Space	Unsolvable Time	Unsolvable Space
BFS	Yes	Efficient	Inefficient	Efficient	Inefficient	Inefficient	Inefficient
DFS	No	Inefficient	Efficient	Inefficient	Efficient	Inefficient	Efficient
IDS	Yes	Efficient	Efficient	Efficient	Efficient	Inefficient	Efficient
A*S	Yes	Depend on Heuristic	Inefficient	Depend on Heuristic	Inefficient	Inefficient	Inefficient

2. For the A* search, how did you choose an appropriate heuristic?

> I compared `Hamming Distance`, `Levenshtein Distance`, `Matching Prefix Length`, and `Jaccard Similarity`. First, I ruled out the last two distances, since they both ignored certain informations in the compared strings, resulting in a much similar heuristic. Aferwards, I chose Hamming Distance over Levenshtein Distance, since they give the same heuristic when the start word and end word are of the same length, and Levenshtein Distance takes a much longer time to calculate.

3. Suppose there is a set of words that you must include at some point in the path between the start_word and the end_word. The order of the words does not matter. How would you implement an algorithm that finds the shortest path from the start_word to the end_word which includes every word in the given set of words? You may describe the algorithm or provide pseudocode.

> Steps:
>
> 1. Calculate the distances betweeen each internal word using BFS
> 2. Calculate the distances between start_word and each internal word using BFS
> 3. Calculate the distances between each internal word AND end word using BFS
> 4. Construct a graph
> 5. Use Dijkstra's algorithm to find the shortest path.

4. How long did this assignment take you? (1 sentence)

> 6 hours

5. Whom did you work with, and how? (1 sentence each)

> No one. Just myself.

6. Which resources did you use? (1 sentence each)

> Class slides

7. A few sentences about:
    - What was the most difficult part of the assignment?
        > Finding difference between different heuristic algorithms.
    - What was the most rewarding part of the assignment?
        > Using the same structure of code to implement all 4 search algorithms.
    - What did you learn doing the assignment?
        > First time implementing DFS using the iterative approach.