CS 564 Database Management Systems

Kirthanaa Raghuraman

January 27, 2017

Assignment 1 : Word Locator

Design Report

Question 1 Explain your choice of the data structure that you implemented. Did you consider any other data structures besides the one that you implemented? How did you arrive at your final choice of the data structure?

The data structure I have implemented is a trie. Trie is a natural choice for this design problem as each node in a trie contains the prefix string and the node can branch out depending on the letters that make up that word. This is very efficient look up as it saves memory in not having to store shared prefixes repeatedly and is pretty fast as the child nodes are indexed. A radix tree is a sophisticated form of the prefix tree but I felt the implementation was complicated so I settled for the trie.

Question 2 What is the best, average, and worst case complexity of your implementation of the locate command in terms of the number of words in the file that you are querying? (you need to provide all three - best,average, and worst-case analysis). For the complexity, I am only interested in the big-Oh analysis.

Worst Case Analysis:
The worst case is accessing a string of length N. In this case, the *locate* command will take N indexing operations and hence the worst case will be *O(N)*.

Best Case Analysis:
The best case is a word consisting of a single letter. It can be obtained in the first indexing and hence the time complexity would be *O(1)*.

Average Case Analysis:
The average case is the same as the worst case, i.e. *O(N)*. We will have to perform N indexing operations for a word of length N and hence the average case is the same as the worst case.

Question 3 What is the average case space complexity of your data structure in terms of the number of words in the input file? In other words, using the big-Oh notation what is the expected average size of your data structure in terms of the number of words.

Assuming there are N nodes in the trie and the number of alphabets is K, then the space complexity is *O(N*K)*. However, it is not possible to determine the number of nodes from the number of words in the file as it depends on several factors , i.e.

- Number of letters in the word
- Distribution of letters in the word
- Number of words with common prefixes.

The trie I've implemented has 37 alphabets (26 alphabets + 10 digits + 1 apostrophe). So space complexity in terms of number of nodes is *O(37N)*.