ECE385: Homework 03

Farbod Mohammadzadeh
(1008360462)
Connor Ludwig
(1008333028)

13 October 2023

5 Pages

Problem 1

The best way to check which data structure is best for a dictionary is to see which data structure has the best time complexity for the operations.

Linked List

Linked lists are a time inefficient data structure for a dictionary, because they have a time complexity of $O(n)$ for *insert* and *rank*, since you have to go through the entire list each time. Additionally, *search* and *delete* could be $O(n)$ if the list is not sorted, or $O(\log n)$ if it is sorted. However, they do provide a space efficient data structure, since they only require $O(n)$ space.

Array

Arrays are a space inefficient data structure, while they do require $O(n)$ space, it is a non-sparse data structure, so it is not as space efficient as a linked list. However, arrays are time efficient for a dictionary, since they have $O(1)$ time complexity for *insert*, *delete*, and *search*. However, *rank* is $O(n)$, since you have to go through the entire array to find the rank, which is worse than a linked list since the structure could be sparse and have a lot of empty space.

Hashmap

A hashmap is a space efficient data structure, since it can be sparse. However, it is not as space efficient as a linked list. *insert*, *delete*, and *search* are all $O(1)$, which is the best time complexity for a dictionary. However, *rank* is $O(n)$, since you have to go through the entire hashmap to find the rank, which is worse than a linked list since the structure could be sparse and have a lot of empty space.

Self-Balancing Binary Search Tree

A self-balancing BST would provide great time complexity for *insert*, *delete*, *search*, and *rank*, since they are all $O(\log n)$. However, it is more space inefficient than a linked list.

Problem 2

Problem 3

Problem 4