Leah Benitez

Professor Fakhouri

CMP 338

October 6, 2020

Homework 3 Analysis

1. **Create graphs from the test times you measured.**
2. **Discuss the difference in run times between the array based list and the linked list for all the test cases. Did the results match your expectations? Why or why not? Be as specific as possible.**

It seems like the array based list and the linked list generally had similar test times, but there were definitely cases where the array based list did better. This is especially evident in the average graph that I included.

I actually thought that the linked list would be more efficient than the array based list because of its use of nodes. Conceptually, it seemed like they would make it easier to do functions like adding and removing new items because all you have to do is update heads and tails. However, now I understand why in some cases, the array based list is more efficient. You have the ability to access specific indices with an array and you cannot do that with a linked list.

1. **Are the run times ever similar for both the array based list and the linked lists? Why or why not? Be as specific as possible.**

Outside of some outliers, there were cases where the array based list and the linked list had similar test times. More specifically: AddSortedOdd, AddSortedEven and AddAll. I believe the runtimes were similar in these cases because the new elements were simply being added to the end. For each list, this is done in different ways but the general idea is the same.

1. **In which test cases is the array based list faster than the linked list? Please explain the reason why or why not. Be as specific as possible.**

The array based list did better than the linked list in these test cases: RemoveAllOdd, RemoveAllEven (kind of) and AddAllAtIndexZero. I am actually surprised about AddAllAtIndexZero. I would have thought the linked list would do better because adding elements to the beginning of the list is just a matter of updating the head of the list, whereas with the array based list, you have to shift the whole array in order to insert a new element at index zero.

But in RemoveAllOdd and RemoveAllEven, I think the array based list was faster than the linked list because of the ability to access a specific index. You also do not have to shift the whole array to the left in most cases. You just have to shift the indices after the one you removed.

In the linked list, you do not have to shift anything but you do have to search through the nodes to find the one you are trying to remove.