Anik Barua 04-18-2021

Homework 3.1 - Explanation of Performance of STL Containers

We know a Vector is a dynamic array, a List is a linked list, a Set is a self-balancing binary search tree and an Unordered Set is a hash table. Based on the data structure implemented, the time complexity will vary with insertion (front and end) and search for each type of container.

As we can see, for insertion at the end, the relative percentage for list to vector is 254.545%. This is 154.545% slower compared to vector. It is because list's time complexity of insertion at the end is O(n) compared to the vector which is O(1). For vector the spaces are already made so it is faster compared to list where it creates spaces as its adding elements. That's why the insertion is faster in vector compared to list. Next, the relative percentage of set to vector for inserting at the end we have 854.545%, i.e. it is 754.545% slower compared to vector. This is because the time complexity of set which is a binary search tree is O(logn) + rebalanced time (time for sorting). Compared this with the vector which is O(1), we can see why the vector is faster compared to set. Now for relative percentage of insertion at end in unordered set to vector we get 527.273% which is 427.273% slower than vector. It is because for unordered set the time complexity is O(n) (worst case) compared to O(1) for the vector. That's how we are getting these results for insertion at end.

For insertion at begin, the relative percentage for list to vector is 2.8536%. This time list is faster because the time complexity is O(1) for insert at begin compared to vector which is O(n). Next, the relative percentage for set compared to vector is 10.5459%. It means set insertion at begin which is a binary search tree is 10.5459% faster because the time complexity is O(logn) compared to vector which is O(n). Then for unordered set which is a hashtable has a relative percentage of 5.08685% at end, i.e. it is 5.08685% faster. This is because the time complexity of unordered set (insertion at begin) is O(1) (average case) compared to O(n) for the vector. That's how we are getting these results for insertion at begin.

For find, the relative percentage for list to vector is 124.683% i.e. 24.683% slower compared to vector. It is because the time complexity of find in list which is a linked list is O(n) compared to vector which is O(logn). Next for the relative percentage for set compared to vector we have 0.0362713%. Set is faster because for find in set and vector, even though the time complexity is the same, O(logn), set is sorted i.e. balanced. Then for an unordered set which is a hash table, has a relative percentage of 0.0181375% compared to vector. This is because the time complexity for the unordered set is O(1) compared to O(logn) for vector. That's how we are getting these results for find in different containers.