Report for Algorithms & Analysis Assignment 1

Experimental Setup

Data Scenarios –

The data that we used for our test cases were-

1) Sorted numbers
2) Unsorted numbers
3) Unordered strings

For addition and removal, the following ratios were used:

1) Only addition- where items were only added. We added 200 items in this case. Therefore, there was no addition to removal ratio.
2) Equal number of additions and removals- where approximately equal number of items were added and removed. The number of additions and removals are 200 each. So, the ratio is 1:1.
3) Only removals, where 200 items are removed from the multiset.
   The reason to choose this ratio was to show the difference in performance of each data structures, namely, linked list, sorted linked list, BST, hash and balance tree, in terms of time they take to compute the results. The reason that we chose the size to be 200 was to see the significant difference in performance of each of the above-mentioned data structures.

For searches, we followed the following ratios:

1) With addition only, the addition to search ratio we have taken is 1:2 to show how efficient the search for a large set of data is in each data structure mentioned above.
2) With removals only, the addition to search ratio is again 1:2 for the same reason as above.
3) For equal number of additions and removals of items, the ratio of (Addition and removal) to search is 1:2 to check for the efficiency of finding a large set of items in the data structures mentioned above

Multiset Sizes – We used a big size of 200 for the multiset for adding and deleting the items to and from the multiset. We used this size because this size allowed us to contrast the performance for adding, deleting and searching operations for the data structures mentioned above.

Generation of scenarios –

We decided to generate these scenarios based on the basic utility that the above mentioned data structures are normally used for.

Timing –

We used the function *System.nanotime()* to compute the time. We performed 5 tests each for each generated data set and selected the best time for a given data structure.

Fixed sets – Which approach did you use to generate these? Briefly describe.

Evaluation

Scenario 1 (varying ratio of additions to removals)

Scenario 2 (varying number of searches vs additions/removals)

Sorted numbers:

Addition only-

Linked list run time = 0.446402329 sec
Sorted Linked List = 0.662534417 sec

Bst = 0.411111313 sec

Hash = 0.495257956 sec

Bal tree = 0.392409331 sec

Equal additions and removals-

Linked list = 0.54062171 sec

Sorted linked list = 0.469854284 sec

Bst = 0.394162819 sec

Hash = 0.470513918 sec

Bal Tree = 0.380045431 sec

Removals only-

Linked list = 0.113773941 sec

Sorted Linked list = java.lang.NullPointerException

Bst = 0.202461754 sec

Hash = 0.096557364 sec
Bal Tree = 0.010073492 sec

Additions + Double the search-

Linked list = 0.115436809 sec
Sorted linked list = 0.181444822 sec
Bst = 0.051560997 sec

Hash = 0.038589945 sec

Bal tree = 0.034979135 sec

Removals only+double the search-

Linked list = 0.36680101 sec

Sorted Linked List = java.lang.NullPointerException

Bst = 0.180906013 sec

Hash = 0.022925983 sec

Bal Tree = 0.039601863 sec

Addition + Removals + Double the search-

Linked list = 0.064089146 sec

Sorted linked list = 0.049372158 sec

Bst = 0.34507347 sec

Hash = 0.337868086 sec

Bal Tree = 0.220111039 sec

Removals + Double the search-

Linked list = 0.397920886 sec

Sorted linked list = Null pointer exception

Bst = 0.034404834 sec

Hash = 0.027751473 sec
Bal Tree = 0.035284598 sec

Unsorted numbers:

Addition only-

Linked list = 0.232417925 sec

Sorted linked list = 0.255140014 sec

Bst = 0.018629863 sec

Hash = 0.034404079 sec

Bal Tree = 0.020148118 sec

Equal additions+removals:

Linked list = 0.343275805 sec

Sorted linked list = 0.276494127 sec

Bst = 0.03419301 sec

Hash = 0.030854059 sec

Bal tree = 0.062092118 sec

Equal additions+ removals+ double search:

Linkedlist = 0.221351771 sec

Sorted linked list = 0.365945411 sec

Bst = 0.180975866 sec
Hash = 0.262007464 sec
Bal tree = 0.194706234 sec

Addition only + double the search-

Linkedlist = 0.258624713 sec

Sortedlinkedlist = 0.037008635 sec

Bst = 0.104649688 sec

Hash = 0.029347509 sec

Bal Tree = 0.048515426 sec

Unordered string:

Unordered string (addition only)

Linked list = 0.014980917 sec

Sorted linked list = 7.656353788 sec

Bst = 0.078045307 sec

Hash = 0.020179079 sec

Bal Tree = 7.940057831 sec

Unordered string (equal additions + equal removals)

Linked list = 0.024122161 sec

Sorted linked list = 8.871619703 sec

Bst = 0.54074518 sec

Hash = 0.014695088 sec

Bal Tree = 0.023972639 sec

Unordered string (removals only)

Linked list = 0.325532881 sec

Sorted linked list = Null pointer exception

Bst = 0.07197984 sec

Hash = 0.16128651 sec

Bal tree = 0.274153878 sec

Unordered string (additions only + double the search)

Linked list = 0.272959966 sec

Sorted linked list = 0.26256553 sec

Bst = 0.227302454 se

Hash = 0.275760109 sec

Bal tree = 0.244206015 sec

Unordered string (removals only + double the search)

Linked list = 0.292533782 sec

Sorted linked list = Null pointer exception

Bst = 0.303684137 sec

Hash = 0.275760109 sec

Bal tree = 0.259974189 sec

Unordered string (Equal Additions+removals and double search)

Linked list = 0.330604553 sec

Sorted linked list = 0.297747803 sec

Bst = 0.309014075 sec

Hash = 0.292574561 sec

Bal tree = 0.256366776 sec