German Education
2014-19498 Eunjoo Jung

Number 3-1 Performance Explanation

Q. Did performance for each operation (add(), remove(), and get()) differ depending on the index? Why is that?

```
               ArrayList      LinkedList      Index = 1
-------------------------------------------------------------
Add            |15378527      |69984

Get            |10357         |10033

Remove         |6978425       |18845
               ArrayList      LinkedList      Index = 49999
-------------------------------------------------------------
Add            |623332        |4197259

Get            |1612          |1741371

Remove         |573848        |1241927
               ArrayList      LinkedList      Index = 99999
-------------------------------------------------------------
Add            |1489          |5294682

Get            |1495          |4260339

Remove         |2861          |3215490
```

Each Operation(add(), remove(), and get()) differs depending on the index(1, 14999 or 99999). MyLinkedList is linear list, it has ‘head’ and ‘next’ Node. When we approach a specific index, the linkedlist implements this operation with head and next Node. For ‘i’th index of each operation of LinkedList such as get(i), add(i), remove(i), it takes O(i), so for the ‘n’ length of the list, then it takes about O(n) time complexity as a worst time Complexity. So, for each index 1, 14999 and 99999, as you can see above the results, the time Complexity gets increasing for the operation add, get and remove, and so forth O(n) time Complexity.

In case of Array, we can simply approach the element with index number, so the time Complexity of get() operation (for each 1, 14999 and 99999 index) is O(1), the constant time. However, for the add(i) and remove(i) it takes about O(n−i) time Complexity and as a result O(n) time Complexity for the worst case. Consequently, as you can see the results above, the time gets decreasing for the operation add and remove, and so forth O(n) time Complexity.

Q. How would a doubly−linked list implementation of myLinkedList change the performance?

‘myLinkedList’ is singly linked list and it uses just one direction Node which is head(linked with next), however, doubly linked list has two pointers, which are ‘prev’ and ‘tail’(previous and next). Therefore, it is possible that can be accessed in both sides of a Node, when the linked list is doubly linked list. On the other hand, ‘myLinkedList’ has only one direction from head to next. Since myLinkedList is linear List, it takes O(n) time Complexity for implementing the operations. Doubly linked list uses slightly a bit more memory because of the two way directions(next and previous) and still the time Complexity for the operations is also O(n). Because it is linked list, that can be operated with Node and next Node(or previous Node).

worst Time Complexity	add()	remove()	get()
Array Linear List	O(n)	O(n)	O(1)
Linked Linear List	O(n)	O(n)	O(n)

Number 3−2 Arrange the following functions in the non−decreasing order of their asymptotic growth

$$n^2(\log n)^6 < n^{2.5}, < 6^{\log n} < 2^{3\log n} < (\log n)^{(1/2)\log n} < n^{\log n} = 2^{2^{2\log\log n}} < 4^{\sqrt{n}}, < (1.1)^n < (\log n)^{n/3},$$