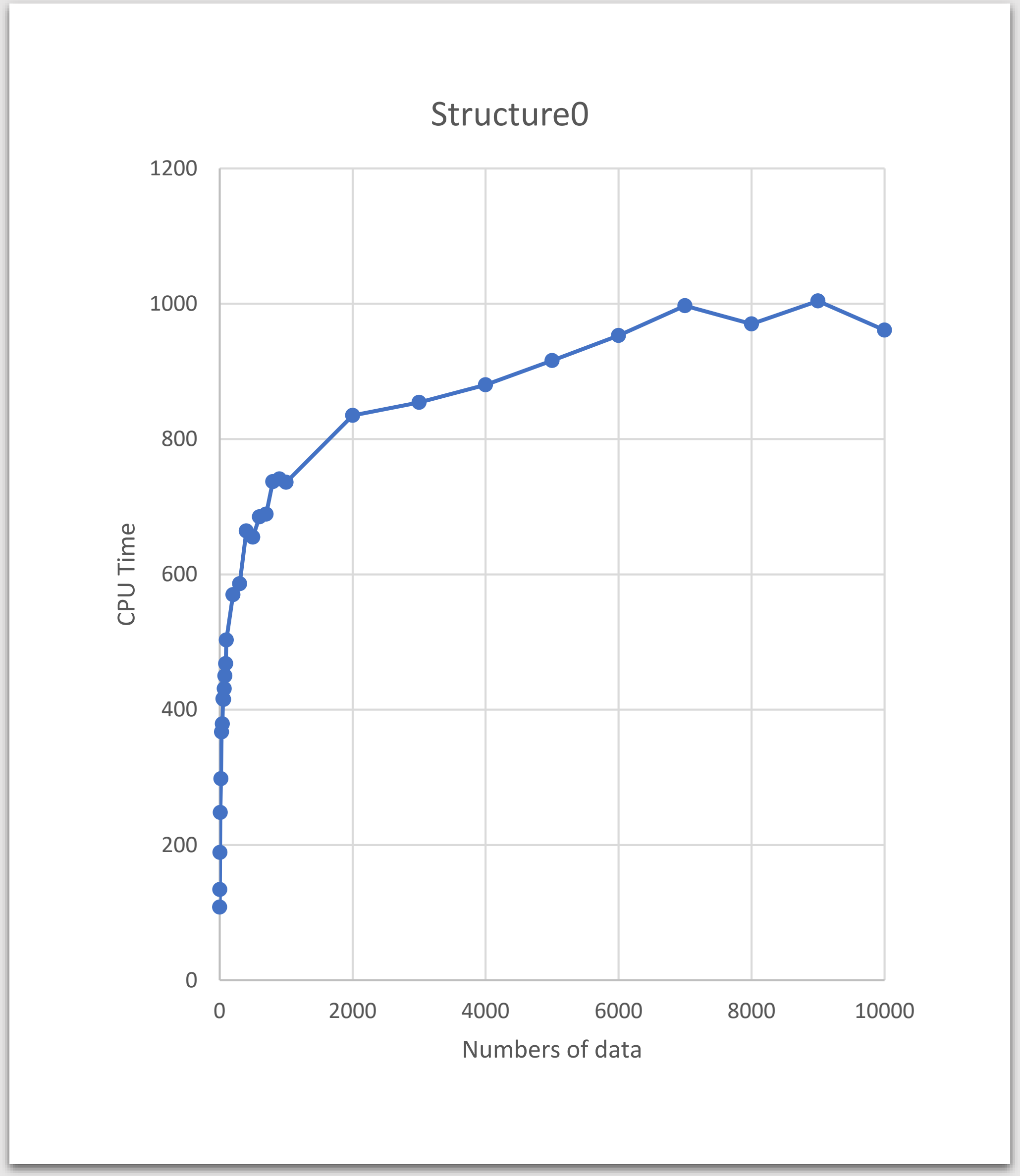
Project 4

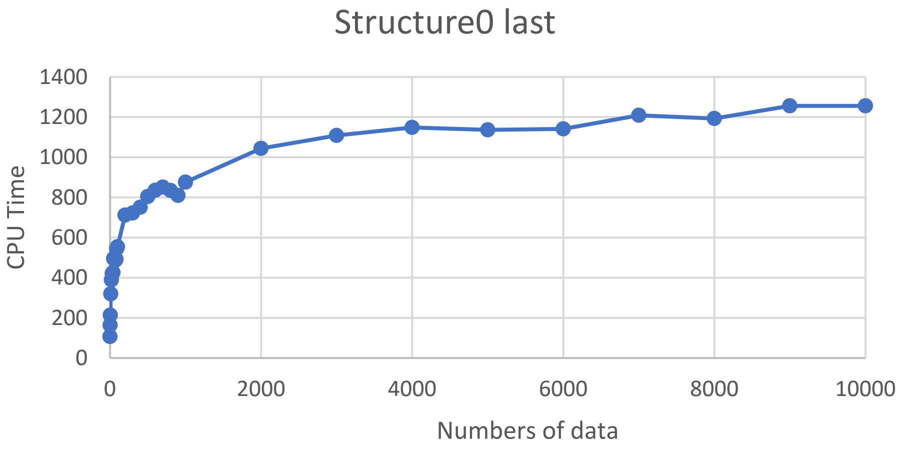
Kepei Lei & PJ Mara

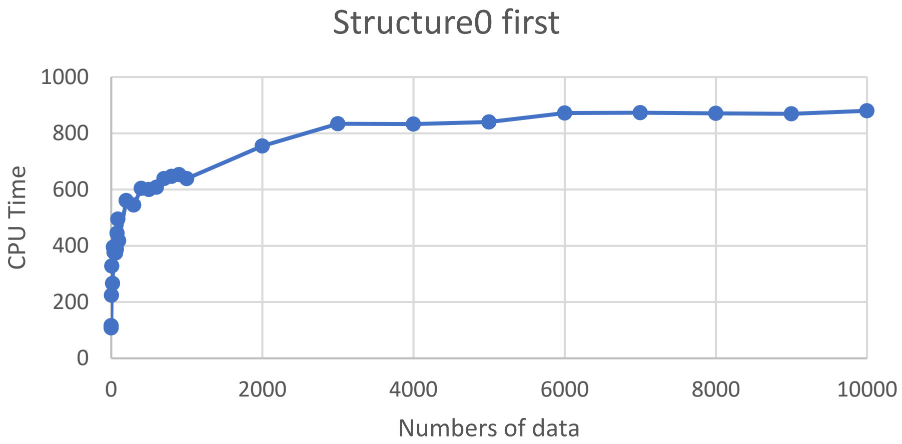


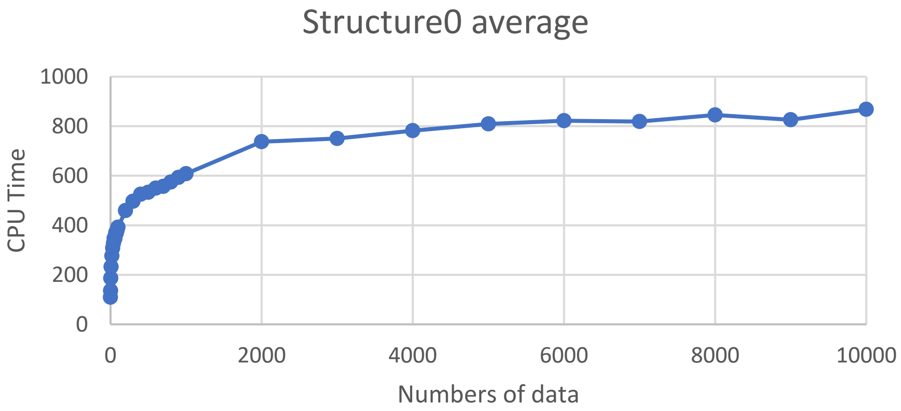
Structure 0: add

O(log(n))

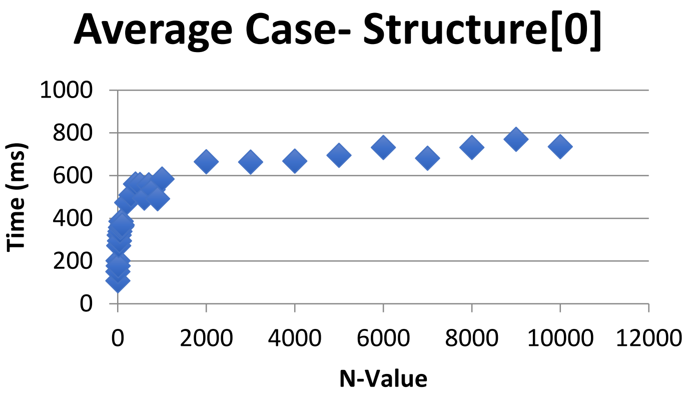
Structure 0: remove



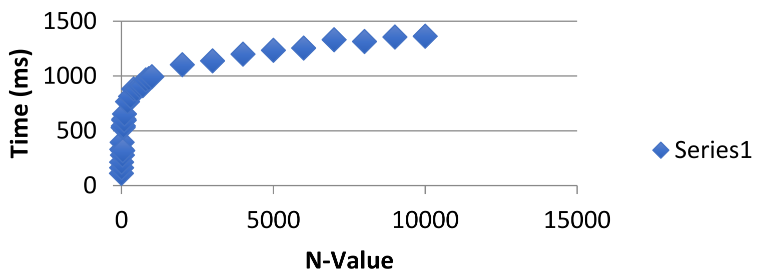




- Last: O(log(n))
- First: O(log(n))
- Average: O(log(n))


Average Case- Structure[0]

Biggest - Structure[0]



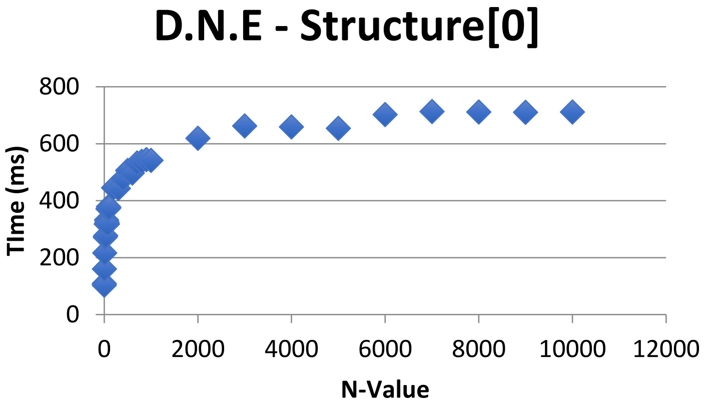
Structure 0: contains

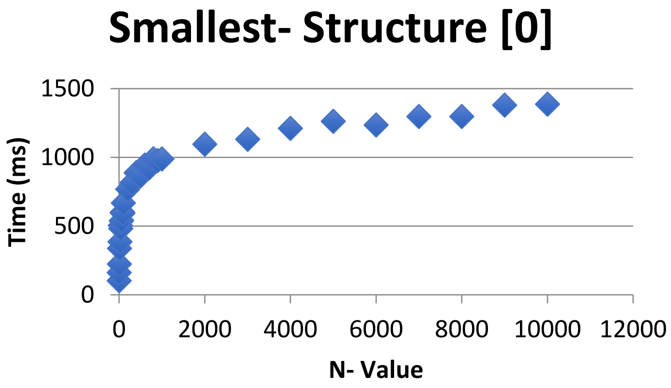
Average: O(log(n))

Last(biggest): O(log(n))

First(smallest): O(log(n))

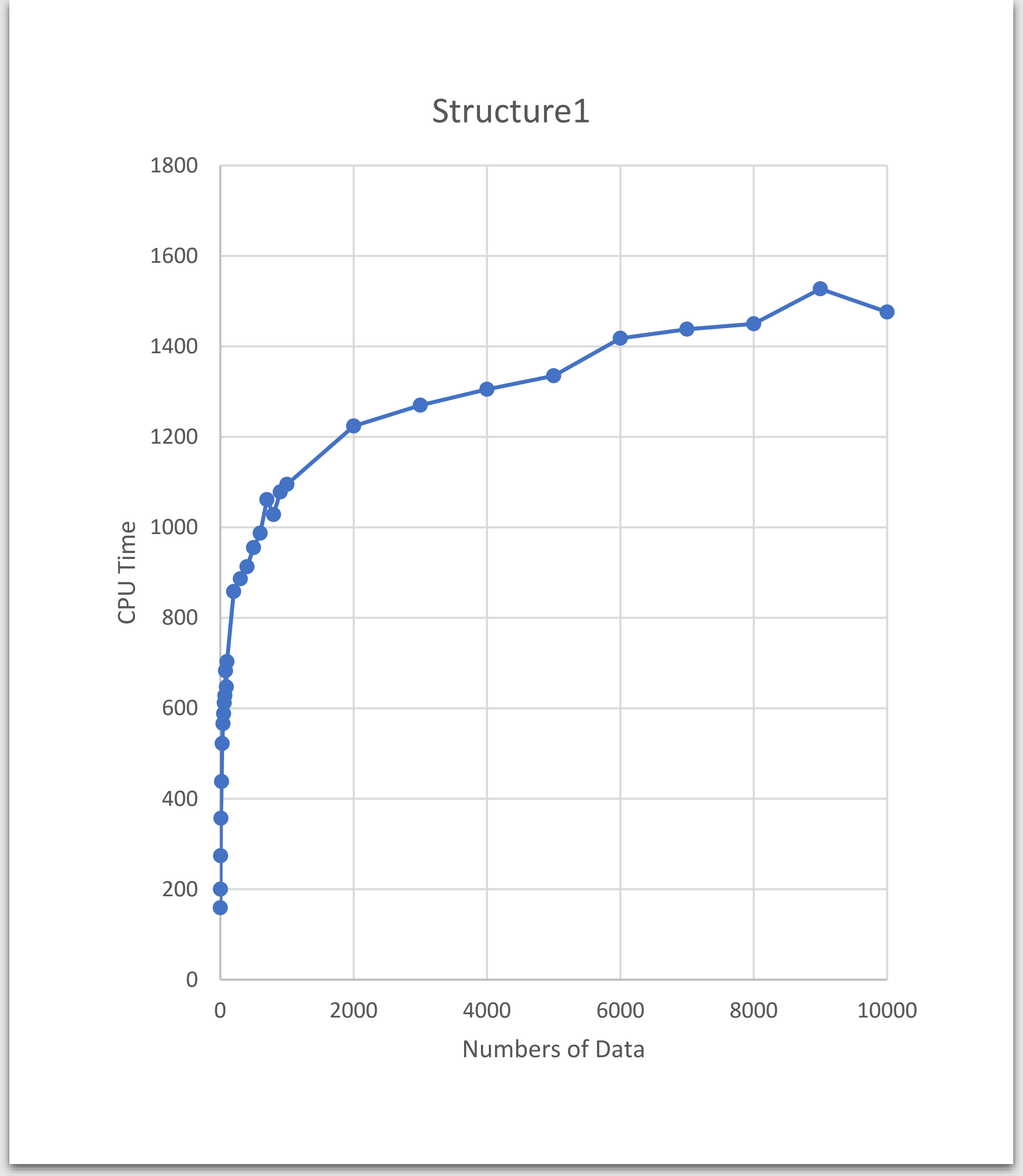
N+1(D.N.E): O(log(n))


D.N.E - Structure[0]


Smallest- Structure [0]

Conclusion

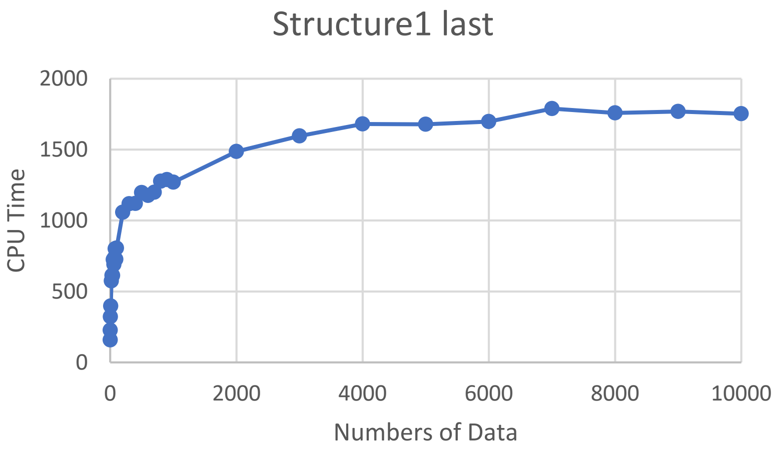
- Structure 0 is a self-balancing Binary Search Tree
- O(log(n)) everything

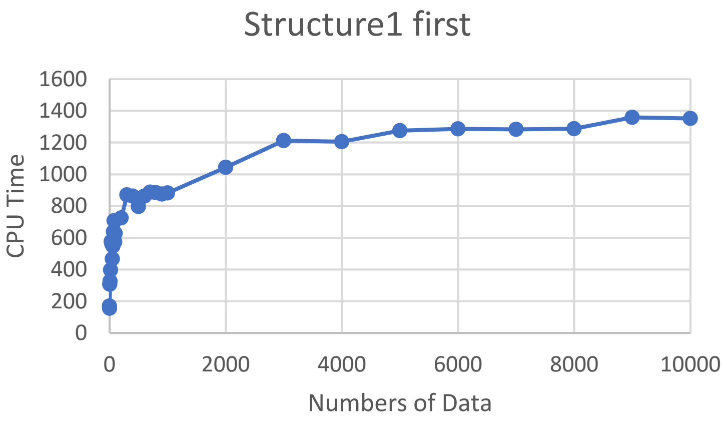


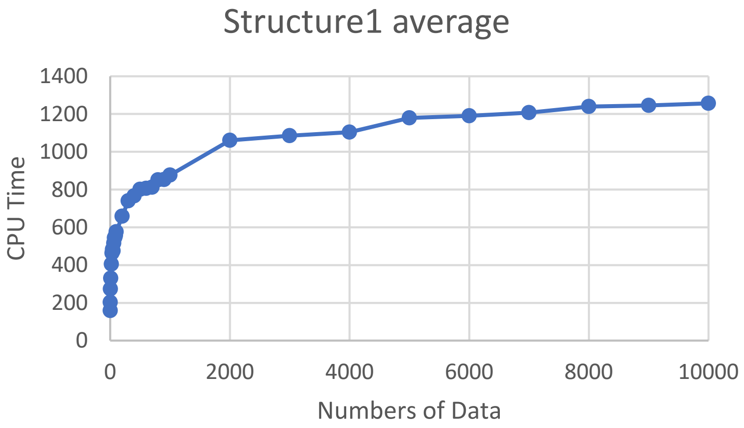
Structure 1: add

O(log(n))

Structure 1: remove



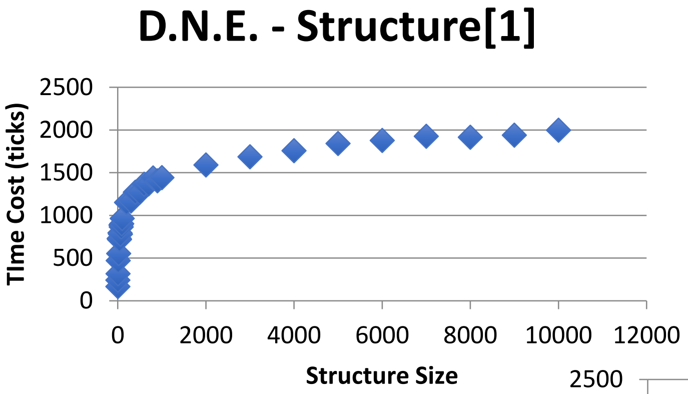




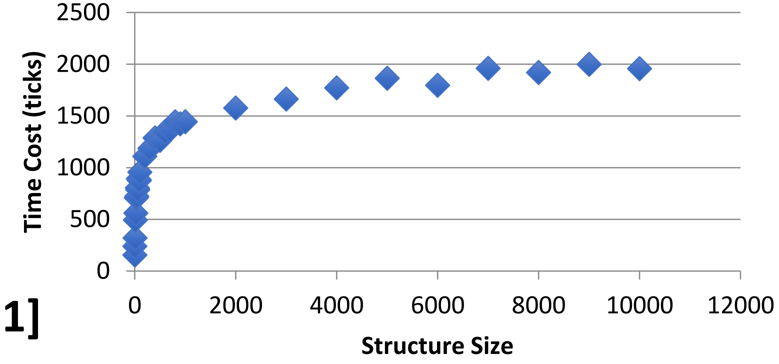
- Last: O(log(n))
- First: O(log(n))
- Average: O(log(n))

Structure 1: contains

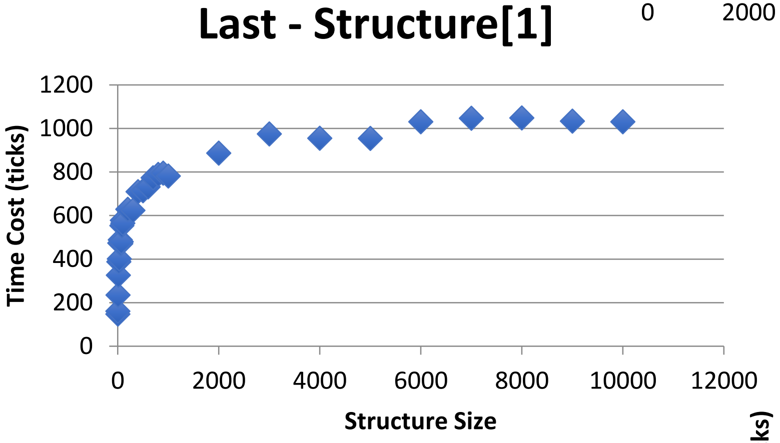
D.N.E. - Structure[1]



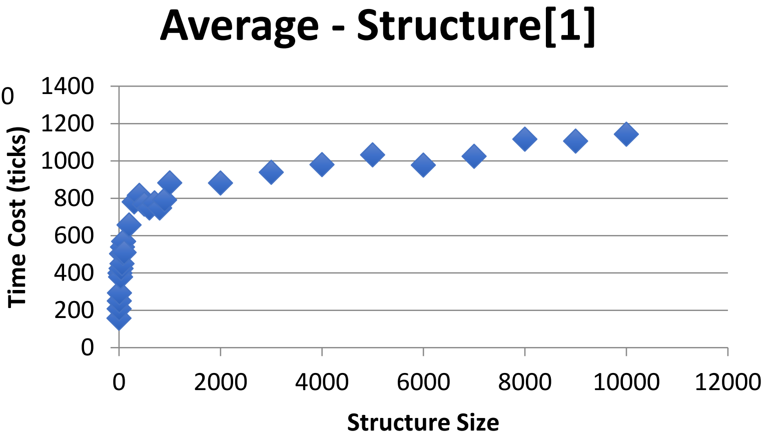
First - Structure[1]



Last - Structure[1]



Average - Structure[1]



Average: O(log(n))

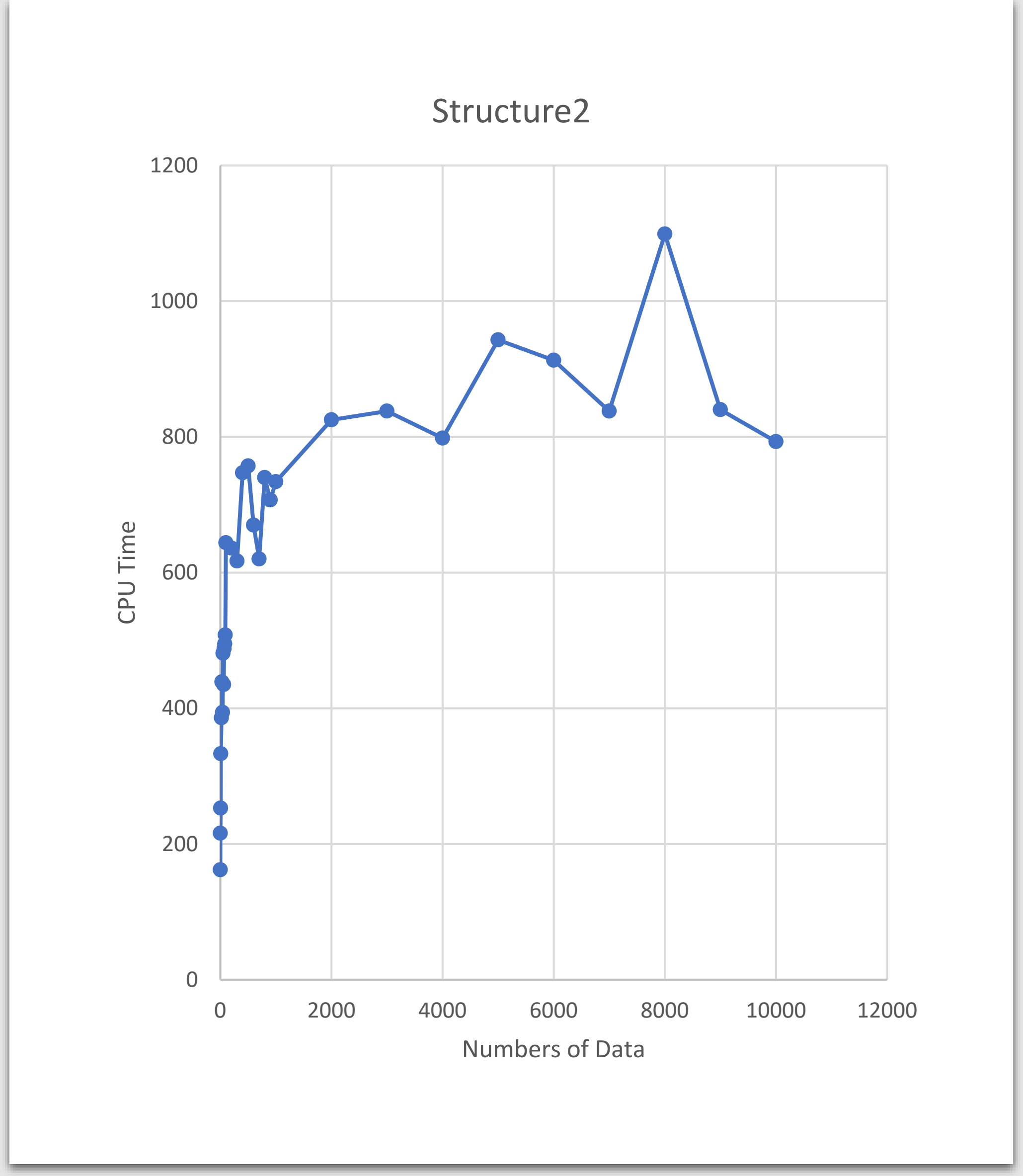
Remove last(biggest): O(log(n))

Remove first(smallest): O(log(n))

Remove N+1(D.N.E): O(log(n))

Conclusion

- Structure 1 is a self-balancing Binary Search Tree
- Still O(log(n)) everything
- Another one!?

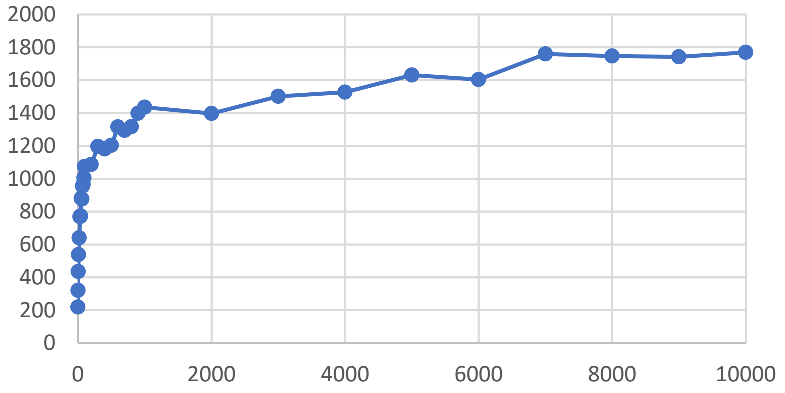


Structure 2: add

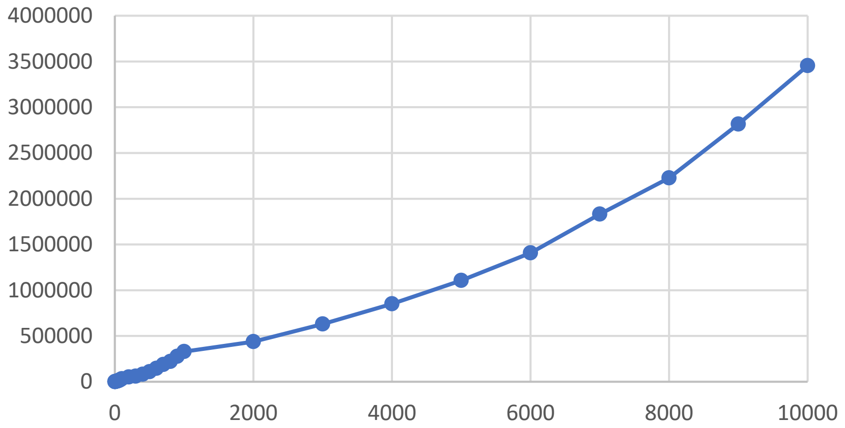
O(log(n))

Structure 2: remove

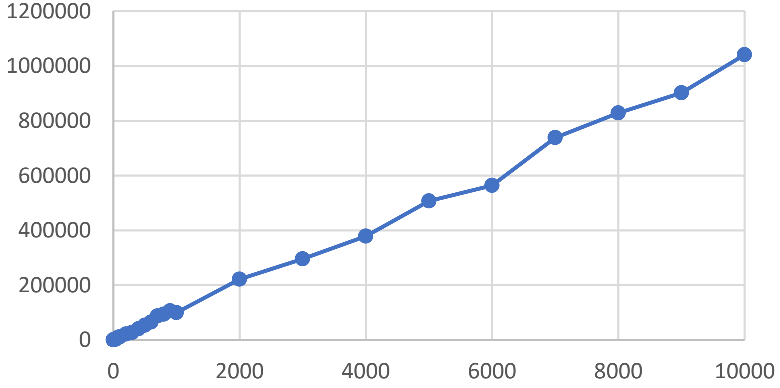
Structure2 last



Structure2 first

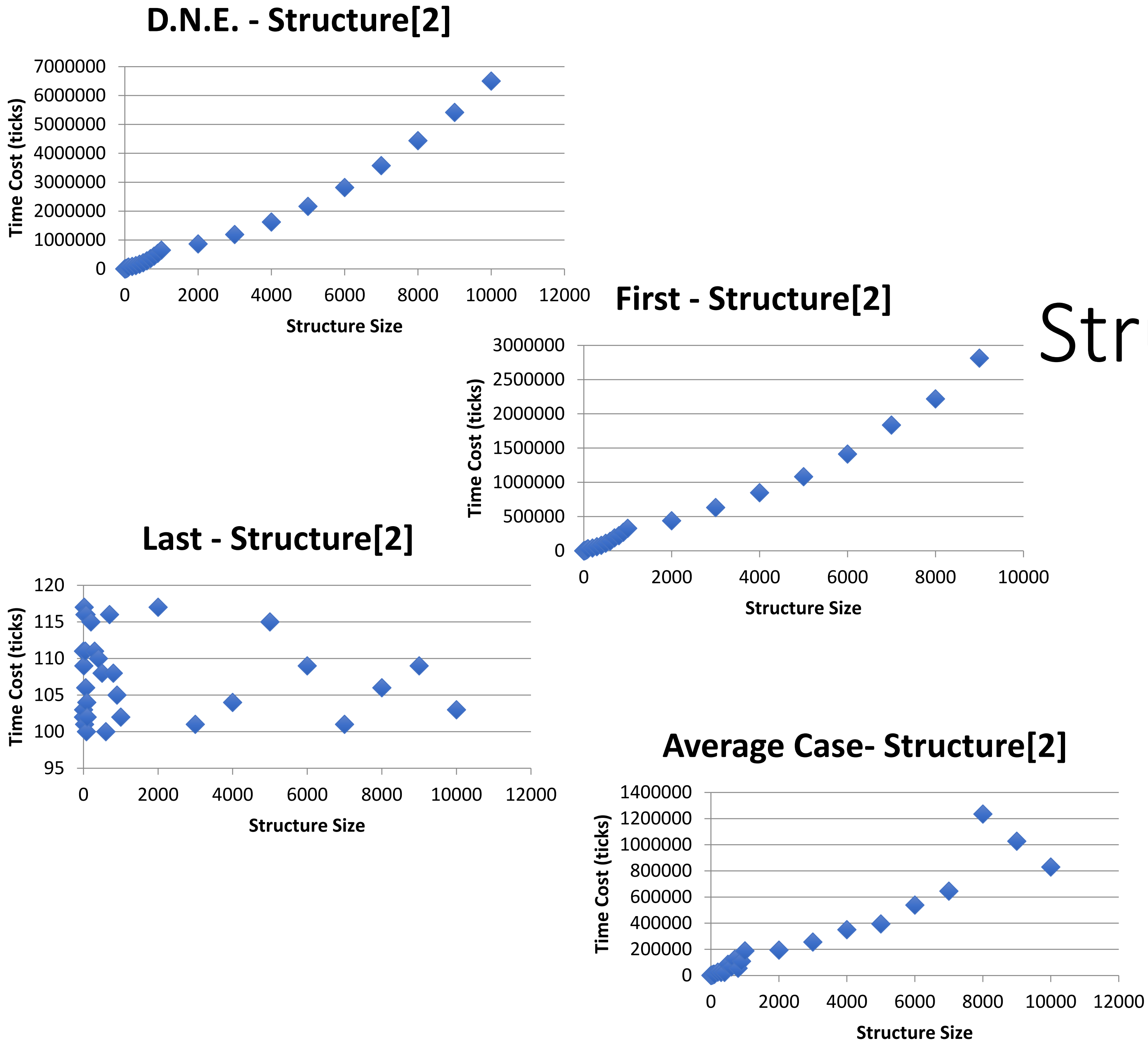


Structure2 average



- Last: O(log(n))
  - First: O(n)
- Average: O(n)

Structure 2: contains



Average: O(n)

(with a jump)

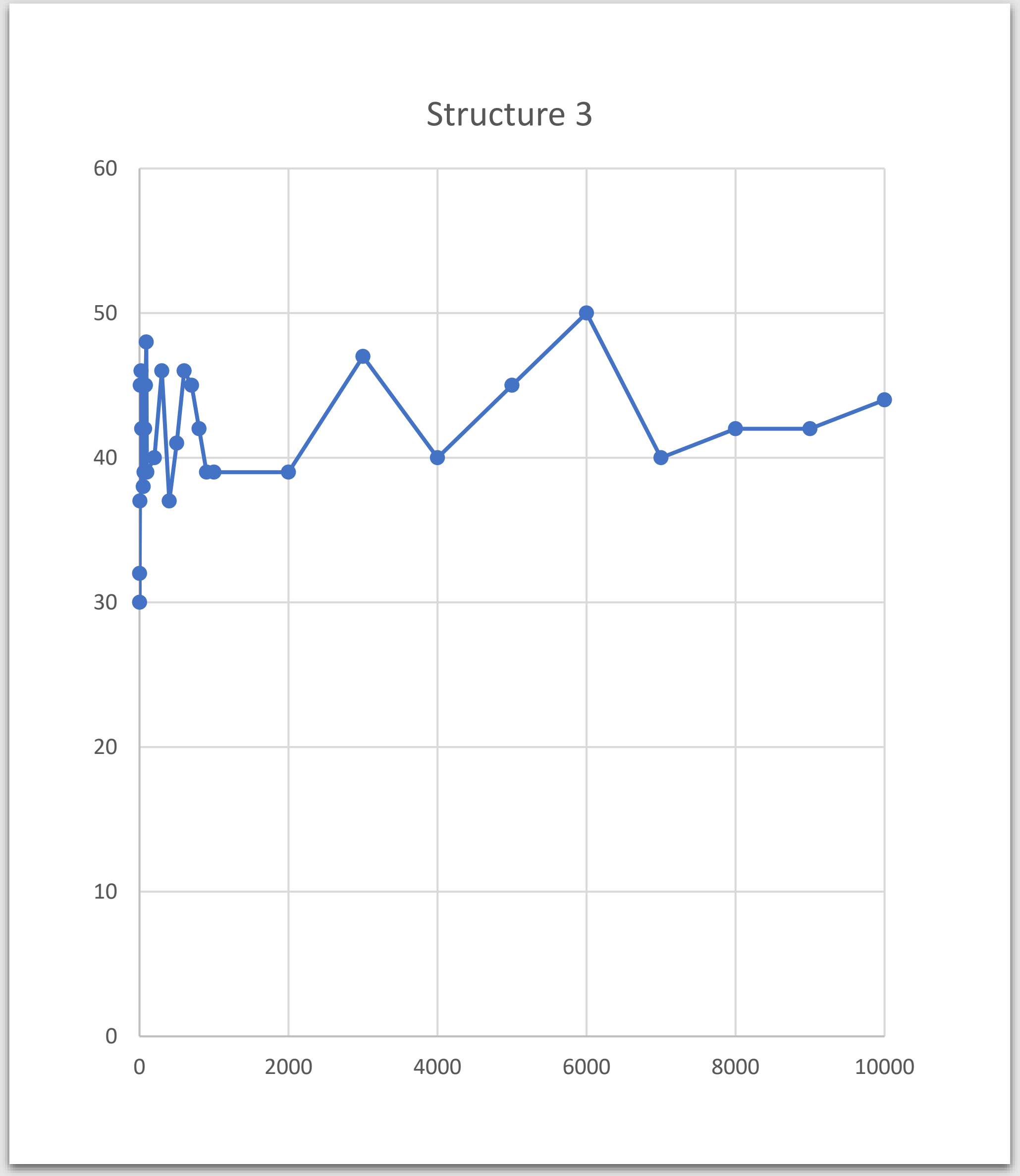
Remove last(biggest): O(n)

Remove first(smallest): O(1)

Remove N+1(D.N.E): O(n)

Conclusion

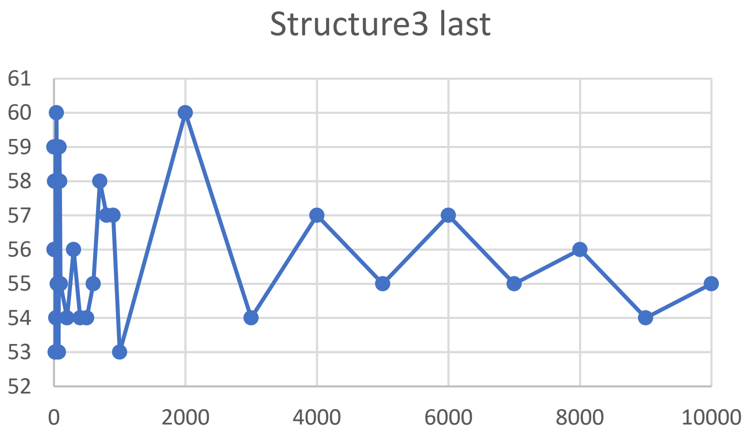
- Structure 2 is a Heap
- O(1) for finding the largest
- O(log n) for removing the largest
- O(n) for other find and remove cases

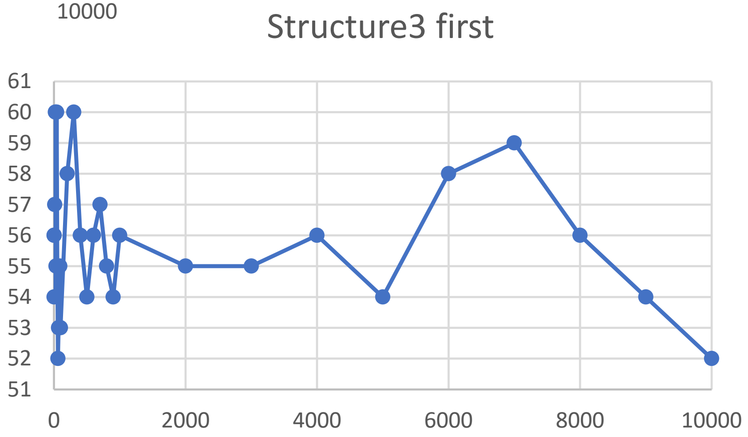


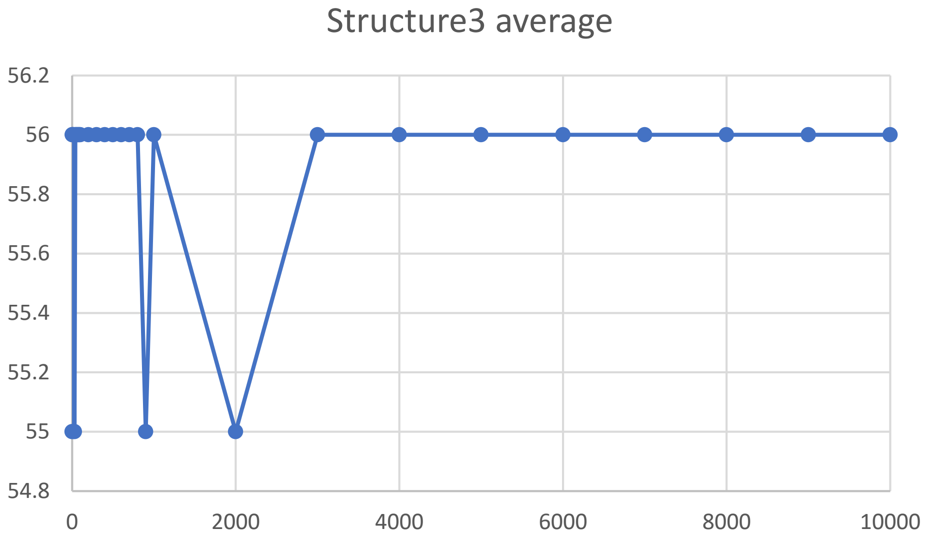
Structure 3: add

O(1)

Structure 3: remove



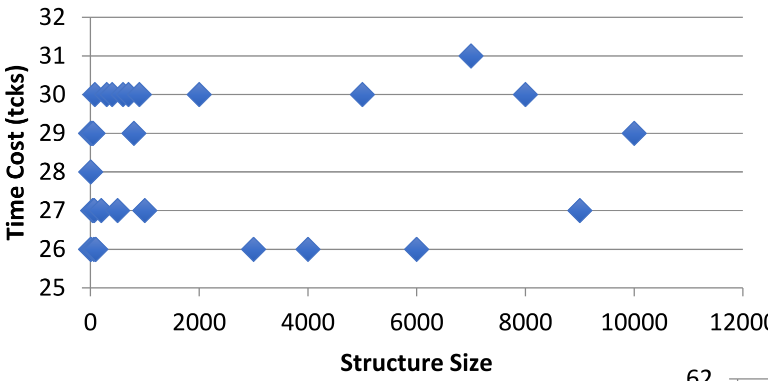




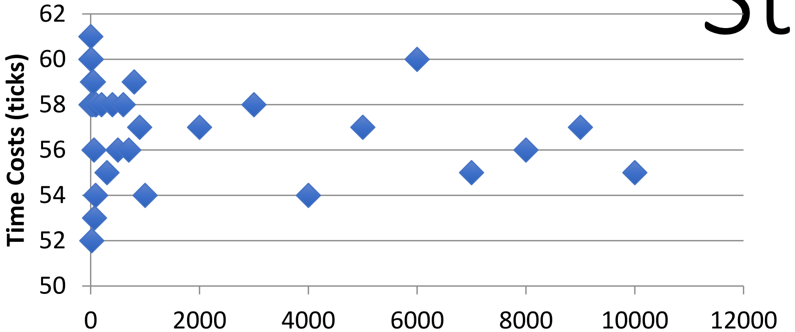
- Last: O(1)
- First: O(1)
- Average: O(1)

Structure 3: contains

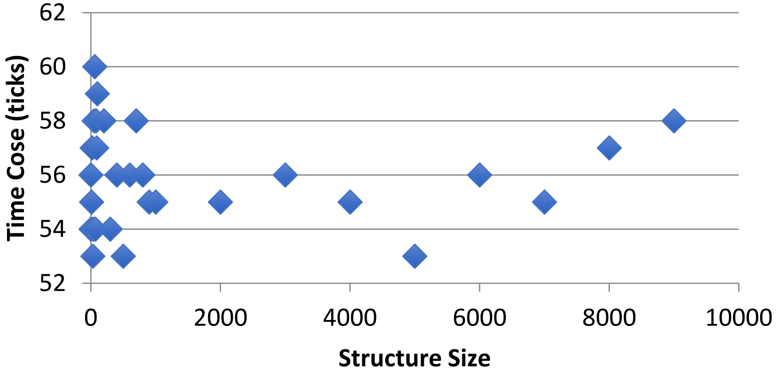
D.N.E. Case- Structure[3]



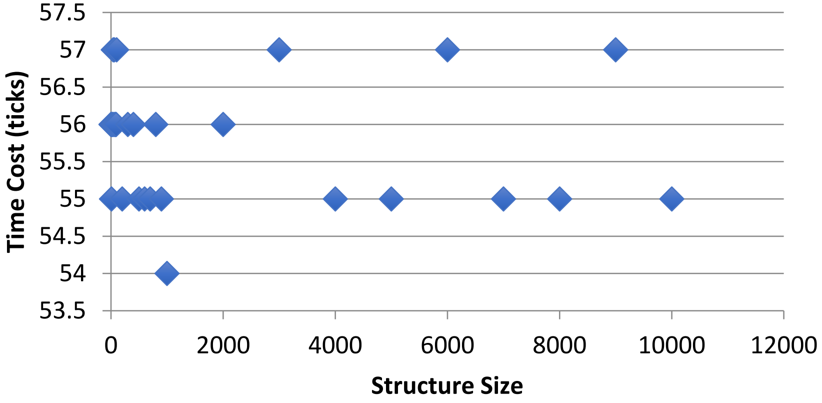
Last - Structure[3]



First- Structure[3]



Average Case- Structure[3]



Average: O(1)

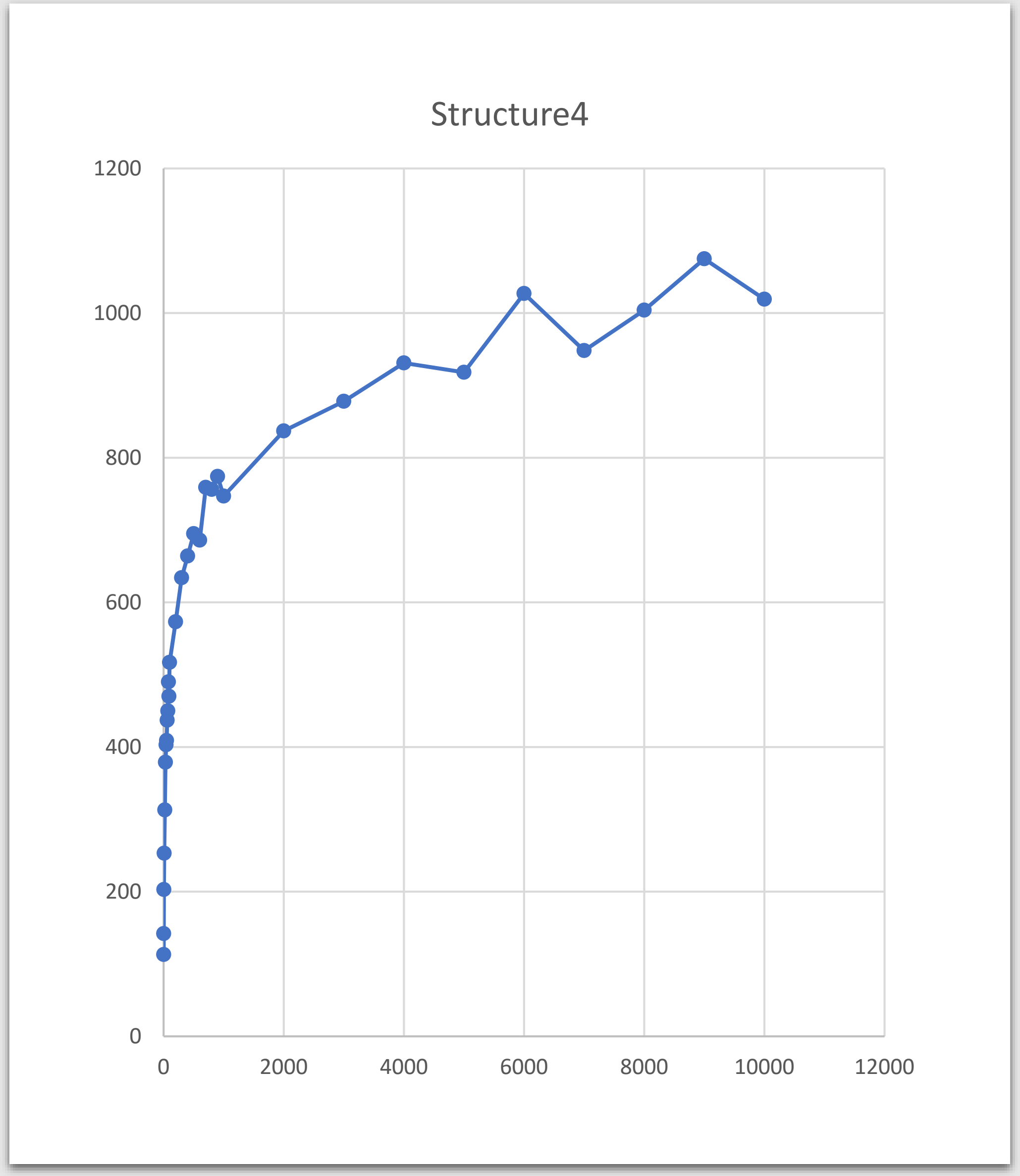
Remove last(biggest): O(1)

Remove first(smallest): O(1)

Remove N+1(D.N.E): O(1)

Conclusion

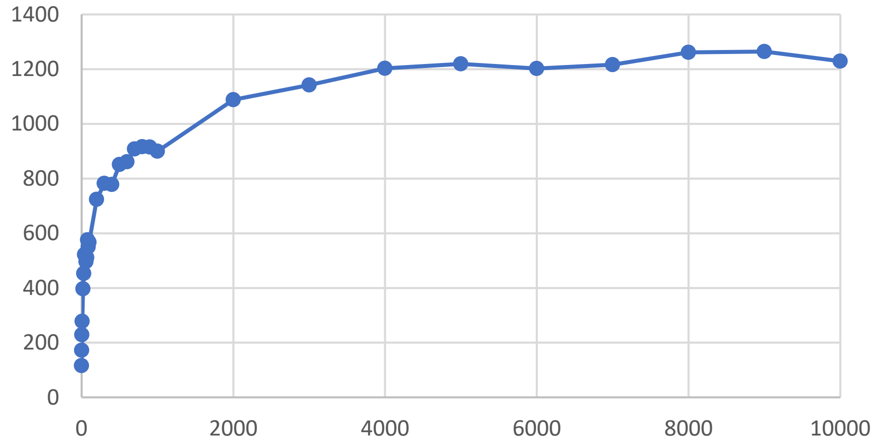
- Structure 3 is a HashSet
- Everything is O(1)!!!

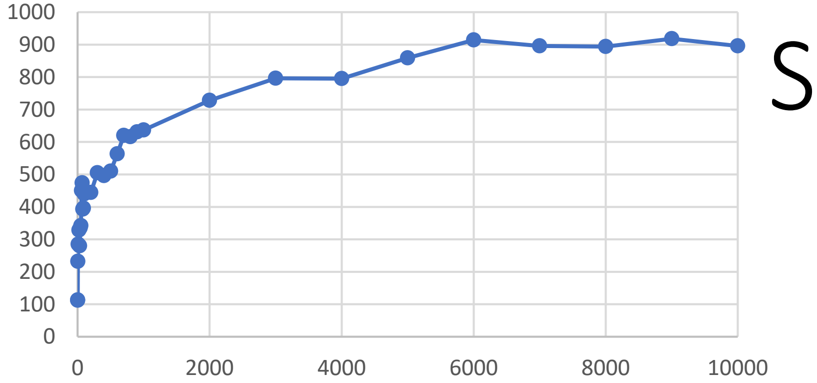


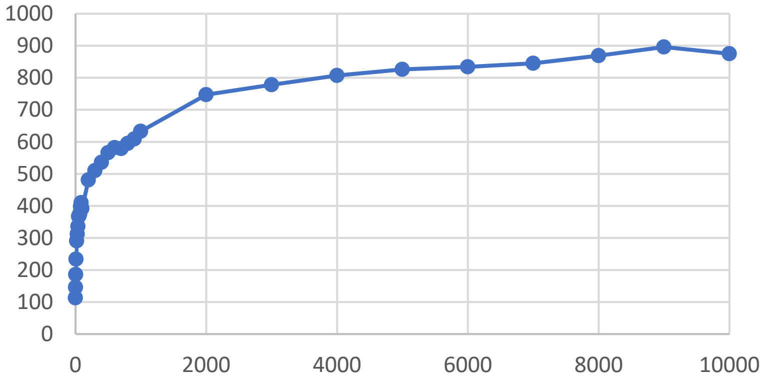
Structure 4: add

O(log(n))

Structure 4: remove


Structure4 last


Structure4 first


Structure4 average

- Last: O(log(n))
- First: O(log(n))
- Average: O(log(n))

Structure 4: contains

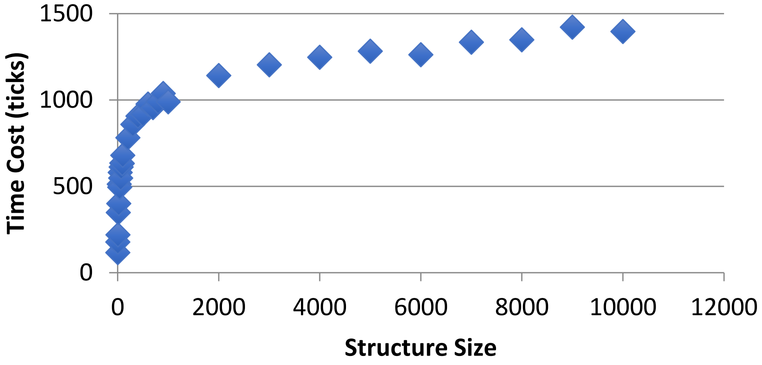
Average: O(log(n))

Remove last(biggest): O(log(n))

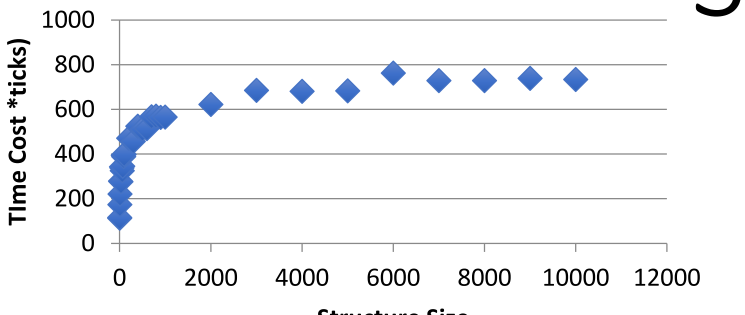
Remove first(smallest): O(log(n))

Remove N+1(D.N.E): O(log(n))

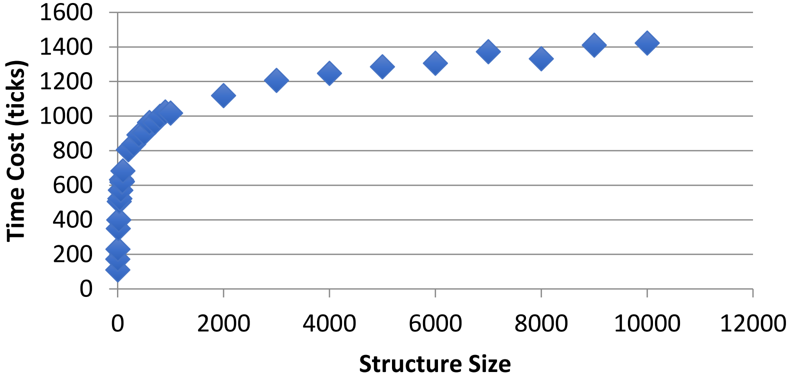
D.N.E. Case- Structure[4]



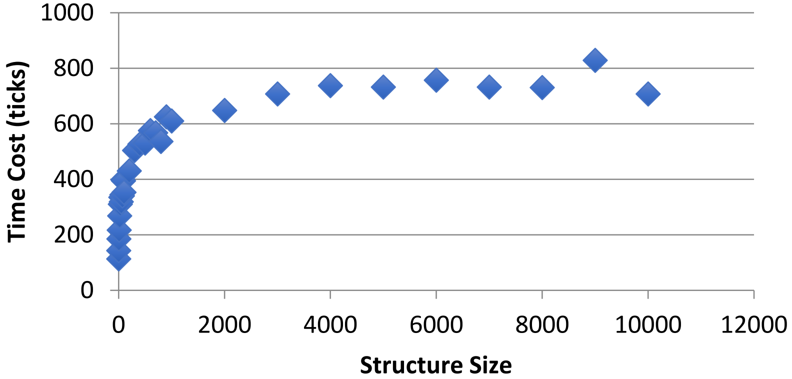
Last - Structure[4]



First - Structure[4]



Average Case- Structure[4]



Conclusion

- Structure 4 is a Binary Search Tree
- Everything is O(log(n))
- Again?????????