Aneesh Abhyankar
RUID – 166006555
Net Id – ana85

Assignment 3

Q1. Develop an implementation of the basic symbol-table API that uses 2-3 trees that are not necessarily balanced as the underlying data structure. Allow 3-nodes to lean either way. Hook the new node onto the bottom with a black link when inserting into a 3-node at the bottom.

=> Each node can be either a two node or a three node. A two node has a single key value and two children. A three node has two key values (the left one less than the right one) and three children.

In this implementation, the tree may be unbalanced unlike the actual implementation of a two-three tree.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Q2. Run experiments to develop a hypothesis estimating the average path length in a tree built from (i) N-random insertions. (ii) N-sorted insertions?

=> The best case height of a 2-3 tree is $\log_3(N)$ and the worst case height is $\log_2(N)$, where N is the total number of nodes in the tree. The best case arises when all the nodes are three nodes and the worst case is when all the nodes are two nodes.

Following is the Average Path Length calculated for doubling data sizes from 1024 up to 8192, for random as well as sorted insertions. The graph plots the values of average path length against data size.

Random Insertions		
Data Size	**Average Path Length**	**Tree Size**
1024	7.50805	620
2048	8.05714	1225
4096	10.2621	2465
8192	9.88253	4192

Sorted Insertions		
Data Size	**Average Path Length**	**Tree Size**
1024	256.5	512
2048	512.5	1024
4096	1024.5	2048
8192	2048.5	4096

Figure 1 – Tables showing average path length of 2-3 trees

Aneesh Abhyankar
RUID – 166006555
Net Id – ana85

We can verify the average length by considering the upper and lower bounds of the path length possible. For example, for data size of N=2048, the best case height is $\log_3(2048) = 6.94$ and worst case height is $\log_2(2048) = 11$. So, our observation, i.e. 8.05714 is correct.

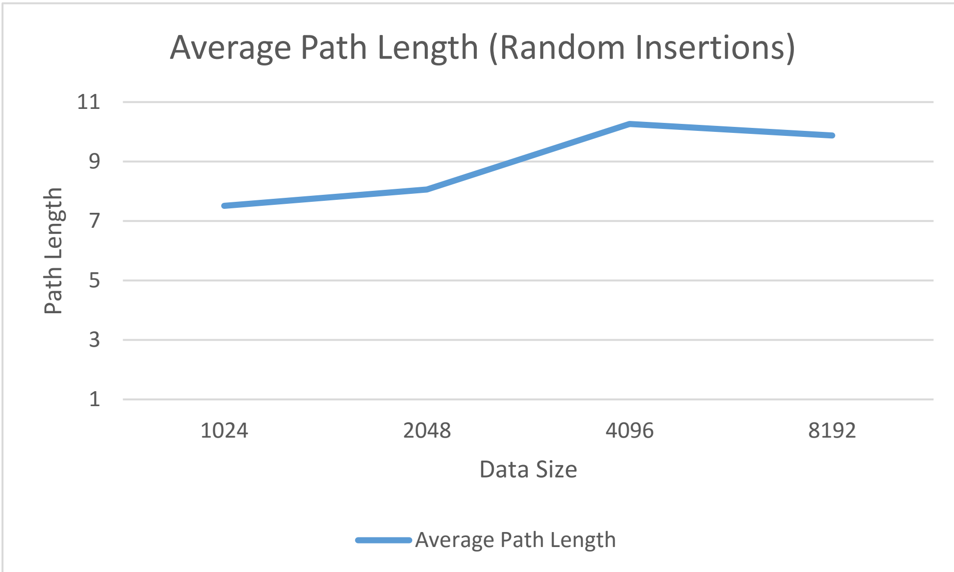

Figure 2 – Plot of Average Path Length for Random Insertions

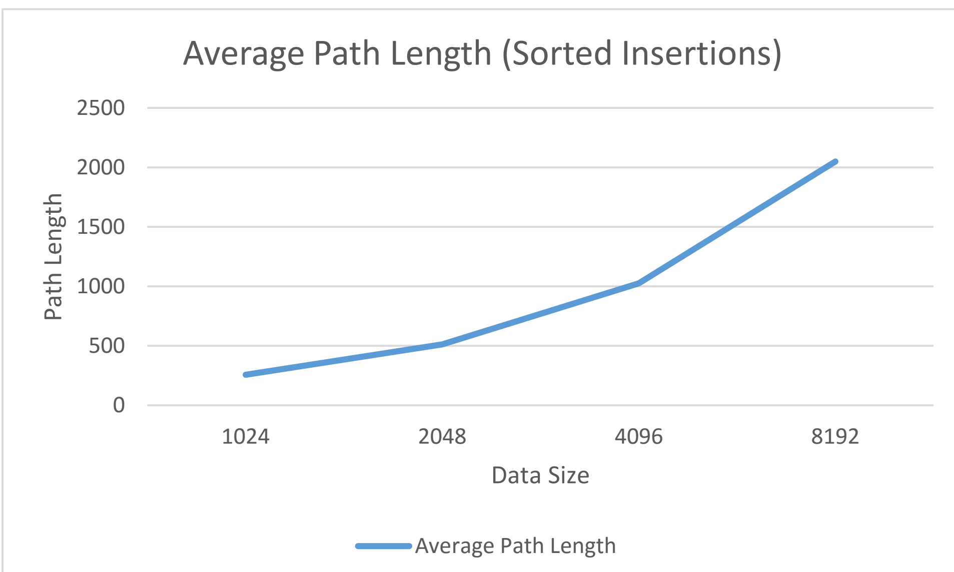

Figure 3 – Plot of Average Path Length for Sorted Insertions

Hypothesis =>

$$P = c * N^b$$

For Random Insertions,

Here, P1 = 8.05714 and P2 = 7.50805

N1 = 2048 and N2 = 1024

Therefore, P1/P2 = 1.073 = $(2048/1024)^b$

b = 1.0731 and c = 0.00225

Therefore, $\mathbf{P = 0.00225*N^{1.0731}}$

Similarly, for Sorted Insertions,

We get, b = 0.998 and c = 0.25

Therefore, $\mathbf{P = 0.25*N^{0.998}}$

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Q3. Write a program that computes the percentage of red nodes in a given red-black tree. Test program by running at least 100 trials of the experiment of increasing N random keys into an initially empty tree for N=10^4, 10^5 and 10^6 and formulate a hypothesis.

=> Red Black Tree implementation includes insertion of node and coloring it Red and then fixing the violation that may arise due to this insertion and moving towards the root while doing so. Tests were run against increasing data size of 10^4, 10^5, and 10^6. Following is the average percentage of red nodes calculated as (# of Red Nodes) / (# of Red Nodes + # of Black Nodes).

Data Size	Average Percentage of Red Nodes
10000	0.484877
100000	0.486498
1000000	0.486583

Figure 4 – Table showing average percentage of red nodes

Aneesh Abhyankar
RUID – 166006555
Net Id – ana85

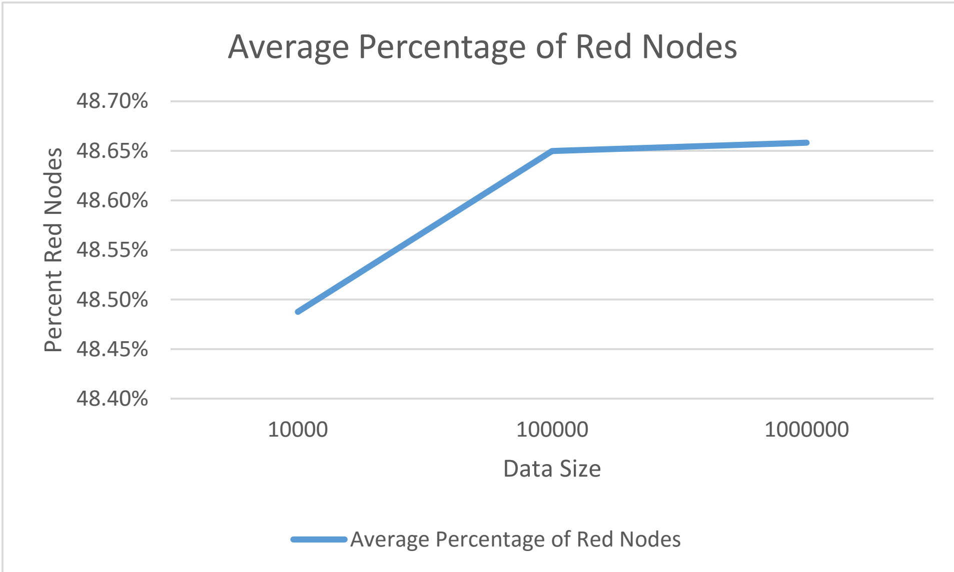

Figure 5 – Plot of Average Percentage of Red Nodes against Data Size

We can see that the average percent of red nodes remains *more or less constant in the range of 48.4% to 48.7%.*

Hypothesis =>

$$P = c * N^b$$

Here, P1 = 0.486498 and P2 = 0.484877

N1 = 100000 and N2 = 10000

Therefore, $P1/P2 = 1.003 = (100000/10000)^b$

b = 0 and c = 0.48

Therefore, **P = 0.48** (constant)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Q4. Run empirical studies to compute the average and standard deviation of the average length of a path to a random node (internal path length divided by tree size) in a red-black BST built by insertion of N random keys into an initially empty tree, for N from 1 to 10,000. Do at least 1,000 trials for each size.

=> Following is the table that shows average and standard deviation of path length for different data sizes between 1 to 10,000 computed by executing 1000 trials. We can see that the average keeps on increasing (rapidly in the beginning and slowly afterwards) as expected with data size, but the standard deviation decreases slowly and settles eventually.

Data Size	**Average**	**Standard Deviation**
4	1	0
8	1.696875	0.084856
16	2.488563	0.053846
32	3.363313	0.046502
64	4.310659	0.055133
128	5.279415	0.045988
256	6.271172	0.039916
512	7.276039	0.038935
1024	8.287318	0.036058
2048	9.302272	0.034241
4096	10.31876	0.033071
8192	11.33704	0.033035

Figure 6 – Table showing average and standard deviation of path length

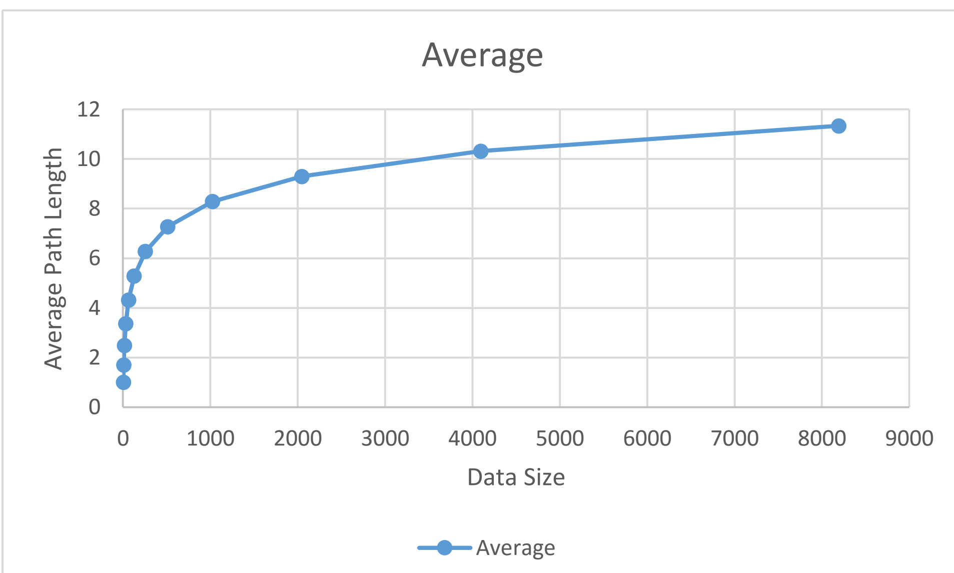

Figure 7 – Scatter plot - the trend average path length follows with increase in data size

Hypothesis => for average of path length

$$P = c * N^b$$

Here, P1 = 10.31876 and P2 = 9.302272

N1 = 4096 and N2 = 2048

Therefore, $P1/P2 = 1.1093 = (4096 / 2048)^b$

b = 0.1496 and c = 2.9732

Therefore, $\mathbf{P = 2.9732*N^{0.1496}}$ (constant)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx