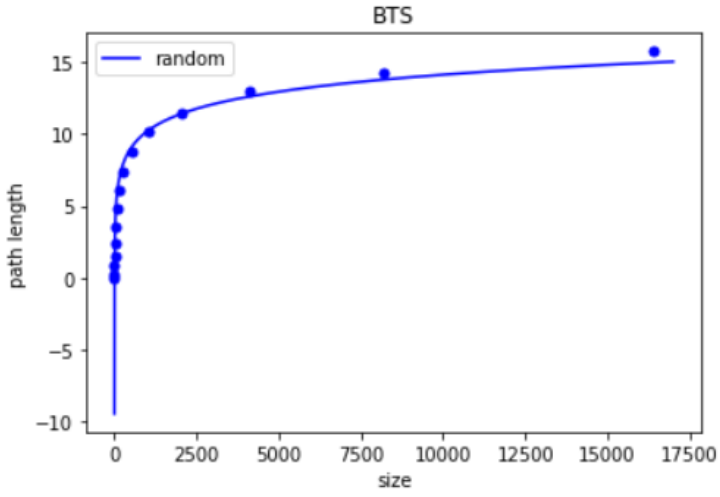
Data Structure & Algorithm Homework 3

Zhuohang Li

Q2.

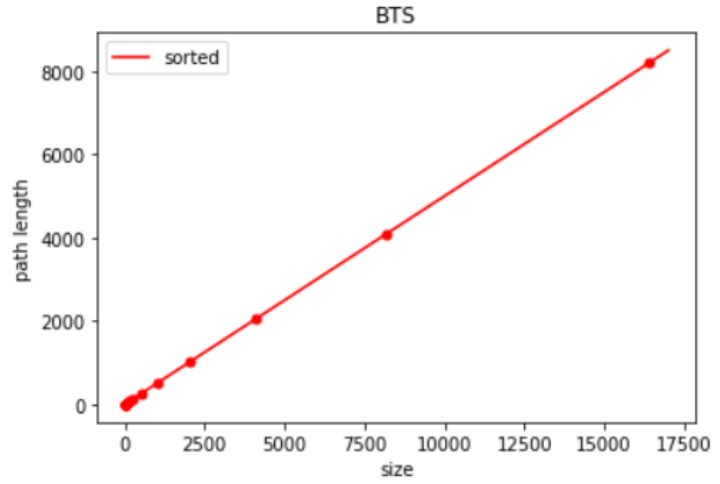
As we can see from tables below, an ordinary binary search tree works pretty well. For average path length, it has the growth rate of roughly O(log N). But it is extremely bad when it comes to sorted input data. The growth rate becomes linear. Since the 2-3 built-in Q1 is not balanced, it has similar performance as BTS (I'm counting both black and red links. If count only black links, the average path length for sorted input shall be reduced by half, since the tree is a linear structure and every red link is followed by a black link). The red-black tree is completely balanced therefore it is guaranteed to have the growth rate of O(log N) in both cases.

Path length of BTS		
Size of data	avg path length (Random)	avg path length (Sorted)
1	0	0
2	0.215	0.5
4	0.835	1.5
8	1.55619	3.5
16	2.466694	7.5
32	3.607877	15.5
64	4.769213	31.5
128	6.140596	63.5
256	7.34659	127.5
512	8.776372	255.5
1024	10.21508	511.5
2048	11.49384	1023.5
4096	12.91616	2047.5
8192	14.2426	4095.5
16384	15.72582	8191.5



y=1.1805795178575367log(x)+(−1.5769211583389544)

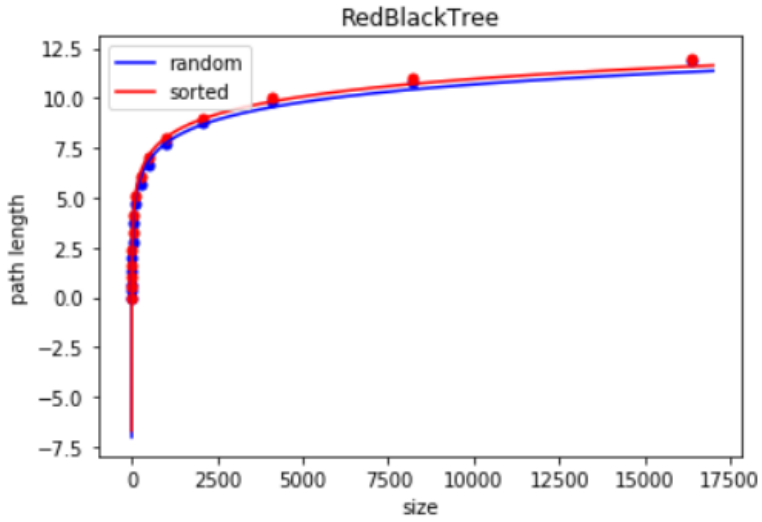
Path length of 2-3 tree from Q1		
Size of data	avg path length (Random)	avg path length (Sorted)
1	0	0
2	0.28	0.5
4	0.805	1.5
8	1.502952	3.5
16	2.504539	7.5



y=0.5x+(−0.5000000000000125)

32	3.574878	15.5
64	4.761676	31.5
128	6.048246	63.5
256	7.35551	127.5
512	8.730289	255.5
1024	10.02256	511.5
2048	11.4746	1023.5
4096	12.90684	2047.5
8192	14.16514	4095.5
16384	15.54943	8191.5

Path length of Red Black Tree		
Size of data	avg path length (Random)	avg path length (Sorted)
1	0	0
2	0.31	0.5
4	0.638333	1
8	1.286869	1.625
16	1.960048	2.375
32	2.785067	3.21875
64	3.721565	4.125
128	4.68982	5.070313
256	5.658734	6.039063
512	6.675271	7.021484
1024	7.696864	8.011719
2048	8.741088	9.006348
4096	9.809582	10.00342
8192	10.8267	11.00183
16384	11.84398	12.00098



Random: y=0.8878900214283271log(x)+(−1.1056354166712592)
Sorted: y=0.8856786428568936log(x)+(−0.7998233666705927)

Q3.

Data Size	Percentage of Red Nodes
10000	0.253888
100000	0.253879
1000000	0.253983

Based on the result as shown in the table above, a proper hypothesis would be that the percentage of red nodes in a Red Black Tree built by random inputs is constant and does not change with the size of the tree.

Q4.

The data is uploaded in Sakai.

Q5.

The value of select(7) for the data set is 8.

The value of rank(7) for the data set is 6.

```
"C:\Program Files\Java\jdk1.8.0_152\bin\java" ...
select(7): 8
rank(7): 6

Process finished with exit code 0
```