Naveen Kumar Buddhala (001582394)

Program Structures & Algorithms

Spring 2021

Assignment No. 4

Task

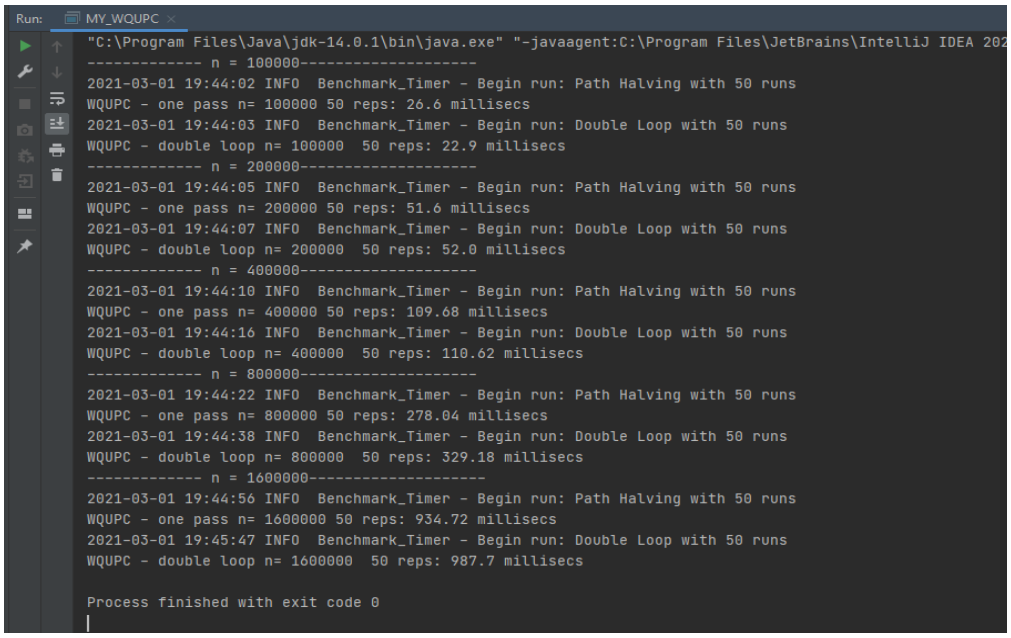
Union-Find alternatives – to code and benchmark two alternatives for implementing Union-Find

1. Depth weighted quick union
2. Weighted quick union with path compression

Output

Weighted Quick Union with path compression –

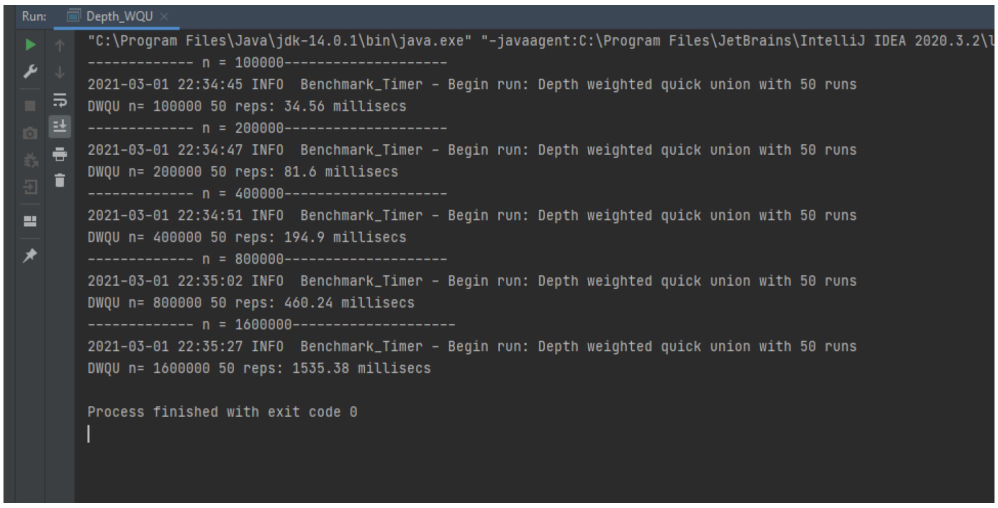
1. Single pass method – path halving
2. Double loop – linking every intermediate nodes to the root



```
"C:\Program Files\Java\jdk-14.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 202
------------- n = 100000--------------------
2021-03-01 19:44:02 INFO  Benchmark_Timer - Begin run: Path Halving with 50 runs
WQUPC - one pass n= 100000 50 reps: 26.6 millisecs
2021-03-01 19:44:03 INFO  Benchmark_Timer - Begin run: Double Loop with 50 runs
WQUPC - double loop n= 100000  50 reps: 22.9 millisecs
------------- n = 200000--------------------
2021-03-01 19:44:05 INFO  Benchmark_Timer - Begin run: Path Halving with 50 runs
WQUPC - one pass n= 200000 50 reps: 51.6 millisecs
2021-03-01 19:44:07 INFO  Benchmark_Timer - Begin run: Double Loop with 50 runs
WQUPC - double loop n= 200000  50 reps: 52.0 millisecs
------------- n = 400000--------------------
2021-03-01 19:44:10 INFO  Benchmark_Timer - Begin run: Path Halving with 50 runs
WQUPC - one pass n= 400000 50 reps: 109.68 millisecs
2021-03-01 19:44:16 INFO  Benchmark_Timer - Begin run: Double Loop with 50 runs
WQUPC - double loop n= 400000  50 reps: 110.62 millisecs
------------- n = 800000--------------------
2021-03-01 19:44:22 INFO  Benchmark_Timer - Begin run: Path Halving with 50 runs
WQUPC - one pass n= 800000 50 reps: 278.04 millisecs
2021-03-01 19:44:38 INFO  Benchmark_Timer - Begin run: Double Loop with 50 runs
WQUPC - double loop n= 800000  50 reps: 329.18 millisecs
------------- n = 1600000--------------------
2021-03-01 19:44:56 INFO  Benchmark_Timer - Begin run: Path Halving with 50 runs
WQUPC - one pass n= 1600000 50 reps: 934.72 millisecs
2021-03-01 19:45:47 INFO  Benchmark_Timer - Begin run: Double Loop with 50 runs
WQUPC - double loop n= 1600000  50 reps: 987.7 millisecs

Process finished with exit code 0
```

Depth Weighted Quick Union –

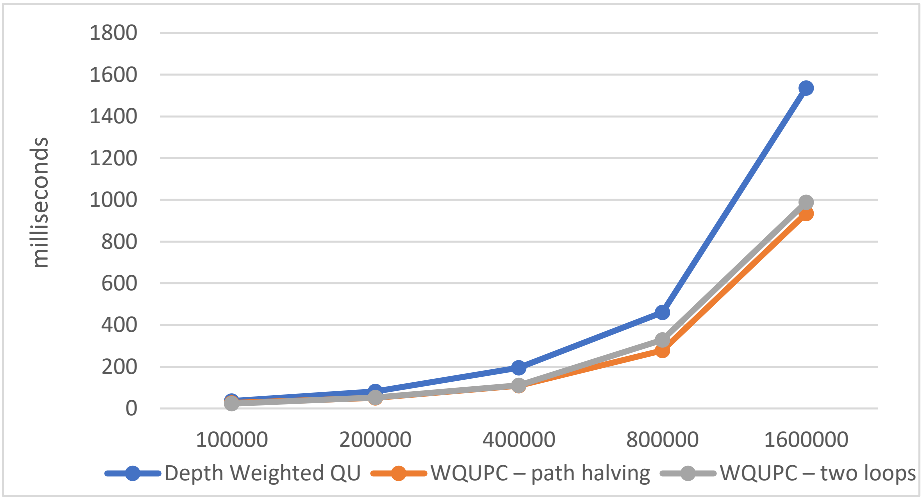


```
"C:\Program Files\Java\jdk-14.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2020.3.2\l
------------- n = 100000--------------------
2021-03-01 22:34:45 INFO  Benchmark_Timer - Begin run: Depth weighted quick union with 50 runs
DWQU n= 100000 50 reps: 34.56 millisecs
------------- n = 200000--------------------
2021-03-01 22:34:47 INFO  Benchmark_Timer - Begin run: Depth weighted quick union with 50 runs
DWQU n= 200000 50 reps: 81.6 millisecs
------------- n = 400000--------------------
2021-03-01 22:34:51 INFO  Benchmark_Timer - Begin run: Depth weighted quick union with 50 runs
DWQU n= 400000 50 reps: 194.9 millisecs
------------- n = 800000--------------------
2021-03-01 22:35:02 INFO  Benchmark_Timer - Begin run: Depth weighted quick union with 50 runs
DWQU n= 800000 50 reps: 460.24 millisecs
------------- n = 1600000--------------------
2021-03-01 22:35:27 INFO  Benchmark_Timer - Begin run: Depth weighted quick union with 50 runs
DWQU n= 1600000 50 reps: 1535.38 millisecs

Process finished with exit code 0
```

Results

No of objects (n)	Depth Weighted QU (millisecs)	WQUPC – path halving (millisecs)	WQUPC – two loops (millisecs)
100000	35.56	26.6	22.9
200000	81.6	51.6	52.0
400000	194.9	109.68	110.62
800000	460.24	278.04	329.18
1600000	1535.38	934.72	987.7



As we can see by the benchmark results, the depth weighted quick union is not great for bigger values of n i.e. size of a set.

There is no performance gain for depth based compared to size based weighted quick union, only a few cases might be faster but all in all it is O(logN) as at most depth of N nodes is logN