

# Gautham Rajsimha Pulipati (001572432)

## TASK

Union Find Alternatives

## OUTPUT:

Part 1, Weighted Quick Union with depth vs size

```
Console
<terminated> New_configuration (1) [Java Application] /Users/gauthamrajsimhapulipati/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre
2021-03-02 01:10:21 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:10000Weighted on size 1.9801305555555555
2021-03-02 01:10:21 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:10000Weighted on height/depth 1.4784259333333334
2021-03-02 01:10:21 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:50000Weighted on size 8.884415688888888
2021-03-02 01:10:21 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:50000Weighted on height/depth 8.745663044444445
2021-03-02 01:10:22 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:90000Weighted on size 17.626570488888888
2021-03-02 01:10:23 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:90000Weighted on height/depth 17.562071244444446
2021-03-02 01:10:23 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:130000Weighted on size 27.497594488888889
2021-03-02 01:10:25 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:130000Weighted on height/depth 27.698577800000002
2021-03-02 01:10:26 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:170000Weighted on size 38.802537955555556
2021-03-02 01:10:28 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:170000Weighted on height/depth 38.020113822222222
```

Part 2, Weighted Quick Union with Path Compression on 1 pass vs 2 pass

```
Console
<terminated> New_configuration [Java Application] /Users/gauthamrajsimhapulipati/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_15.0.1.v20201027-0507
2021-03-02 00:54:12 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:10000Path compression with 1 pass 1.0198491111111112
2021-03-02 00:54:12 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:10000Path compression with 2 pass 1.2109823555555554
2021-03-02 00:54:12 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:50000Path compression with 1 pass 4.958075888888889
2021-03-02 00:54:12 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:50000Path compression with 2 pass 5.445700911111111
2021-03-02 00:54:13 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:90000Path compression with 1 pass 9.101237933333334
2021-03-02 00:54:13 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:90000Path compression with 2 pass 10.139097288888889
2021-03-02 00:54:14 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:130000Path compression with 1 pass 14.185725066666667
2021-03-02 00:54:14 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:130000Path compression with 2 pass 15.730276822222221
2021-03-02 00:54:15 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:170000Path compression with 1 pass 19.742737088888889
2021-03-02 00:54:16 INFO Benchmark_Timer - Begin run: Benchmarking with 45 runs
n=:170000Path compression with 2 pass 21.407923155555554
```

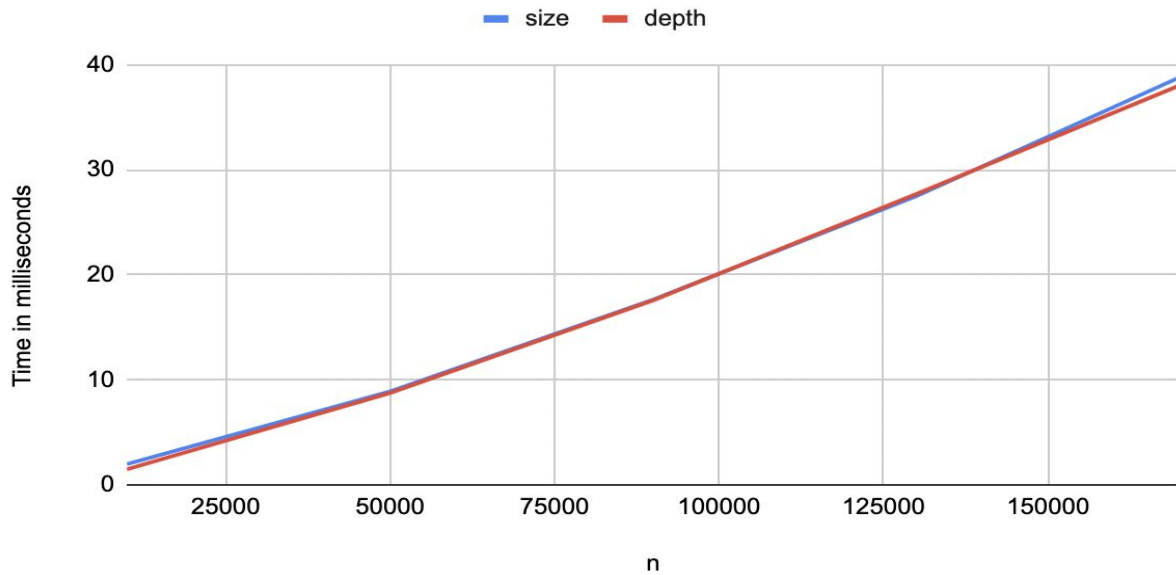
## RELATIONSHIPS AND CONCLUSIONS

Part1: When we use depth and Size, the depth of a root is always 0, so I considered the max depth of any node in the tree, which is height. On benchmarking this with various n values, I can conclude that **it doesn't affect the run time** on either of these cases.

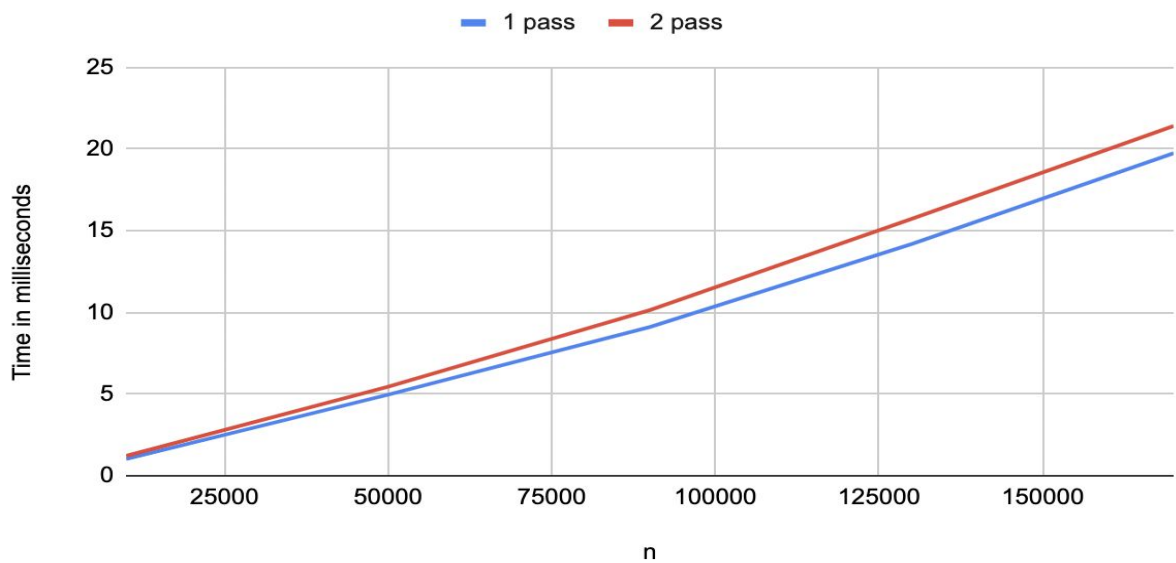
Part2: For 1 pass, we update the parent of the node as grandparent during the find operation and 2 pass where we write a 2nd loop on the function to update the parent of every element from the given to the root. Judging by the values, **1 pass looked faster than 2 pass**, but on a small margin.

## EVIDENCE FOR THE CONCLUSIONS

size vs depth



1 pass vs 2 pass



- Files added in union\_find folder: UF\_alternatives1\_benchmarking, WQU\_SIZE, UF\_alternatives2\_benchmarking, WQUPC\_SIZE