Program Structures & Algorithms

Spring 2022

Assignment No. 3

WQUPC

Thomas John
NEU ID: 002933800

Task

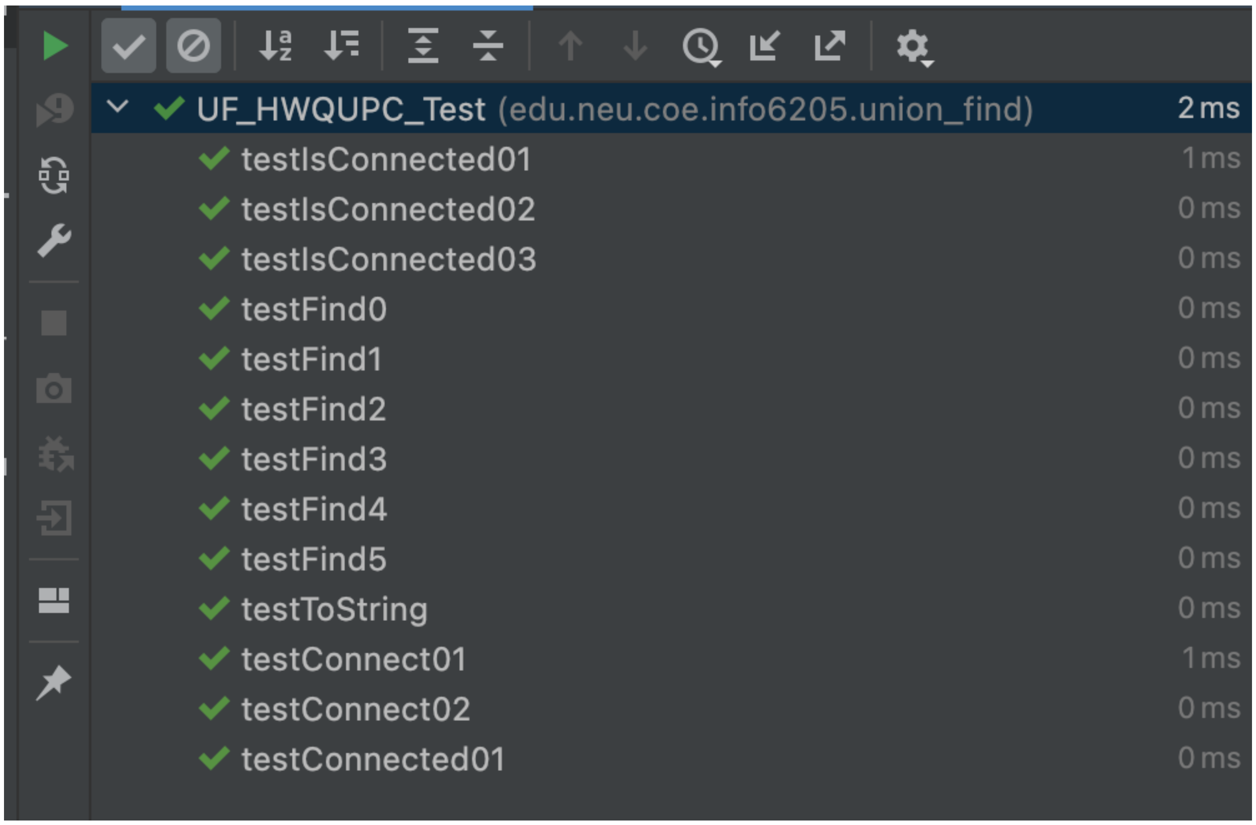
Step 1:

(a) Implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF_HWQUPC. All you have to do is to fill in the sections marked with // TO BE IMPLEMENTED ... // ...END IMPLEMENTATION.

(b) Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).

Github URL:
https://github.com/thomasjohn-neu/INFO6205/commit/544e25c8f7f26e4ebdb92be90310076fbd109a73

Unit Tests



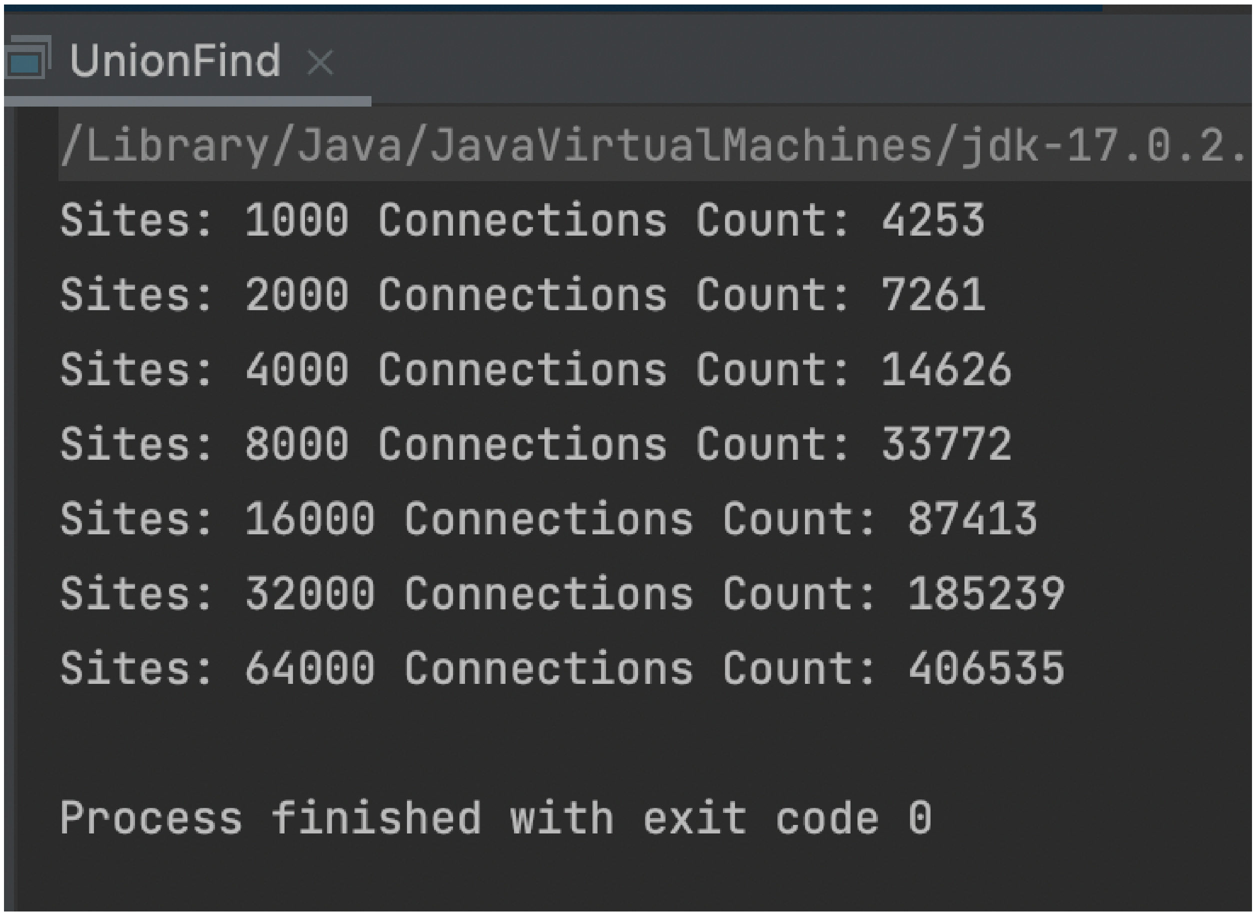
```
✓ UF_HWQUPC_Test (edu.neu.coe.info6205.union_find)   2 ms
    ✓ testIsConnected01                               1 ms
    ✓ testIsConnected02                               0 ms
    ✓ testIsConnected03                               0 ms
    ✓ testFind0                                       0 ms
    ✓ testFind1                                       0 ms
    ✓ testFind2                                       0 ms
    ✓ testFind3                                       0 ms
    ✓ testFind4                                       0 ms
    ✓ testFind5                                       0 ms
    ✓ testToString                                    0 ms
    ✓ testConnect01                                   1 ms
    ✓ testConnect02                                   0 ms
    ✓ testConnected01                                 0 ms
```

Step 2:

Using your implementation of UF_HWQUPC, develop a UF ("union-find") client that takes an integer value n from the command line to determine the number of "sites." Then generates random pairs of integers between 0 and n-1, calling connected() to determine if they are connected and union() if not. Loop until all sites are connected then print the number of connections generated. Package your program as a static method count() that takes n as the argument and returns the number of connections; and a main() that takes n from the command line, calls count(), and prints the returned value. If you prefer, you can create a main program that doesn't require any input and run the experiment for a fixed set of n values. Show evidence of your run(s).

Output Screenshot

UnionFind.java triggers the UF_HWQUPC object and tests the values for multiple values of sites.



```
/Library/Java/JavaVirtualMachines/jdk-17.0.2.
Sites: 1000 Connections Count: 4253
Sites: 2000 Connections Count: 7261
Sites: 4000 Connections Count: 14626
Sites: 8000 Connections Count: 33772
Sites: 16000 Connections Count: 87413
Sites: 32000 Connections Count: 185239
Sites: 64000 Connections Count: 406535

Process finished with exit code 0
```

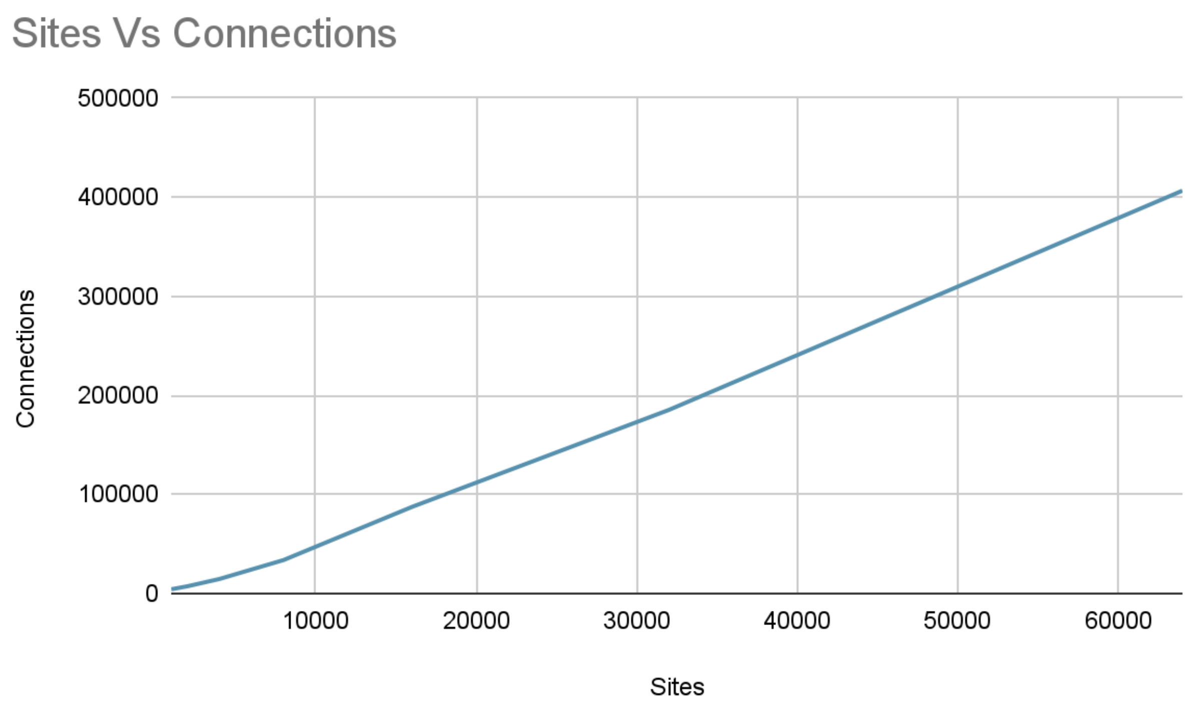
Step 3:

Determine the relationship between the number of objects (*n*) and the number of pairs (*m*) generated to accomplish this (i.e. to reduce the number of components from *n* to 1). Justify your conclusion in terms of your observations and what you think might be going on.

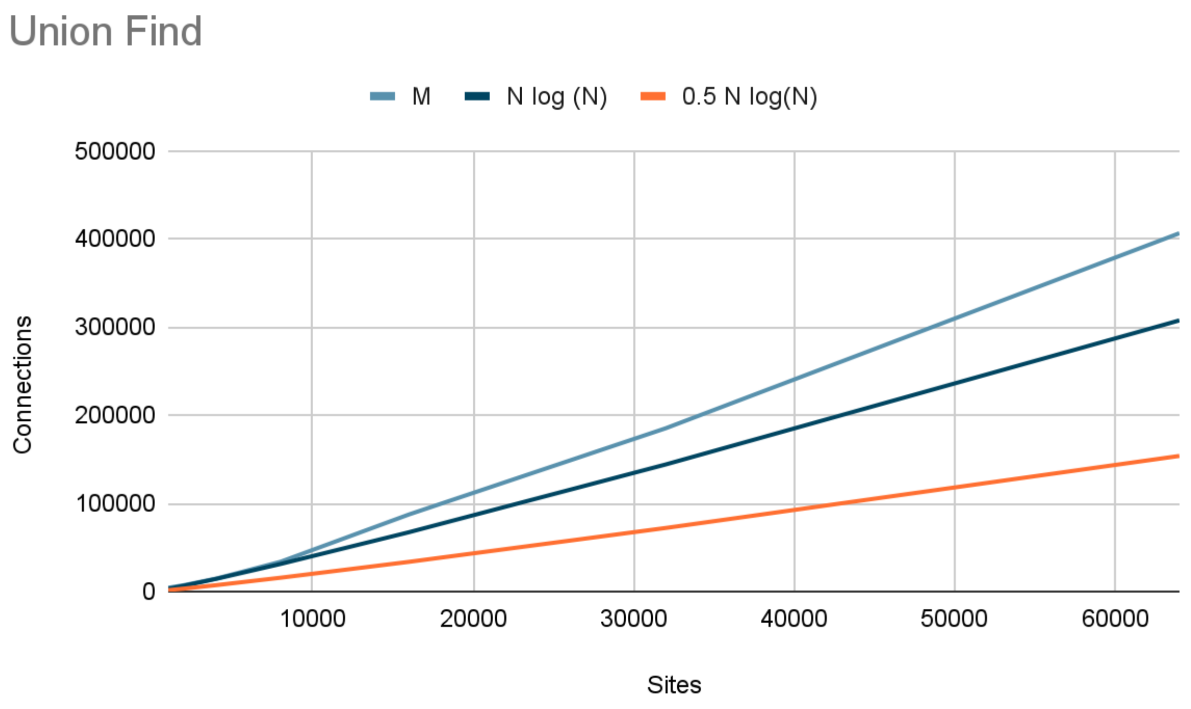
Relationship Conclusion

M = (N * log (N)) / 2

where M = Connection Count, N = Sites



N	M	N log N	0.5 N log N
1000	4253	3000	1500
2000	7261	6602	3301
4000	14626	14408	7204
8000	33772	31225	15612
16000	87413	67266	33633
32000	185239	144165	72083
64000	406535	307596	153798



Code Repository

https://github.com/thomasjohn-neu/INFO6205/commit/544e25c8f7f26e4ebdb92be90310076fbd109a73