Program Structures and Algorithms

Spring 2023

Assignment No: 3

Name: Srikanth Nandikonda

NUID: 002737724

Task:

Step 1: Implement height-weighted Quick Union with Path Compression and check that the unit tests for this class.

Step 2: Create a main program that doesn't require any input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

Step 3: Create a main program that doesn't require any input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

Relationship Conclusion:

The actual number of connections required to connect all the sites in a union-find implementation can vary depending on the random order of union operations. Based on the value of m(number of pairs) by running the program 50 times for multiple values of n starting from 100 and doubling it till the value reaches 3276800, derived an relation between m and n.

m is proportional to n*log(n)

m = 0.5175 * n* ln(n)

m is number of random pairs to make the number of connections from n to 1

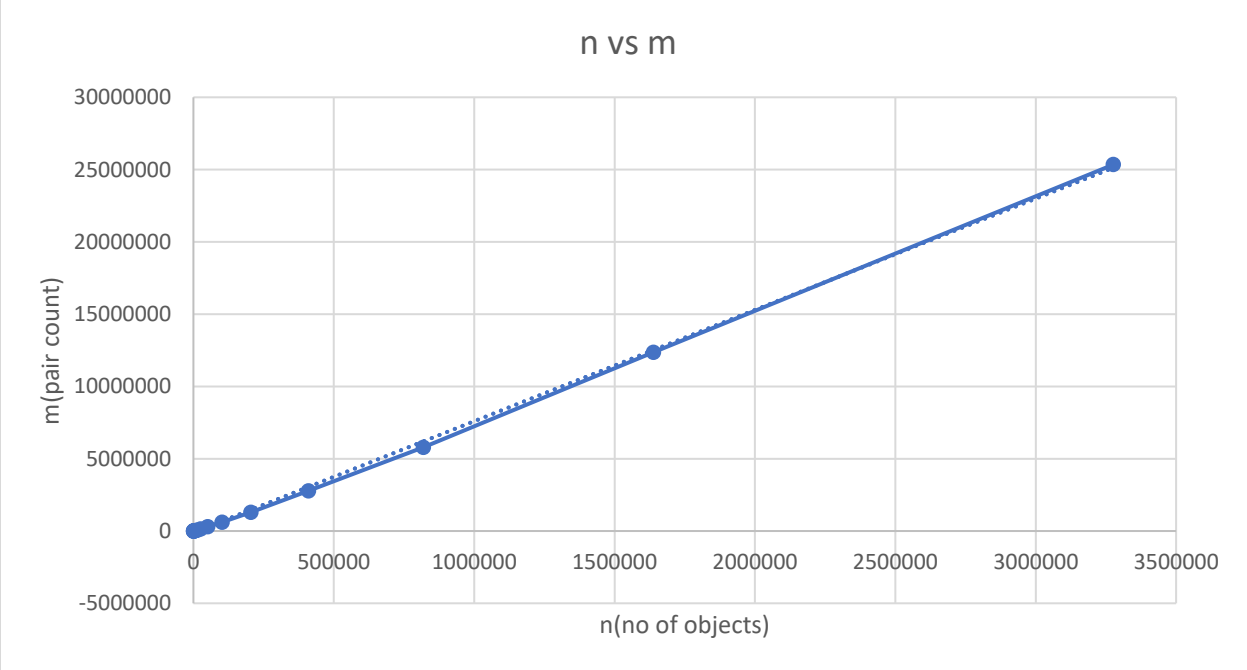
n is the number of elements in the union find

Above provides a good approximation for the average number of connections required and can be useful in predicting the performance of union-find algorithms. It is also worth noting that as n increases, the actual number of connections required becomes closer to the estimated value, as the random variations in the number of connections become smaller in comparison to the overall number of connections. This estimate is based on the observation that, on average, a logarithmic number of connections are required to connect all the sites in a balanced tree structure.

Evidence:

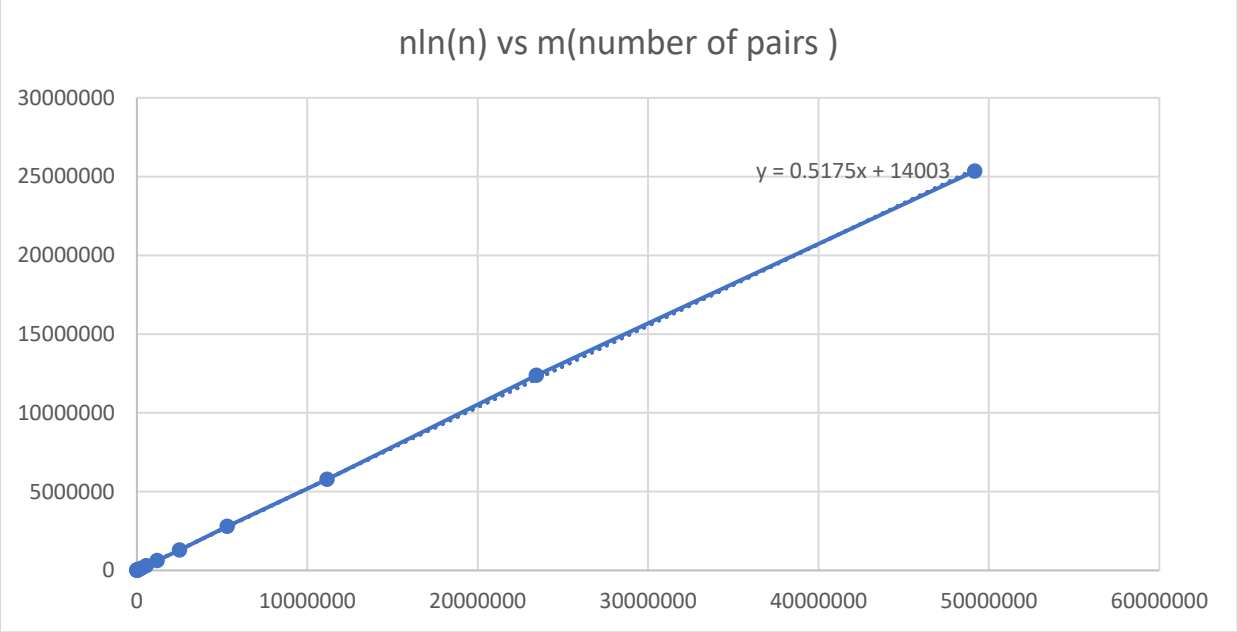
n(number of elements	nln(n)(value	m(value obtained from progra	Equation obtained from the graph
100	460.5170186	274	238.3175571
200	1059.663473	601	548.3758474
400	2396.585819	1352	1240.233161
800	5347.689382	2770	2767.429255
1600	11804.41425	6204	6108.784376
3200	25826.89948	13891	13365.42048
6400	56089.94092	29681	29026.54443
12800	121052.1658	64643	62644.49578
25600	259848.8993	137661	134471.8054
51200	555186.9343	299784	287309.2385
102400	1181352.14	620829	611349.7324
204800	2504660.822	1289655	1296161.976
409600	5293234.73	2784870	2739248.973
819200	11154295.63	5783480	5772347.989
1638400	23444243.6	12373986	12132396.06
3276800	49159791.88	25345048	25440192.3

Graph of n (number of elements) vs m (number of connections to make it from n to 1)



➔ Graph of the n and m is almost linear, but its not a perfect linear function, so mostly it may be a n*ln(n) relation. To get the exact relation , we will plot the graph for nlogn and m

Graph of n*ln(n) vs m(number of pairs obtained from code)

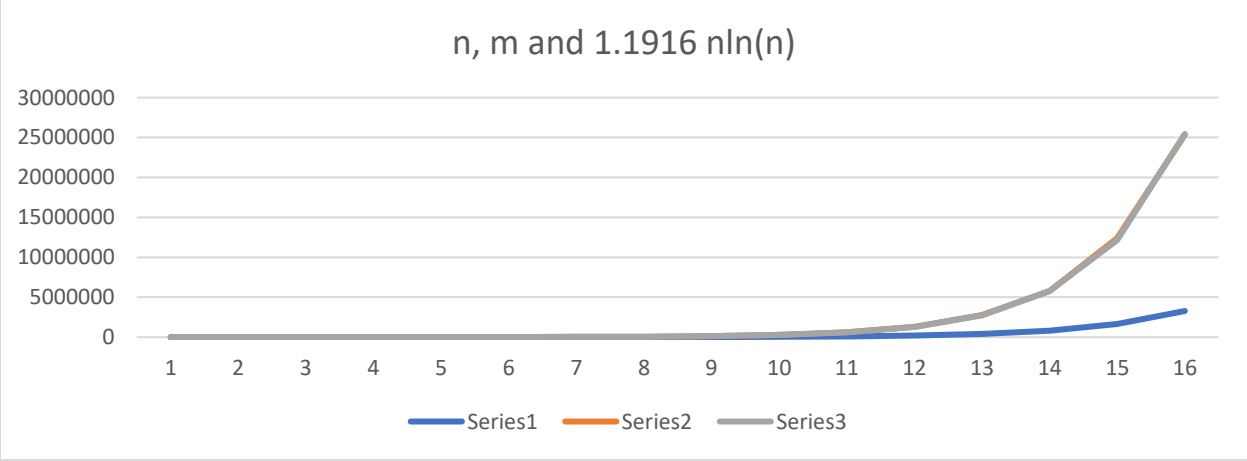


Using the trendline feature, obtain the linear equation.

$$y = 0.5175x + 14003$$

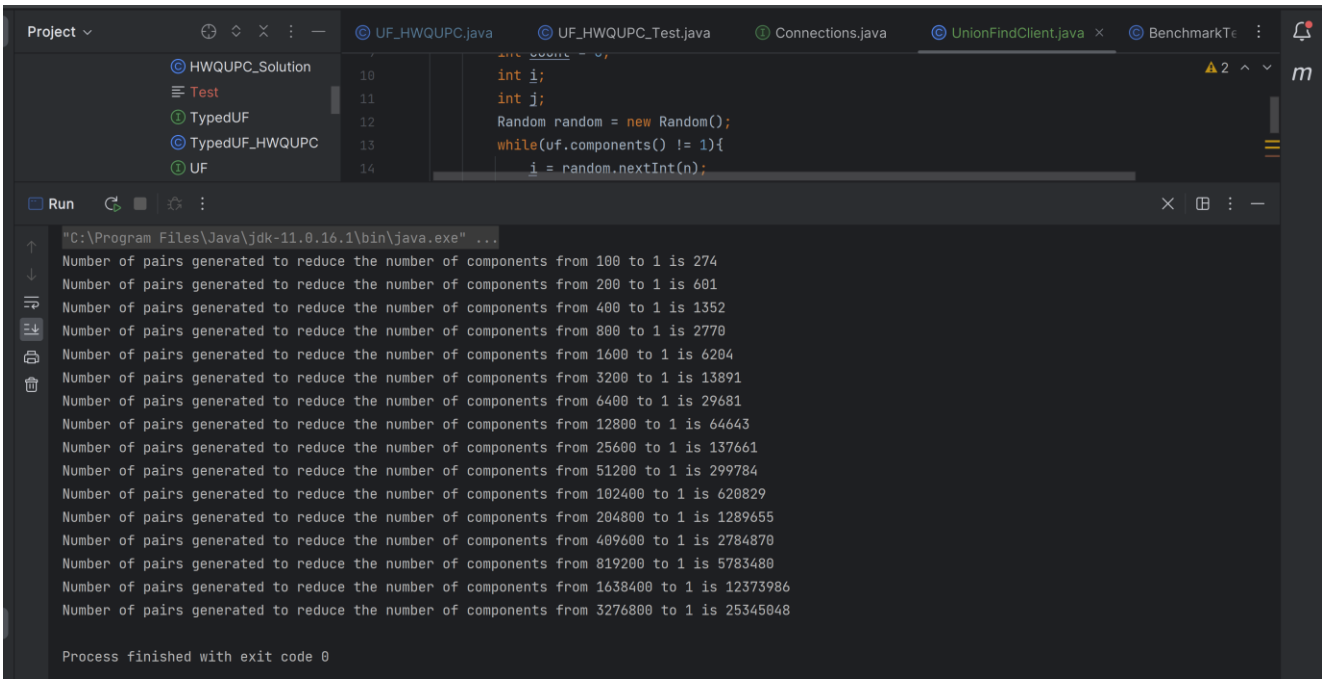
$m = 0.5175 * n * \ln(n)$

Graph three values on a single graph



Plot of m and 1.11916 n log(n) are very correlated, would be the best fit for out data points

Output Screenshot



Output

Number of pairs generated to reduce the number of components from 100 to 1 is 274

Number of pairs generated to reduce the number of components from 200 to 1 is 601

Number of pairs generated to reduce the number of components from 400 to 1 is 1352

Number of pairs generated to reduce the number of components from 800 to 1 is 2770

Number of pairs generated to reduce the number of components from 1600 to 1 is 6204

Number of pairs generated to reduce the number of components from 3200 to 1 is 13891

Number of pairs generated to reduce the number of components from 6400 to 1 is 29681

Number of pairs generated to reduce the number of components from 12800 to 1 is 64643

Number of pairs generated to reduce the number of components from 25600 to 1 is 137661

Number of pairs generated to reduce the number of components from 51200 to 1 is 299784

Number of pairs generated to reduce the number of components from 102400 to 1 is 620829

Number of pairs generated to reduce the number of components from 204800 to 1 is 1289655

Number of pairs generated to reduce the number of components from 409600 to 1 is 2784870

Number of pairs generated to reduce the number of components from 819200 to 1 is 5783480

Number of pairs generated to reduce the number of components from 1638400 to 1 is 12373986

Number of pairs generated to reduce the number of components from 3276800 to 1 is 25345048

Process finished with exit code 0

Unit Test

UF_HWQUPC_Test.java