

Aishwarya Balyaya

001586556

Program Structure & Algorithms

Spring 2022

Assignment 3

Task: To implement height-weighted Quick union with Path compression.

For this task, UF_HWQUPC java class was used and following methods were implemented :

- **find ():** to update the root of input object if path compression is performed
- **mergeComponents():** to merge 2 subtrees such that smaller root points to larger root
- **doPathCompression():** a method that implements the single-pass process of path compression method.

Executed the **Union Find code for height-weighted quick union**. Summarized the relationship between a **number of random pairs generated** to reduce the number of components from **n to 1** for various different components of n.

To perform and test the implementation of UF_HWQUPC class, the UnionFind java class was created

```
union_find > UF_HWQUPC > doPathCompression
WQUPC.java x UF_HWQUPC.java x UF_HWQUPC_Test.java x UnionFind.java x
68      * @return the number of components (between {@code 1} and
69      */
70      public int components() { return count; }
73
74      /**
75       * Returns the component identifier for the component conta
76       *
77       * @param p the integer representing one site
78       * @return the component identifier for the component conta
79       * @throws IllegalArgumentException unless {@code 0 <= p <
80       */
81      public int find(int p) {
82          validate(p);
83          int root = p;
84          while(root != parent[root]){
85              root = parent[root];
86          }
87          if(pathCompression){
88              doPathCompression(p);
89          }
90          return root;
91      }
92
```

```
union_find > UF_HWQUPC > doPathCompression
WQUPC.java x UF_HWQUPC.java x UF_HWQUPC_Test.java x UnionFind.java x
169
170     private final int[] parent; // parent[i] = parent of i
171     private final int[] height; // height[i] = height of subtree rooted at i
172     private int count; // number of components
173     private boolean pathCompression;
174
175     private void mergeComponents(int i, int j) {
176         // FIXME make shorter root point to taller one
177         // END
178         if (i==j) return;
179         if (height[i] < height[j]) {
180             parent[i]=j;
181             height[j] += height[i];
182         }
183         else {
184             parent[j] = i;
185             height[i] += height[j];
186         }
187     }
188
189     /**
190      * This implements the single-pass path-halving mechanism of path compression
191      */
192     private void doPathCompression(int i) {
193         // FIXME update parent to value of grandparent
194         // END
195         while(i != parent[i]){
196             parent[i] = parent[parent[i]];
197             i = parent[i];
198         }
199     }
200 }
```

Console Output:

n:8 Total connections generated: 7
n:16 Total connections generated: 42
n:32 Total connections generated: 83
n:64 Total connections generated: 179
n:128 Total connections generated: 230
n:256 Total connections generated: 666

n:512 Total connections generated: 1935
n:1024 Total connections generated: 4655
n:2048 Total connections generated: 9103
n:4096 Total connections generated: 18000
n:8192 Total connections generated: 40339
n:16384 Total connections generated: 83646

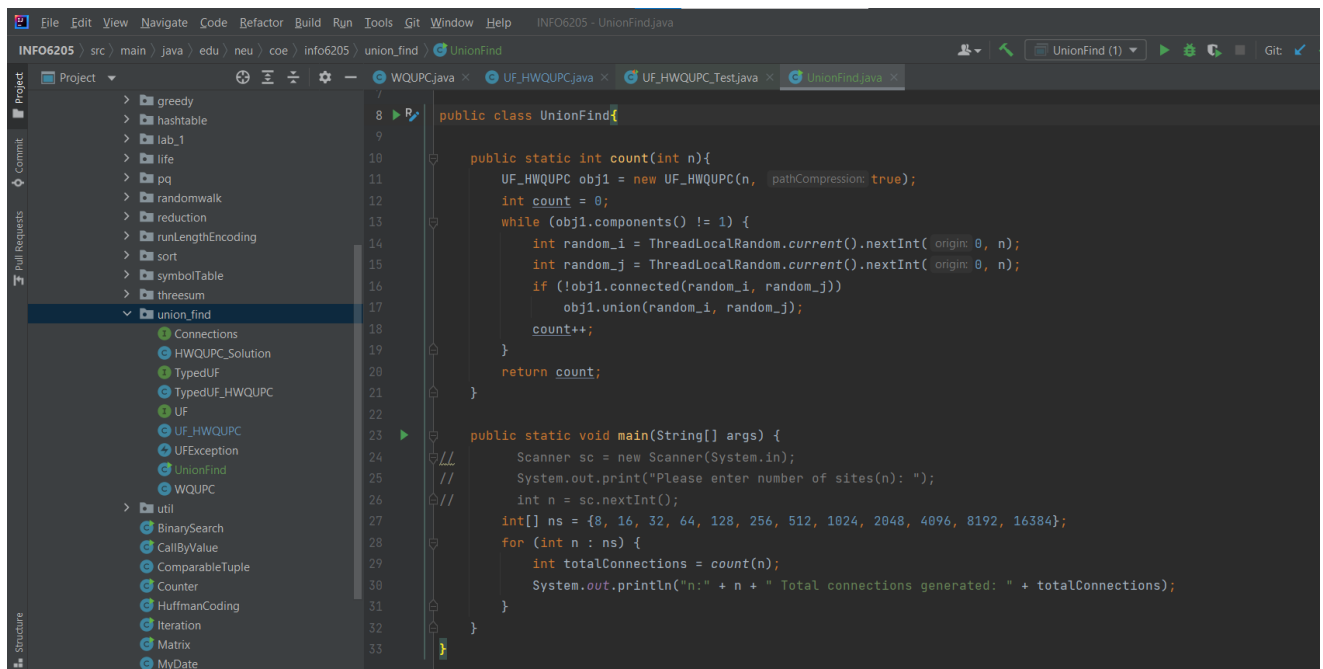
Relationship:

From the results stated above, It can be concluded that to reduce the number of components from **n to 1**, the relationship between **Number of Objects** and the **Number of Pairs** generated is $N/2 * \log(N)$.

i.e. For a number of **32 components** in Quick Union, approximately **55 random pairs** are generated to accomplish the singular component condition.

Evidence:

I have attached a table and a chart to show the relationship between the number of objects and a number of pairs generated

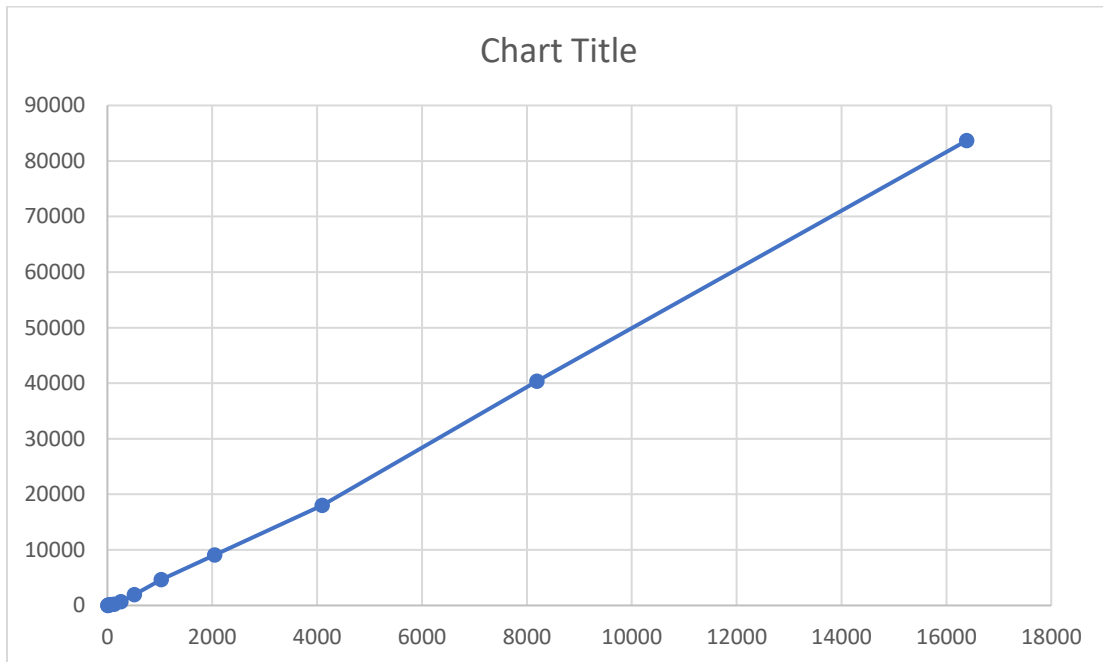


```
File Edit View Navigate Code Refactor Build Run Tools Git Window Help INFO6205 - UnionFind.java
INFO6205 / src / main / java / edu / neu / coe / info6205 / union_find / UnionFind
Project
  greedy
  hashtable
  lab_1
  life
  pq
  randomwalk
  reduction
  runLengthEncoding
  sort
  symbolTable
  threesum
  union_find
    Connections
    HWQUPC_Solution
    TypedUF
    TypedUF_HWQUPC
    UF
    UF_HWQUPC
    UFException
    UnionFind
    WQUPC
  util
    BinarySearch
    CallByValue
    ComparableTuple
    Counter
    HuffmanCoding
    Iteration
    Matrix
    MyDate
WQUPC.java x UF_HWQUPC.java x UF_HWQUPC_Test.java x UnionFind.java x
UnionFind (1)
public class UnionFind {
    public static int count(int n) {
        UF_HWQUPC obj1 = new UF_HWQUPC(n, pathCompression: true);
        int count = 0;
        while (obj1.components() != 1) {
            int random_i = ThreadLocalRandom.current().nextInt(0, n);
            int random_j = ThreadLocalRandom.current().nextInt(0, n);
            if (!obj1.connected(random_i, random_j)) {
                obj1.union(random_i, random_j);
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Please enter number of sites(n): ");
        int n = sc.nextInt();
        int[] ns = {8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384};
        for (int n : ns) {
            int totalConnections = count(n);
            System.out.println("n: " + n + " Total connections generated: " + totalConnections);
        }
    }
}
```

```
INFO6205 - UnionFind.java
INFO6205 > src > main > java > edu > neu > coe > info6205 > union_find > UnionFind
Project
Run: UnionFind (1) x
"C:\Program Files\Java\jdk-12.0.2\bin\java.exe" ...
n:8 Total connections generated: 7
n:16 Total connections generated: 42
n:32 Total connections generated: 83
n:64 Total connections generated: 179
n:128 Total connections generated: 230
n:256 Total connections generated: 666
n:512 Total connections generated: 1935
n:1024 Total connections generated: 4655
n:2048 Total connections generated: 9103
n:4096 Total connections generated: 18000
n:8192 Total connections generated: 40339
n:16384 Total connections generated: 83646
Process finished with exit code 0
```

Graphical Representation: N vs $N/2 \ln(N)$



Unit Test:

