Yiqing Huang (001525629)

**Program Structures & Algorithms**

**Fall 2021**

**Assignment No. 3**

* **Task (List down the tasks performed in the Assignment)**
* **Step 1:**
  + **Implement height-weighted Quick Union with Path Compression;**
  + **Check that the unit tests all work;**
* **Step 2:**
  + **Develop a UF (“union-find”) client that takes an integer value from the command line as the number of “sites”;**
  + **Repeatedly generate a pair of integers between to and connect the two sites if they are not connected until there is only one component;**
  + **Encapsulate the client and package the algorithm above as a static method *count()*;**
  + **Create a main function that takes from the command line;**
* **Step 3:**
  + **Use sufficient (and sufficiently large) different values of to determine the relationship between the number of objects/sites and the number of pairs/connections**
  + **Justify the conclusion in terms of the observations and give a reasonable explanation.**
* **Step 1**

***\* src/main/java/edu/neu/coe/info6205/union\_find/UF\_HWQUPC.java***

* + **Unit tests result: All PASS.**
  + ***UF\_HWQUPC***
  + ***Graphical user interface, text

    Description automatically generated***
* **Step 2**

***+ src/main/java/edu/neu/coe/info6205/union\_find/UFClient.java***

***+ src/main/java/edu/neu/coe/info6205/union\_find/Pair.java***

* ***When there is one and only one input argument from the command line, call count() once and print the result.***
* ***Text

  Description automatically generated***
* ***Here is the result when :***
* ***Text

  Description automatically generated***
* **Step 3**

***\* src/main/java/edu/neu/coe/info6205/union\_find/UFClient.java***

***+ src/main/java/edu/neu/coe/info6205/union\_find/UFClientAnalysis.py***

* **Relationship Conclusion:**
* **Evidence to support the conclusion:** 
  + **Design a *multiCount(int start, int end, int step)* function to call *count(int n)* function for multiple times to satisfy the requirement of using sufficient (and sufficiently large) different values of .**
  + **Modify the main function of *UFClient*. When there are 3 input arguments from the command line, parse the first one as the initial , the second one as the maximum , and the third one as the step size.**
  + **Save the result into a csv file for the sake of analyzing the dataset in *Python*.**
  + **Let , , and here is the result:**
  + **A screenshot of a computer

    Description automatically generated with low confidence**
  + **In *Python* script**
    - **Split the dataset equally into 3 groups**
    - **Apply the *Least Squares Method* to find the fitted curve**
    - **Use various function to fit to find the best fitted function**
  + **In the experiment, I choose , , and to fit the curve.**
  + **A black screen with white text

    Description automatically generated with low confidence**
  + **From the result, it is obvious that using the *power* or *log* form is overfitted, which means when applying the result from one group to another group, it does not have a good fit. However, the *nlog* form result seems a good fit.**
  + **Thus, in conclusion, .**
  + **Chart, line chart

    Description automatically generated**
  + **What is more, from the plot, this conclusion seems acceptable.**