INFO 6205

Program Structures & Algorithms

Fall 2020

Assignment No.4

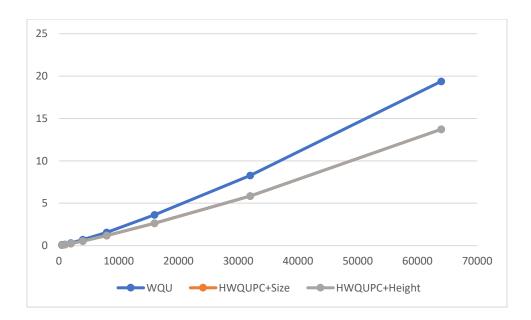
- Task: Quick Union with Path Compression with Benchmark
- **Output:** The union find algorithm based on height after path compression is faster than the original algorithm without path compression.

• Relationship conclusion

The time complexity of find () function of height based quick union algorithm without path compression is O (IgN), and the time complexity of union () function is O (IgN).

The time complexity of find () function of height based quick union algorithm with path compression is O (IgN), and the time complexity of union () function is O (IgN).

• Evidence to support relationship



Screenshot of Unit test passing

Test code

```
↓ UF HWQUPC.java

↓ *UF HWQU.java ♥ ↓ UF HWQUPC Test.java

↓ UF HWQU Test.java

↓ UF HWQ
                          public static void main(String[] args) {
   215
                                  int n =500;
   216
                                  for(int i=0; i<8; i++) {
    217
                                   //findTest(n);
    218
                                   //findTest1(n);
    219
                                   findTest2(n);
                                   n *=2;
   220
   221
                                    }
                         }
   222
   223
    224⊖
                         public static void findTest(int n) {
    225
                                    Integer[] xs = new Integer[n];

№226

                                    Benchmark_Timer a=new Benchmark_Timer<>(
    227
                                                         "WQU",
                                                         t -> {
   228
   229
                                                                   count(n);
   230
                                                         },null);

№231

                                   double time= a.run(xs,1000);
                                    System.out.println("n " + n + "
                                                                                                                                 Time " +time);
   232
   233
   234⊕
                     public static void findTest1(int n) {
   235
                                    Integer[] xs = new Integer[n];
   236
                                    UF HWQUPC pc = new UF HWQUPC(n);
Q<sub>1</sub>237
                                    Benchmark_Timer a=new Benchmark_Timer<>(
   238
                                                         "HWQUPC with Size",
   239
                                                         t -> {
M240
                                                                   pc.count(n);
   241
                                                        },null);

№242

                                   double time= a.run(xs,1000);
                                    System.out.println("n " + n + "
                                                                                                                                                 Time " +time);
   243
   244
   245⊖ public static void findTest2 (int n) {
   246
                                    Integer[] xs = new Integer[n];
   247
                                    UF_HWQUPC1 pc = new UF_HWQUPC1 (n);

№248

                                    Benchmark_Timer a=new Benchmark_Timer<>(
   249
                                                         "HWQUPC with height",
   250
                                                         t -> {

№251

                                                                   pc.count(n);
   252
                                                         },null);
Q<sub>1</sub>253
                                   double time= a.run(xs,1000);
   254
                                    System.out.println("n " + n + "
                                                                                                                                                Time " +time);
   255
    256
  257 }
```

UF_HWQUPC.java -> compass with size UF_HWQUPC1.java -> compass with height UF_HWQU -> without compass