Assignment No. 8: Disjoint Sets

Allocated time: 2 hours

Implementation

You are required to implement **correctly** and **efficiently** the basic operations for disjoint sets: `Make-Set(x)`, `Union(x, y)` and `Find-Set(x)` (section 21.1 from Cormen), using trees as underlying structures for each disjoint set (i.e. disjoint set forests – see section 21.3 from Cormen). Also, you should use *path compression* and *union by rank* heuristics in your implementations, to improve the running time.
Moreover, you are required to employ these data structures in an algorithm for finding the *connected components* of a graph (see section 21.1 from Cormen for details and pseudo-code).

Evaluation

! Before you start to work on the algorithms evaluation code, make sure you have a correct algorithm! You will have to prove your algorithm (i.e. the connected components algorithm) works on a small-sized graph (which you may hardcode in your main function).

Once you are sure your program works correctly, set the number of vertices $V = 10000$ and vary the number of edges E from 10000 to 60000; in each case, generate random graphs, apply the connected components algorithm and count the number of calls to `Make-Set(x)`, `Union(x, y)` and `Find-Set(x)` performed in the algorithm. Generate a chart which shows how the number of operations varies with E.

What is the running time of your algorithm?