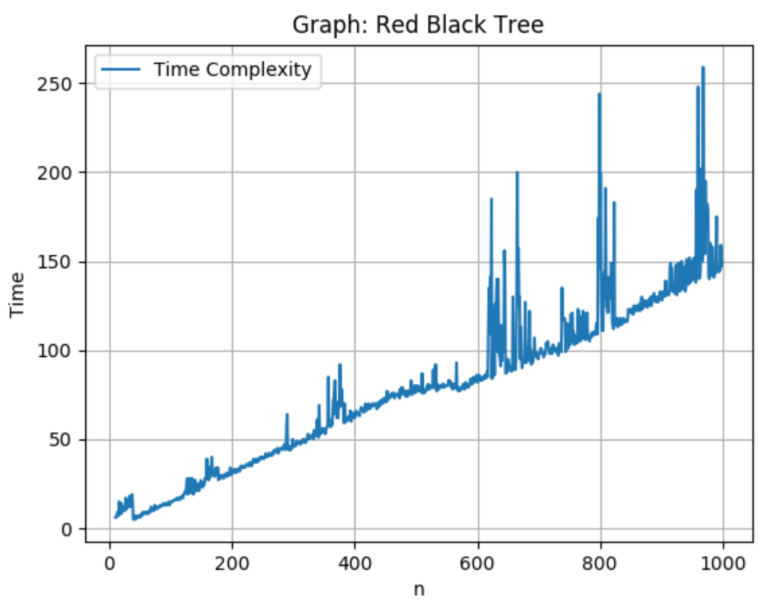
Analysis and Design of Algorithms
Homework 4

Arturo Fornés Arvayo A01227071

April 16, 2017

1. Red-Black Tree

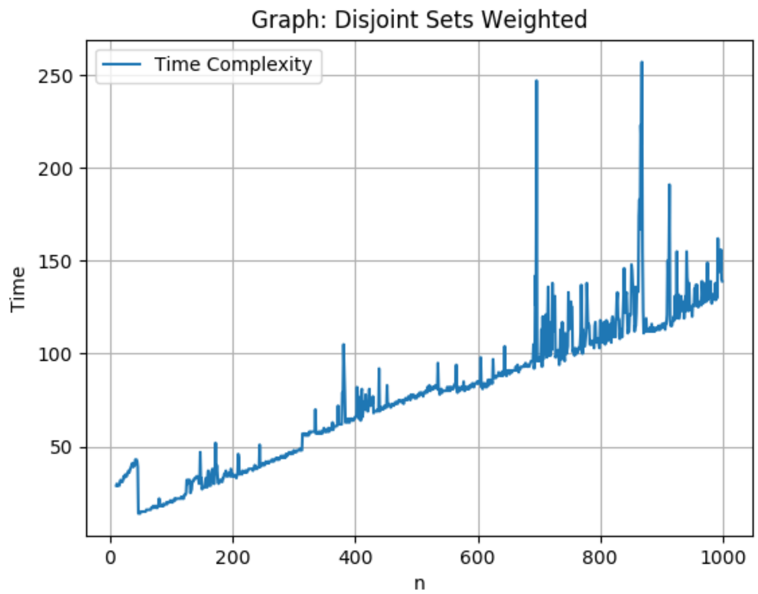
   Red-Black Tree code included in *rbtree.cpp*. An insertion in this data structure takes $O(lgn)$.

   This graph demonstrates the time it takes to execute n insertions in a Red-Black Tree, which follows a curve similar to $O(lgn)$.

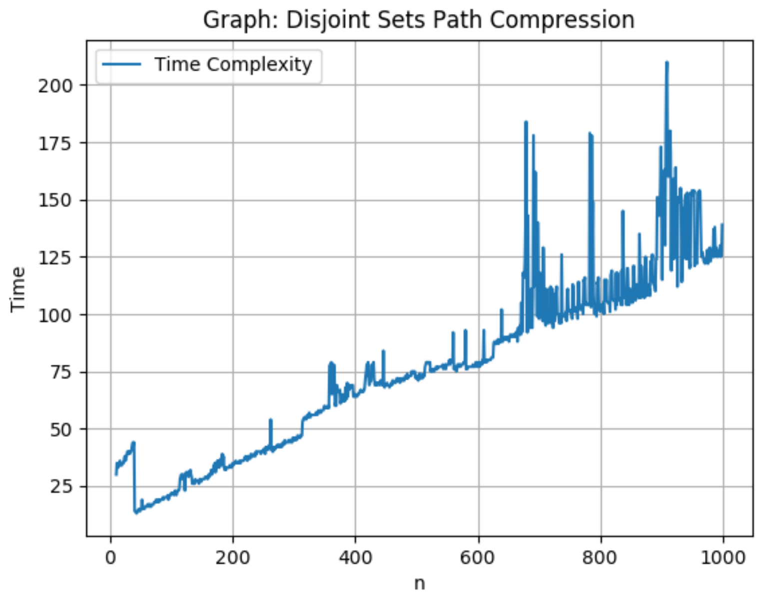
2. Disjoint Set

   (a) Weighted Union by Rank Representation. Code included in *ds_weighted.cpp*. Makeset operation takes O(1), Find-Set takes O(1) and Union takes $O(min\{\mid A \mid, \mid B \mid\})$.



This graph represents the time taken to execute a mix of n Union and Find-Set operations. Roughly approximating the complexity $O(m + nlgn)$ where m are Union and Find-Set Operations and n is the number of singletons at the beginning.

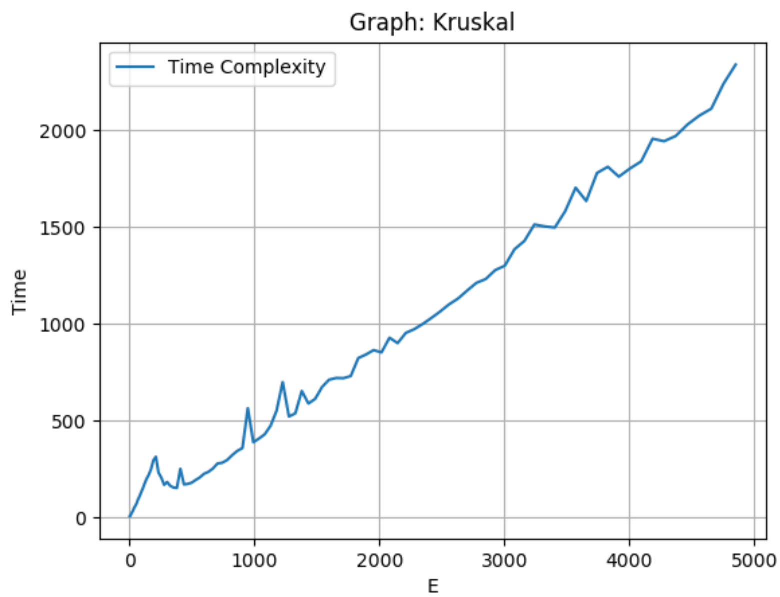
(b) Union by Rank + Path Compression Representation. Code included in *ds_path_compression.cpp*. With this representation, Union is said to take roughly $O(1)$.



This graph represents the time taken to execute a mix of n Union and Find-Set operations. Roughly approximating the complexity $O(m + nlgn)$ where m are Union and Find-Set Operations and n is the number of singletons at the beginning. Differently to the past graph, this one's Time-axis reaches a smaller value and better approximates $O(m + nlgn)$.

3. Kruskal

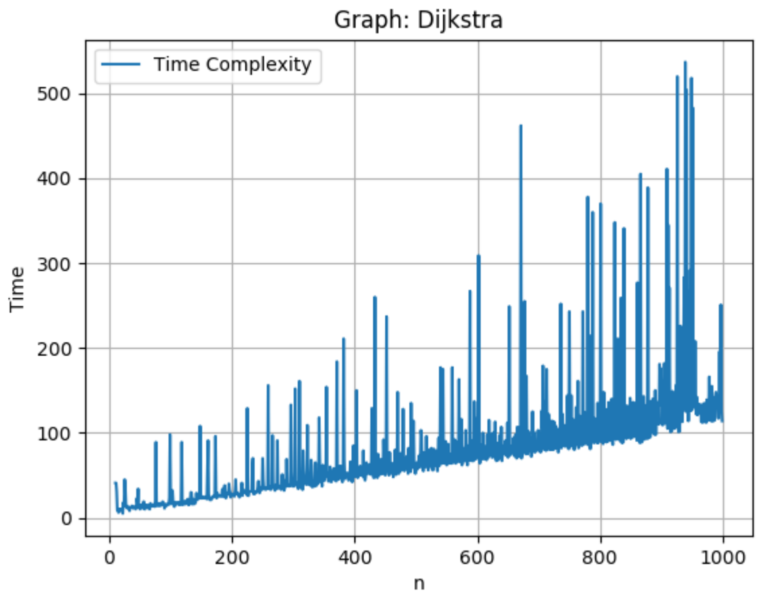
   Kruskal implementation code included in *kruskal.cpp*. The complexity of the Kruskal algorithm is $O(E\,lgV)$.



This graph represents Kruskal on a randomly generated complete graph of E edges. This approximates $O(E\,lgV)$.

4. Dijkstra

   Dijkstra implementation code included in *dijkstra.cpp*. The complexity of Dijkstra's algorithm is $O(E + V\,lgV)$.



This graph represents Dijkstra's algorith on a randomly generated graph of n vertices. The general curve of the graph approximates the complexity described of $O(E + V\,lgV)$.