Computer Science 2300: Lab 4

Due: April 4, 2012

For this lab you will be implementing a specific version of Kruskal's algorithm and comparing its performance with Prim's algorithm.

1 Implementing Kruskal's algorithm

Write a program that implements Kruskal's algorithm **using a disjoint set forest, union-by-rank, and path compression**. Run Kruskal's algorithm on random graphs starting from 100 vertices and increasing in orders of 2 up to 6400 vertices. Calculate the time taken to generate the MST's over an **average of 10 runs**. Ensure that your implementation of Kruskal's does not run excessively slowly as compared to the provided implementation of Prim's algorithm. Compute two sets of timings:

1. The time taken to run Kruskal's algorithm in its entirety.
2. The time taken after the sorting of edges has been finished.

Also compute the running time for the provided complete implementation of Prim's algorithm.
Note: Most of the major functions you need have been implemented for you already. Specifically, we are giving you code for random graph generation and a comparator so that you can use STL sort. There is also a header file for the disjoint set forest implementation. Spend some time reading and becoming familiar with the existing code base for this lab. You may modify anything you like, but it is not recommended.

2 Analysis and Comparison

To receive full credit, show the TA two graphs. The first one should show the average runtime of Prim's algorithm, Kruskal's algorithm, and Kruskal's algorithm after the sort phase is finished, on complete graphs (with random edge weights) of size 100, 200, 400, 800, 1600, 3200, and 6400. The second should show the average weight of an edge that is included in the MST for graphs of these sizes.