

CS 4720/5720 Design and Analysis of Algorithms

Homework #5a

Due: Tuesday, November 27, 2018

If written: due in class

If typed and submitted on Canvas: 11:59 pm

Submission requirements:

1. **5% extra credit** if you submit *digital, typed* writeups in PDF format to the “Homework 4 Writeup” assignment on the Canvas site. However, you may also turn in handwritten assignments in class – but assignments submitted in person will not receive the 5% extra credit. The due dates are the same for either submission method, but I will need handwritten assignments turned in to me in class.
2. Note: this is a “short” homework with no programming assignment, and will be worth 50 points, half as much as a normal homework.

Assignment:

1. *Comparing Minimum Spanning Tree Algorithms:* In class we went over two algorithms which find the minimum spanning tree of a weighted, connected graph. In this exercise, you will compare the worst-case time efficiencies of these algorithms using various graph and data structure representations.
 - (a) **Show that Prim’s Algorithm has $\Theta(|V|^2)$ time complexity** when the graph is represented by a weight matrix and the priority queue is implemented as an un-ordered array (that is, if the queue has n items, inserting is $\Theta(1)$ but deleting and updating are $\Theta(n)$). (Hint: you might find it useful to refer to the book’s pseudocode for Dijkstra’s algorithm; this pseudocode shows the priority queue operations explicitly)
 - (b) **Show that Prim’s Algorithm has $\Theta(|E| \log |V|)$ time complexity** when the graph is represented by adjacency lists and the priority queue is implemented as a min-heap (that is, if the queue has n items, inserting, updating, and deleting are all $\Theta(\log n)$). (Hint: you might find it useful to refer to the book’s pseudocode for Dijkstra’s algorithm; this pseudocode shows the priority queue operations explicitly)

Now, Kruskal's algorithm can be as fast as $\Theta(|E| \log |E|)$ when the graph is represented by adjacency lists and a fast sorting algorithm is used to sort the edge weights (to be precise, this also requires another optimization which the book explains in the section "Disjoint Subsets and Union-Find Algorithms; read that section if you're interested). So we have this list:

- $\Theta(|V|^2)$: Prim with weight matrix and un-ordered array.
 - $\Theta(|E| \log |V|)$: Prim with adjacency lists and min-heap.
 - $\Theta(|E| \log |E|)$: Kruskal with adjacency lists and fast sort.
- (c) A graph that has few edges is called *sparse*, and has $|E| \in \Theta(|V|)$. **Which of the above combinations of algorithms and data structures is asymptotically fastest for finding a MST in a sparse graph?**
- (d) What about a graph that has a moderate number of edges with $|E| \in \Theta(|V| \log |V|)$? **Which of the above combinations of algorithms and data structures is asymptotically fastest for finding a MST in this type of graph?**
- (e) A graph that has many edges is called *dense*, and has $|E| \in \Theta(|V|^2)$. **Which of the above combinations of algorithms and data structures is asymptotically fastest for finding a MST in a dense graph?**