WEEK 09

1. Preparation for Assignment

If, and *only if* you can truthfully assert the truthfulness of each statement below are you ready to start the exercises.

1.1. Reading Comprehension Self-Check.

- I understand that the **greedy technique** suggests constructing a solution to an optimization problem through a sequence of steps, each expanding a partially constructed solution obtained so far, until a complete solution to the problem is reached.
- I know why it is **false** to say that on each step of a greedy algorithm, the choice made must be *feasible, globally optimal, and irrevocable.*
- I know why it is **false** to say that Prim's and Kruskal's algorithms are two different greedy algorithms for constructing minimum spanning trees of weighted *unconnected* graphs.
- I know why it is **false** to say that the problem of scheduling n jobs of unknown durations for execution by a single processor is ideal for solution with a greedy algorithm.
- I have begun figuring out how a greedy algorithm can be used to find an optimal set of n weights so that it would be possible to weigh on a balance scale any integer load in the largest possible range from 1 to W, provided weights can be put only on the free cup of the scale, or provided weights can be put on both cups of the scale.
- I remember what a digraph is.
- I know that Dijkstra?s algorithm solves the single-source shortest-path problem of finding shortest paths from a given vertex (the source) to all the other vertices of a weighted graph or digraph.
- I know that Dijkstra's algorithm works like Prim?s algorithm but compares path lengths rather than edge lengths.
- I know why Dijkstra?s algorithm always yields a correct solution for a graph with nonnegative weights.
- I know that a Huffman tree is a binary tree that minimizes the weighted path length from the root to the leaves of predefined weights.
- I know that the most important application of Huffman trees is Huffman codes.

Date: June 16, 2020.

- I know that a Huffman code is an optimal prefix-free variable-length encoding scheme that assigns bit strings to symbols based on their frequencies in a given text.
- I know that a Huffman code is created by a greedy construction of a binary tree whose leaves represent the alphabet symbols and whose edges are labeled with 0?s and 1?s.

1.2. **Memory Self-Check.** How many algorithms for finding minimum spanning trees of undirected connected graphs, with their names, do you remember from the reading for this week? How many others, including their names, can you find on the internet?

2. Week 09 Exercises

2.1. **Exercise 1 on page 323.** Write code in a language of your choice instead of pseudocode.

2.2. **Exercise 1 on page 332.**

2.3. **Exercise 2 on page 332.**

2.4. **Exercise 3 on page 338.**

2.5. **Exercise 1 on page 342.**

2.6. **Exercise 3 on page 343.**

3. Week 09 Problems

3.1. **Exercise 3 on page 322.**

3.2. **Not in the Book.** Prim's and Kruskal's algorithms are quite similar. The only real difference is in the set of edges available for inclusion in the tree at each step. In Prim's algorithm, only those edges that are adjacent to edges already in the tree (and not completing simple cycles) may be added (so that, as a result, all the intermediate stages are trees). In Kruskal's algorithm, any edge that does not complete a simple cycle may be added (so that the intermediate stages may be forests and not trees).

Create minimum spanning trees for this and this graph. When applicable, use lexicographical order to break ties. List the edges in the order chosen by each algorithm, and find the total weight.