Minimum-Cost Spanning Trees

Read the book chapters first, then make sure you can answer the questions in the notes. Following that, work on some skills-check problems and exercises. Then take the online quizzes.

Reading 2.4
Skills Check 17, 22, 23, 24, 25, 26, 27
Exercises 54, 55, 57 (also do via Prim's), 58, 60, 62, 66 (Challenge), 69
Quiz (Not ready) Take the quiz[1]

This day's reading assignment also includes a file on traveling salesman problem. Don't miss it.

2.4

- When do we call a subgraph a *tree*? When do we call it a *spanning tree*?
- Does every graph have a spanning tree subgraph?
- Are there many possible spanning tree subgraphs?
- What are we after in *minimum-cost spanning tree* problems?
- Provide some real-world situations from your experience that you might approach as minimum-cost spanning tree problems.
- How do Minimum-cost Spanning Tree problems differ from Hamiltonian Circuit problems and from Traveling Salesman Problems?
- Describe how *Kruskal's algorithm* works.
- Demonstrate Kruskal's algorithm in some examples.
- Does Kruskal's algorithm always produce the minimum-cost spanning tree?
- There is another algorithm not discussed in the text, called *Prim's algorithm*, that works as follows:
  - Start from an arbitrary vertex. Look at all edges that start from that vertex, and pick the one with the smallest cost. So now you have selected two vertices and an edge between them.
  - At the next step, look at all the vertices you have selected, and look at all the edges that go from one of these vertices to a vertex you haven't yet selected. Choose among these the edge with the smallest cost. Then add that edge, and the new vertex it connected, to your set.
  - Continue until you have included all vertices.
- Demonstrate Prim's algorithm in some examples.

[1] https://moodle.hanover.edu/mod/quiz/view.php?id=