CSE 326: Data Structures

Topic #17:
Let’s get connected… minimally!

Ashish Sabharwal
Autumn, 2003

Today’s Outline

• Discuss Quiz #5
• Finish Shortest Path Problems
• Minimum Spanning Trees

Before we move on…

• Dijkstra’s algorithm, as we saw, gives the minimum distance between s and t.
• Can we modify it to output the shortest path between s and t?

<table>
<thead>
<tr>
<th>node</th>
<th>known</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An Application:
Moving Around Washington

What’s the fastest way from Seattle to Pullman?

Answer:

A Different Application:
Communication in Washington

What’s the cheapest inter-city network?

Is This Problem Really Different?

• Is knowing Dijkstra’s algorithm enough to solve the latter application?

Yes? Then how?
No? Then why?
Spanning Tree, MST

*Spanning tree*: a subgraph of a connected, undirected graph that
1. touches all vertices in the graph (*spans* the graph)
2. forms a tree (is connected and contains no cycles)

Minimum spanning tree: the spanning tree with the least total edge cost.

Two Different Approaches

Prim’s Algorithm
Almost identical to Dijkstra’s

Kruskals’s Algorithm
Completely different!

Prim’s Algorithm for MST

*A node-based greedy algorithm*
Builds MST by greedily adding nodes

1. Select a node to be the “root”
   - mark it as known
   - Update cost of all its neighbors
2. While there are unknown nodes left in the graph
   a. Select an unknown node $b$ with the smallest cost from some known node $a$
   b. Mark $b$ as known
   c. Add $(a, b)$ to MST
   d. Update cost of all nodes adjacent to $b$

Prim’s Algorithm: Example

Prim’s Algorithm: Complexity

- Depends on what?

- How long does each step take?

Runtime:

Prim’s Algorithm: Correctness

- A proof very similar to that of Dijkstra’s algorithm works!

(Left as exercise)
Kruskal’s Algorithm for MST

An edge-based greedy algorithm
Builds MST by greedily adding edges

1. Initialize with
   • empty MST
   • all vertices marked unconnected
   • all edges unmarked
2. While there are still unmarked edges
   a. Pick the lowest cost edge \((u, v)\) and mark it
   b. If \(u\) and \(v\) are not already connected, add \((u, v)\) to the
      MST and mark \(u\) and \(v\) as connected to each other

Doesn’t it sound familiar?

Kruskal’s Algorithm: Complexity

• Depends, of course, on the data structures/ADT used.
  What should we use?

• How long does each step take?

Kruskal’s Algorithm: Example

Play at Home with Prim

1. Starting at node A, find the MST using Prim’s method.
   (continue on next slide)

Play at Home with Kruskal

2. Now find the MST using Kruskal’s method.
3. Under what conditions will these methods give the same result?
4. What data structures should be used for Kruskal’s? Running time?
To Do

• Read sections 9.1 – 9.3, 9.5