Kruskal’s Running Time Analysis

We have multiple choices on which underlying data structure to use to build the Priority Queue used in Kruskal’s Algorithm:

<table>
<thead>
<tr>
<th>Priority Queue Implementations</th>
<th>Heap</th>
<th>Sorted Array</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>6-8</td>
<td></td>
</tr>
<tr>
<td>Each removeMin</td>
<td>13</td>
<td></td>
</tr>
</tbody>
</table>

Based on our algorithm choice:

<table>
<thead>
<tr>
<th>Priority Queue Implementation</th>
<th>Total Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heap</td>
<td></td>
</tr>
<tr>
<td>Sorted Array</td>
<td></td>
</tr>
</tbody>
</table>

Reflections

Why would we prefer a Heap?

Why would be prefer a Sorted Array?

Partition Property

Consider an arbitrary partition of the vertices on G into two subsets U and V.

Let e be an edge of minimum weight across the partition.

Then e is part of some minimum spanning tree.

*Proof in CS 374!*
Partition Property Algorithm

Prim’s Minimum Spanning Tree Algorithm

```
Pseudocode for Prim’s MST Algorithm

PrimMST(G, s):
1. Input: G, Graph;
2. s, vertex in G, starting vertex of algorithm
3. Output: T, a minimum spanning tree (MST) of G
4. foreach (Vertex v : G):
5.     d[v] = +inf
6.     p[v] = NULL
7. d[s] = 0
8. PriorityQueue Q  // min distance, defined by d[v]
9. Q.buildHeap(G.vertices())
10. Graph T   // "labeled set"
11. repeat n times:
12.     Vertex m = Q.removeMin()
13.     T.add(m)
14.     foreach (Vertex v : neighbors of m not in T):
15.         if cost(v, m) < d[v]:
16.             d[v] = cost(v, m)
17.             p[v] = m
18. return T
```

Running Time of MST Algorithms

Kruskal’s Algorithm:

Prim’s Algorithm:

Q: What must be true about the connectivity of a graph when running an MST algorithm?

...what does this imply about the relationship between n and m?

<table>
<thead>
<tr>
<th></th>
<th>Adj. Matrix</th>
<th>Adj. List</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heap</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unsorted Array</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CS 225 – Things To Be Doing:

1. Programming Exam C is different than usual schedule:
   Exam: Sunday, Dec 2 – Tuesday, Dec 4
2. MP7 Released – Slightly different structure:
   Hard Deadline on Monday, Dec. 3 for Part 1
3. lab_ml in lab this week!
4. Daily POTDs are ongoing for +1 point / problem