Kruskal’s Algorithm

Pseudocode for Kruskal’s MST Algorithm

```java
KruskalMST(G):
    DisjointSets forest
    foreach (Vertex v : G):
        forest.makeSet(v)
    PriorityQueue Q // min edge weight
    foreach (Edge e : G):
        Q.insert(e)
    Graph T = (V, {})
    while |T.edges()| < n-1:
        Vertex (u, v) = Q.removeMin()
        if forest.find(u) == forest.find(v):
            T.addEdge(u, v)
            forest.union( forest.find(u), forest.find(v) )
    return T
```

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></td>
<td></td>
</tr>
<tr>
<td>: 6–8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Each removeMin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>: 13</td>
<td></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):

Input: G, Graph;
      s, vertex in G, starting vertex of algorithm

Output: T, a minimum spanning tree (MST) of G

foreach (Vertex v : G):
  d[v] = +inf
  p[v] = NULL

d[s] = 0

PriorityQueue Q  // min distance, defined by d[v]
Q.buildHeap(G.vertices())

Graph T  // “labeled set”

repeat n times:
  Vertex m = Q.removeMin()
  T.add(m)

  foreach (Vertex v : neighbors of m not in T):
    if cost(v, m) < d[v]:
      d[v] = cost(v, m)
      p[v] = m

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