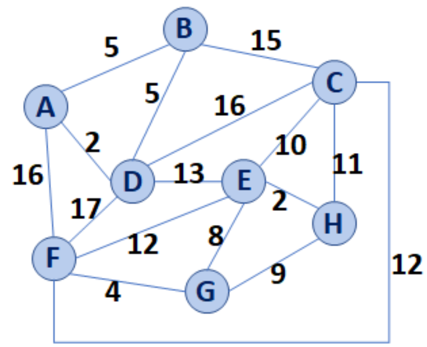
#37: Kruskal + Prim's Algorithm

April 20, 2018 · *Wade Fagen-Ulmschneider*

Kruskal's Algorithm



(A, D)
(E, H)
(F, G)
(A, B)
(B, D)
(G, E)
(G, H)
(E, C)
(C, H)
(E, F)
(F, C)
(D, E)
(B, C)
(C, D)
(A, F)
(D, F)

Pseudocode for Kruskal's MST Algorithm

```
1   KruskalMST(G):
2     DisjointSets forest
3     foreach (Vertex v : G):
4       forest.makeSet(v)
5
6     PriorityQueue Q    // min edge weight
7     foreach (Edge e : G):
8       Q.insert(e)
9
10    Graph T = (V, {})
11
12    while |T.edges()| < n-1:
13      Vertex (u, v) = Q.removeMin()
14      if forest.find(u) == forest.find(v):
15         T.addEdge(u, v)
16         forest.union( forest.find(u),
17                       forest.find(v) )
18
19    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:

Priority Queue Implementations:	Heap	Sorted Array
Building :6-8		
Each removeMin :13		

Based on our algorithm choice:

Priority Queue Implementation:	Total Running Time
Heap	
Sorted Array	

Reflections

Why would we prefer a Heap?

Why would be prefer a Sorted Array?

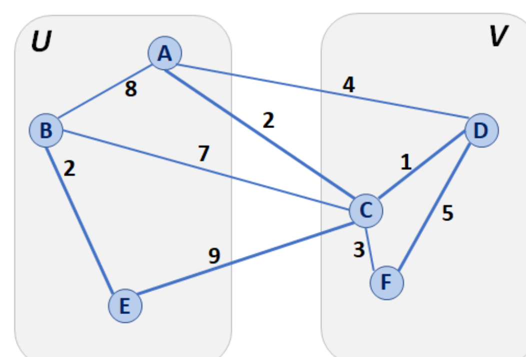
Partition Property

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

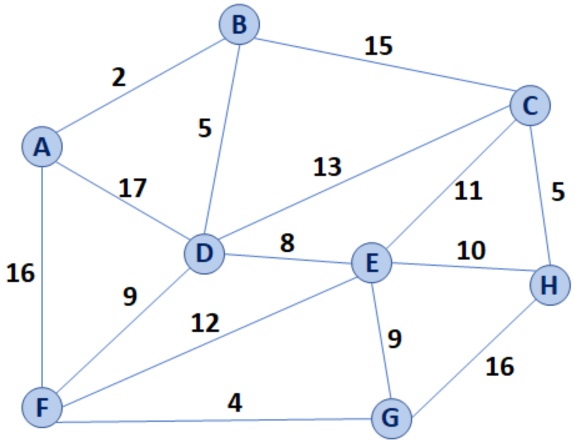
Let **e** be an edge of minimum weight across the partition.

Then **e** is part of some minimum spanning tree.

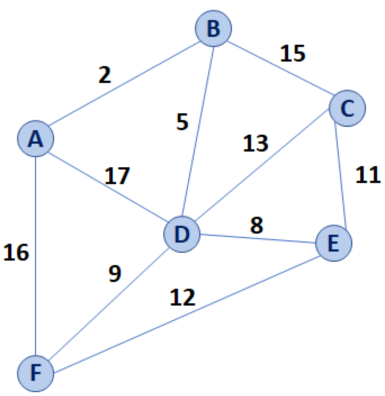
Proof in CS 374!



Partition Property Algorithm



Prim's Minimum Spanning Tree Algorithm



	Adj. Matrix	Adj. List
Heap		
Unsorted Array		

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**?

CS 225 – Things To Be Doing:

1. Programming Exam C ongoing
2. MP7 is released; EC due tonight, Monday, April 23th
3. lab_graphs available; due Sunday, April 22nd
4. Daily POTDs end **next** Friday (April 27th); +6 left to complete!