#37: MSTs: Prim's Algorithm

November 17, 2021 · *G Carl Evans*

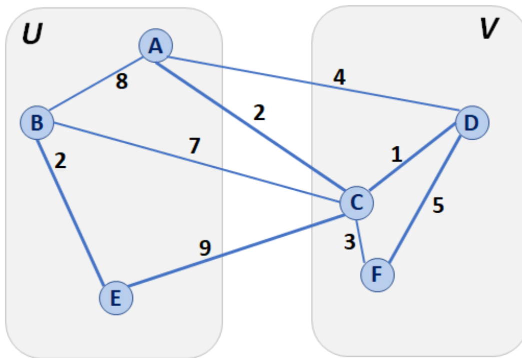
Partition Property

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

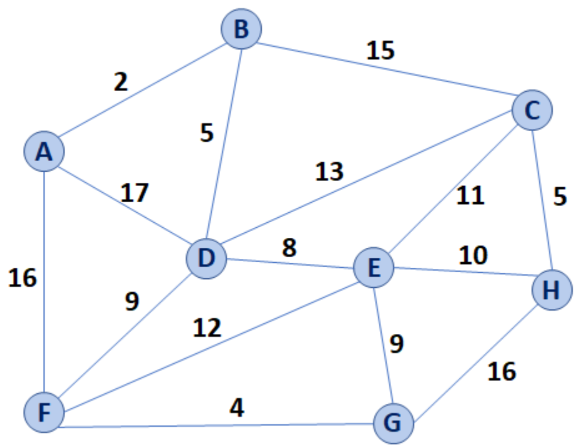
Let **e** be an edge of minimum weight across the partition.

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

Proof in CS 374!

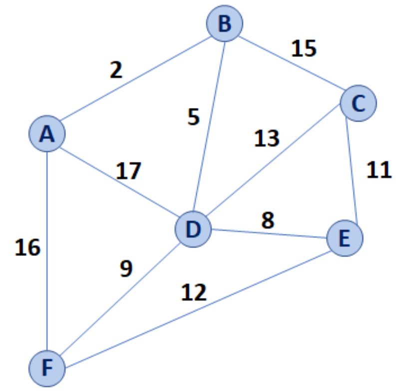


Partition Property Algorithm



	Adj. Matrix	Adj. List
Heap		
Unsorted Array		

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**?

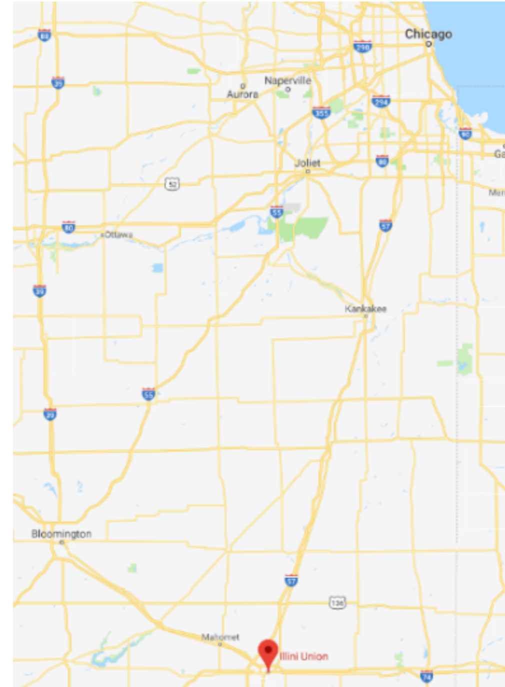
Kruskal's MST	Prim's MST

Q: Suppose we built a new heap that optimized the decrease-key operation, where decreasing the value of a key in a heap updates the heap in amortized constant time, or O(1)*. How does that change Prim's Algorithm runtime?

Final big-O Running Times of classical MST algorithms:

Kruskal's MST	Prim's MST

Shortest Path Home:



CS 225 – Things To Be Doing:
1. Get your projects approved and start work on them. 2. Daily POTDs are ongoing for +1 point /problem but pausing over break