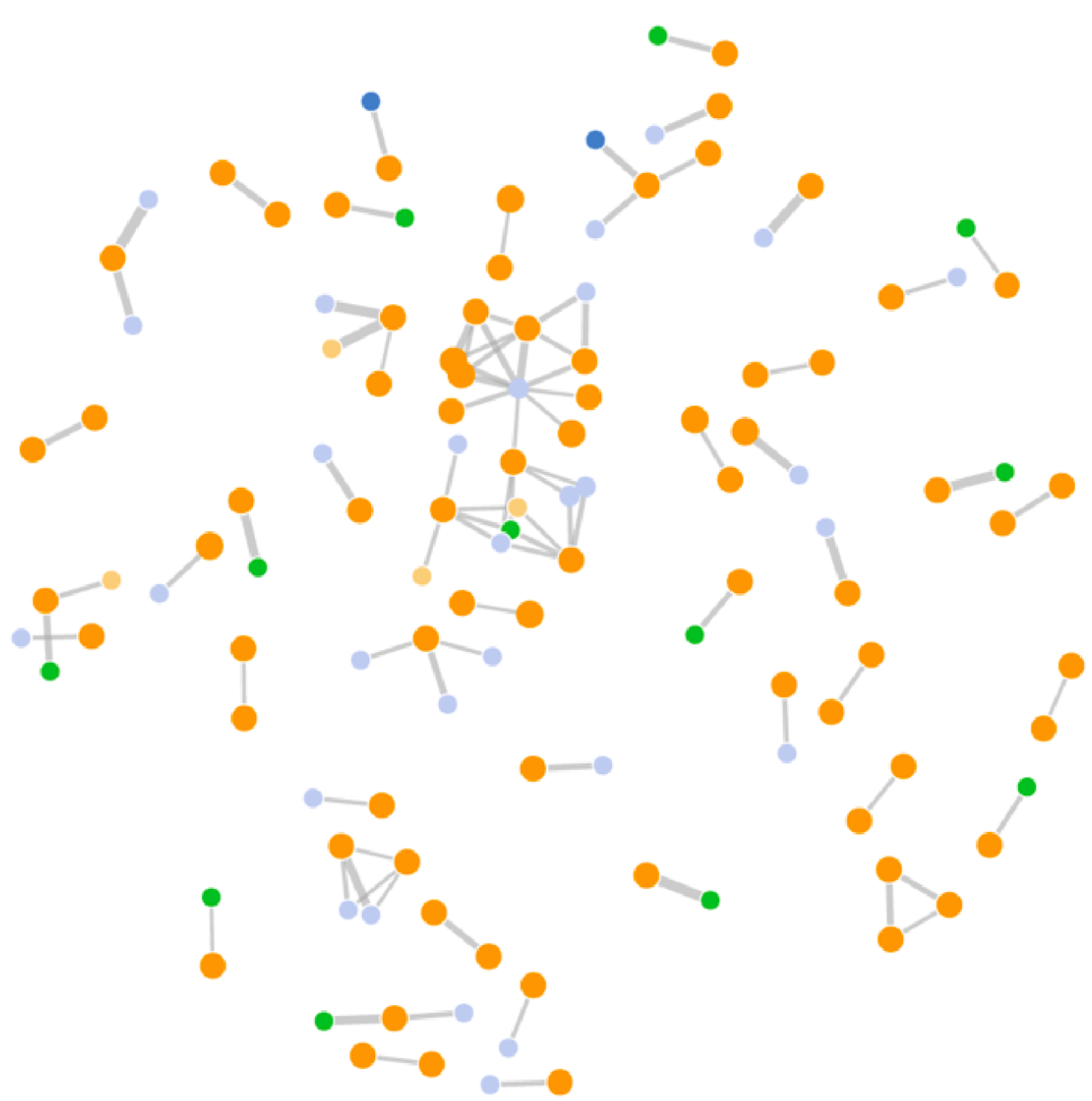
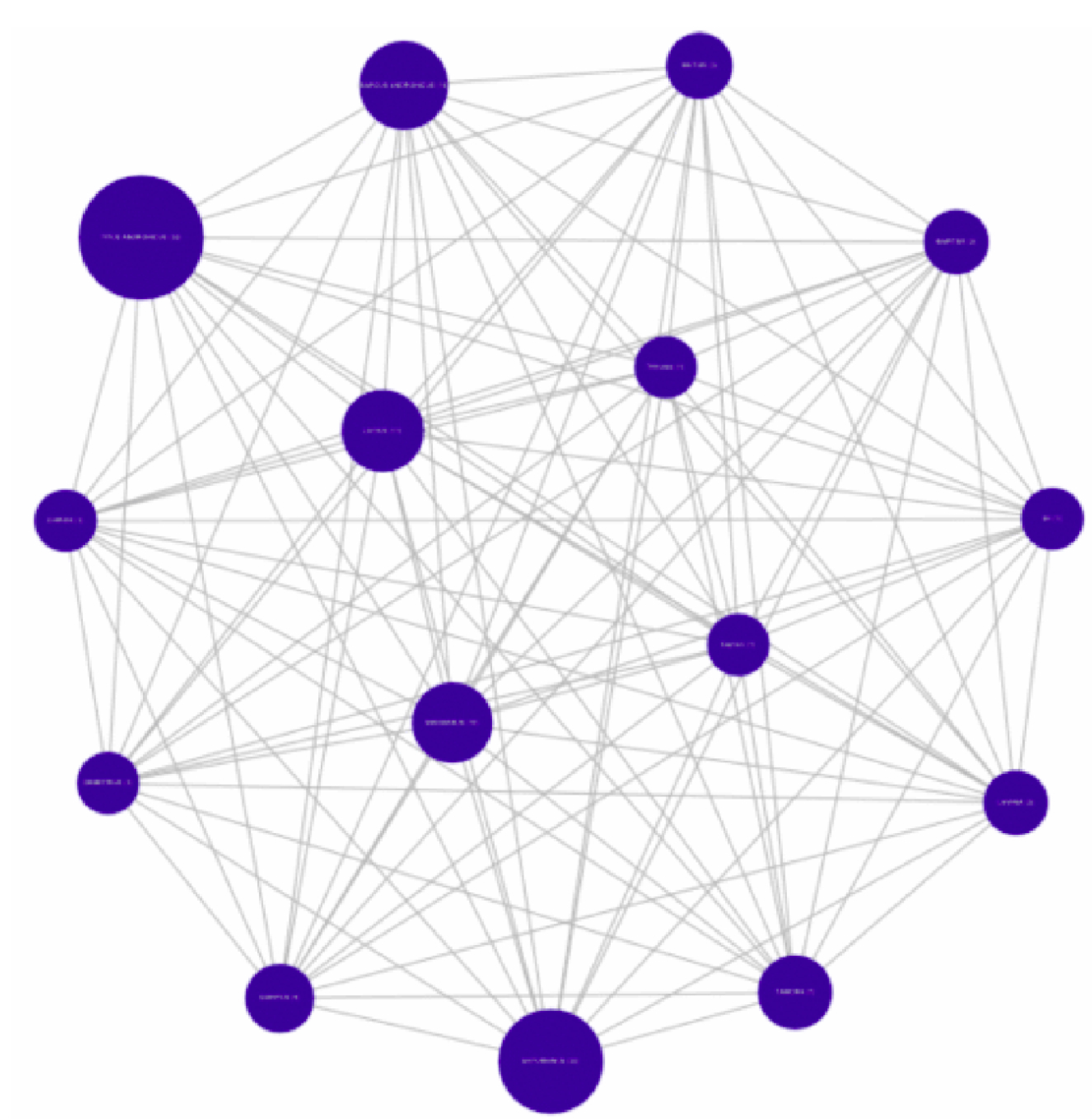
Announcements

* **Final Exam:** 05/09 1:30-4:30pm, location TBD (see website for updates)
Email to Thierry @ ramais@illinois.edu asap with "conflict" in subject line

* Review session: Thursday, 05/05 @ 11am in SC 1404





How would you characterize the difference between these graphs?

Prim's algorithms (1957) is based on the Partition Property:

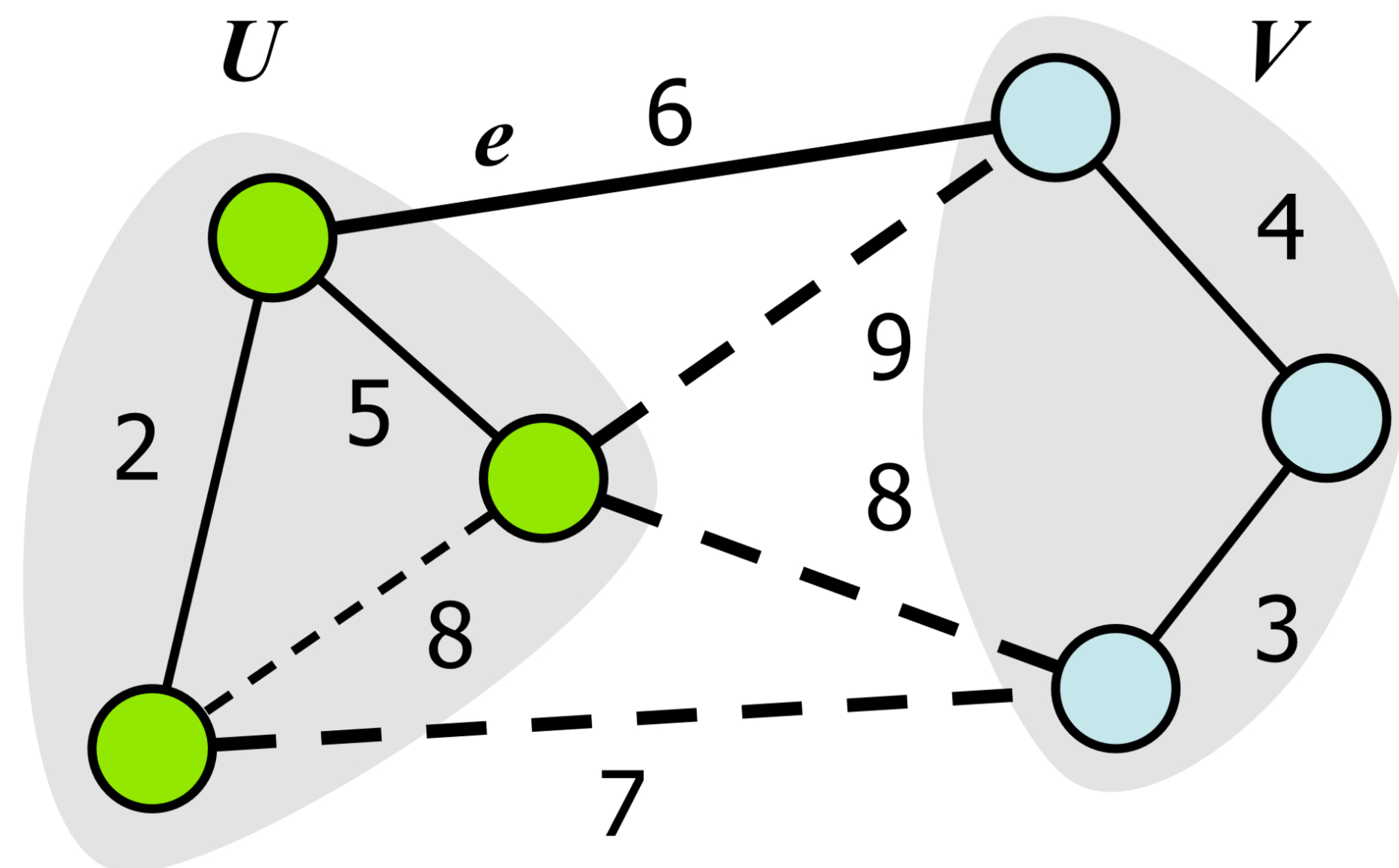
Consider a partition of the vertices of G into subsets U and V.

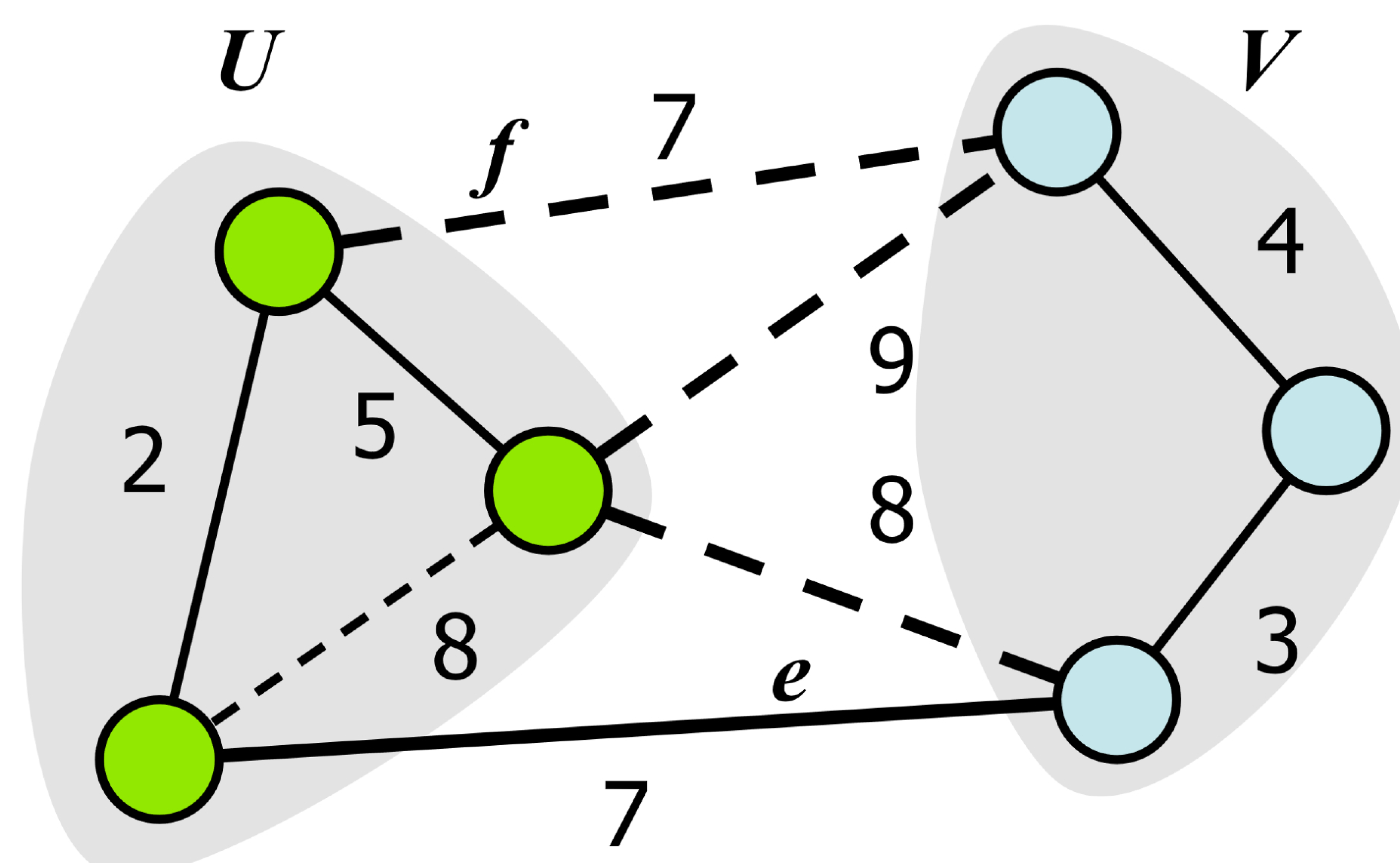
Let e be an edge of minimum weight across the partition.

Then e is part of some minimum spanning tree.

Proof:

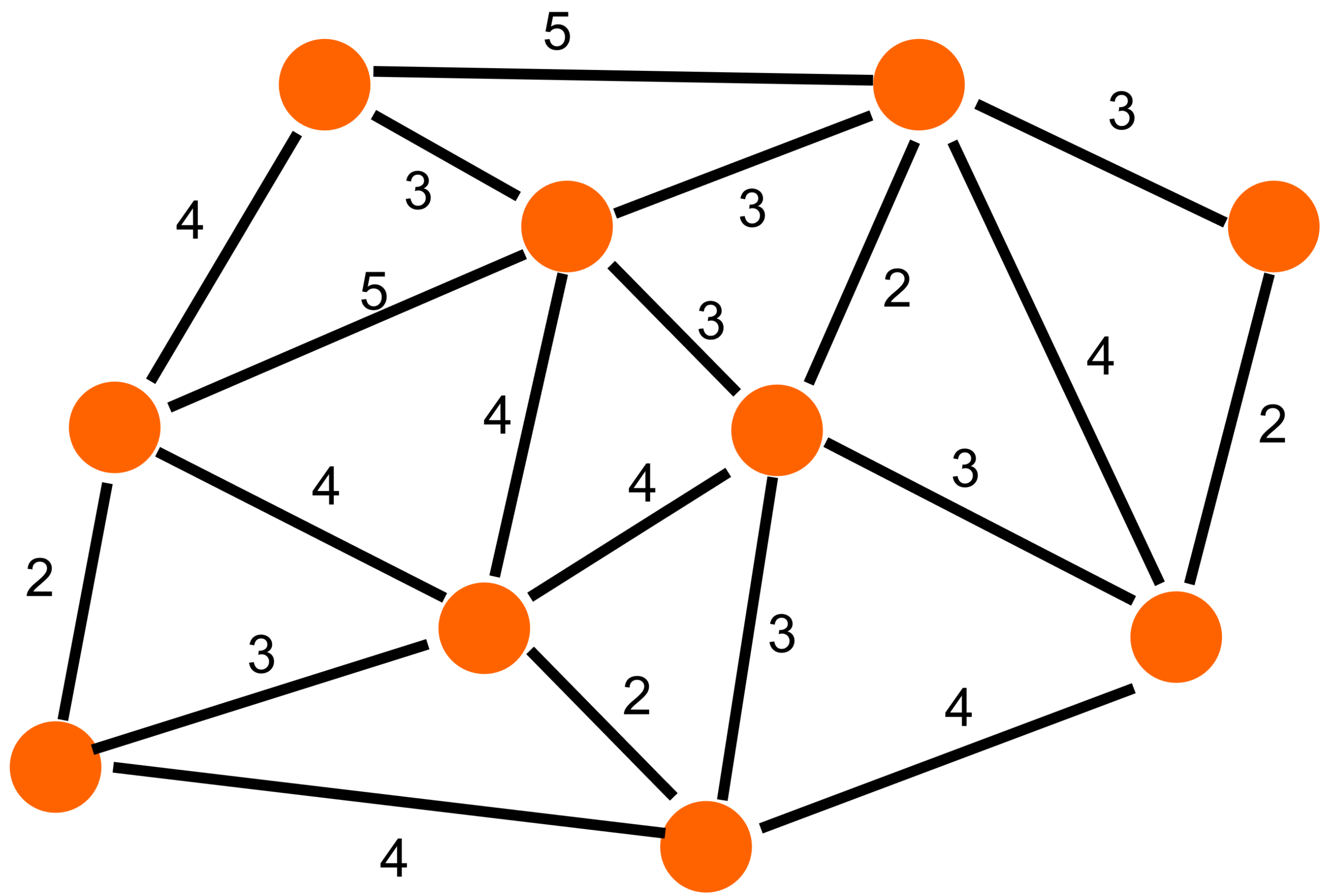
See cs374





MST - minimum total weight spanning tree

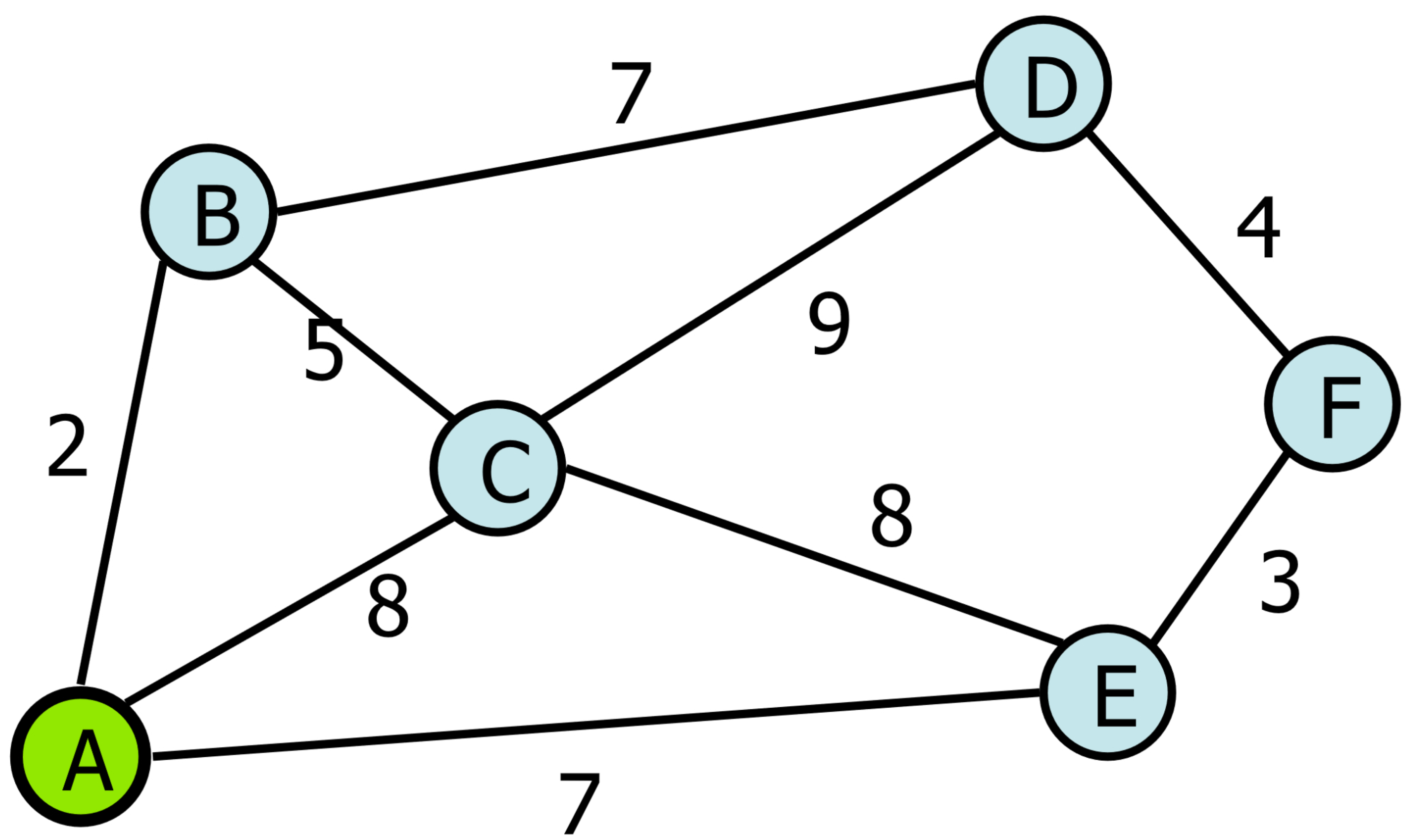
Theorem suggests an algorithm...



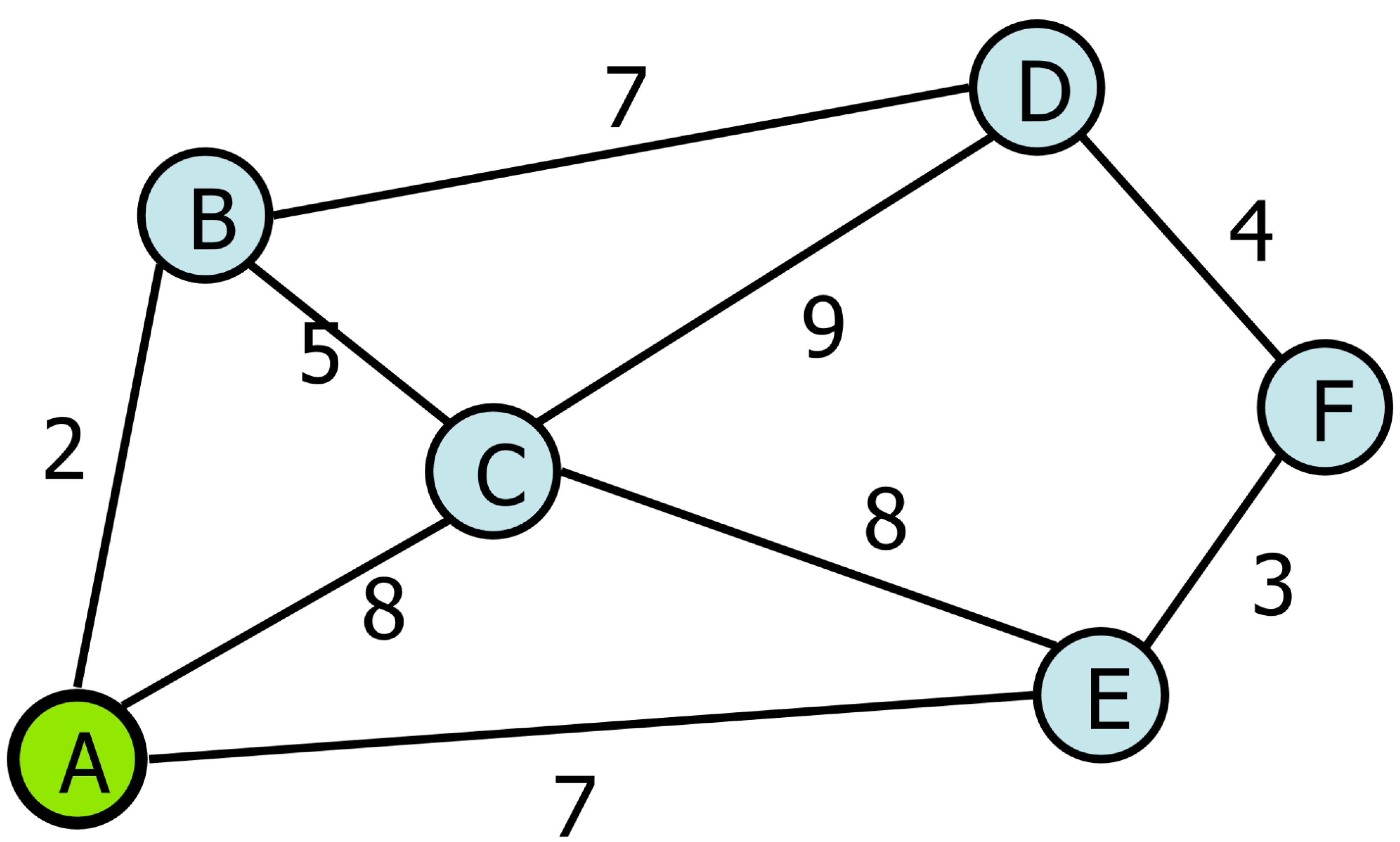
Example of Prim's algorithm -

Initialize structure:

1. For all v, d[v] = "infinity", p[v] = null
2. Initialize source: d[s] = 0
3. Initialize priority (min) queue
4. Initialize set of labeled vertices to ∅.



Example of Prim's algorithm -



Initialize structure:

1. For all v, d[v] = "infinity", p[v] = null
2. Initialize source: d[s] = 0
3. Initialize priority (min) queue
4. Initialize set of labeled vertices to ∅.

Repeat these steps n times:

- Find & remove minimum d[] unlabelled vertex: v
- Label vertex v
- For all unlabelled neighbors w of v,

  If cost(v,w) < d[w]

  d[w] = cost(v,w)

  p[w] = v

Prim's Algorithm (undirected graph with unconstrained edge weights):

Initialize structure:

1. For all v, d[v] = "infinity", p[v] = null
2. Initialize source: d[s] = 0
3. Initialize priority (min) queue
4. Initialize set of labeled vertices to ∅.

Repeat these steps n times:

- Remove minimum d[] unlabeled vertex: v
- Label vertex v (set a flag)
- For all unlabeled neighbors w of v,

  If cost(v,w) < d[w]

    d[w] = cost(v,w)

    p[w] = v

	adj mtx	adj list
heap	O(n	O(n log n + m log n)
Unsorted array	O(n	O(n

Which is best?

Depends on density of the graph:

Sparse

Dense