Graph applications and computing an MST

10/11/24

Administrivia

- HW 4 (Dynamic programming and Graph applications) due Tuesday night

Graph algorithms/problems so far

- BFS: Traverse and find distance in unweighted graph
- DFS: Traverse and handle multiple components
- MST: Find minimum spanning tree (connect at minimum cost)
- Shortest path

Which of the following algorithms is most appropriate for the Skyscraper floors problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Skyscraper floors problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Domino game problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Domino game problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Highways problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Highways problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Monitoring the Amazon problem?

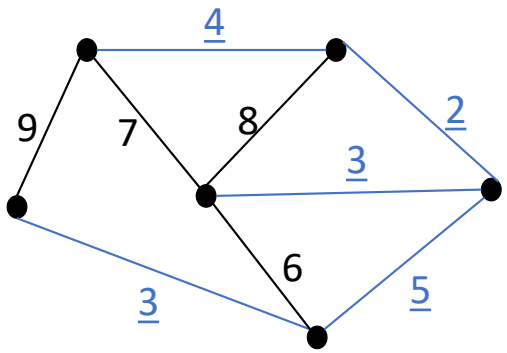
A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Which of the following algorithms is most appropriate for the Monitoring the Amazon problem?

A. BFS
B. DFS
C. Either BFS or DFS
D. MST
E. Shortest path

Recall: Minimum Spanning Tree (MST)

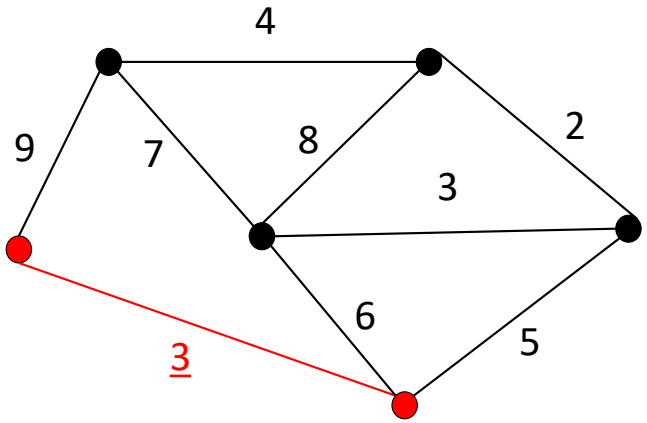
- A subset of edges that connects all vertices and has minimum total weight



Total cost: 17

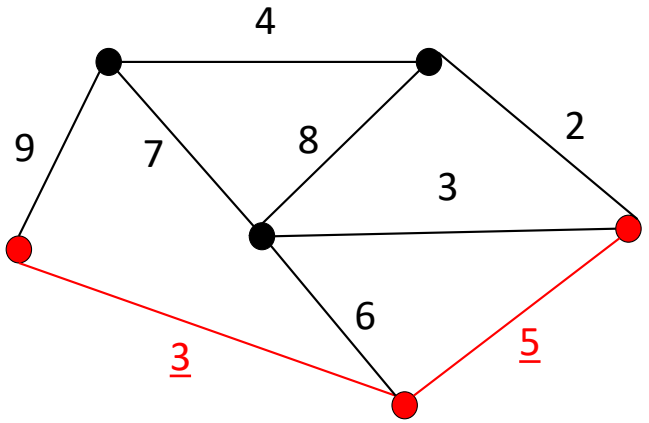
Prim-Jarnik algorithm

- Repeatedly add the cheapest edge leaving the set of vertices reachable from a root vertex r



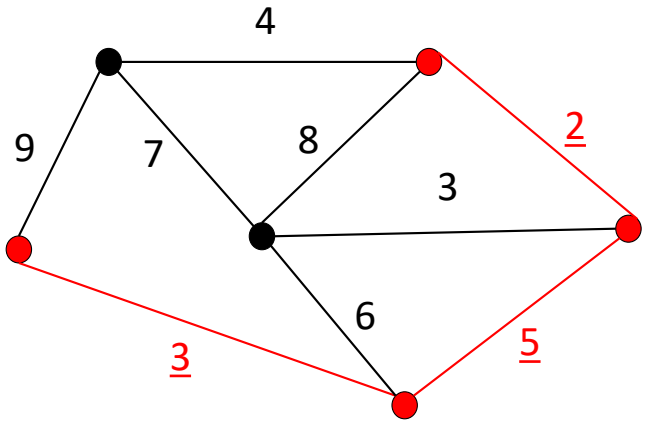
Prim-Jarnik algorithm

- Repeatedly add the cheapest edge leaving the set of vertices reachable from a root vertex r



Prim-Jarnik algorithm

- Repeatedly add the cheapest edge leaving the set of vertices reachable from a root vertex r



Prim-Jarnik algorithm

- Repeatedly add the cheapest edge leaving the set of vertices reachable from a root vertex r

```
MST-Prim(G,w, r)
        set key of r to 0, set keys for other vertices to ∞
        priority queue Q = G.V
        while Q != ∅
                u = Extract-min(Q)
                for each neighbor v of u
                        if v in Q and w(u,v) < v.key
                                v.key = w(u,v)
```

Cut property

Let G=(V,E) be a connected undirected graph with non-negative edge weights. If A is a subset of E contained in a MST, (S,V-S) is a cut respecting A, and e is one of the cheapest edges from S to V-S, then there is a MST containing A+e.

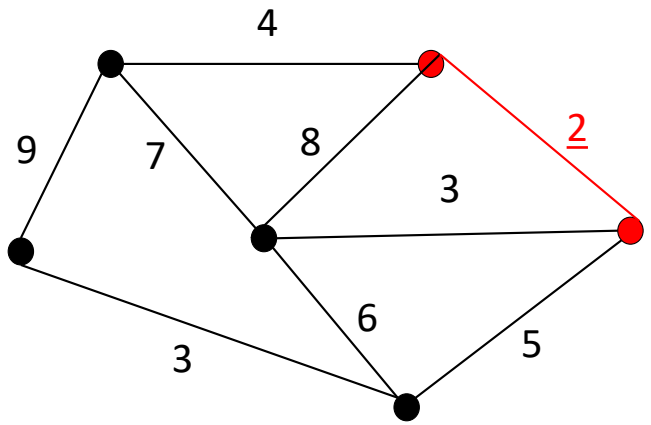
Applying the cut property

Let G=(V,E) be a connected undirected graph with non-negative edge weights. If A is a subset of E contained in a MST, (S,V-S) is a cut respecting A, and e is one of the cheapest edges from S to V-S, then there is a MST containing A+e.

Prim-Jarnik: Repeatedly add the cheapest edge leaving the set of vertices reachable from a root vertex r

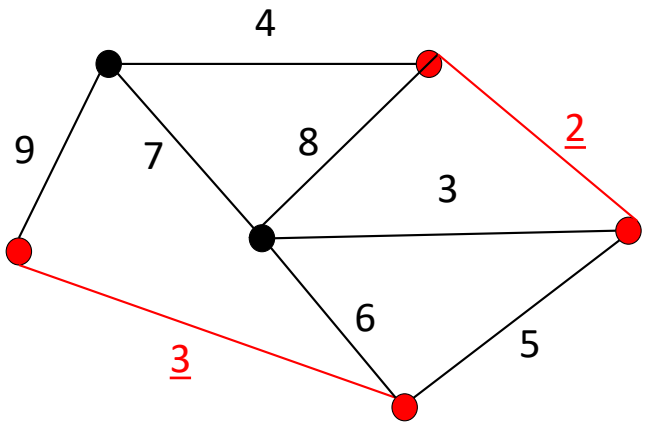
Kruskal's algorithm

- Repeatedly add the cheapest edge whose endpoints aren't already connected



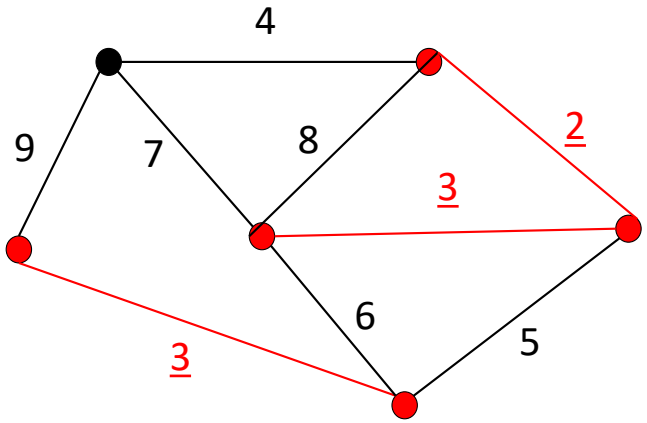
Kruskal's algorithm

- Repeatedly add the cheapest edge whose endpoints aren't already connected



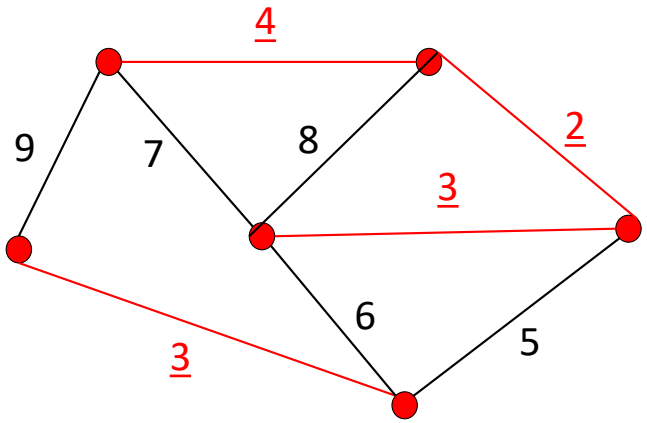
Kruskal's algorithm

- Repeatedly add the cheapest edge whose endpoints aren't already connected



Kruskal's algorithm

- Repeatedly add the cheapest edge whose endpoints aren't already connected



Kruskal's algorithm

- Repeatedly add the cheapest edge whose endpoints aren't already connected

```
MST-Kruskal(G,w)
        A = empty set
        for each vertex v in G.V
                Make-set(v)
        sort edges of G.E into nondecreasing order
        for each edge (u,v) in this order
                if Find-set(u) != Find-set(v)
                        A = A + (u,v)
                        Union(u,v)
        return A
```

Kruskal's algorithm

- Repeatedly add the cheapest edge whose endpoints aren't already connected

Cut property: Let G=(V,E) be a connected undirected graph with non-negative edge weights. If A is a subset of E contained in a MST, (S,V-S) is a cut respecting A, and e is one of the cheapest edges from S to V-S, then there is a MST containing A+e.