Week 12 – MST, Shortest Path

Algorithm Analysis

School of CSEE

MST

Exercise 2

Analyze Prim's algorithm by answering the following.

1 $Q = V[G]$;
2 for each $u \in Q$ $key[u] = \infty$; $\pi[u]$ = NIL;
3 $key[r] = 0$; $\pi[r]$ = NULL;
4 while (Q not empty)
5 u = ExtractMin(Q);
6 for each $v \in \text{Adj}[u]$
7 if ($v \in Q$ and $w(u,v) < key[v]$)
8 $\pi[v] = u$; $key[v] = w(u,v)$;

1) What is the running time for lines 1-3?
2) How many times 'while' loop at line 4 is executed?
3) What is the total running time?

Answer 2

Assuming, (binary) heap is used to implement PQ,

1) What is the running time for lines 1-3?
   O(VlogV)

   or Θ(V) (since, all element has same value)

2) How many times 'while' loop at line 4 is executed?
   |V| times

3) What is the total running time?
   O(VlogV + ElogV) = O(ElogV)

Exercise 2

```
5    u = ExtractMin(Q);
6    for each v ∈ Adj[u]
7        if (v ∈ Q and w(u,v) < key[v])
8            π[v] = u;        key[v] = w(u,v);
```

1. Line 5 ExtractMin will take O(logV) times. Thus with while loop, it will take (VlogV)
2. For loop in lines 6-7 is executed O(|E|) times. (not O(|V|*|E|) times)
   Increasing key value in line 8, will restructure heap which takes O(|logV|) times. ➔ Thus, line 6-7 will take O(ElogV)
3. Thus it will takes O(VlogV + ElogV) = O(ElogV)

Question 2

Kruskal's algorithm can return different spanning trees for the same input graph G, depending on how it breaks ties when the edges are sorted in order.

1) In what condition, Kruskal's algorithm return same minimum spanning tree T of G with?
2) What is data structure used?
3) What is the time complexity of the algorithm?

Basic idea of Kruskal's algorithm

- Sort edges into nondecreasing order by *w.*
- The algorithm maintains *A*, a forest of trees.
- Repeatedly merges two components into one by choosing the light edge that connects them.

  i.e.,

  1. Choose the light edge crossing the cut between them.

  2. (If it forms a cycle, the edge is discarded.)

1) In what condition, Kruskal's algorithm return same minimum spanning tree T of G with?

Ans) Stable sorting ?

Basic idea of Kruskal's algorithm

- Sort edges into nondecreasing order by *w.*
- The algorithm maintains *A*, a forest of trees.
- Repeatedly merges two components into one by choosing the light edge that connects them.

  i.e.,

  1. Choose the light edge crossing the cut between them.

  2. (If it forms a cycle, the edge is discarded.)

2) What is data structure used?

Ans) Usually, Disjoint set / Union-find

Basic idea of Kruskal's algorithm

- Sort edges into nondecreasing order by *w.*
- The algorithm maintains *A,* a forest of trees.
- Repeatedly merges two components into one by choosing the light edge that connects them.

  i.e.,

  1. Choose the light edge crossing the cut between them.
  2. (If it forms a cycle, the edge is discarded.)

3) What is the time complexity of the algorithm?

- Depending on data structure.
- If disjoint-set data structure is used, it is safe to say O(ElogE).

Exercise 3

What is the design strategy of Kruskal’s algorithm and Prim’s algorithm?

Ans) Greedy

MST applications

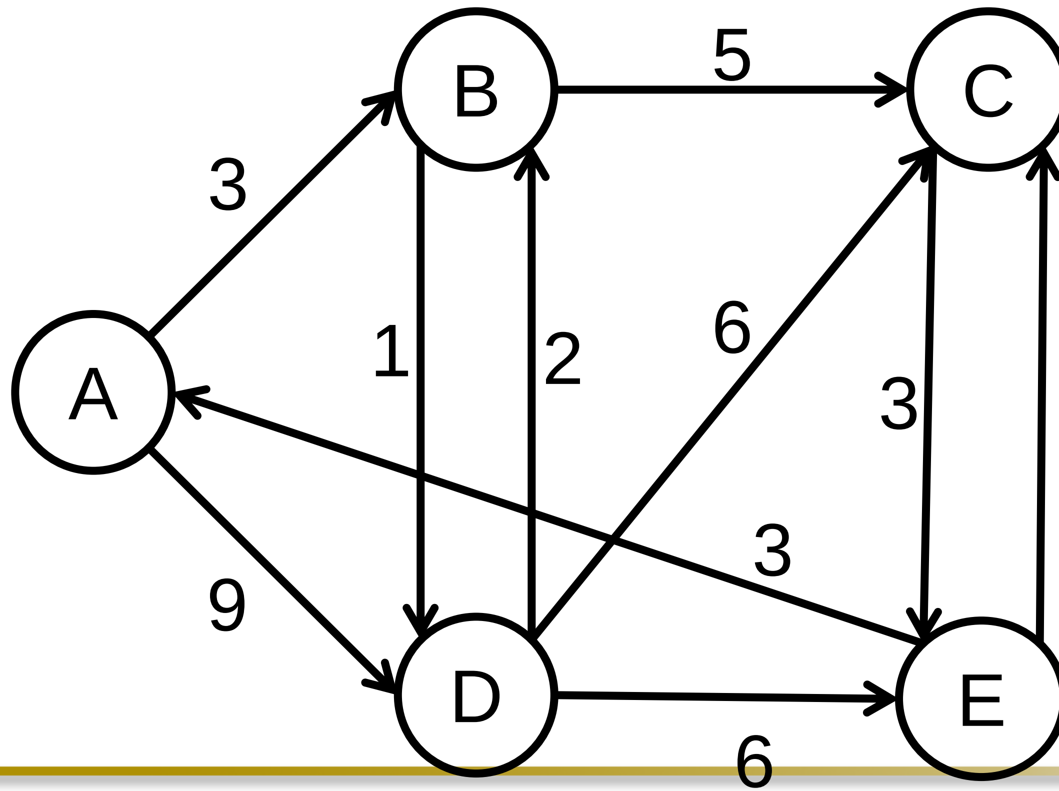
- Minimum spanning trees are used for network designs (i.e. telephone or cable networks). They are also used to find approximate solutions for complex mathematical problems like the Traveling Salesman Problem. Other, diverse applications include:
  - Cluster Analysis.
  - Real-time face tracking and verification (i.e. locating human faces in a video stream).
  - Protocols in computer science to avoid network cycles.
  - Entropy based image registration.
  - Max bottleneck paths.
  - Dithering (adding white noise to a digital recording in order to reduce distortion).

From https://www.statisticshowto.com/

Shortest Path

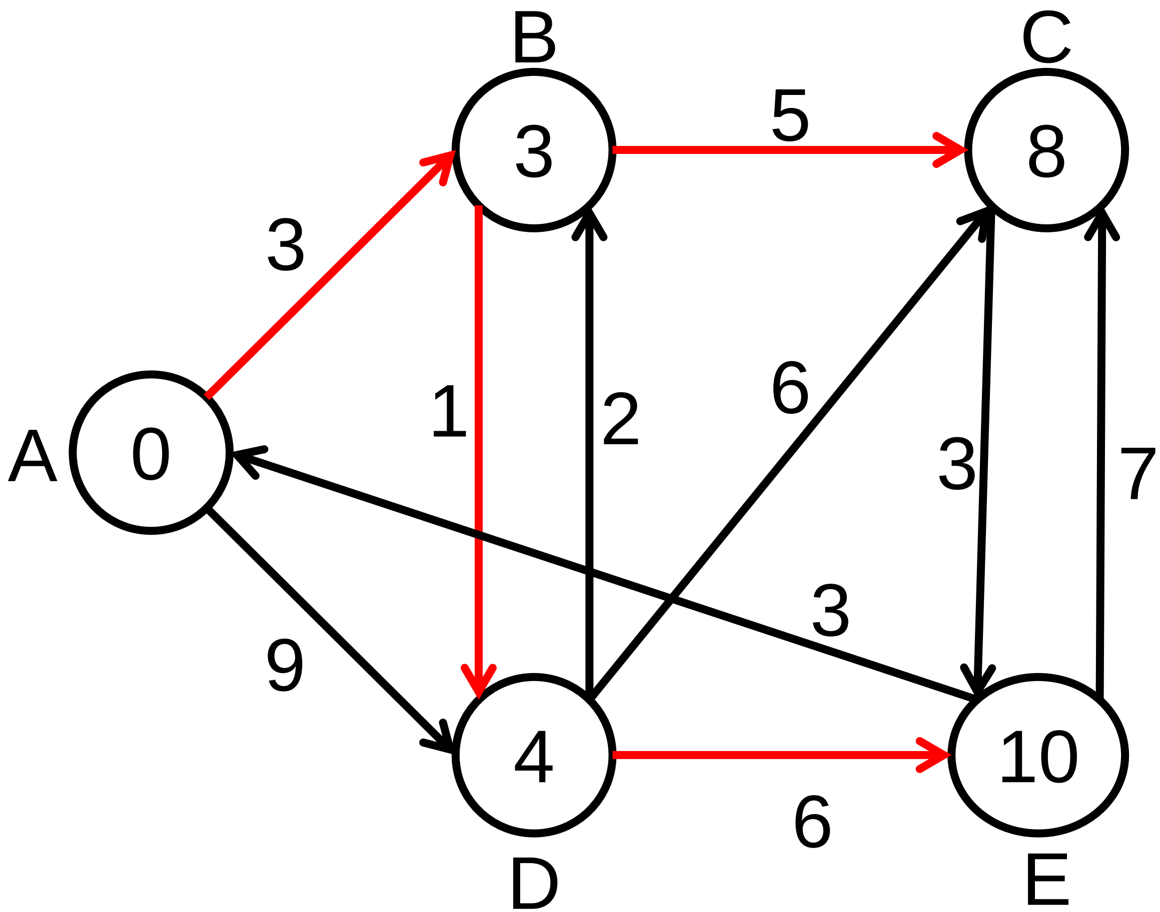
Exercise 1

Run Dijkstra's algorithm on the following directed graph, using vertex *A* as the source. After vertex D is extracted from priority queue and all edges incident from vertex D are relaxed, what is the distance of vertex B, vertex C, and vertex E?



HGU 한동대학교 HANDONG GLOBAL UNIVERSITY School of CSEE

Final result.



Answer 1

Order of relaxation: A, B, D, C, E

After vertex D is extracted from priority queue and all edges incident from vertex D are relaxed,

B: 3(A)

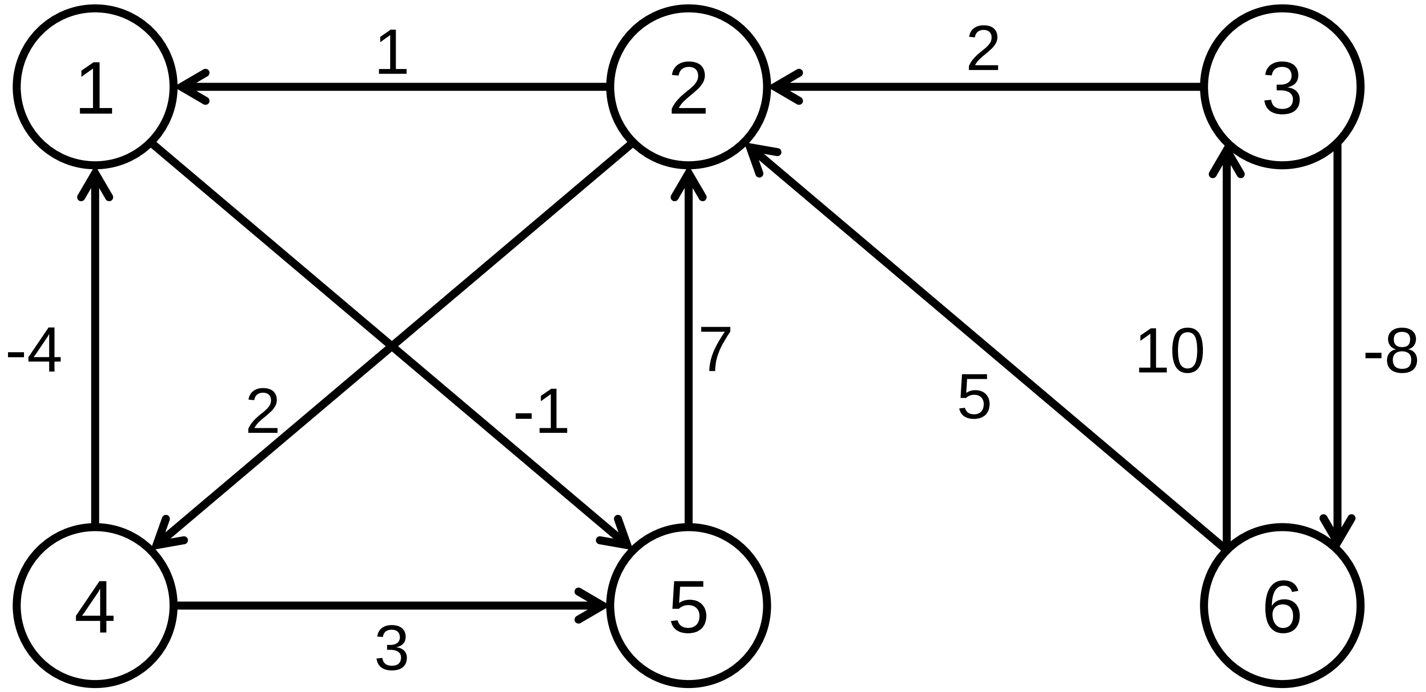
C: 8(B)

D: 9(A) → 4(B)

E: 10(D)

Exercise 2

Run Floyd Warshall algorithm on the following weighted, directed graph. Show the last resulting matrix D.



A recursive solution

- $$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{(if } k=0\text{)} \\ \min\left(d_{ij}^{(k-1)},\ d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{(if } k\geq 1\text{)} \end{cases}$$

- The Matrix $D^{(n)} = (d_{ij}^{(n)})$ gives the final answer:

  $d_{ij}^{(n)} = \delta(i,j)$ for all $i,j \in V$.

Answer 2

V2➔V5: V2➔V1➔V5

$$D^0 = \begin{bmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & \infty & \infty \\ \infty & 2 & 0 & \infty & \infty & -8 \\ -4 & \infty & \infty & 0 & 3 & \infty \\ \infty & 7 & \infty & \infty & 0 & \infty \\ \infty & 5 & 10 & \infty & \infty & 0 \end{bmatrix}$$

Answer 2

$$D^1 = \begin{bmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & 0 & \infty \\ \infty & 2 & 0 & \infty & \infty & -8 \\ -4 & \infty & \infty & 0 & -5 & \infty \\ \infty & 7 & \infty & \infty & 0 & \infty \\ \infty & 5 & 10 & \infty & \infty & 0 \end{bmatrix}$$

Answer 2

$$D^2 = \begin{bmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & 0 & \infty \\ \color{red}{3} & 2 & 0 & \color{red}{4} & \color{red}{2} & -8 \\ -4 & \infty & \infty & 0 & -5 & \infty \\ \color{red}{8} & 7 & \infty & \color{red}{9} & 0 & \infty \\ \color{red}{6} & 5 & 10 & \color{red}{7} & \color{red}{5} & 0 \end{bmatrix}$$

Answer 2

$$D^6 = \begin{bmatrix} 0 & 6 & \infty & 8 & -1 & \infty \\ -2 & 0 & \infty & 2 & -3 & \infty \\ -5 & -3 & 0 & -1 & -6 & -8 \\ -4 & 2 & \infty & 0 & -5 & \infty \\ 5 & 7 & \infty & 9 & 0 & \infty \\ 3 & 5 & 10 & 7 & 2 & 0 \end{bmatrix}$$

Exercise 3

a) What is the design strategy of Dijkstra’s algorithm?

b) What is the design strategy of Floyd’s algorithm?

Exercise 4

Compare the time complexity of the following all-pair shortest paths.

(1) Dijkstra's algorithm for nonnegative edge weights

(2) Bellman-Ford for each vertex

(3) Floyd-Warshall algorithm

Answer 4

Compare the time complexity of the following all-pair shortest paths.

(1) Dijkstra's algorithm for nonnegative edge weights
$\Theta(VE+ V^2 \lg V) = \Theta(V^3)$

(2) Bellman-Ford for each vertex
$\Theta(V^2E) = \Theta(V^4)$

(3) Floyd-Warshall algorithm
$\Theta(V^3)$

Then, what is advantage of Floyd alg. Over Dijkstra's alg.?

Ans) It can deal with negative edge.