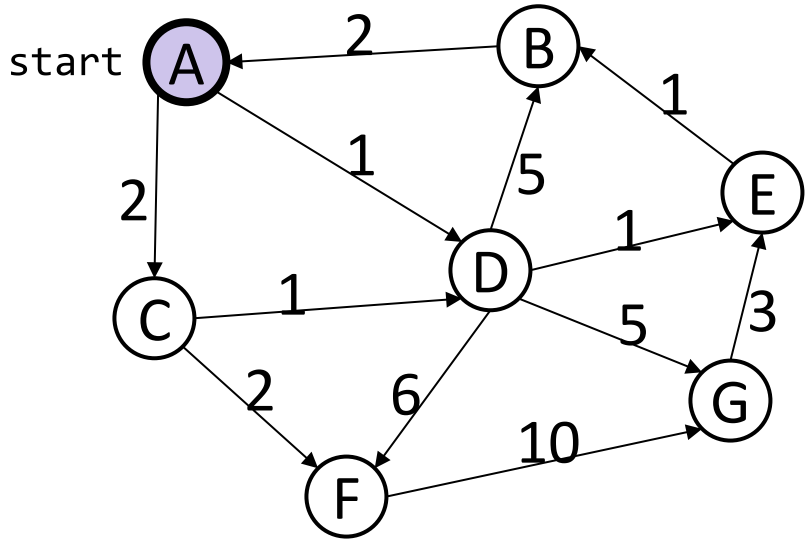
Lecture 13
Shortest Paths
Exercises ANS

Department of Computer Science
Hofstra University

Acknowledgement: Lecture slides based on UofW Course on Data Structures

Q. Dijkstra's Algorithm

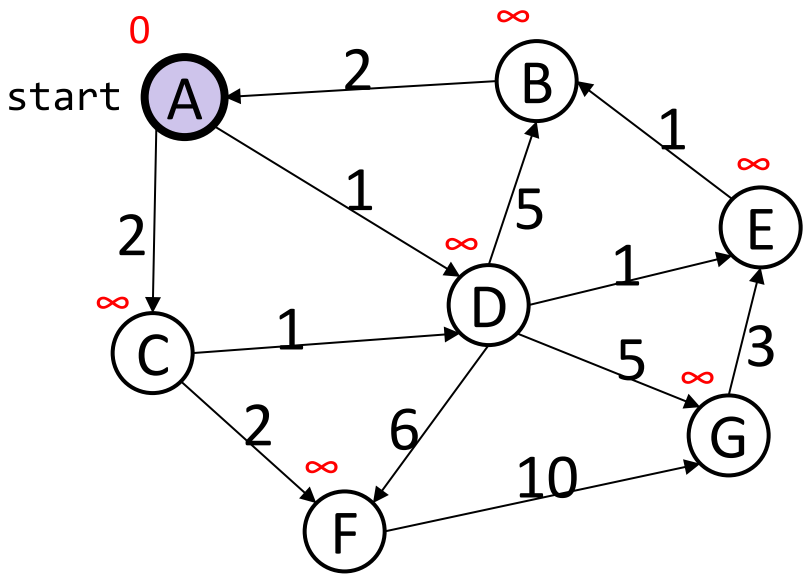
Given this directed graph, run Dijkstra's Algo to find shortest paths starting from source node A. Give the node visit order, and fill in this table of SN (Shortest Distance) and PN (Previous Node), crossing out old SD and PN as you find a shortcut path with smaller SD



Visit Order

Node	SD	PN
A		
B		
C		
D		
E		
F		
G		

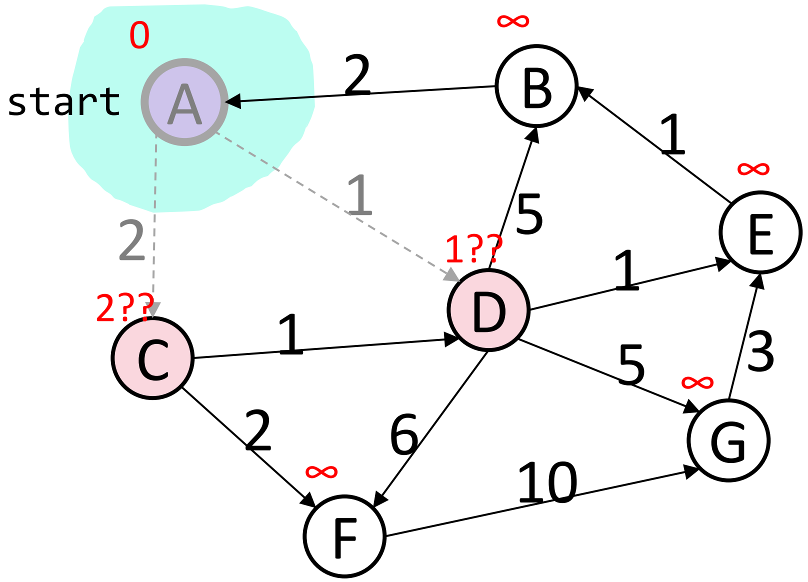
Q. Dijkstra's Algorithm



Node	SD	PN
A	∞	
B	∞	
C	∞	
D	∞	
E	∞	
F	∞	
G	∞	

<u>Visit Order</u>

Q. Dijkstra's Algorithm

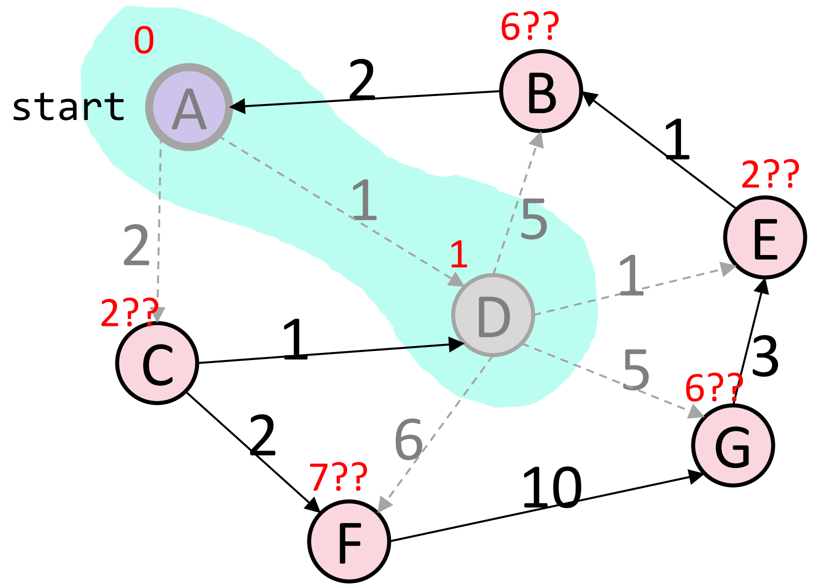


Node	SD	PN
A	0	/
B	∞	
C	2	A
D	1	A
E	∞	
F	∞	
G	∞	

<u>Visit Order</u>

A

Q. Dijkstra's Algorithm



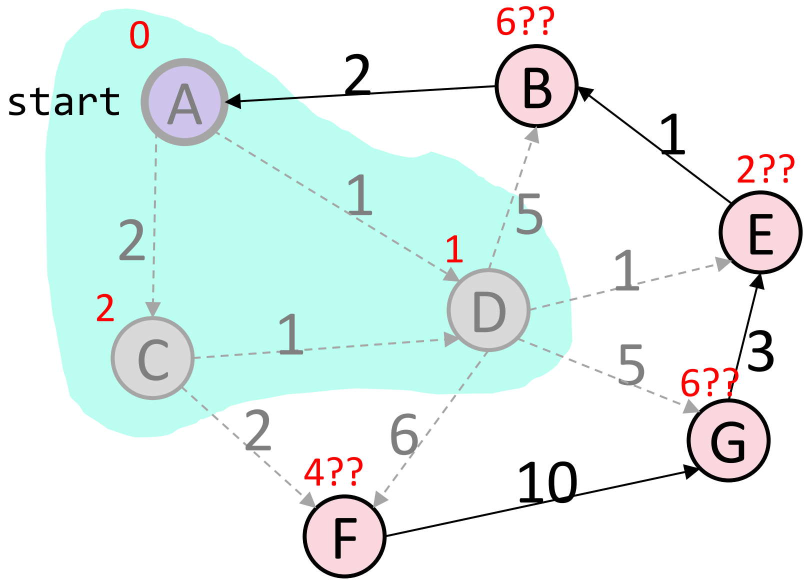
Node	SD	PN
A	0	/
B	**6**	**D**
C	2	A
D	1	A
E	**2**	**D**
F	**7**	**D**
G	**6**	**D**

<u>Visit Order</u>

A, D

We can choose to visit either C or E next, since they have equal smallest SD of 2 among all unvisited nodes. Let's visit C in alphabetical order

Q. Dijkstra's Algorithm

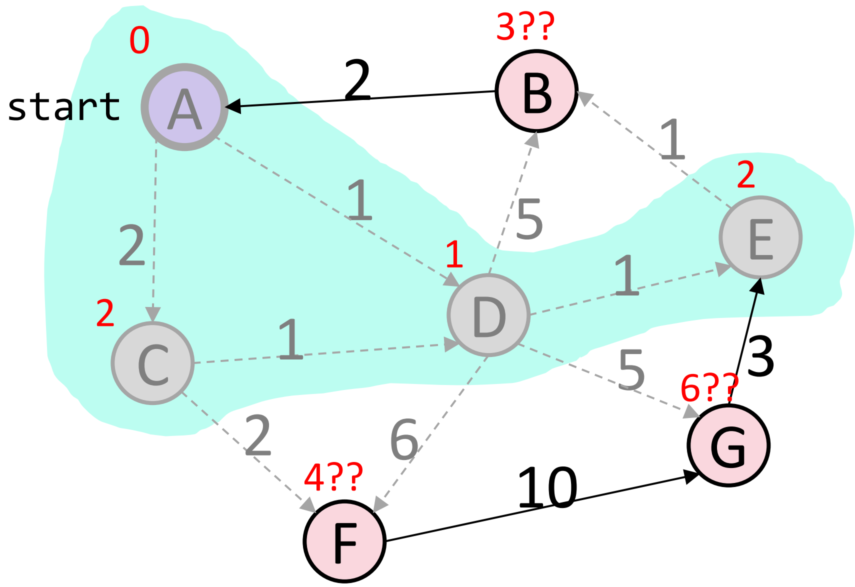


Node	SD	PN
A	0	/
B	6	D
C	2	A
D	1	A
E	2	D
F	~~7~~ **4**	~~D~~ **C**
G	6	D

Visit Order

A, D, C

Q. Dijkstra's Algorithm

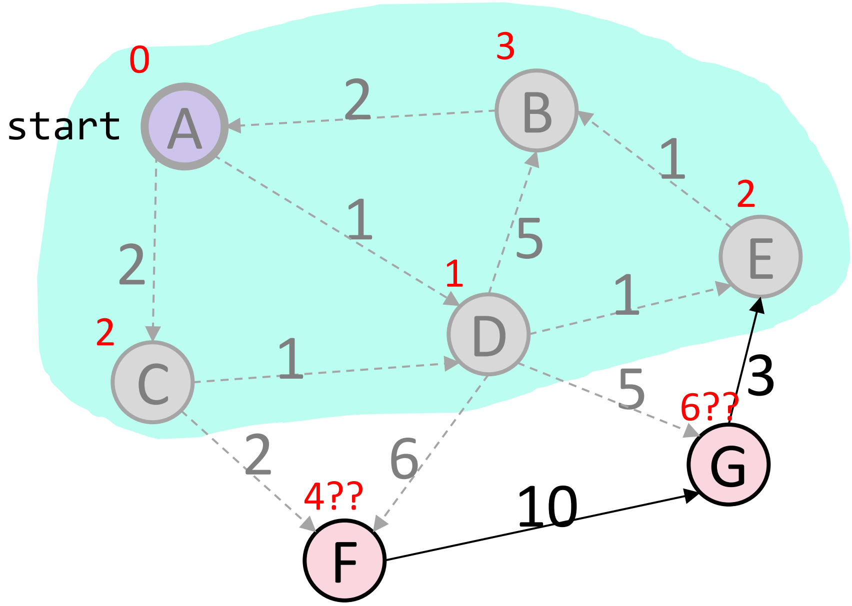


Node	SD	PN
A	0	/
B	~~6~~ **3**	~~D~~ **E**
C	2	A
D	1	A
E	2	D
F	~~7~~ 4	~~D~~ C
G	6	D

<u>Visit Order</u>

A, D, C, E

Q. Dijkstra's Algorithm

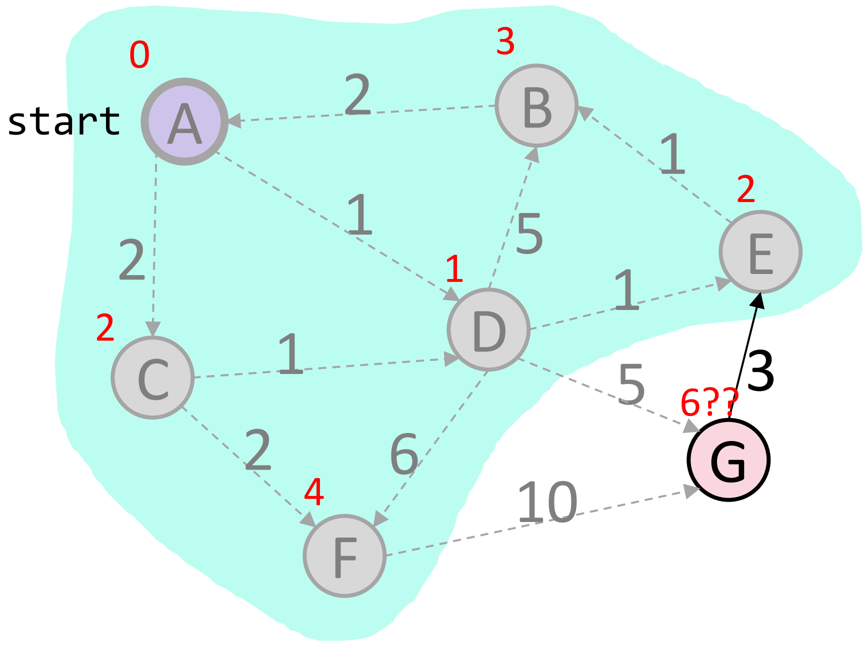


Node	SD	PN
A	0	/
B	~~6~~ 3	~~D~~ E
C	2	A
D	1	A
E	2	D
F	~~7~~ 4	~~D~~ C
G	6	D

<u>Visit Order</u>

A, D, C, E, B

Q. Dijkstra's Algorithm

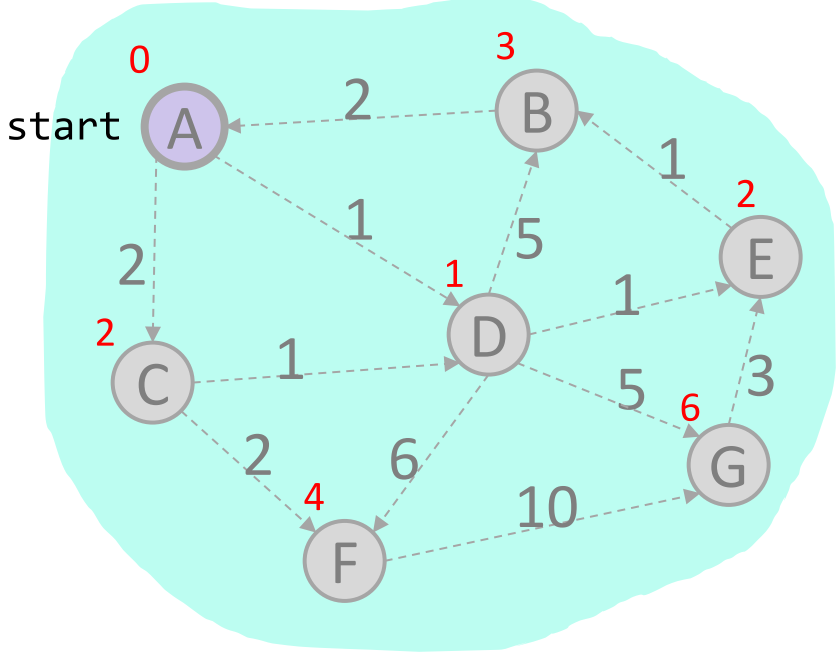


Node	SD	PN
A	0	/
B	~~6~~ 3	~~D~~ E
C	2	A
D	1	A
E	2	D
F	~~7~~ 4	~~D~~ C
G	6	D

<u>Visit Order</u>

A, D, C, E, B, F

Q. Dijkstra's Algorithm ANS

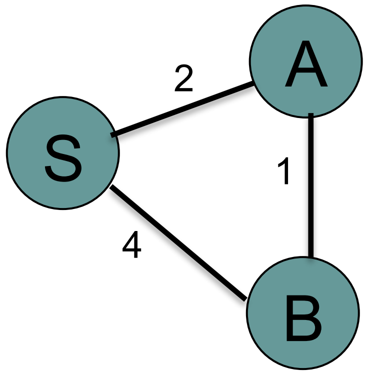


Node	SD	PN
A	0	/
B	~~6~~ 3	~~D~~ E
C	2	A
D	1	A
E	2	D
F	~~7~~ 4	~~D~~ C
G	6	D

Visit Order

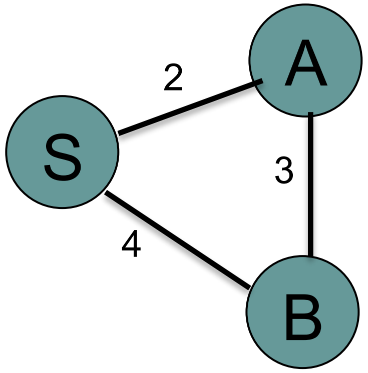
A, D, C, E, B, F, G

Q. Dijkstra's Algorithm (Source Node S)



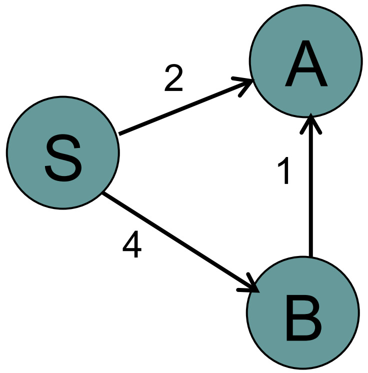
ANS

Node	SD	PN
S	0	/
A		
B		



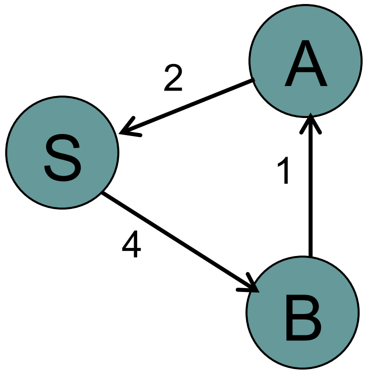
ANS

Node	SD	PN
S	0	/
A		
B		



ANS

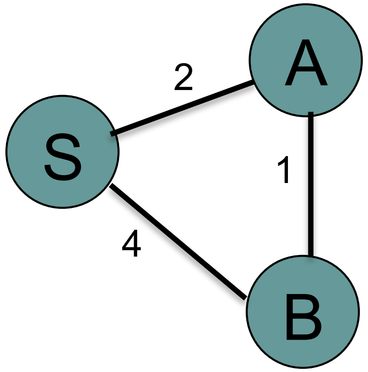
Node	SD	PN
S	0	/
A		
B		



ANS

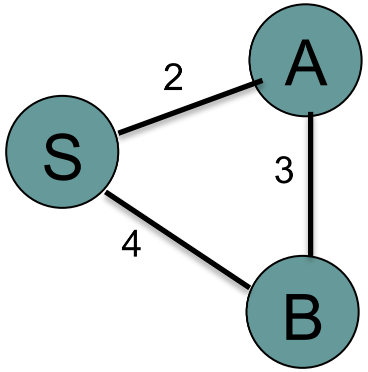
Node	SD	PN
S	0	/
A		
B		

Q. Dijkstra's Algorithm (Source Node S) ANS



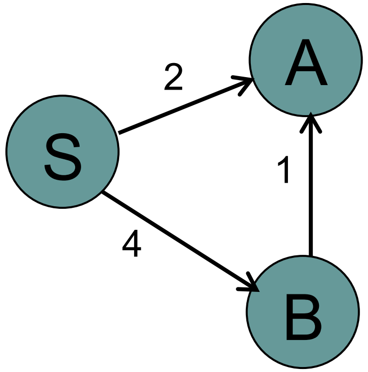
ANS

Node	SD	PN
S	0	/
A	2	S
B	4 3	~~S~~ A



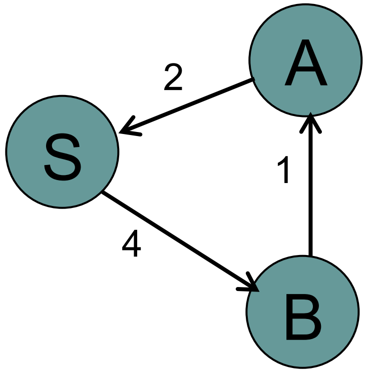
ANS

Node	SD	PN
S	0	/
A	2	S
B	4	S



ANS

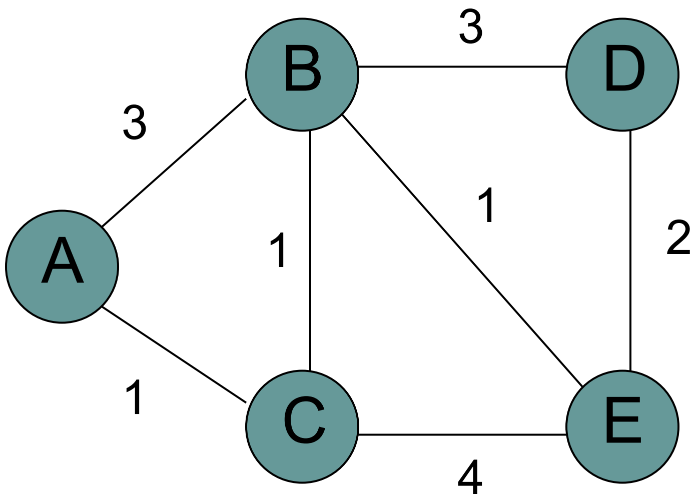
Node	SD	PN
S	0	/
A	2	S
B	4	S



ANS

Node	SD	PN
S	0	/
A	5	B
B	4	S

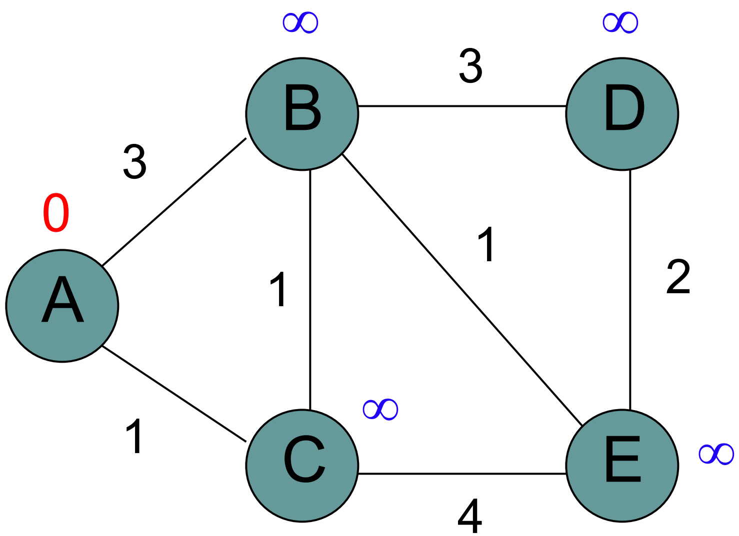
Q. Dijkstra's Algorithm (Source Node A, Undirected Graph)

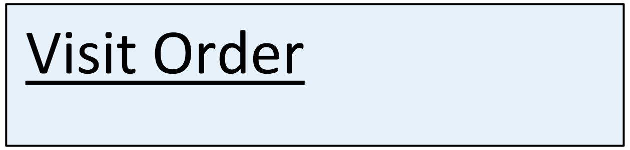


Node	SD	PN
A		
B		
C		
D		
E		

<u>Visit Order</u>

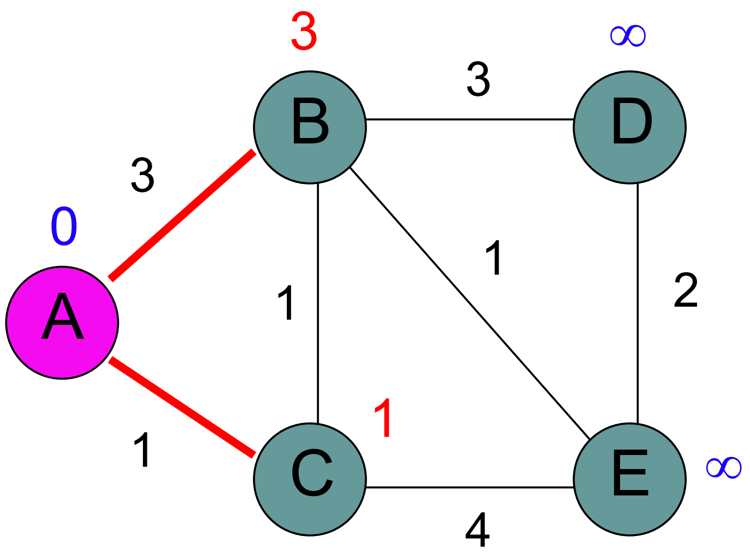
Initialize





Node	SD	PN
A	0	/
B	∞	
C	∞	
D	∞	
E	∞	

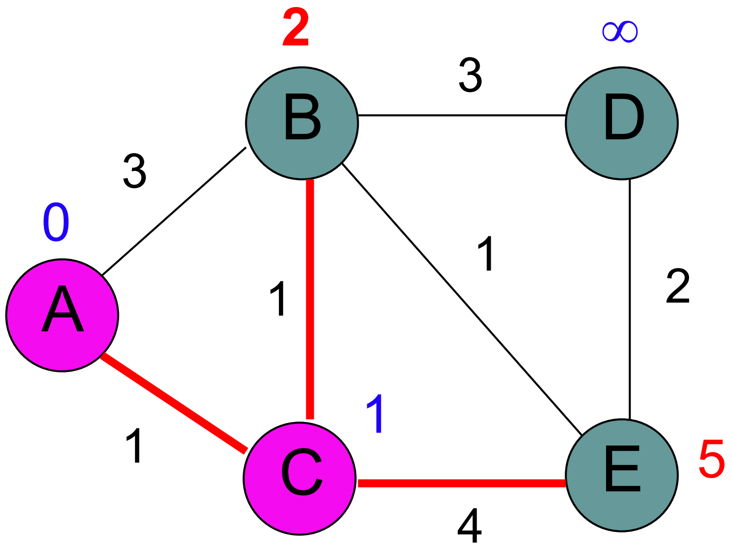
Visit Node A



Visit Order
A

Node	SD	PN
A	0	/
B	3	A
C	1	A
D	∞	
E	∞	

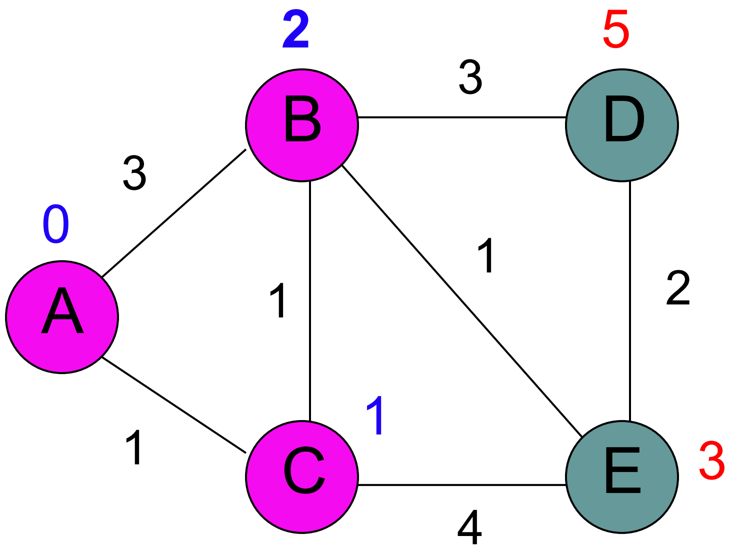
Visit Node C



Visit Order
A, C

Node	SD	PN
A	0	/
B	~~3~~ 2	~~A~~ C
C	1	A
D	∞	
E	5	C

Visit Node B

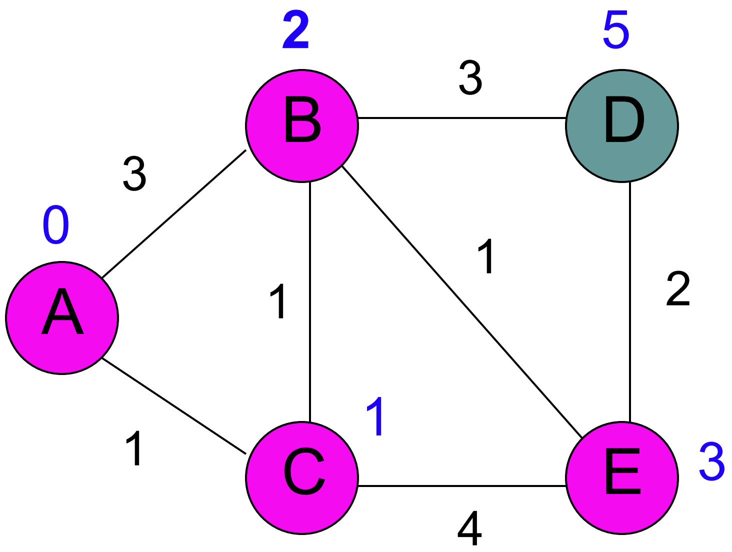


Node	SD	PN
A	0	/
B	~~3~~ 2	~~A~~ C
C	1	A
D	5	B
E	~~5~~ 3	~~C~~ B

Visit Order

A, C, B

Visit Node E



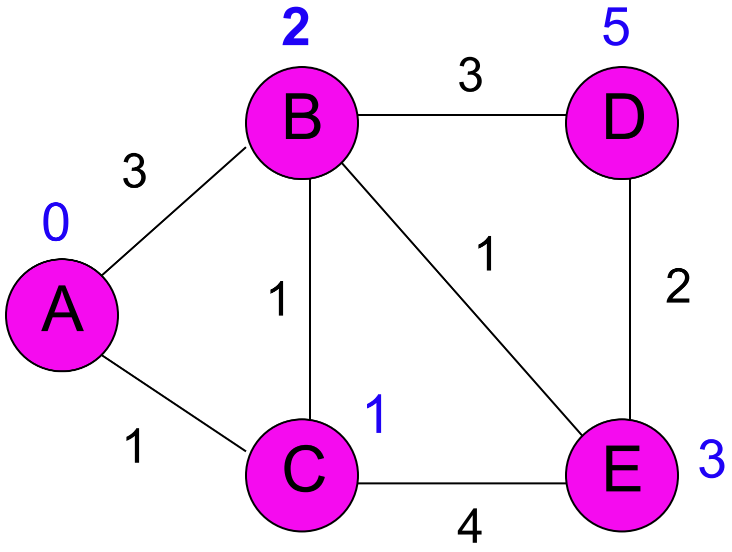
Visit Order

A, C, B, E

Node	SD	PN
A	0	/
B	~~3~~ 2	~~A~~ C
C	1	A
D	5	B
E	~~5~~ 3	~~C~~ B

Nothing changes

Visit Node D



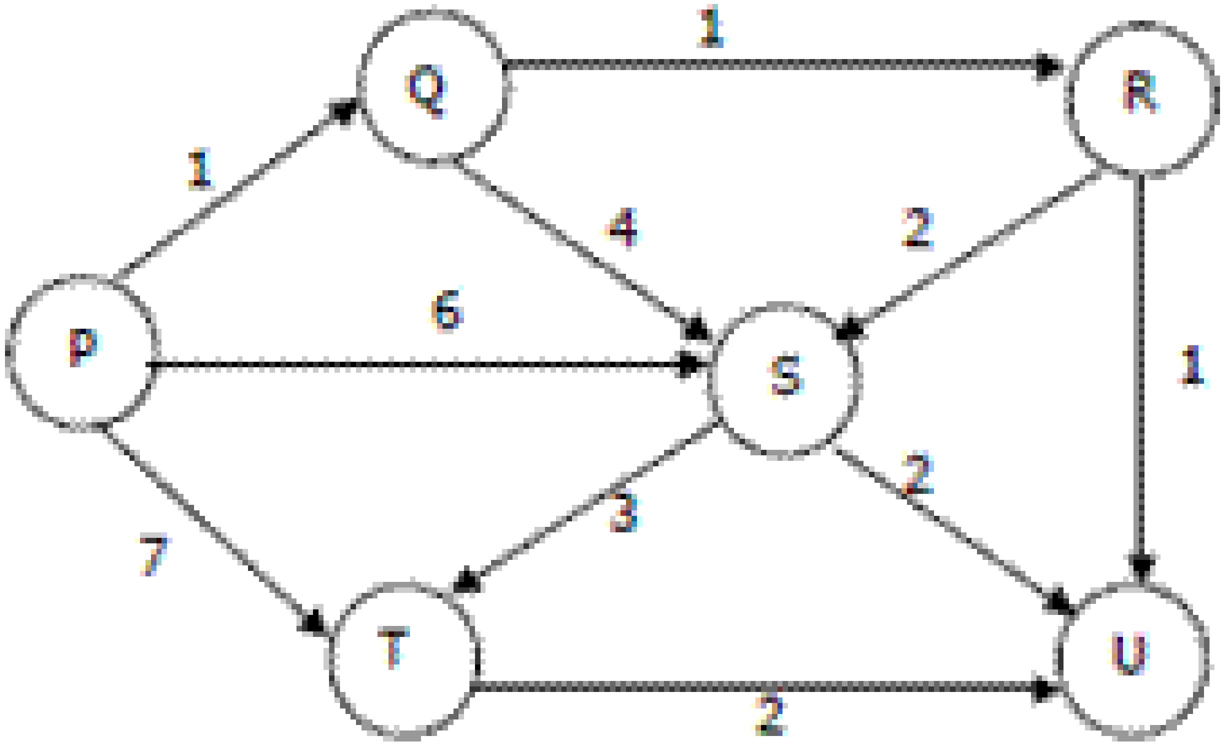
Visit Order

A, C, B, E, D

Node	SD	PN
A	0	/
B	~~3~~ 2	~~A~~ C
C	1	A
D	5	B
E	~~5~~ 3	~~C~~ B

Nothing changes

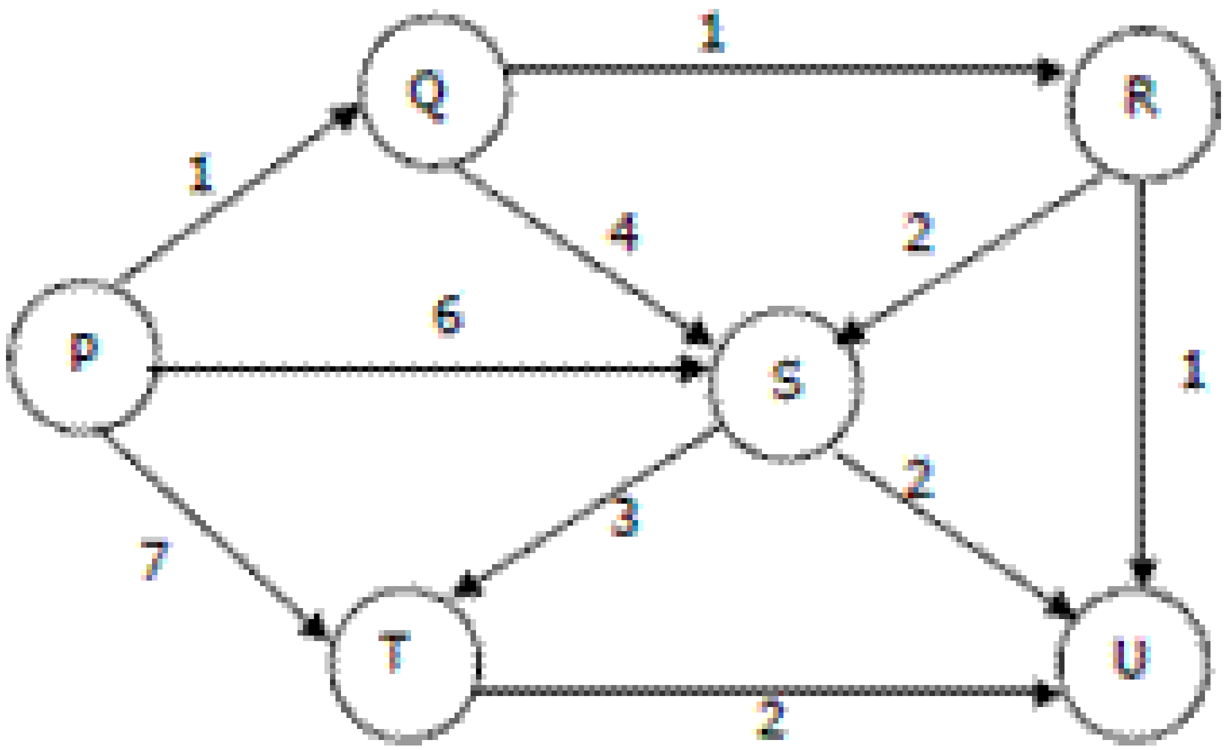
Q. Dijkstra's Algorithm (Source Node P, Directed Graph)



Visit order:

Node	SD	PN
P	0	
Q		
R		
S		
T		
U		

Q. Dijkstra's Algorithm (Source Node P, Directed Graph) ANS

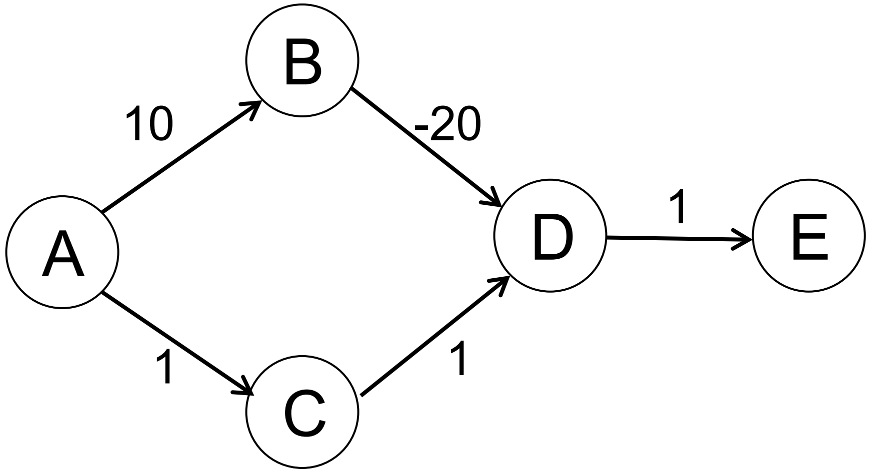


Visit order: P, Q, R, U, S, T

Node	SD	PN
P	0	
Q	1	P
R	2	Q
S	~~6~~ ~~5~~ 4	~~P~~ ~~Q~~ R
T	7	P
U	3	R

Q. Topological Sort

Given this directed graph, run Topological Sort to find shortest paths starting from source node A. Give the node visit order, and fill in this table of SN (Shortest Distance) and PN (Previous Node), crossing out old SD and PN as you find a shortcut path with smaller SD. Considering all possible topological orders.

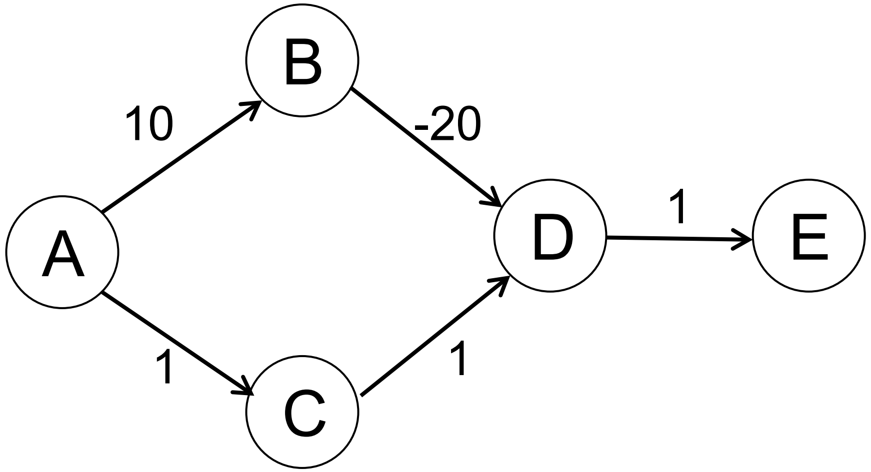


Visit Order

Node	SD	PN
A	0	/
B		
C		
D		
E		

Q. Topological Sort ANS

We consider two possible topological orders A, B, C, D, E, and A, C, B, D, E



Visit Order
A, B, C, D, E

Node	SD	PN
A	0	
B	10	A
C	1	A
D	-10	B
E	-9	D

Visit Order
A, C, B, D, E

Node	SD	PN
A	0	
B	10	A
C	1	A
D	~~2~~ -10	~~C~~ B
E	-9	D

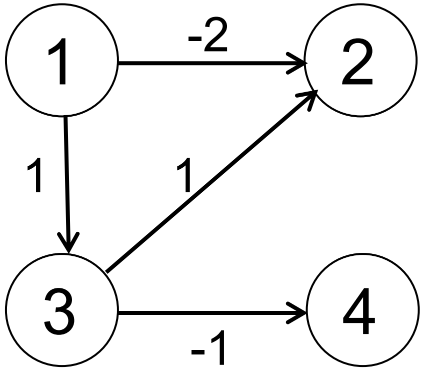
Q. Johnson's algorithm

Consider the following weighted digraph. As part of Johnson's algorithm for All-pairs Shortest Paths, add a dummy source node d, and edges with weight 0 from d to all vertices of G. Let the modified graph be G'.

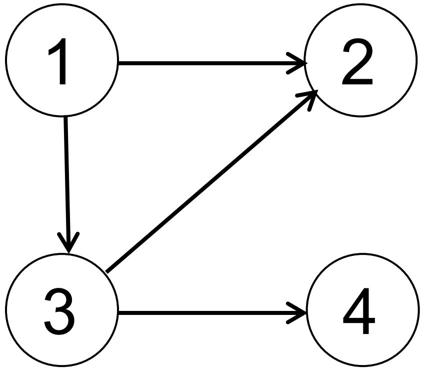
a) Compute the shortest distances from dummy source node d to each node in G' by hand: h[0], h[1], .. h[V-1], then reweight the edges of the original graph to make the edge weights greater than or equal to 0. Draw the reweighted graph G' (without the dummy node d).

b) For the reweighted graph G': run Dijkstra's Algo to find shortest paths starting from source node 1, and compute the shortest paths for the graph with updated positive or zero weights. (Do not show the intermediate steps.)

c) For the original graph G: compute the shortest paths starting from source node 1 with negative weights.



Original graph



Reweighted graph

Node	h()
1	
2	
3	
4	

Shortest paths starting from dummy node

Node	SD	PN
1	0	/
2		
3		
4		

Shortest paths starting from source node 1 in reweighted graph

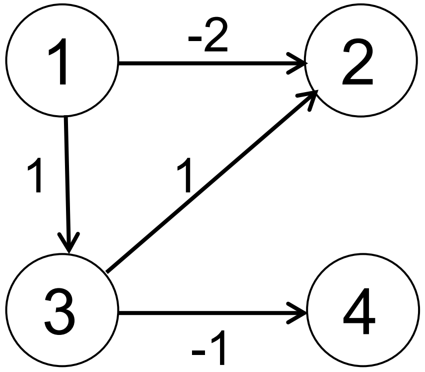
Node	SD	PN
1	0	/
2		
3		
4		

Shortest paths starting from source node 1 in original graph

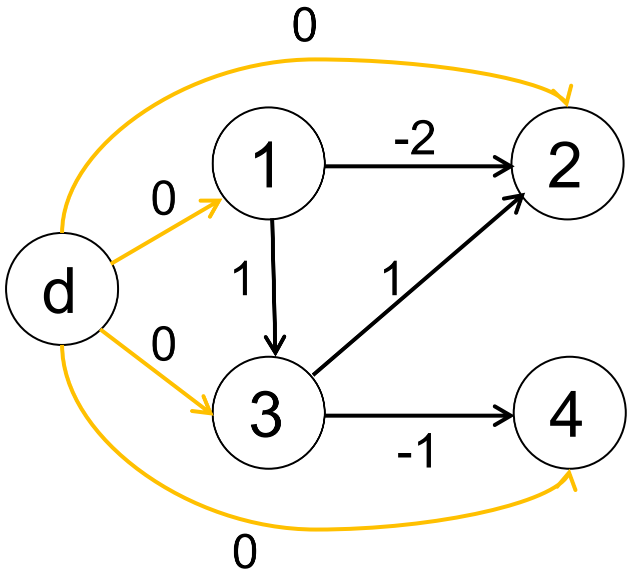
Q. Johnson’s algorithm ANS (a)(b)

Shortest distances from dummy source node d: h[1]=0, h[2]=-2, h[3]=0, h[4]=-1. (Theoretically you should run Bellman-Ford algorithm starting from node d, but the graph is simple enough that you can obtain the h[] values by observation.)

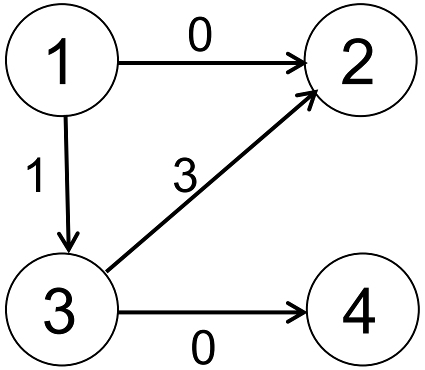
Using $w'(u, v) = w(u, v) + h[u] - h[v]$, we have: $w'[1][2]=-2+0-(-2)=0$, $w'[1][3]=-1+0-(-1)=0$, $w'[3][2]=1+0-(-2)=3$, $w'[3][4]=-1+0-(-1)=0$



Original graph



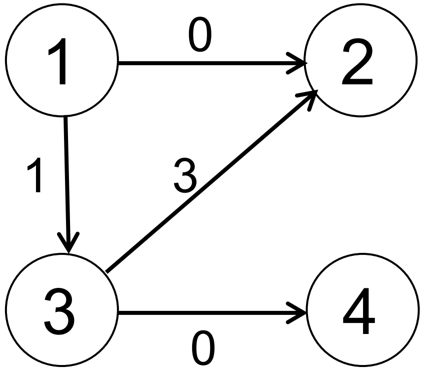
Original graph
w/ dummy node



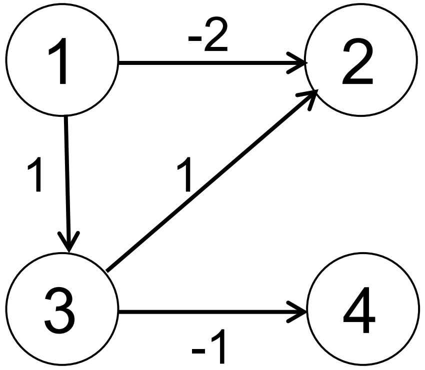
Reweighted graph

Q. Johnson's algorithm ANS (c)

- Let's run Dijkstra's algorithm starting from source node 0, and obtain the shortest paths table for the reweighted graph
- We then subtract h[s] – h[t] from length of each shortest path from s to t to obtain the shortest paths table for the original graph (PN stays the same)
  - SD(2)=0–(0–(–2))=–2
  - SD(3)=1–(0–0)=1
  - SD(4)=1 – (0–(–1))=0



Reweighted graph



Original graph

Node	h()
1	0
2	-2
3	0
4	-1

Shortest paths starting from dummy node

Node	SD'	PN
1	0	/
2	0	1
3	1	1
4	1	3

Shortest paths starting from source node 1 in reweighted graph

Node	SD	PN
1	0	/
2	-2	1
3	1	1
4	0	3

Shortest paths starting from source node 1 in original graph