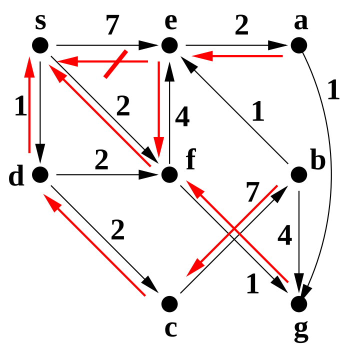
University of Nevada, Las Vegas Computer Science 477/677

Dijkstra's Algorithm Practice Problems and Solutions

1. Use Dijkstra's algorithm to solve the single source shortest path problem for the following weighted directed graph, where **s** is the source. Show the steps.

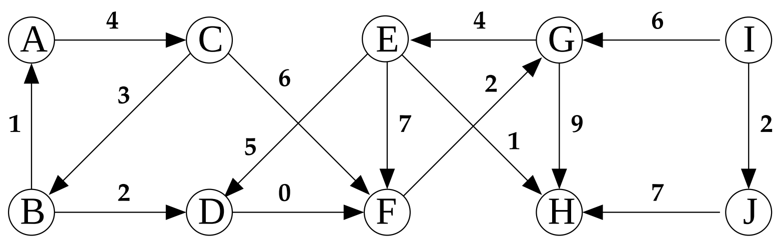


s	a	b	c	d	e	f	g
0	8	10	3	1	~~7~~ 6	2	3
*	e	c	d	s	~~s~~ f	s	f

Let Q be the minqueue consisting of partially processed vertices. We write Q as a list of vertices, sorted by V. Initially, $Q = (\mathbf{s})$.

In the first step, we dequeue **s** and enqueue **d,e,f**. $Q =$ (d,f,e)
In the second step we dequeue **d** and enqueue **c**. $Q =$ (f,c,e)
In the third step, we dequeue **f**, enqueue **g**, and change **back(e)** to **f**, decreasing **V(e)** from 7 to 6. $Q =$ (**c,e,g**)
In the fourth step, we dequeue **c** and enqueue **b**. Now, $Q =$ (**b,e,g**).
In the fifth step, we dequeue **b**. $Q =$ (**e,g**), and change **back(g)** to **b**, decreasing **V(g)** from 9 to 8. $Q =$ (**e,g**).
In the sixth step, we dequeue **e** and enqueue **a**. $Q =$ (**a,g**).
In the seventh step, we dequeue **a**. $Q =$ (**g**).
In the eighth and last step, we dequeue **g**. Q is now empty and we are done.

2. Let G be the directed graph given below. Use Dijkstra's algorithm to solve the single source shortest path problem on G with start vertex A. Show your work.



I will try to use the notation used in the code on page 110 of our textbook, Thus, the backpointer is called prev. The second array is the minqueue, Q, where minimum dist node is on the left. Initially, A is the only member of the queue, and its backpointer prev[A] is undefined.

	A	B	C	D	E	F	G	H	I	J
dist	0	∞	∞	∞	∞	∞	∞	∞	∞	∞
prev	⋆									

A
0

At each step, we execute u = deletemin(Q). Thus u = A. We insert C into Q, since it is the only outneighbor of A.

	A	B	C	D	E	F	G	H	I	J
dist	0	∞	4	∞	∞	∞	∞	∞	∞	∞
prev	⋆		A							

C
4

Now we let u = deletemin(Q) = C, and we insert B and F into Q.

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	∞	∞	10	∞	∞	∞	∞
prev	⋆	C	A			C				

B	F
7	10

Now we let u = deletemin(Q) = B, and we insert D into Q. Note that D is ahead of F in the priority.

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	9	∞	10	∞	∞	∞	∞
prev	⋆	C	A	B		C				

D	F
9	10

Now we let u = deletemin(Q) = D, and we update dist(F) and prev(F).

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	9	∞	9	∞	∞	∞	∞
prev	⋆	C	A	B		D				

F
9

Now we let u = deletemin(Q) = F, and we insert G into Q. Now we let u = deletemin(Q) = F, and we insert G into Q.

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	9	∞	9	11	∞	∞	∞
prev	$\star$	C	A	B		D	F			

G
11

Now we let u = deletemin(Q) = G, and we insert E and H into Q.

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	9	15	9	11	20	∞	∞
prev	$\star$	C	A	B	G	D	F	G		

E	H
15	20

Now we let u = deletemin(E) = G, and we update dist[H] and prev[H]

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	9	15	9	11	16	∞	∞
prev	$\star$	C	A	B	G	D	F	E		

H
16

Now we let u = deletemin(E) = H. Since H has no outneighbors, we do not insert anything into Q. Since Q is empty, we are done. I and J are unreachable from A.

	A	B	C	D	E	F	G	H	I	J
dist	0	7	4	9	15	9	11	16	∞	∞
prev	$\star$	C	A	B	G	D	F	E		