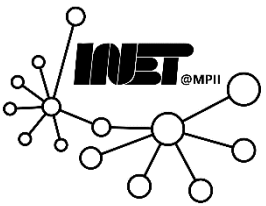
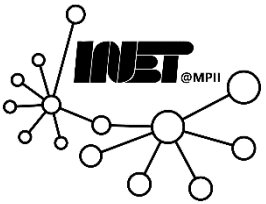
Homework 7

Routing



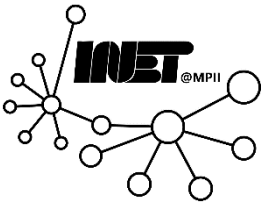
Get the Slides here

tinyurl.com/y7j2d6ap



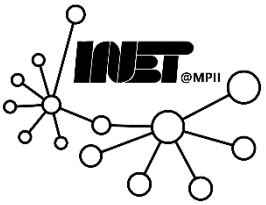
Homework Overview

- Link State Routing
- Distance Vector Routing with RIP
- Inter-AS Routing (BGP)



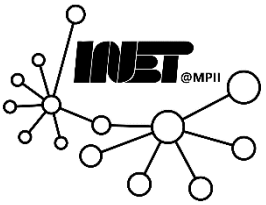
Questions

In this exercise, we will always use routers as start and destination points of any communication. Think of the network as an undirected graph. Routers are modeled as nodes in the graph, links between routers are edges with weights equal to the respective path costs.



Questions

In this exercise, we will always use routers as start and destination points of any communication. Think of the network as an undirected graph. Routers are modeled as nodes in the graph, links between routers are edges with weights equal to the respective path costs.



Question 1 (Link State Routing)

Given is a network with the nodes N-T along with the following undirected weighted links:

(N,O,4), (N,P,3), (N,Q,7), (O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3),(S,T,4)

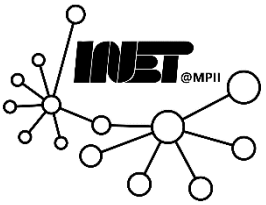
Question 1 (Link State Routing)

Given is a network with the nodes N-T along with the following undirected weighted links:

(N,O,4), (N,P,3), (N,Q,7), (O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3),(S,T,4)

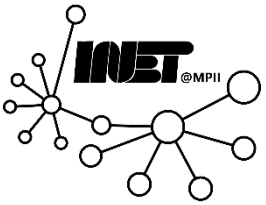
Question 1 (a)

Draw the planar graph in such a way that edges do not overlap.



Question 1 (a)

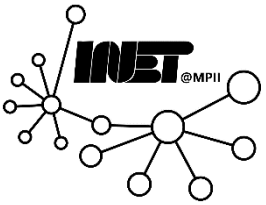
Draw the planar graph in such a way that edges do not overlap.



Question 1 (a)

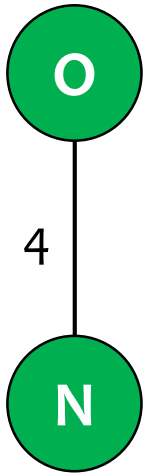
Draw the planar graph in such a way that edges do not overlap.

(N,O,4), (N,P,3), (N,Q,7), (O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



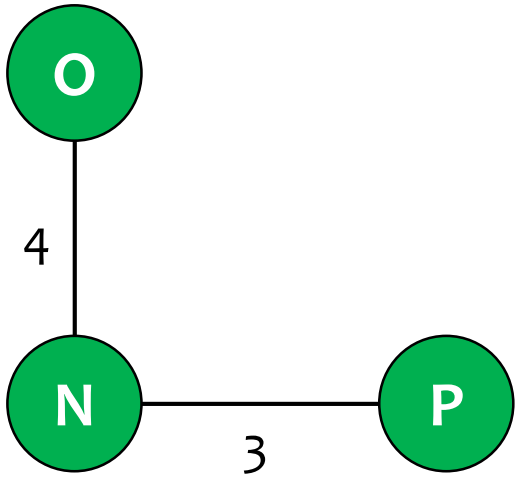
Question 1 (a)

(N,O,4), (N,P,3), (N,Q,7), (O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



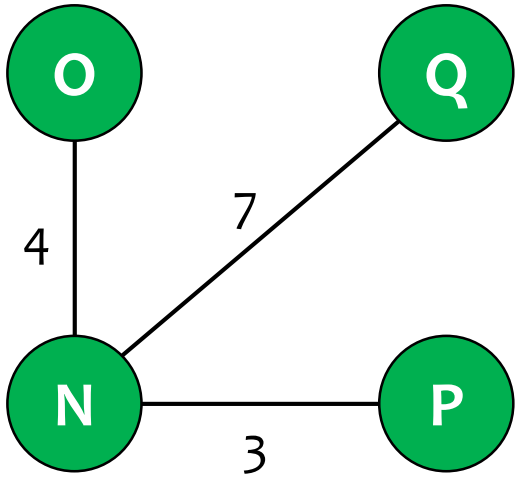
Question 1 (a)

(N,P,3), (N,Q,7), (O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



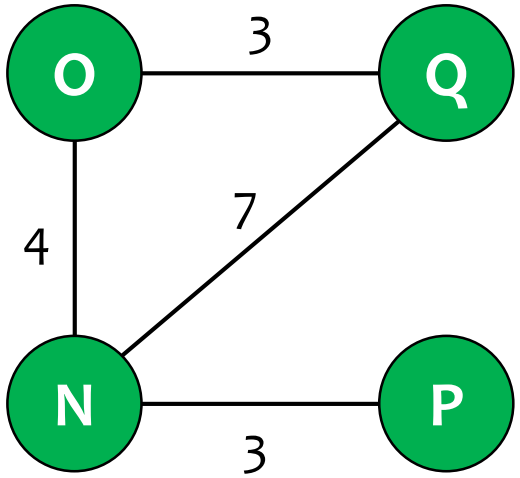
Question 1 (a)

(N,Q,7), (O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



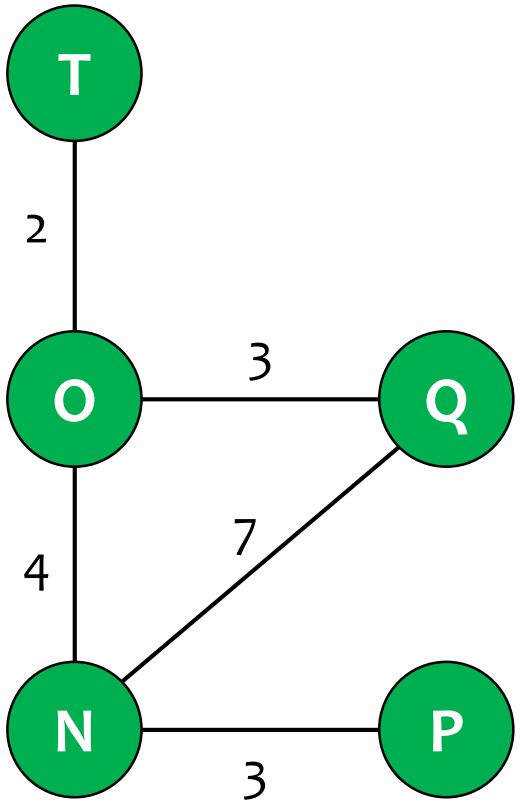
Question 1 (a)

(O,Q,3), (O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



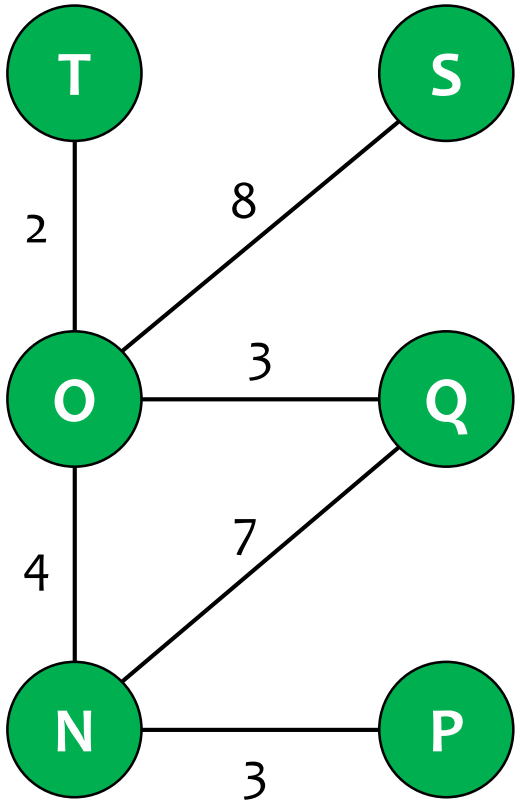
Question 1 (a)

(O,T,2), (O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



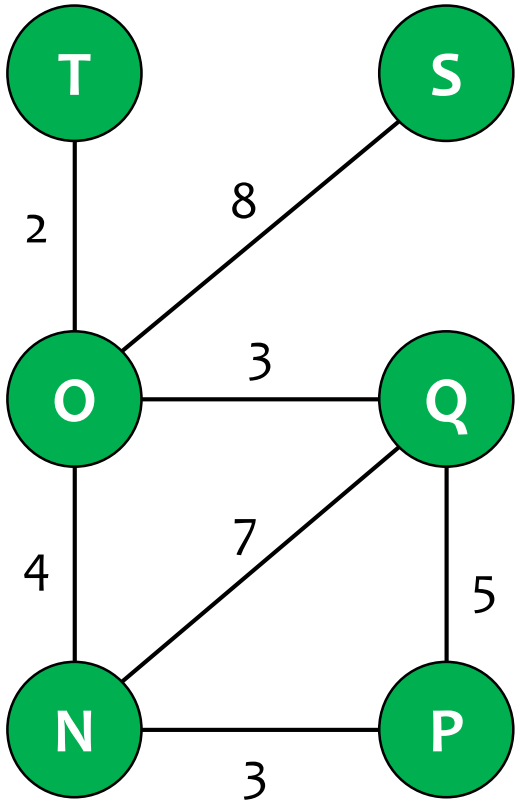
Question 1 (a)

(O,S,8), (P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



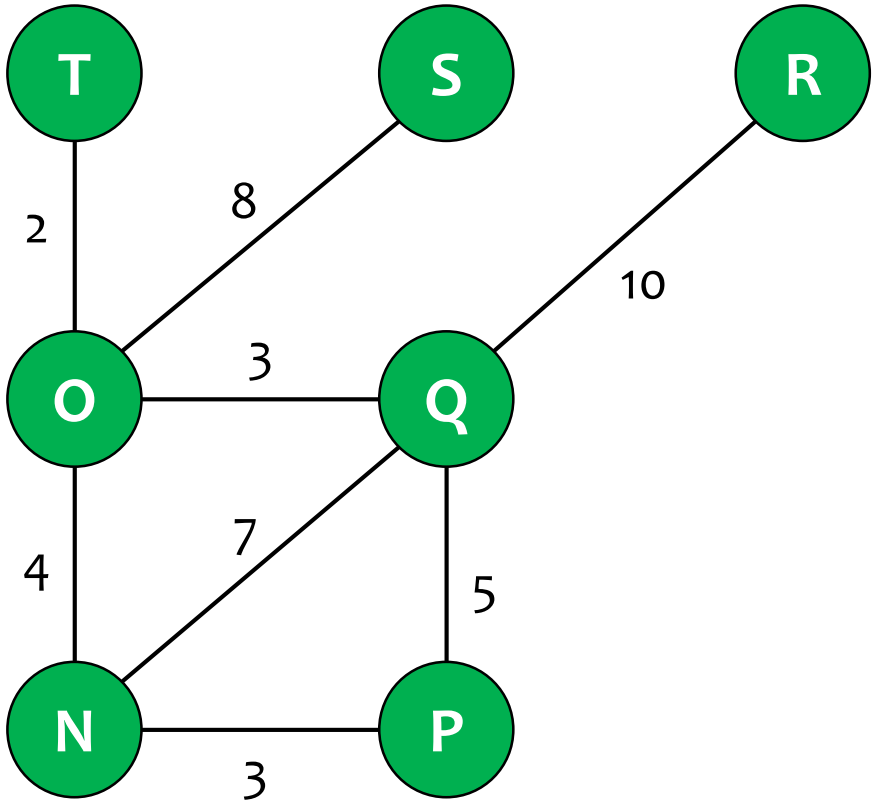
Question 1 (a)

(P,Q,5), (Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



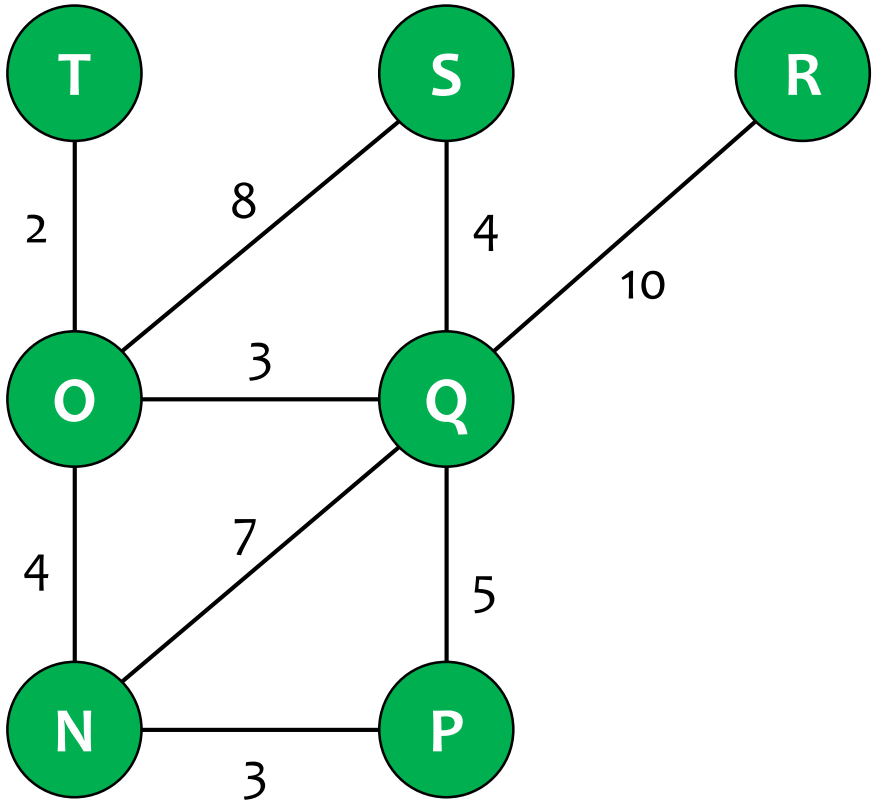
Question 1 (a)

(Q,R,10), (Q,S,4), (R,S,3), (S,T,4)



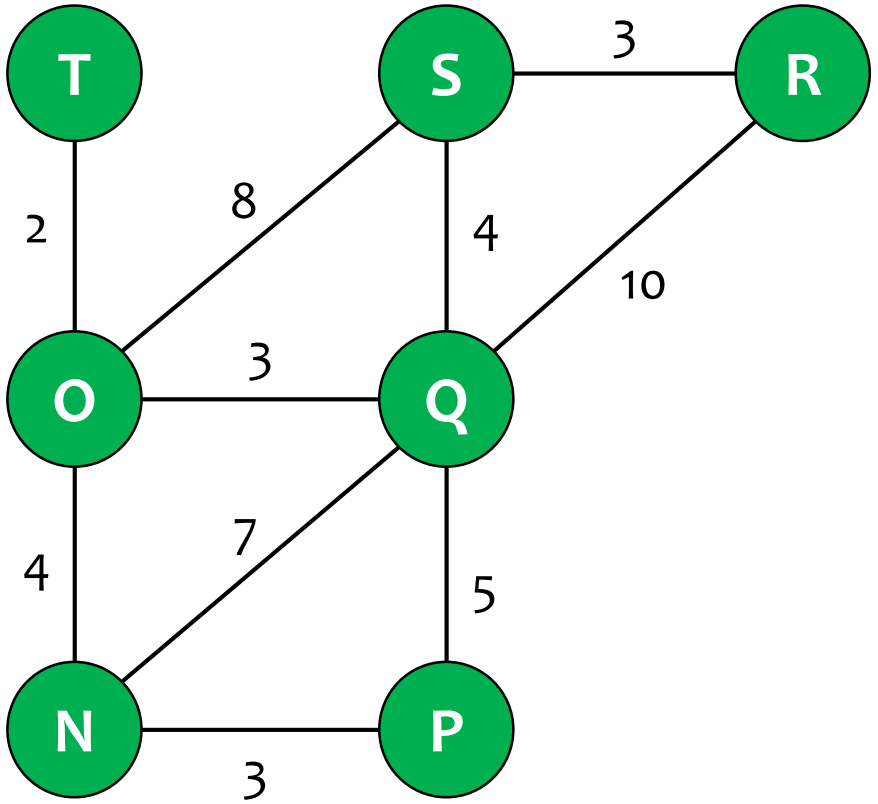
Question 1 (a)

(Q,S,4), (R,S,3), (S,T,4)



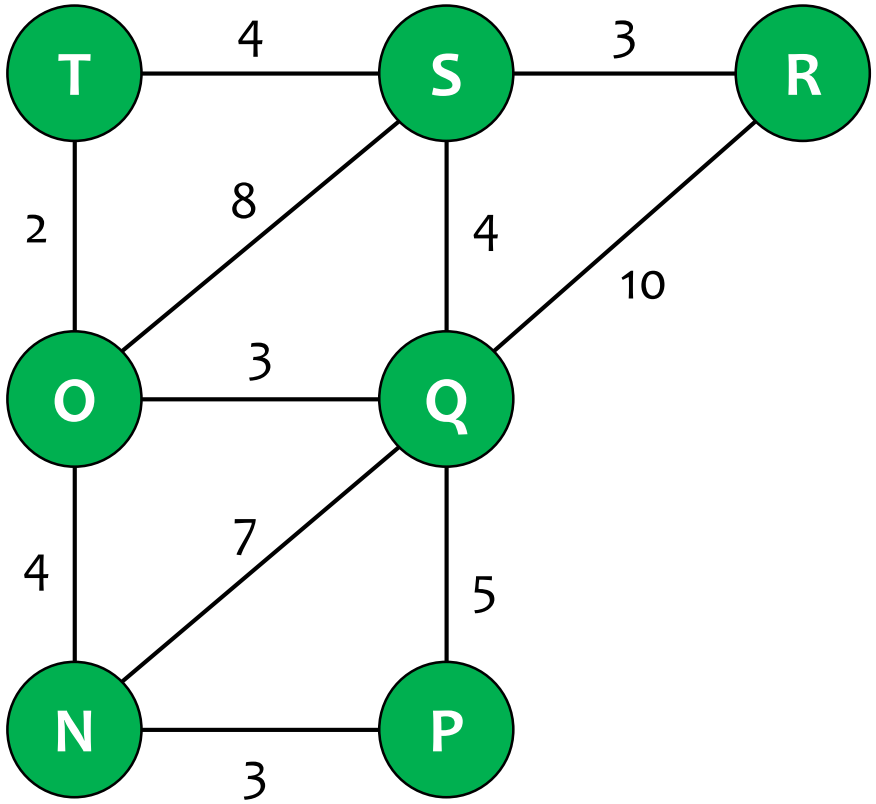
Question 1 (a)

(R,S,3), (S,T,4)



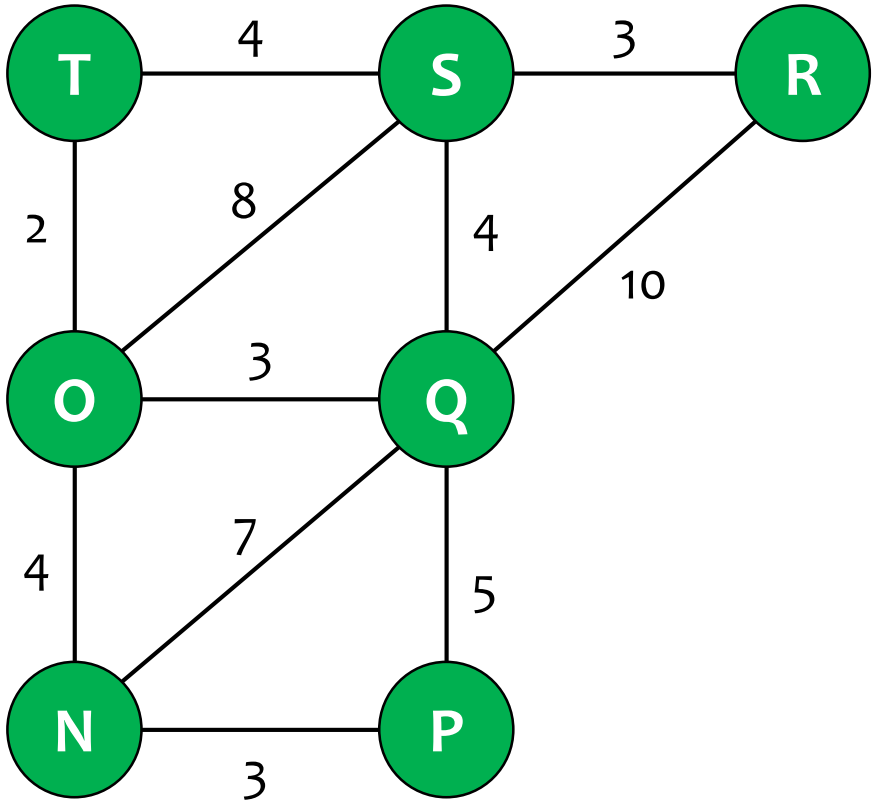
Question 1 (a)

(S,T,4)



Question 1 (a)

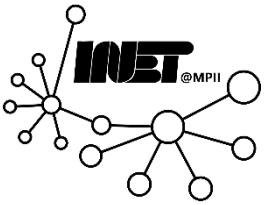
Solution:



Question 1 (b)

Compute the shortest-path-tree of node O. Give all intermediary steps.

Note: For your solution, use the format of the table we used in lecture 12 “Routing-Algorithms”, starting from slide 8 to represent the results of Dijkstra’s algorithm.



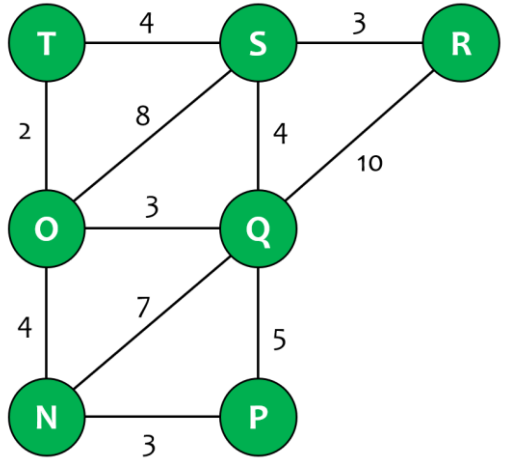
Question 1 (b)

Compute the shortest-path-tree of node O. Give all intermediary steps.

Note: For your solution, use the format of the table we used in lecture 12 “Routing-Algorithms”, starting from slide 8 to represent the results of Dijkstra’s algorithm.

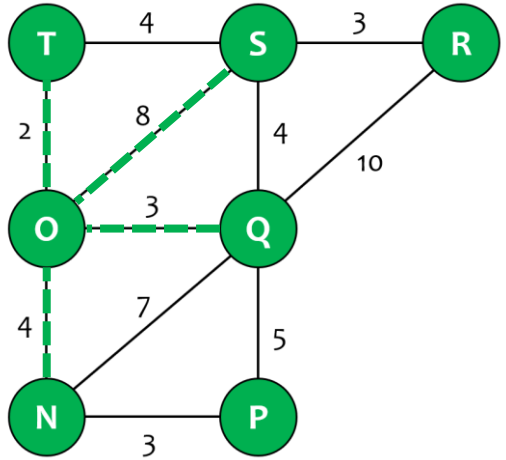
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)



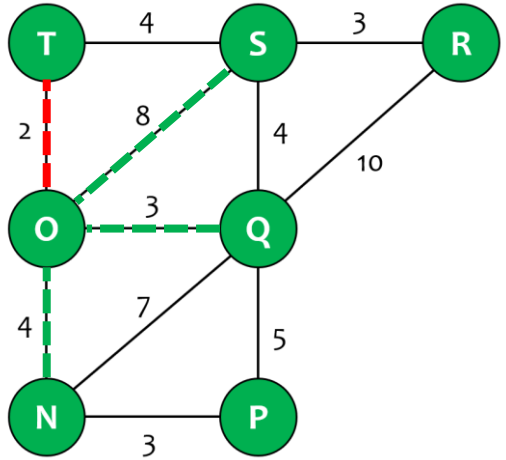
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O



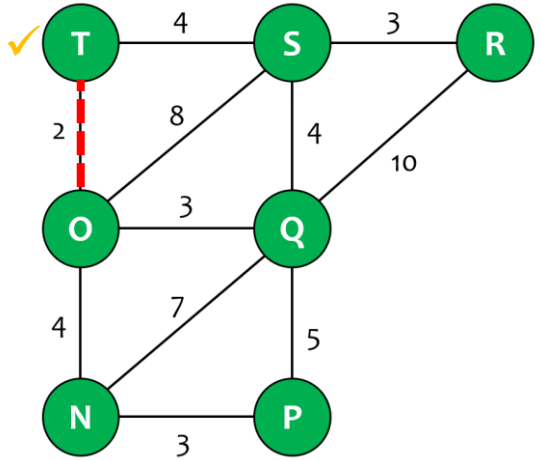
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O



Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT						

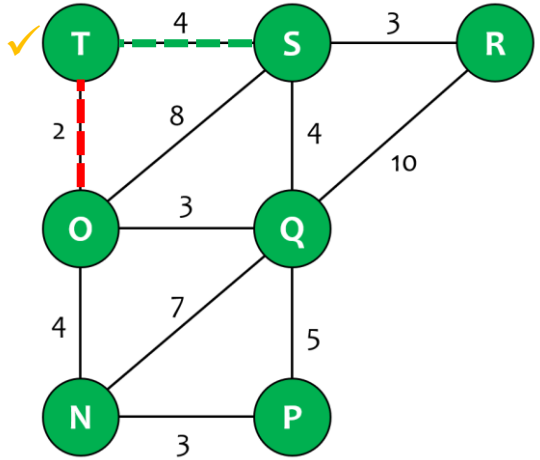


Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O

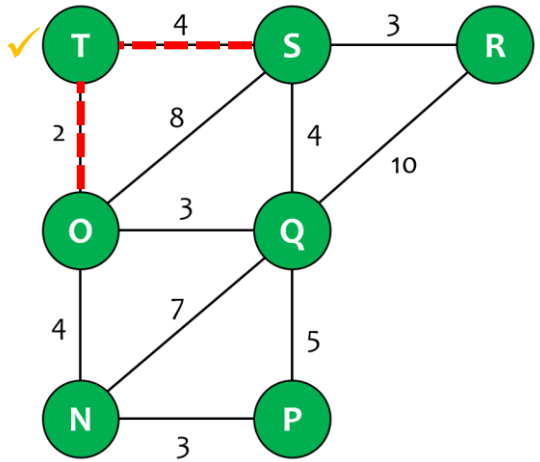
D(S) = min(D(S), (D(T) + c(T,S)))

D(S) = min(8, (2 + 4))



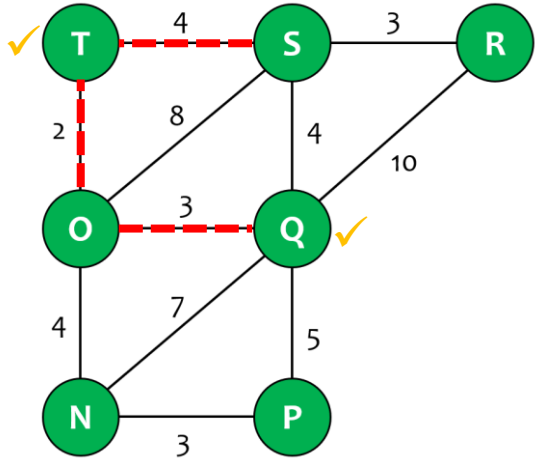
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2							



Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ						



Question 1 (b)

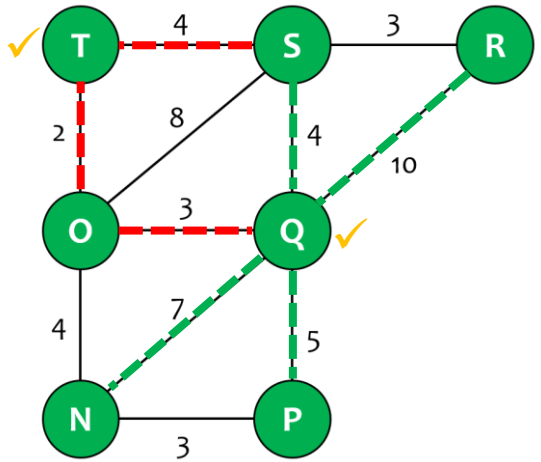
#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q) ✓	D(R),p(R)	D(S),p(S)	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O

D(P) = min(D(P), (D(Q) + c(Q,P)))

D(P) = min(∞, (3 + 5))

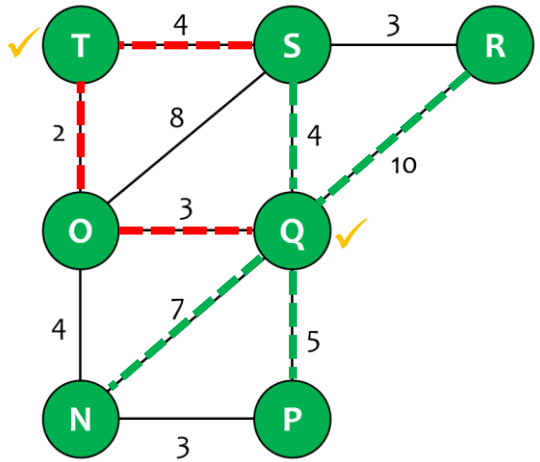
D(R) = min(D(R), (D(Q) + c(Q,R)))

D(R) = min(∞, (3 + 10))



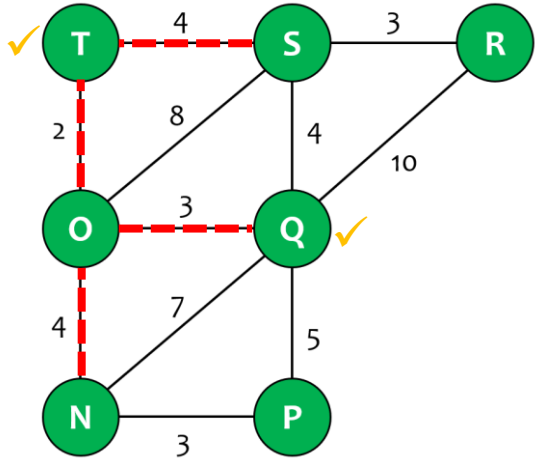
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O



Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN						

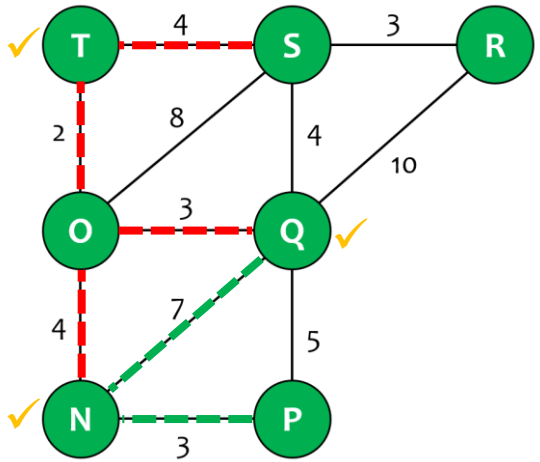


Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O

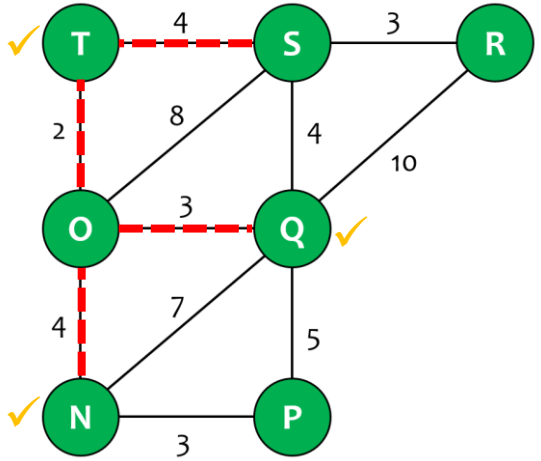
D(P) = min(D(P), (D(N) + c(N,P)))

D(P) = min(8, (4 + 3))



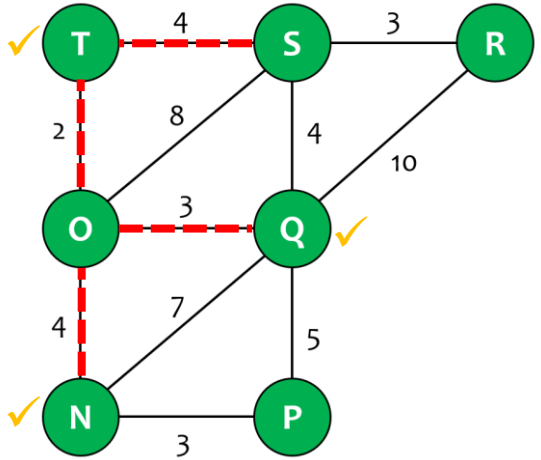
Question 1 (b)

#	start N'	D(N),p(N) ✓	D(P),p(P)	D(Q),p(Q) ✓	D(R),p(R)	D(S),p(S)	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O



Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS						

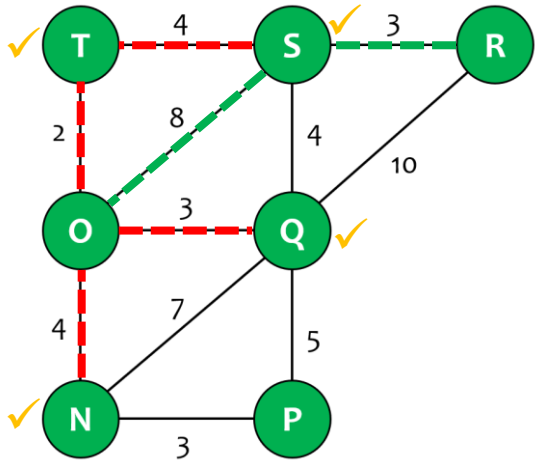


Question 1 (b)

#	start N'	D(N),p(N) ✓	D(P),p(P)	D(Q),p(Q) ✓	D(R),p(R)	D(S),p(S) ✓	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O

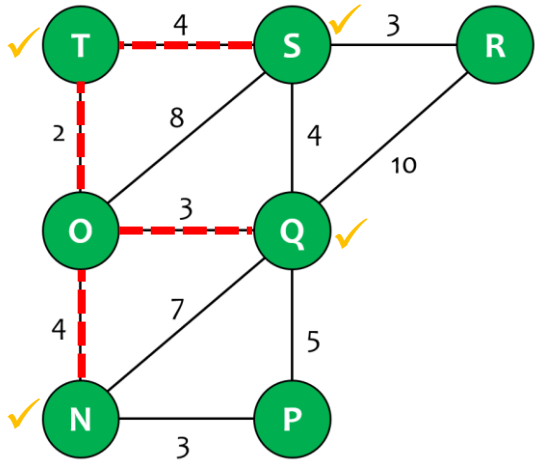
D(R) = min(D(R), (D(S) + c(S,R)))

D(R) = min(13, (6 + 3))



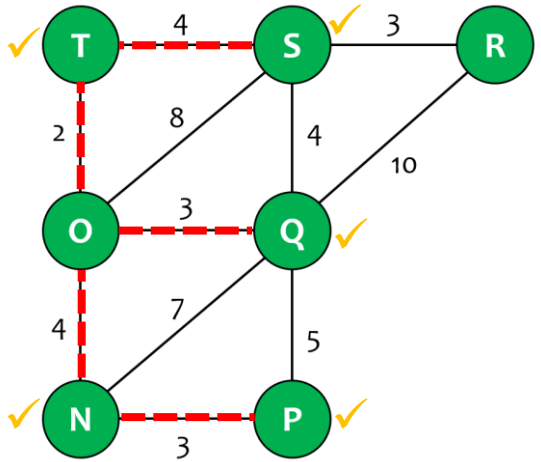
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O



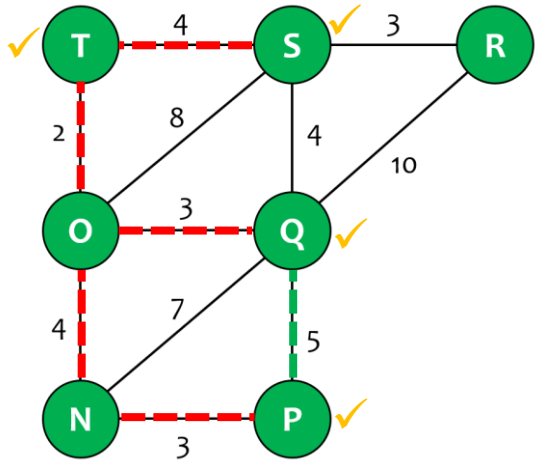
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSP						



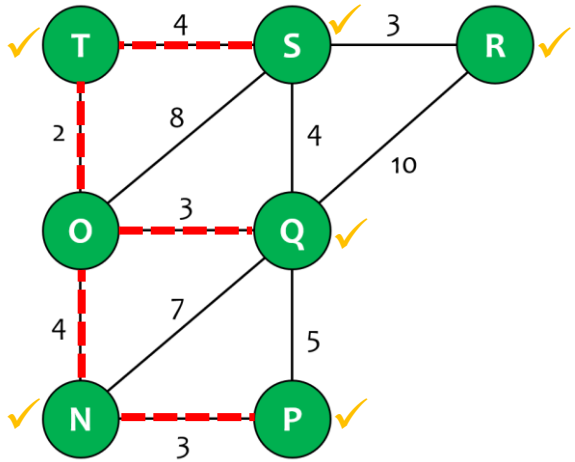
Question 1 (b)

#	start N'	D(N),p(N) ✓	D(P),p(P) ✓	D(Q),p(Q) ✓	D(R),p(R)	D(S),p(S) ✓	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSP	4,O	7,N	3,O	9,S	6,T	2,O



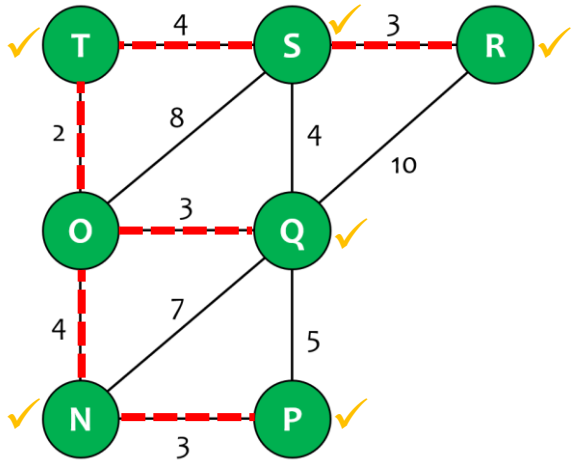
Question 1 (b)

#	start N'	D(N),p(N) ✓	D(P),p(P) ✓	D(Q),p(Q) ✓	D(R),p(R) ✓	D(S),p(S) ✓	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSP	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSPR						



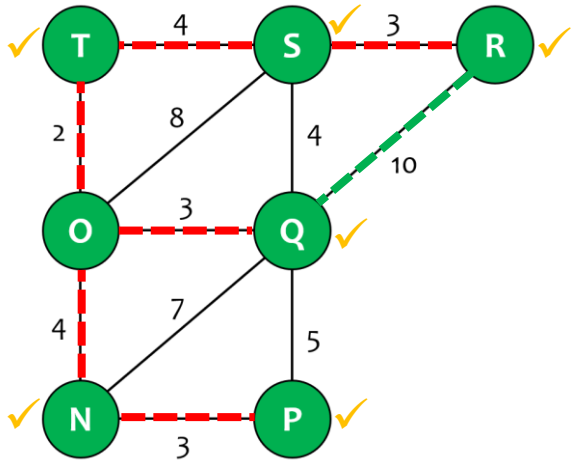
Question 1 (b)

#	start N'	D(N),p(N) ✓	D(P),p(P) ✓	D(Q),p(Q) ✓	D(R),p(R) ✓	D(S),p(S) ✓	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSP	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSPR						



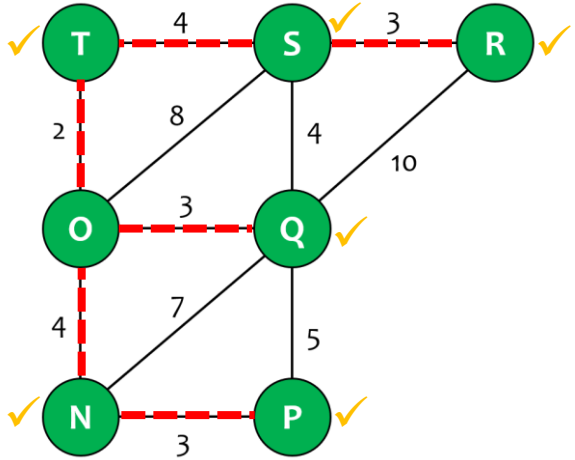
Question 1 (b)

#	start N'	D(N),p(N)	D(P),p(P)	D(Q),p(Q)	D(R),p(R)	D(S),p(S)	D(T),p(T)
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSP	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSPR	4,O	7,N	3,O	9,S	6,T	2,O

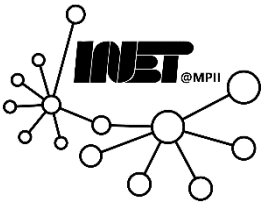


Question 1 (b)

#	start N'	D(N),p(N) ✓	D(P),p(P) ✓	D(Q),p(Q) ✓	D(R),p(R) ✓	D(S),p(S) ✓	D(T),p(T) ✓
0	O	4,O	∞	3,O	∞	8,O	2,O
1	OT	4,O	∞	3,O	∞	6,T	2,O
2	OTQ	4,O	8,Q	3,O	13,Q	6,T	2,O
	OTQN	4,O	7,N	3,O	13,Q	6,T	2,O
	OTQNS	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSP	4,O	7,N	3,O	9,S	6,T	2,O
	OTQNSPR	4,O	7,N	3,O	9,S	6,T	2,O



Questions?

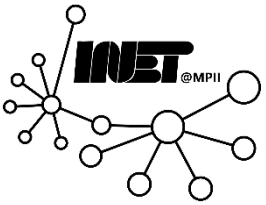


Question 1 (c)

Use your sketch from (a) to draw the shortest-path-tree from node O.

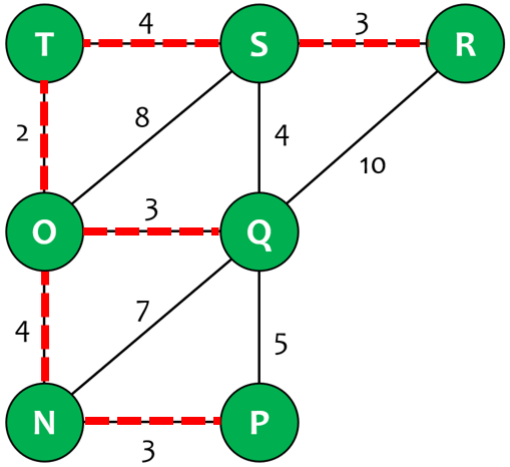
Question 1 (c)

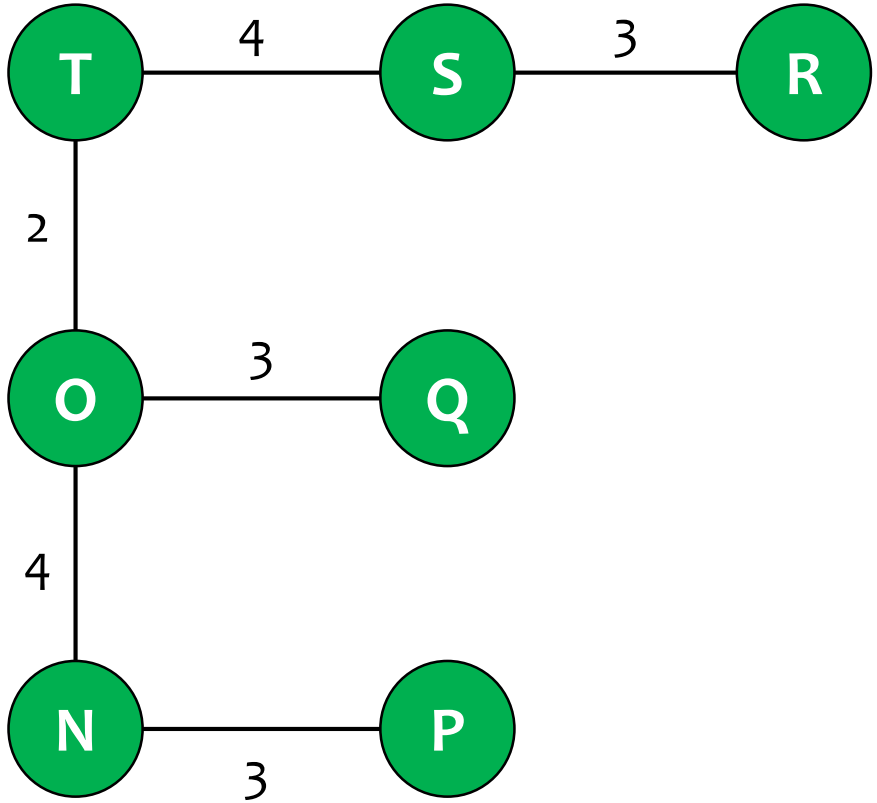
Use your sketch from (a) to draw the shortest-path-tree from node O.



Question 1 (c)

Use your sketch from (a) to draw the shortest-path-tree from node O.



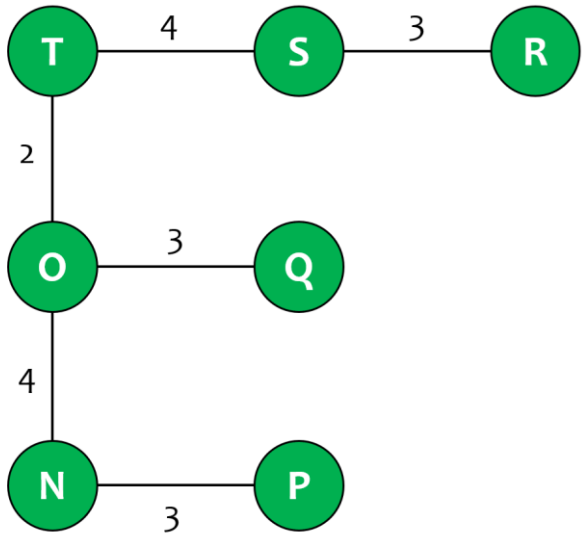


Question 1 (d)

Based on the shortest-path-tree, provide the forwarding table for node O. An example for node N could look like this:

Question 1 (d)

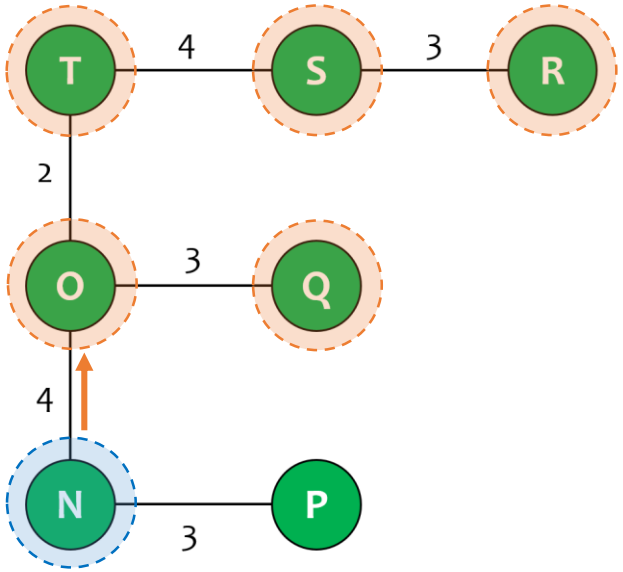
Based on the shortest-path-tree, provide the forwarding table for node O. An example for node N could look like this:



Destination	Next Hop

Question 1 (d)

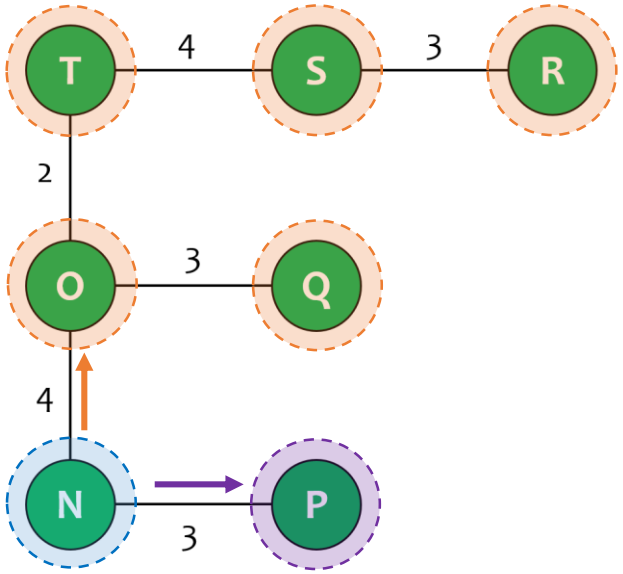
An example for node N could look like this:



Destination	Next Hop
O, T, Q, S, R	O

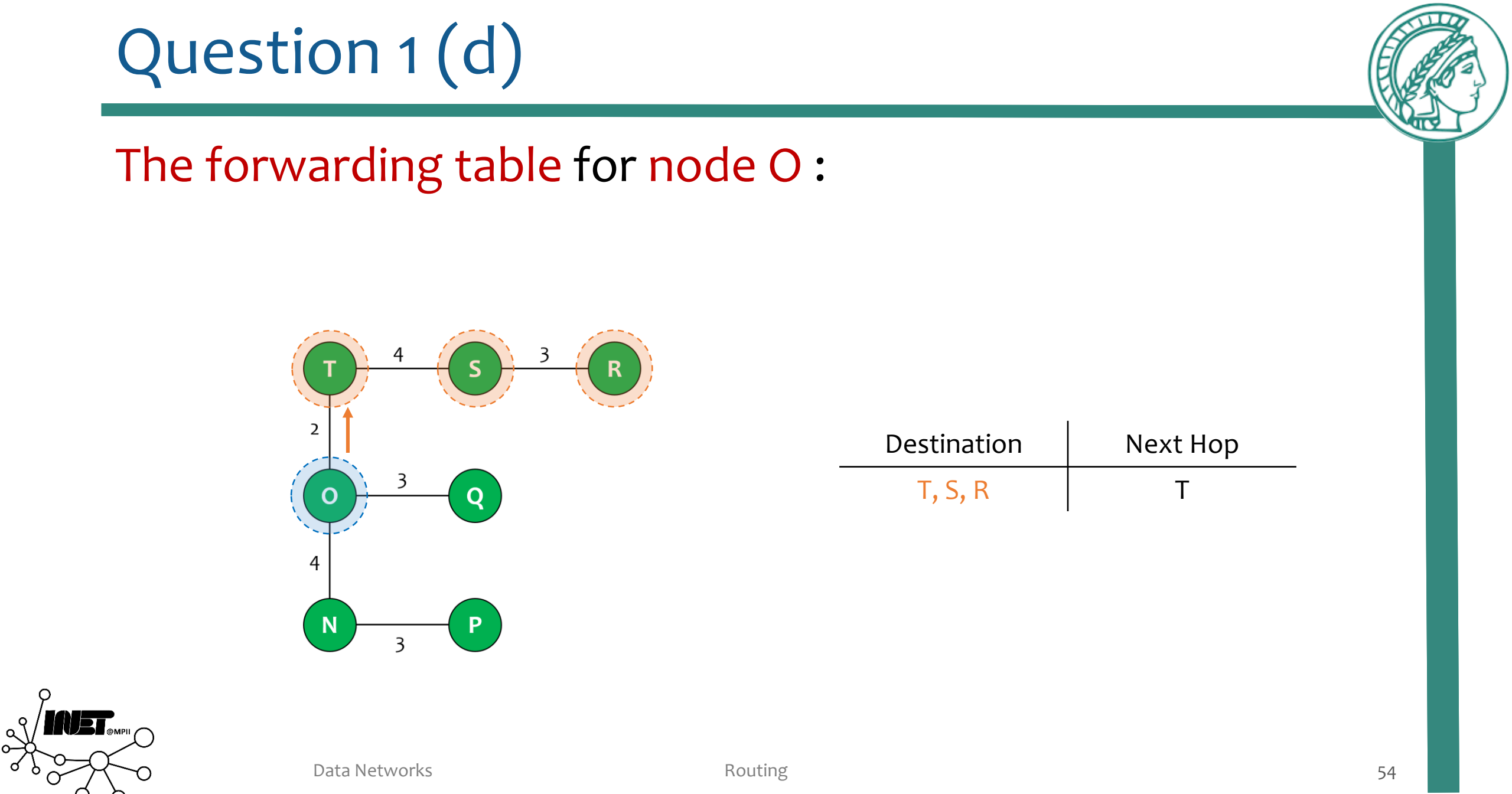
Question 1 (d)

An example for node N could look like this:



Destination	Next Hop
O, T, Q, S, R	O
P	P

Question 1 (d)

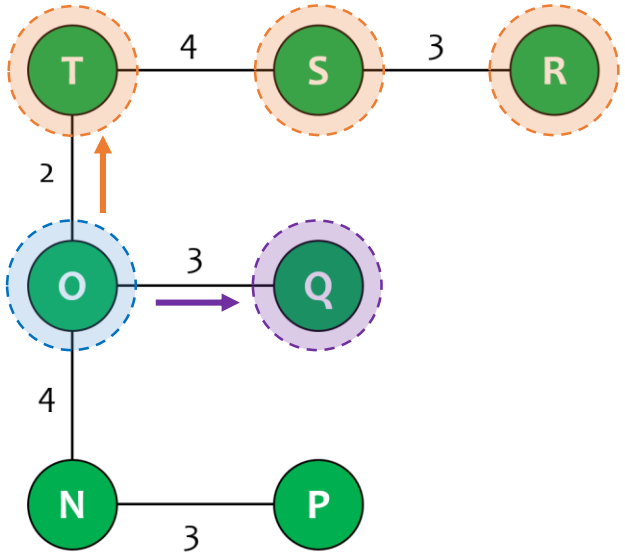


The forwarding table for node O :

Destination	Next Hop
T, S, R	T

Question 1 (d)

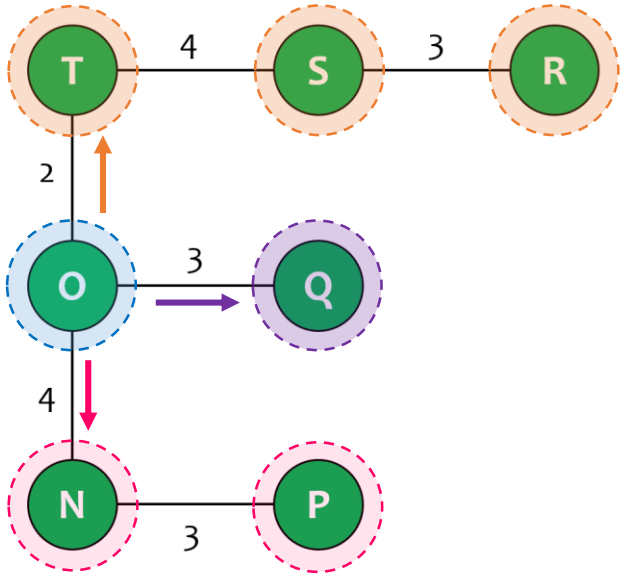
The forwarding table for node O :



Destination	Next Hop
T, S, R	T
Q	Q

Question 1 (d)

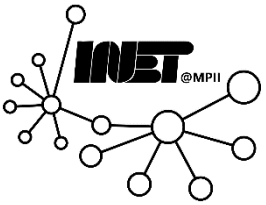
The forwarding table for node O :



Destination	Next Hop
T, S, R	T
Q	Q
N, P	N

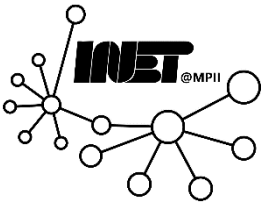
Question 1 (e)

Assume that link (T, S, 4) fails. Use your sketch from (a) and draw the recomputed shortest-path-tree for node O. Provide the recomputed forwarding table for node O in a table.



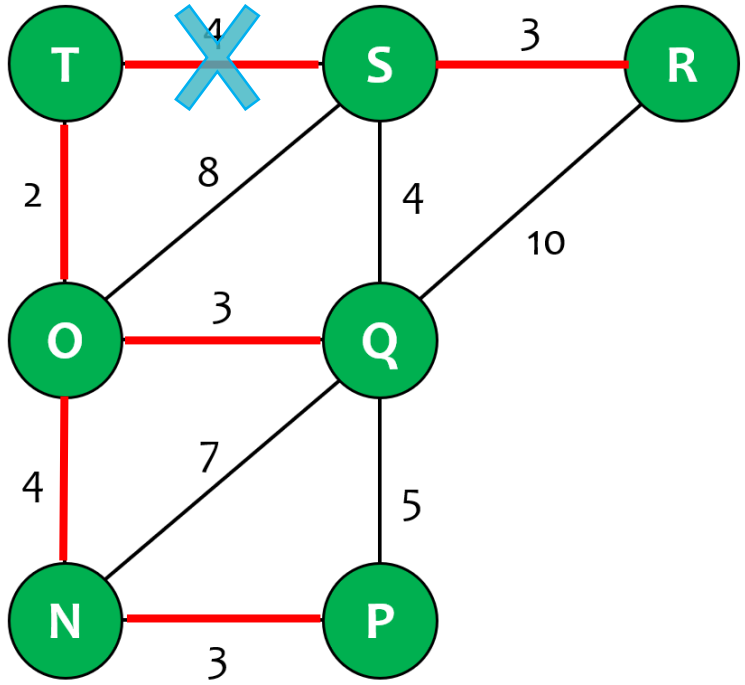
Question 1 (e)

Assume that link (T, S, 4) fails. Use your sketch from (a) and draw the recomputed shortest-path-tree for node O.
Provide the recomputed forwarding table for node O in a table.

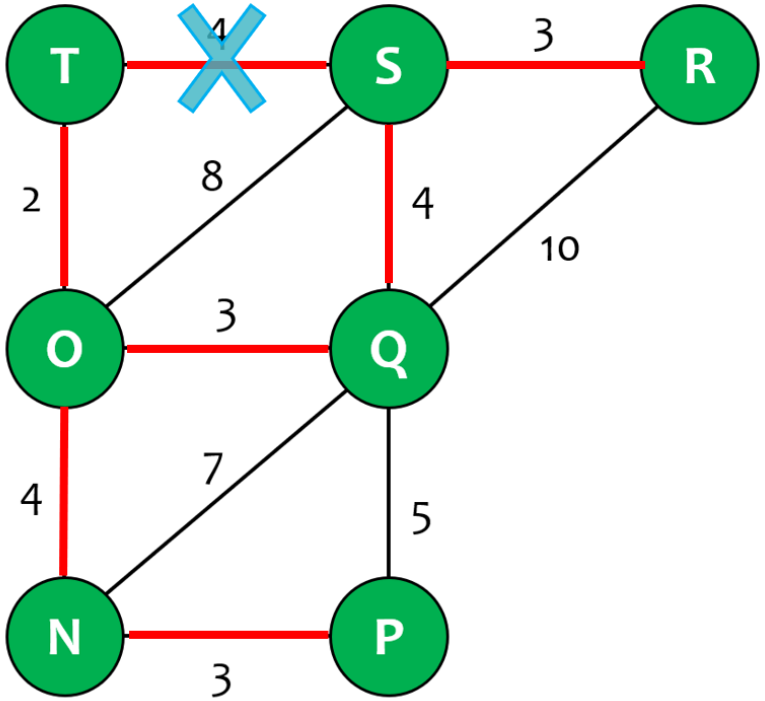


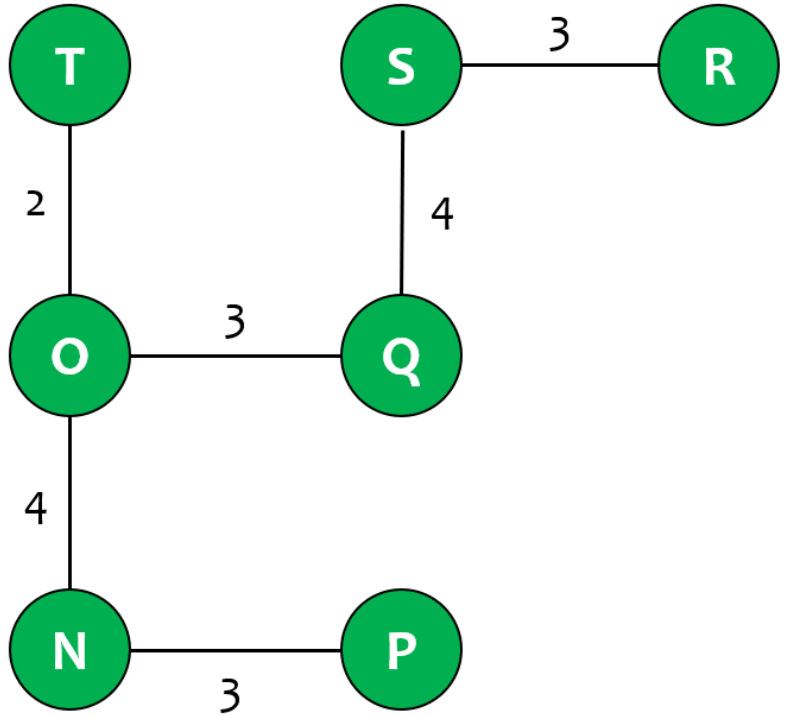
Question 1 (e)

Assume that link (T, S, 4) fails. Use your sketch from (a) and draw the recomputed shortest-path-tree for node O.



Question 1 (e)



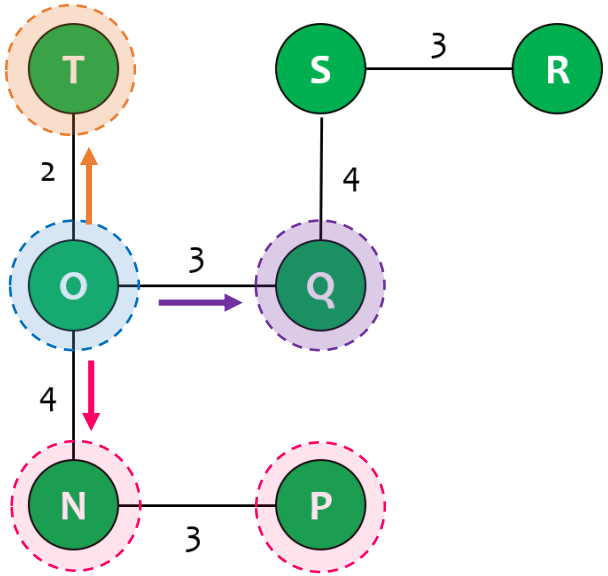


Question 1 (e)

Assume that link (T, S, 4) fails. Use your sketch from (a) and draw the recomputed shortest-path-tree for node O.
Provide the recomputed forwarding table for node O in a table.

Question 1 (e)

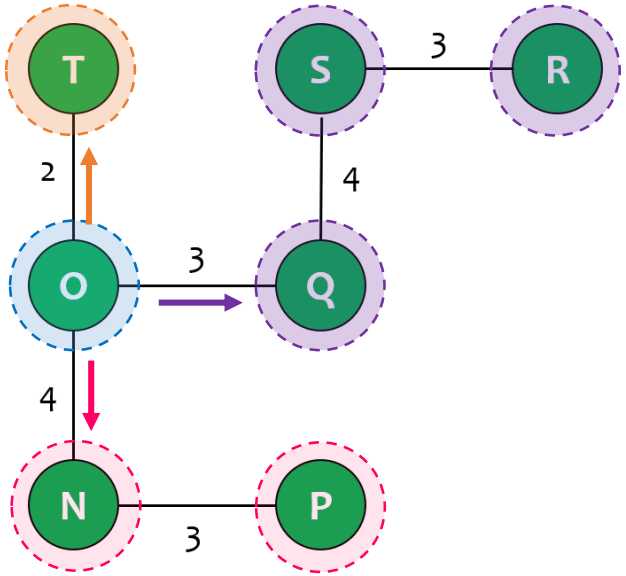
The forwarding table for node O



Destination	Next Hop
T	T
Q	Q
N, P	N

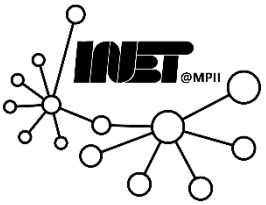
Question 1 (e)

The forwarding table for node O



Destination	Next Hop
T	T
Q, S, R	Q
N, P	N

Questions?



Question 2 (Distance Vector Routing with RIP)

RIP is a distance vector protocol which uses information in updates to calculate the routing table using the Bellman-Ford algorithm.

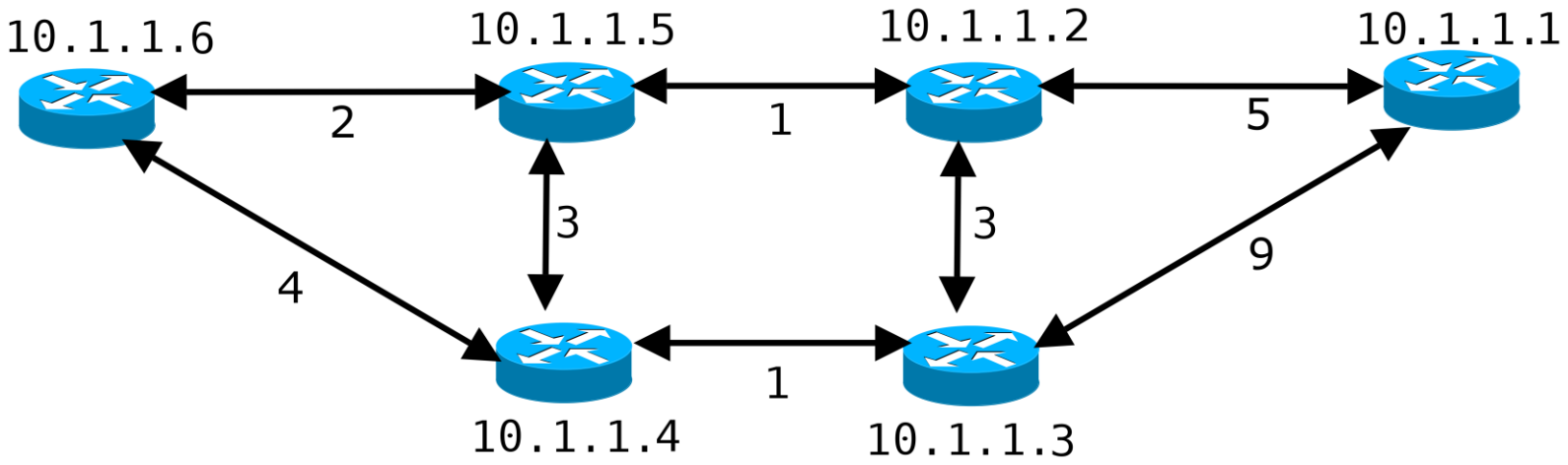
Figure shows a simple topology where each router is identified by one IP address (for simplicity, we don't consider interfaces) and the cost of each link is written in the middle of the arc connecting the routers.

Question 2 (Distance Vector Routing with RIP)

RIP is a distance vector protocol which uses information in updates to calculate the routing table using the Bellman-Ford algorithm.

Figure shows a simple topology where each router is identified by one IP address (for simplicity, we don't consider interfaces) and the cost of each link is written in the middle of the arc connecting the routers.

Question 2 (Distance Vector Routing with RIP)



Question 2 (a)

Compute the routing table for 10.1.1.6 using Bellman-Ford (not RIP). Please write down only the final routing table state using the below Table.

destination	next	#hops
x.x.x.x	y.y.y.y	z

Table 2: A routing table.

Question 2 (a)

Compute the routing table for 10.1.1.6 using Bellman-Ford (not RIP). Please write down only the final routing table state using the below Table.

destination	next	#hops
x.x.x.x	y.y.y.y	z

Table 2: A routing table.

Question 2 (a)

Bellman-Ford Equation (dynamic programming)

Define

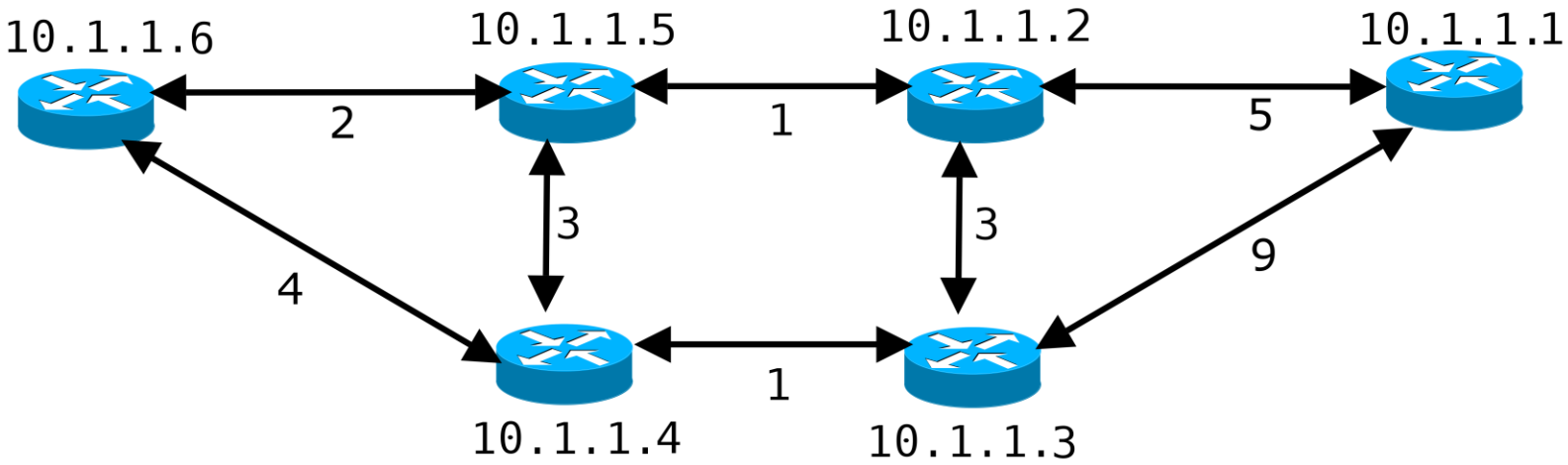
- $d_x(y)$:= cost of least-cost path from x to y

Then

- $d_x(y) = \min_v\{ c(x, v) + d_v(y) \}$

  where min is taken over all neighbors v of x

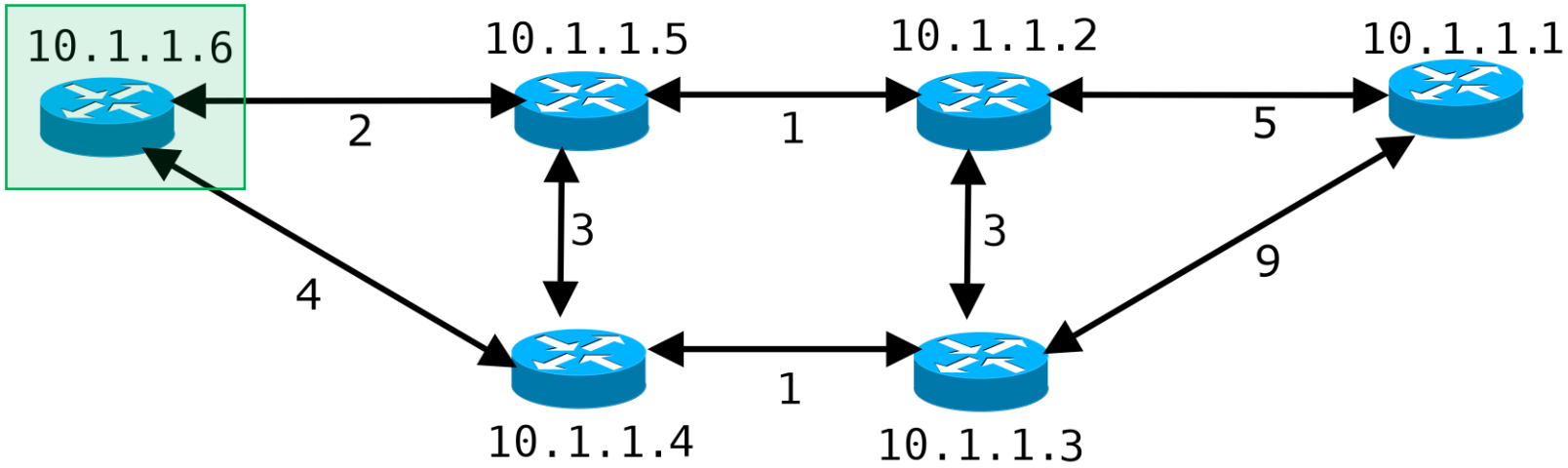
Question 2 (a)



destination	next	#hops	(weight)

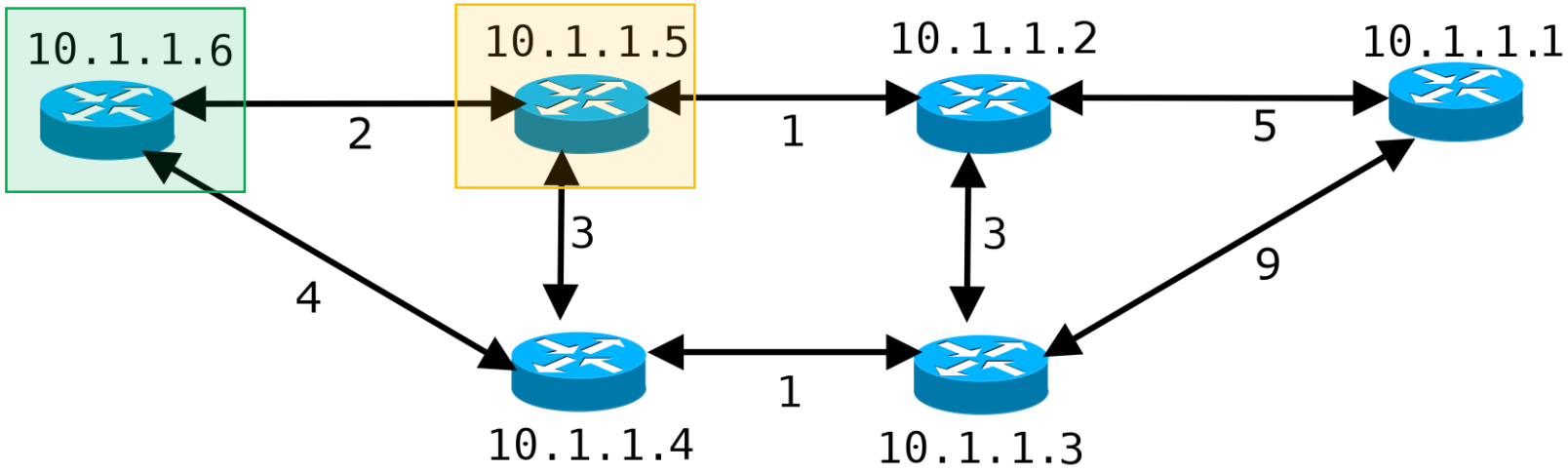
→ Optional

Question 2 (a)



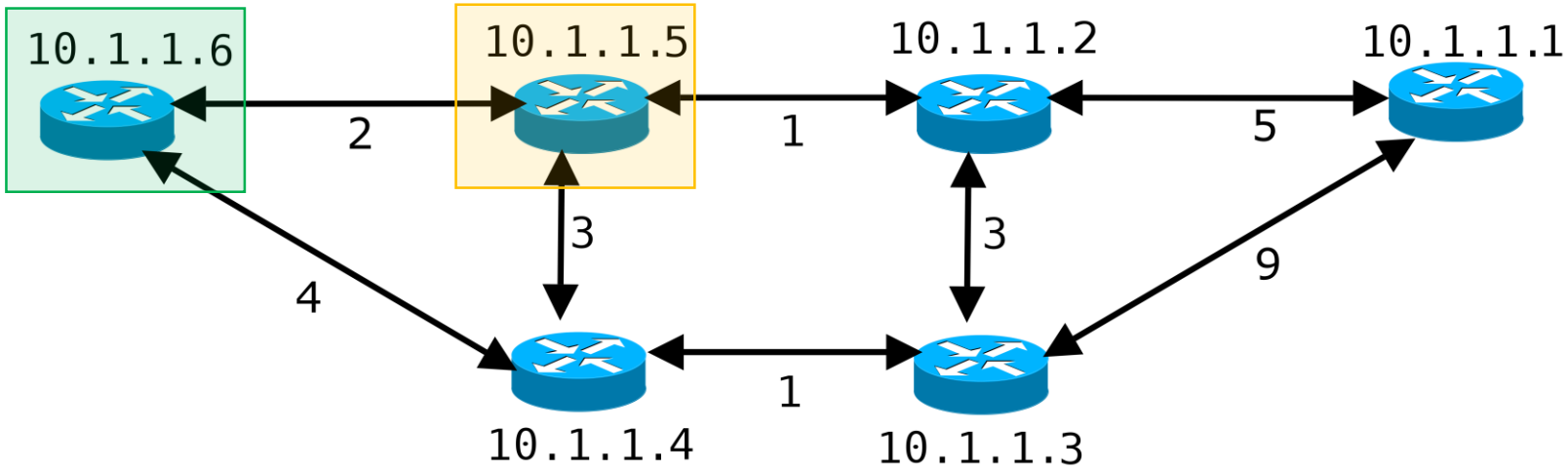
destination	next	#hops	(weight)

Question 2 (a)



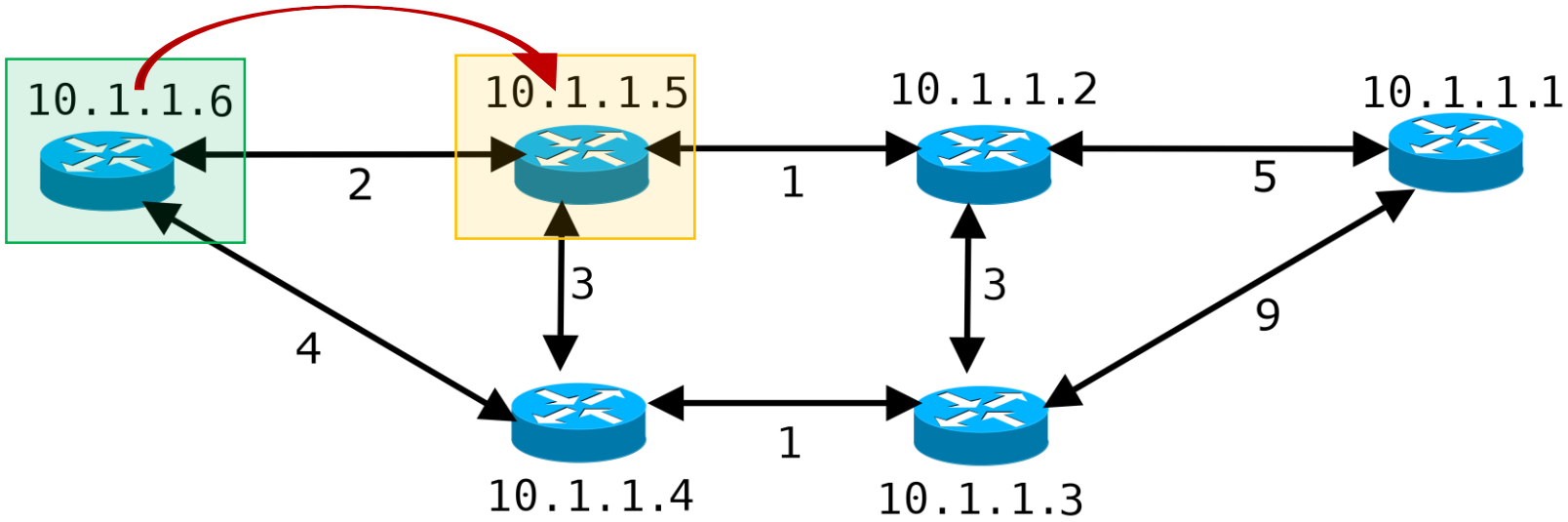
destination	next	#hops	(weight)
10.1.1.5			

Question 2 (a)



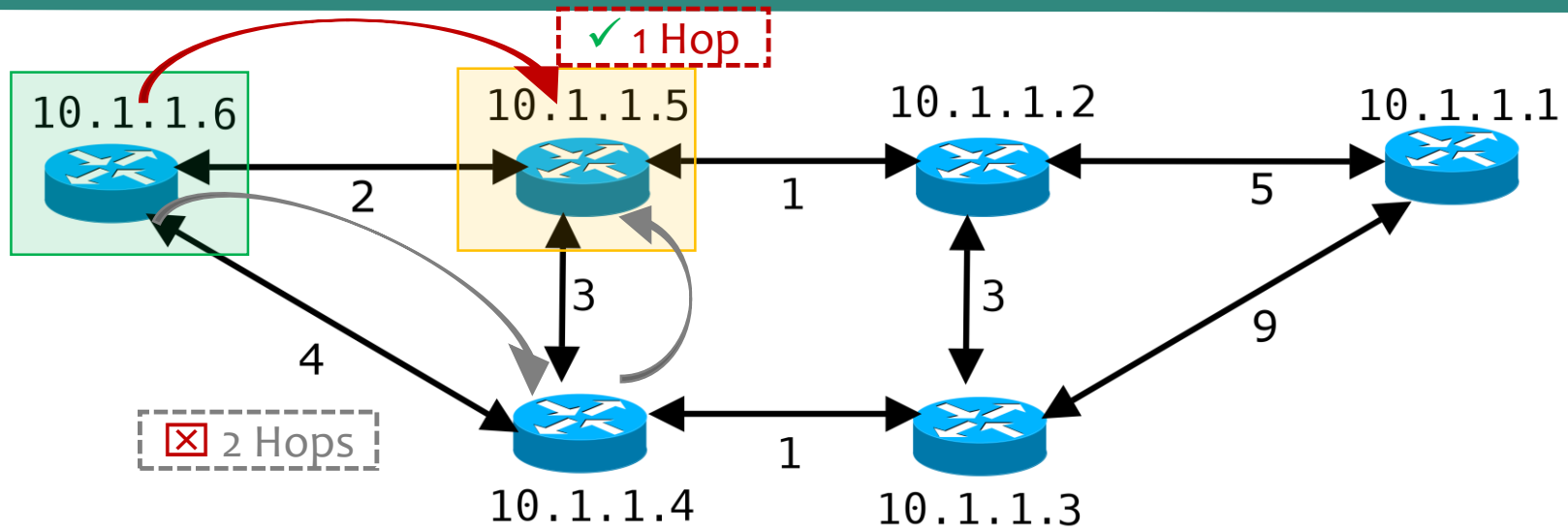
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5		

Question 2 (a)



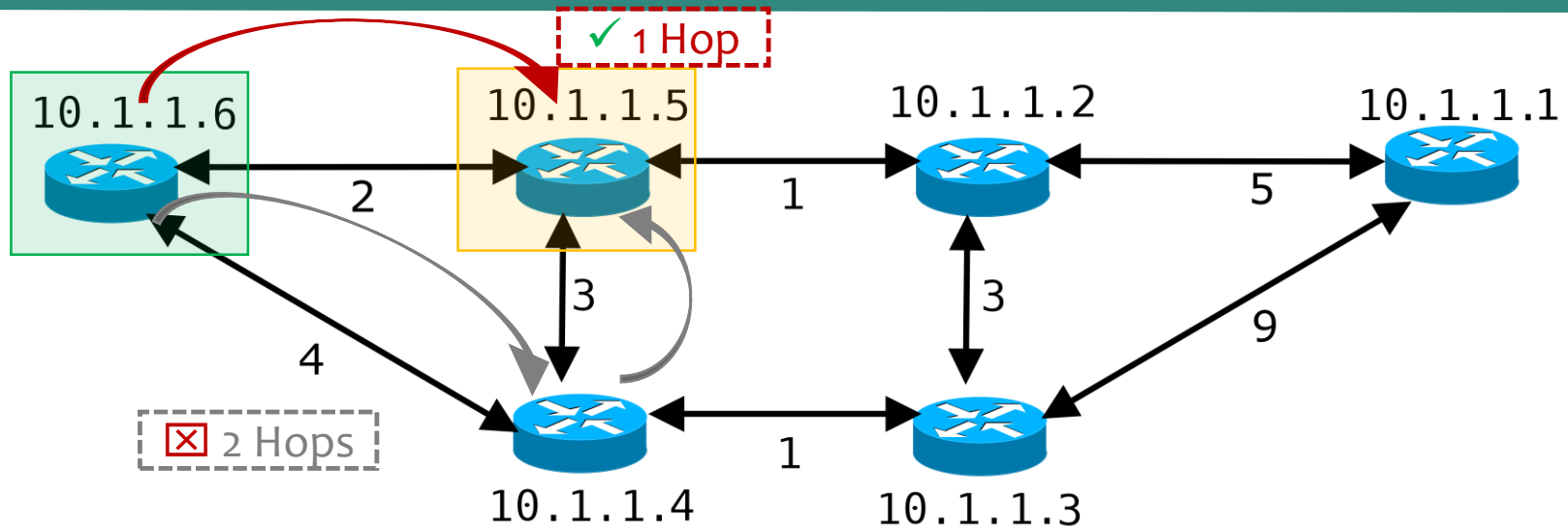
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5		

Question 2 (a)



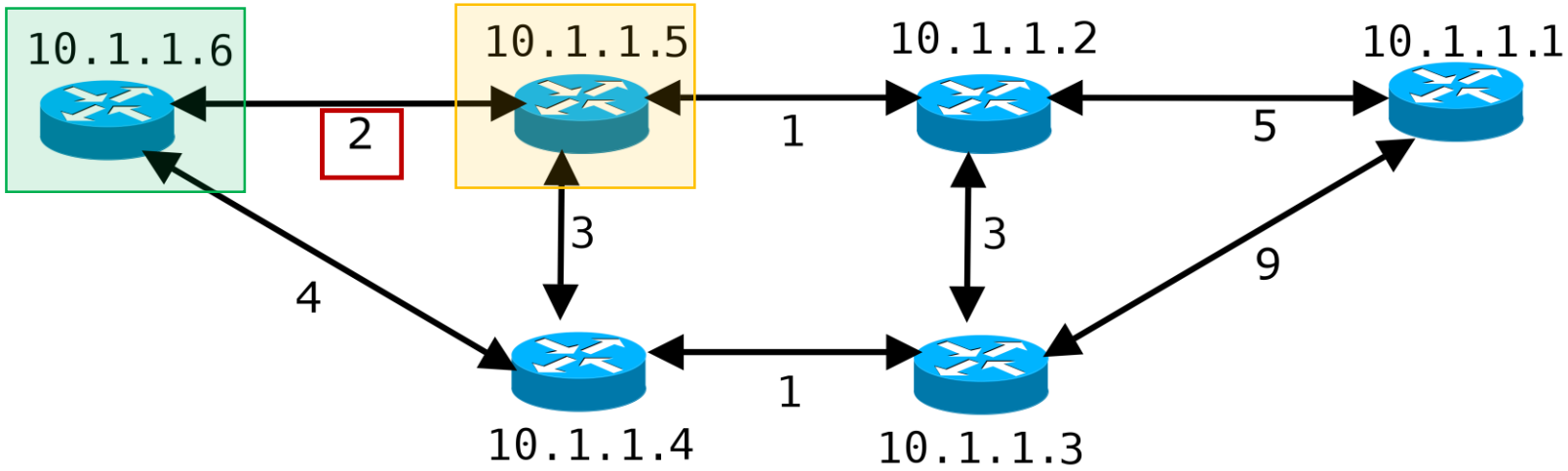
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5		

Question 2 (a)



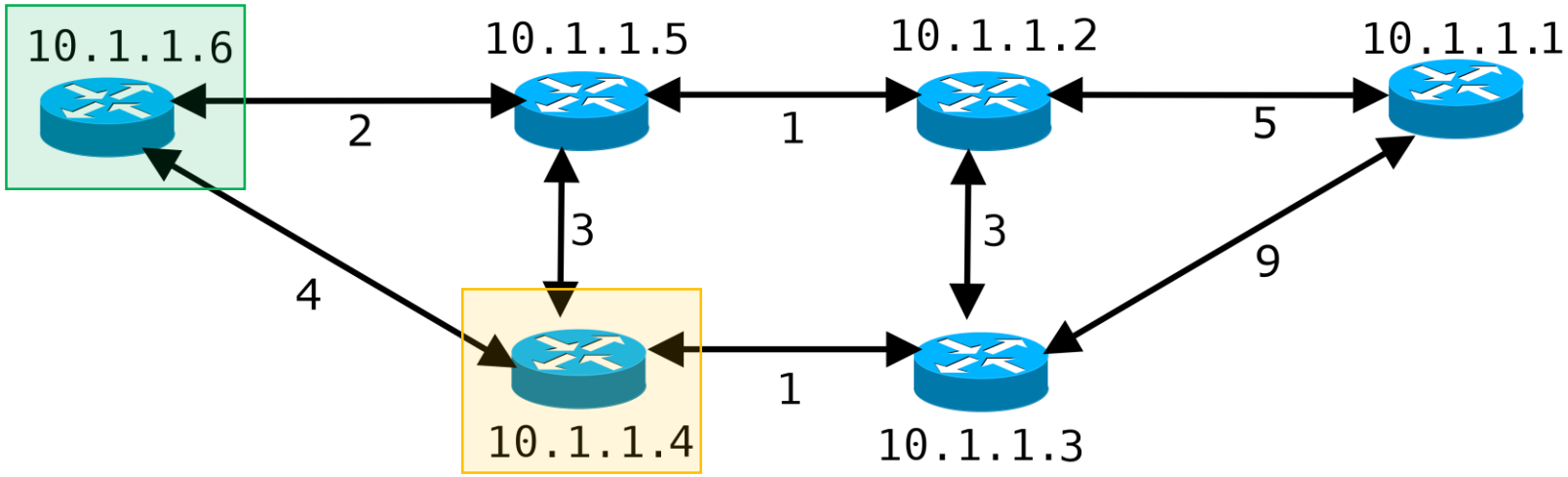
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	

Question 2 (a)



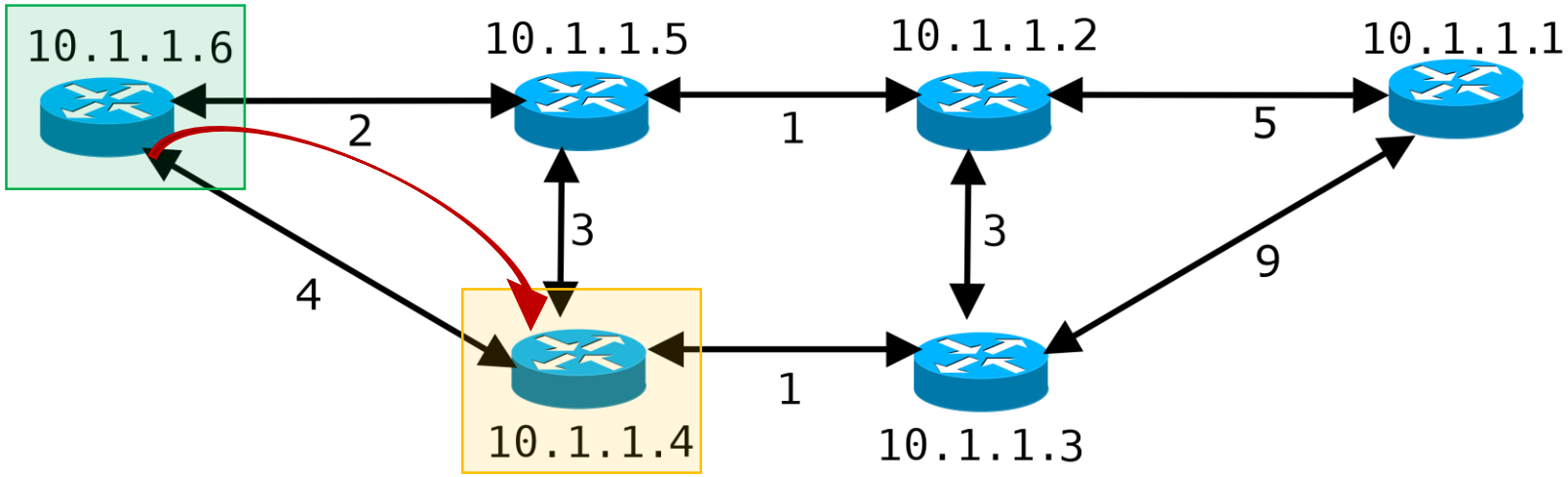
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)

Question 2 (a)



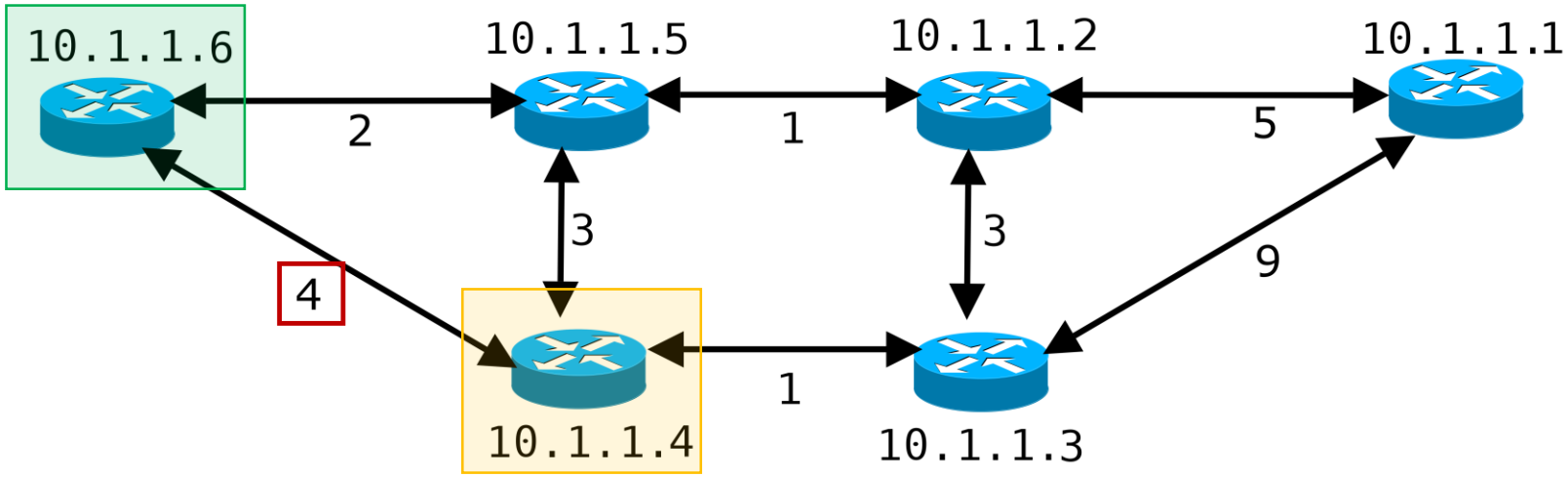
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4			

Question 2 (a)



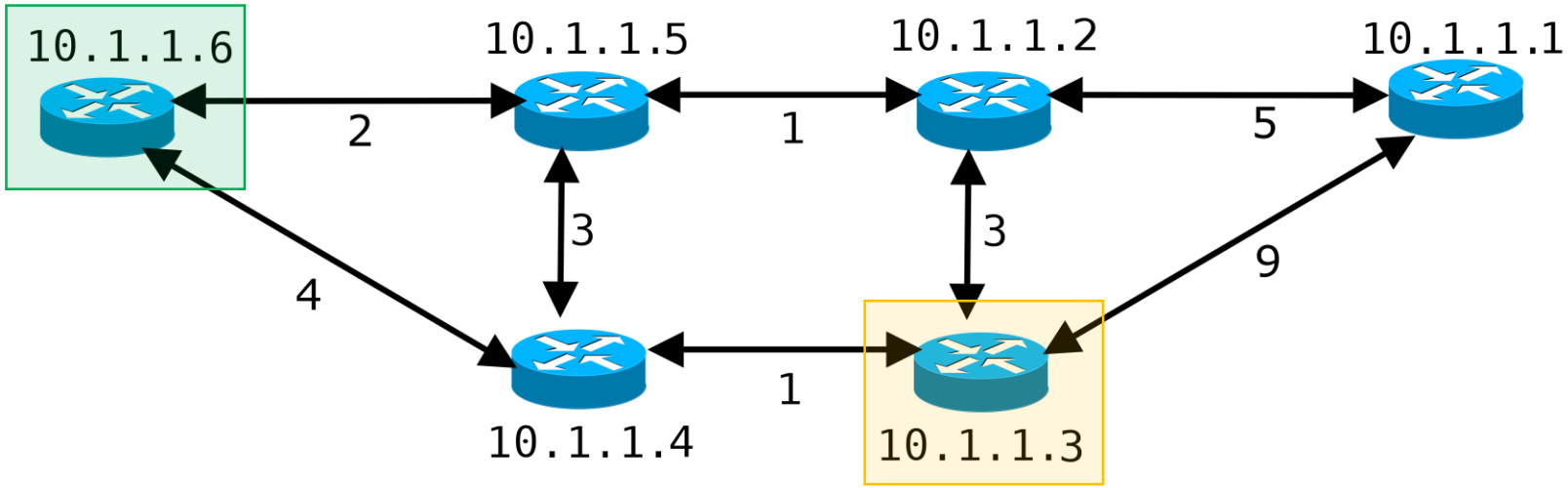
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	

Question 2 (a)



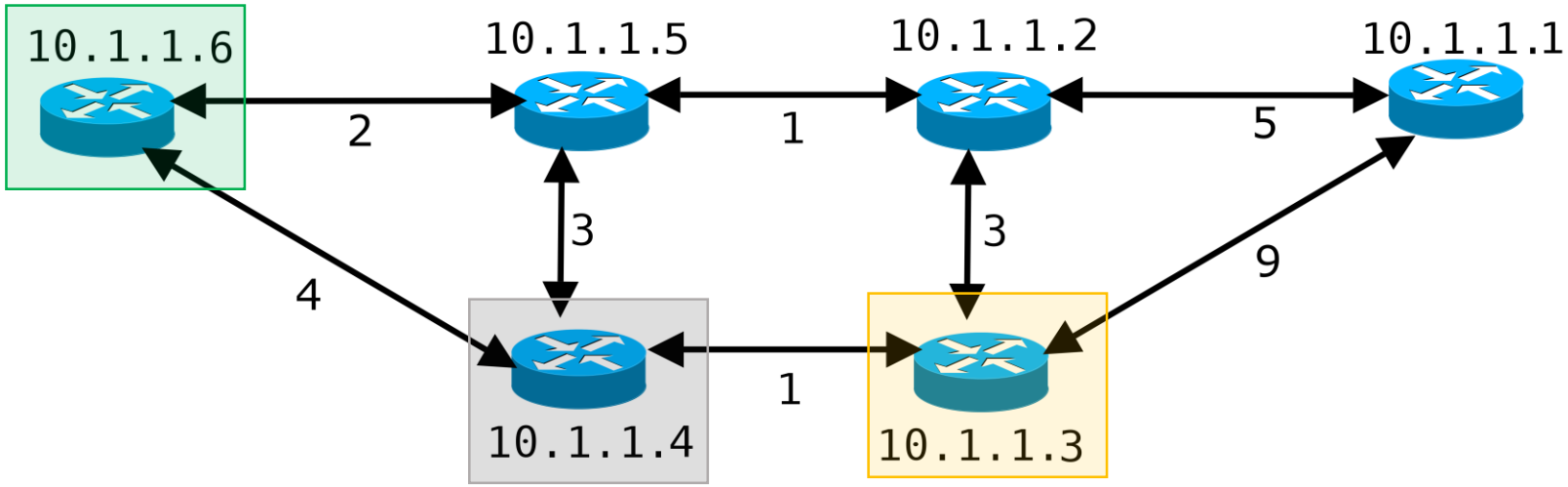
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)

Question 2 (a)



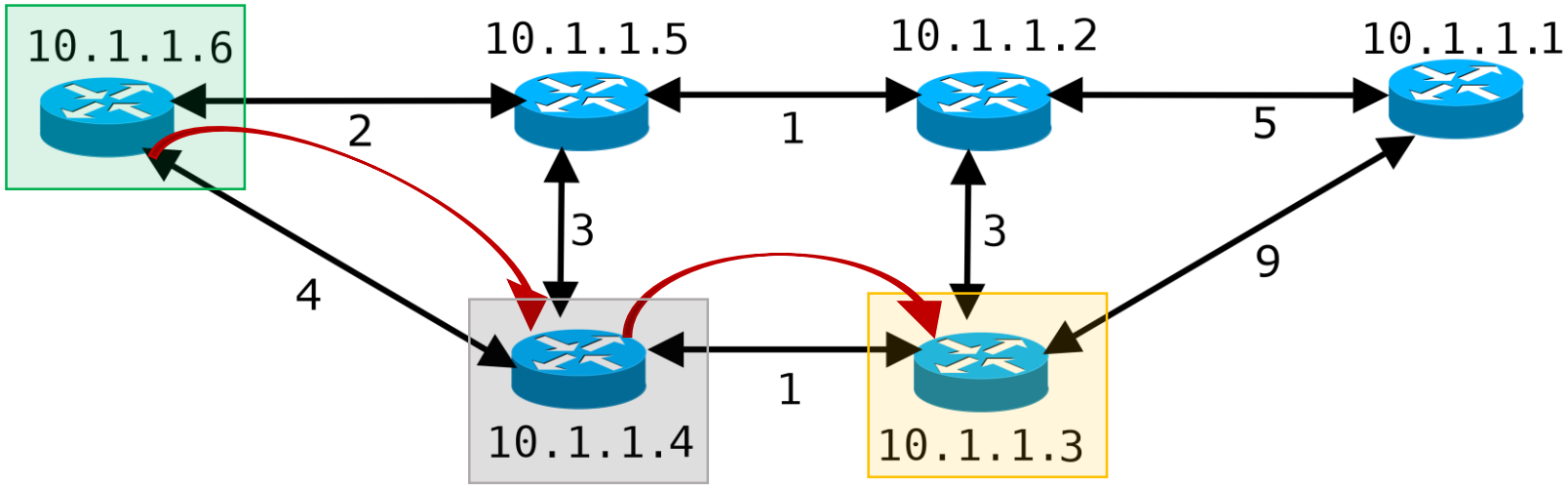
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3			

Question 2 (a)



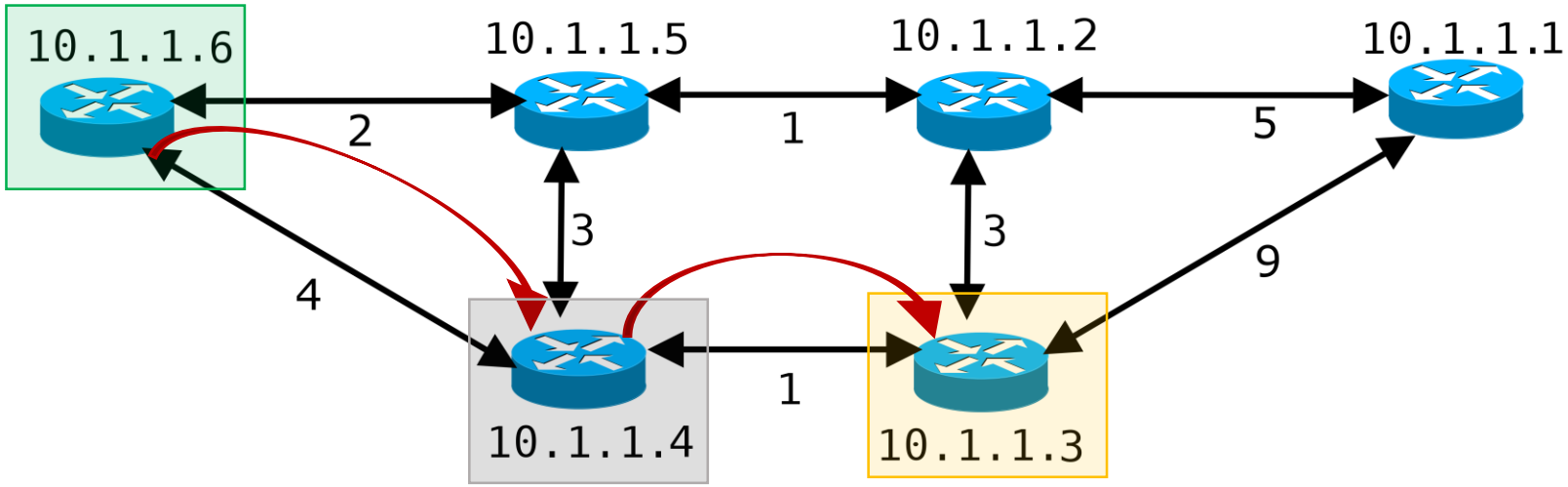
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4		

Question 2 (a)



destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	

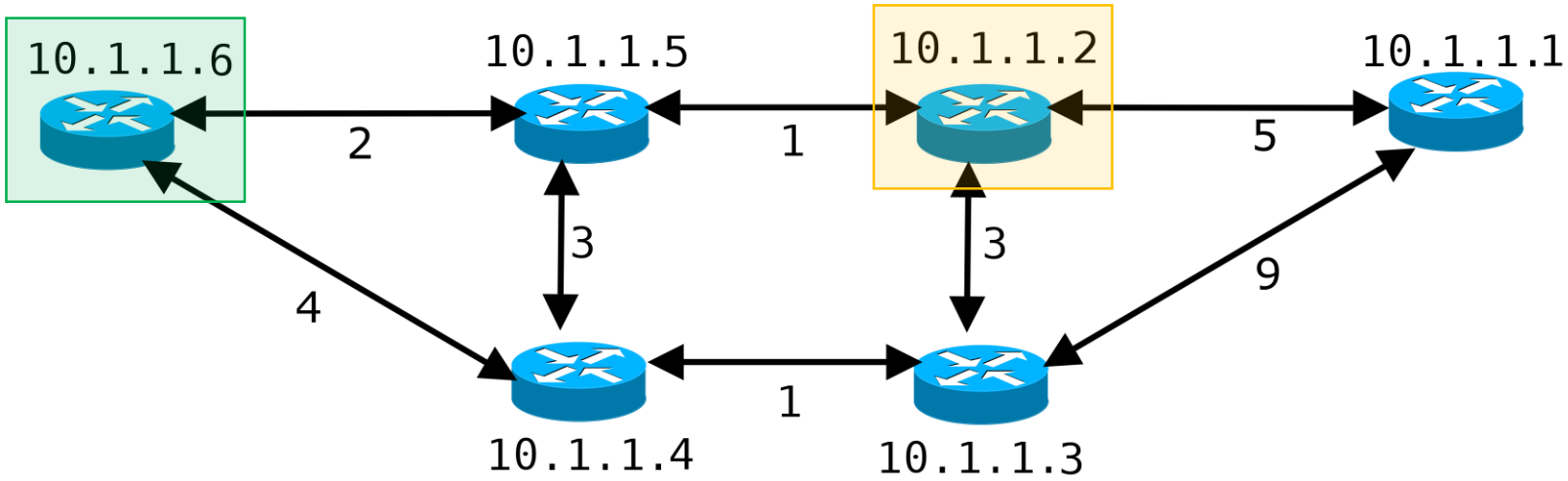
Question 2 (a)



destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)

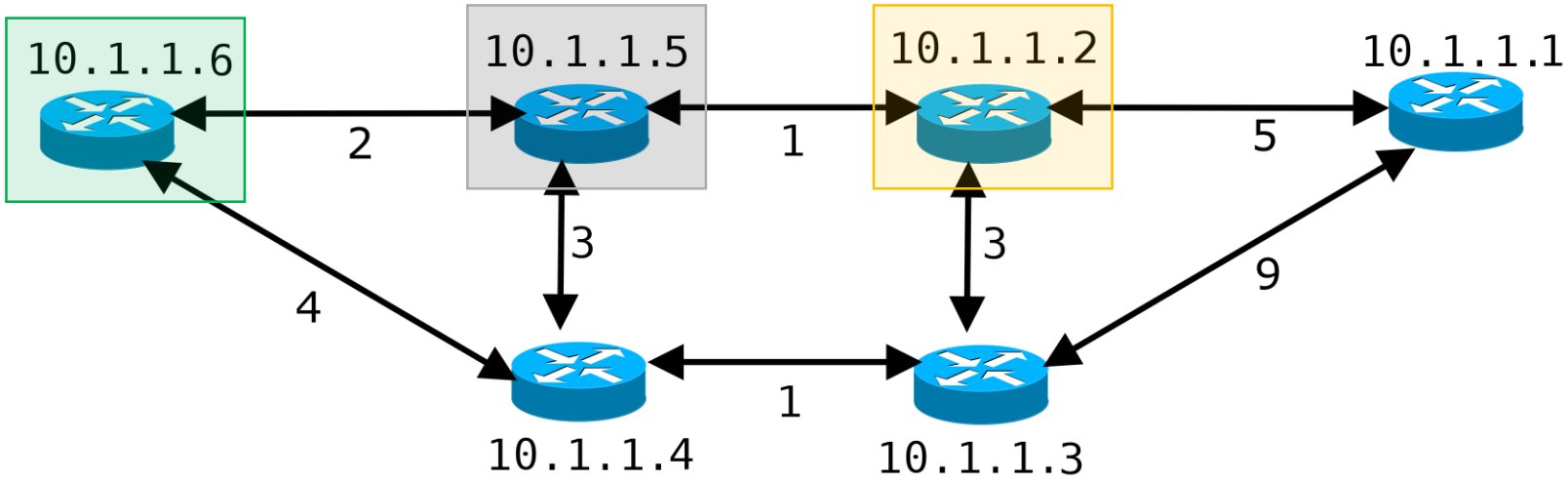
4+1

Question 2 (a)



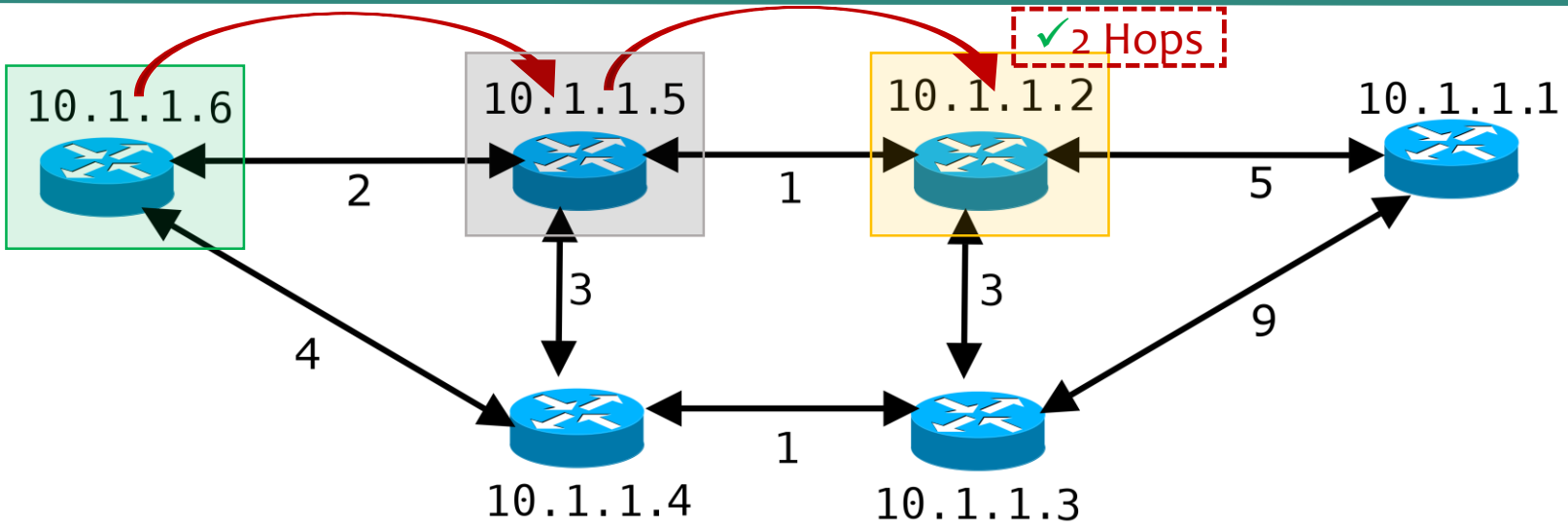
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2			

Question 2 (a)



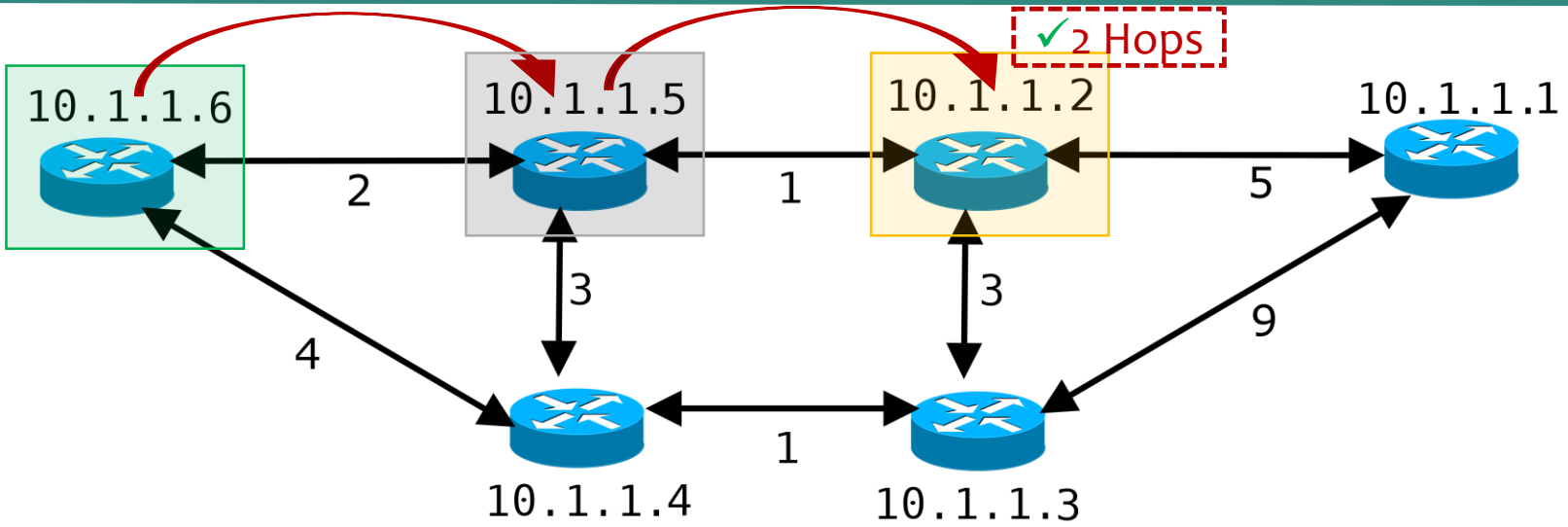
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5		

Question 2 (a)



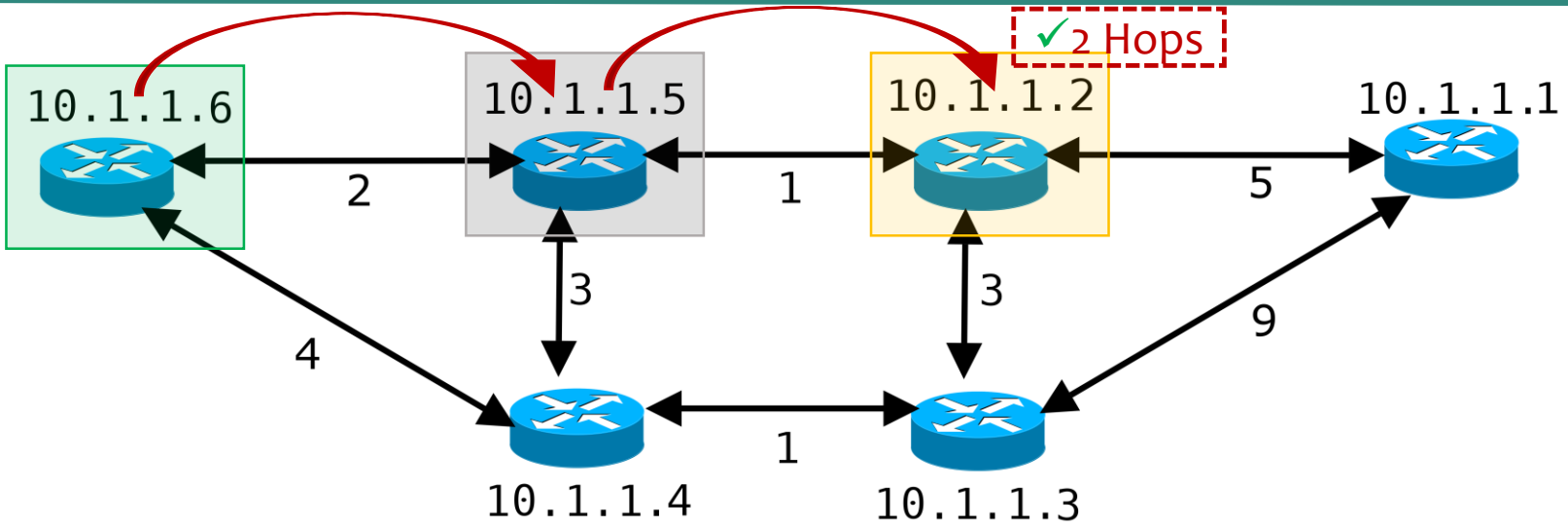
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5		

Question 2 (a)



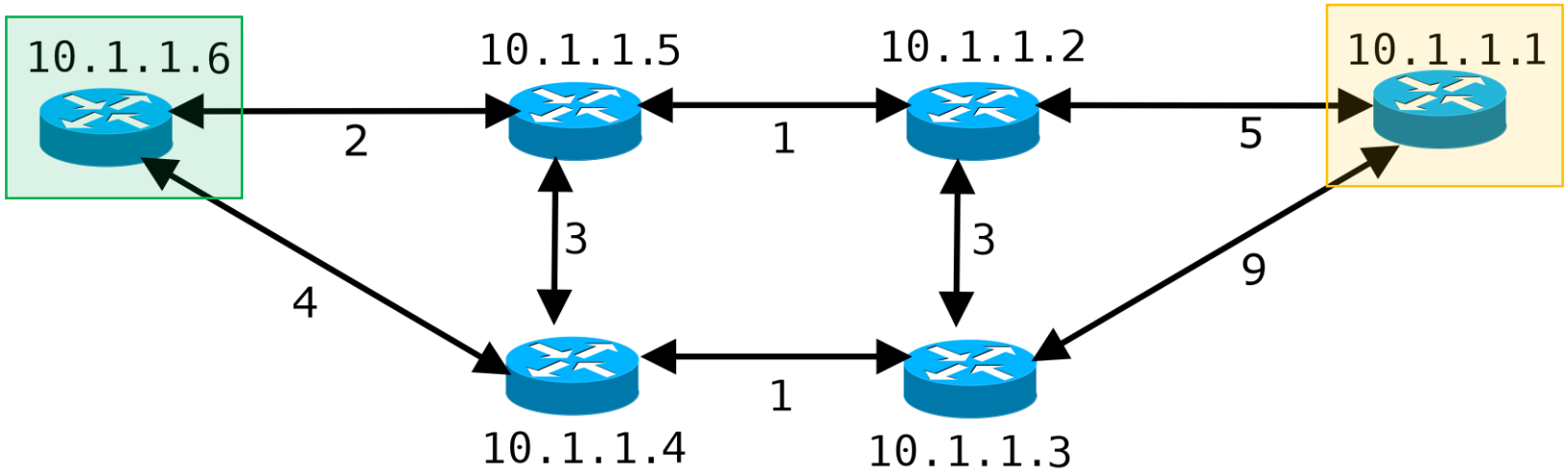
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	

Question 2 (a)



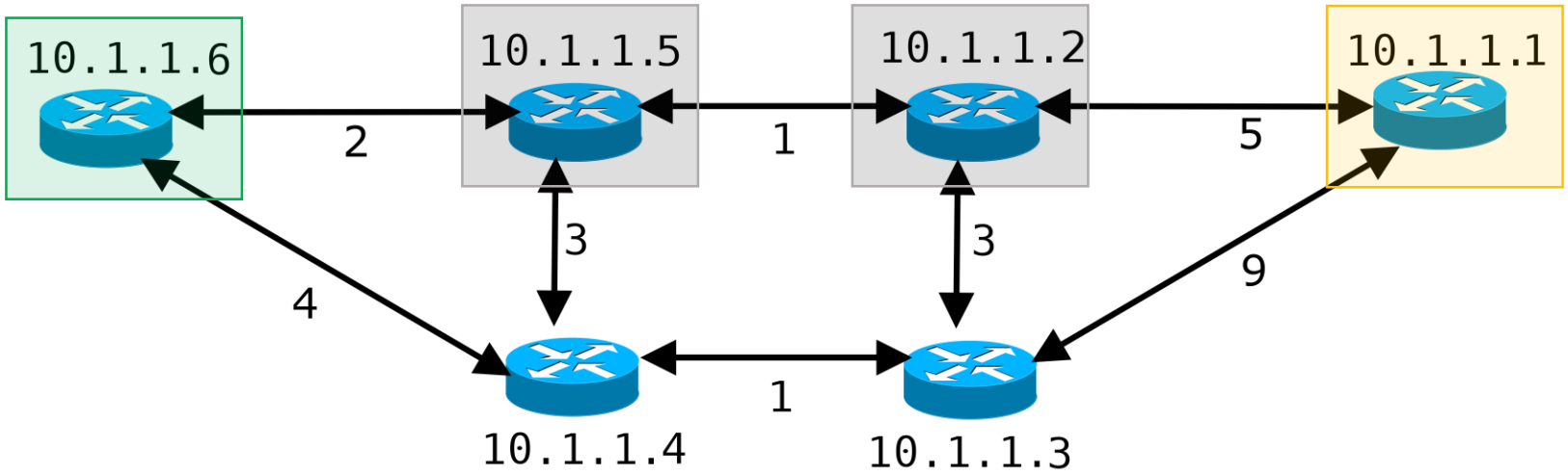
destination	next	#hops	(weight)	
10.1.1.5	10.1.1.5	1	(2)	
10.1.1.4	10.1.1.4	1	(4)	
10.1.1.3	10.1.1.4	2	(5)	
10.1.1.2	10.1.1.5	2	(3)	2+1

Question 2 (a)



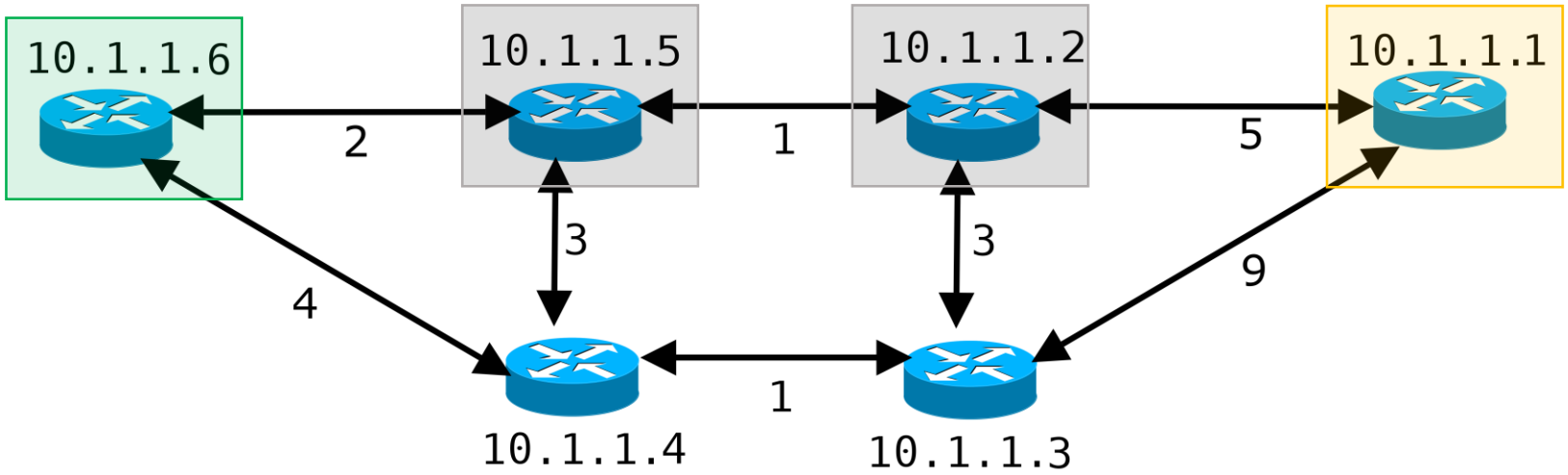
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	(3)
10.1.1.1			

Question 2 (a)



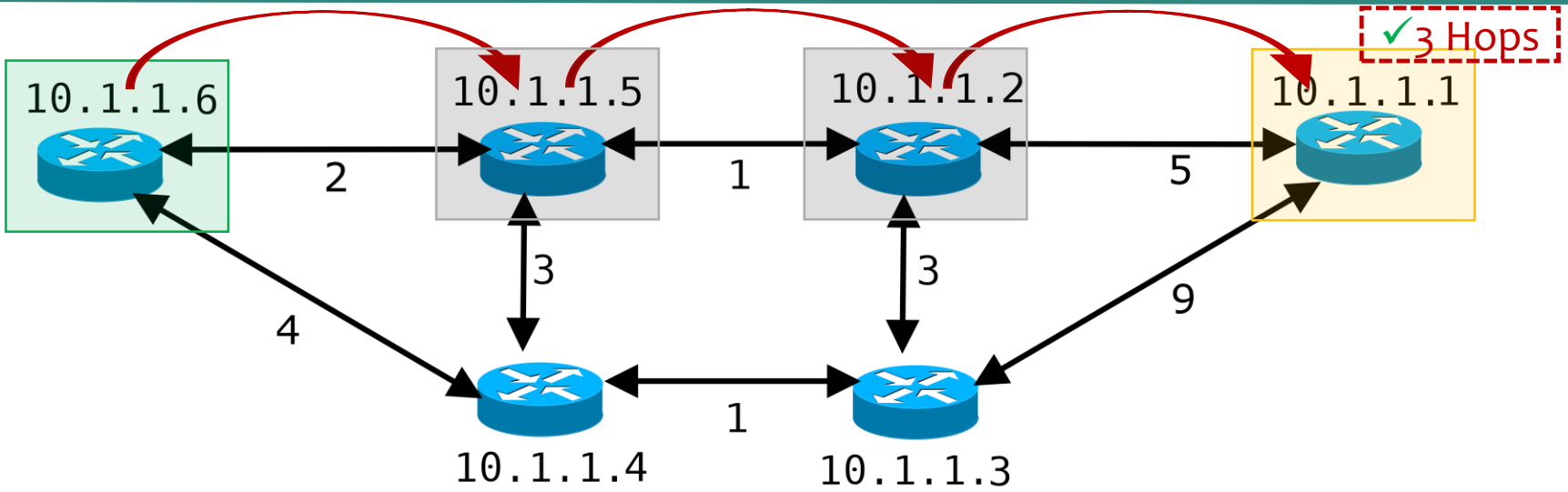
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	(3)
10.1.1.1	10.1.1.5		

Question 2 (a)



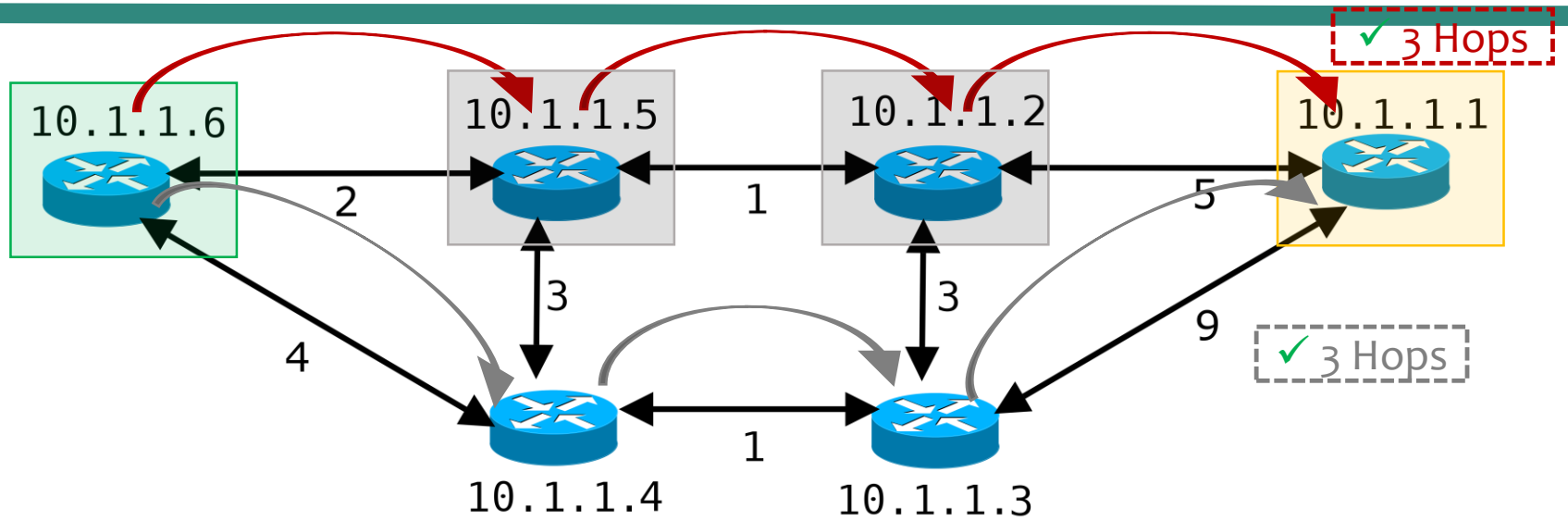
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	(3)
10.1.1.1	10.1.1.5		

Question 2 (a)



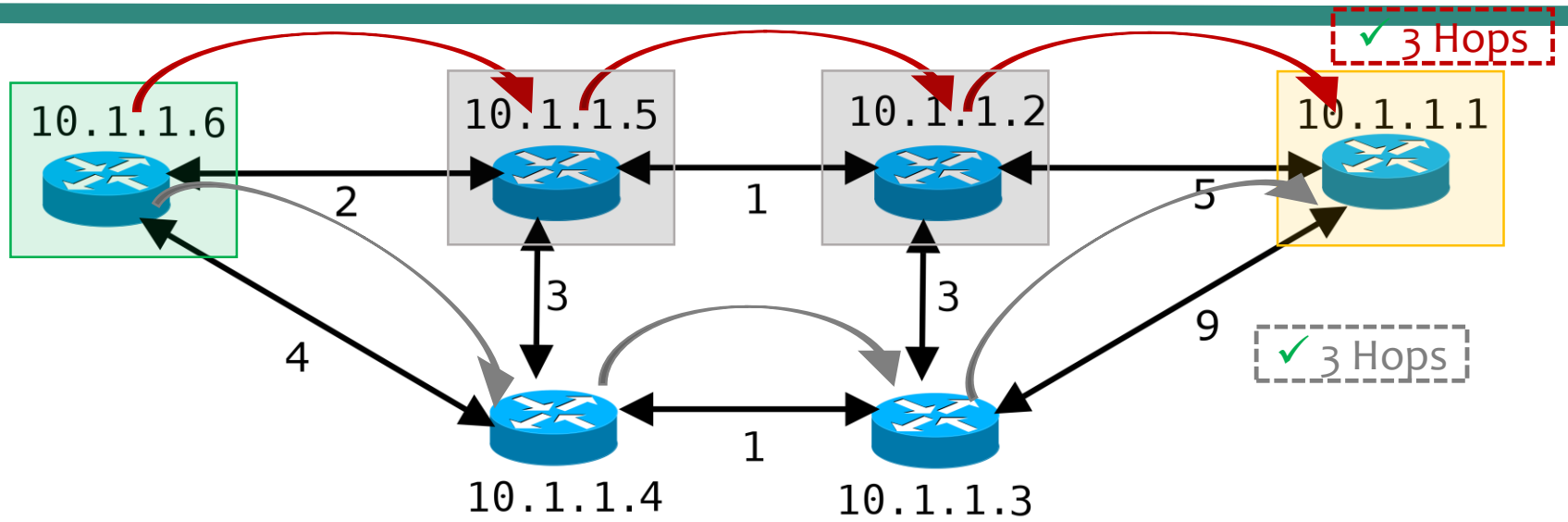
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	(3)
10.1.1.1	10.1.1.5	3	

Question 2 (a)



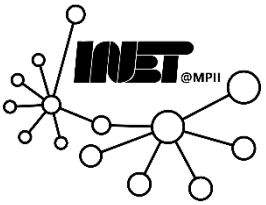
destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	(3)
10.1.1.1	10.1.1.5	3	

Question 2 (a)



destination	next	#hops	(weight)
10.1.1.5	10.1.1.5	1	(2)
10.1.1.4	10.1.1.4	1	(4)
10.1.1.3	10.1.1.4	2	(5)
10.1.1.2	10.1.1.5	2	(3)
10.1.1.1	10.1.1.5	3	(8)

Questions?



Question 2 (b)

Assuming that the link between the router 10.1.1.5 and 10.1.1.2 breaks (infinity= 16), show the evolution of the entry concerning router 10.1.1.1 for the routing tables of both 10.1.1.6 and 10.1.1.5 when using RIP. (Hint: what metrics are being used by RIP?)

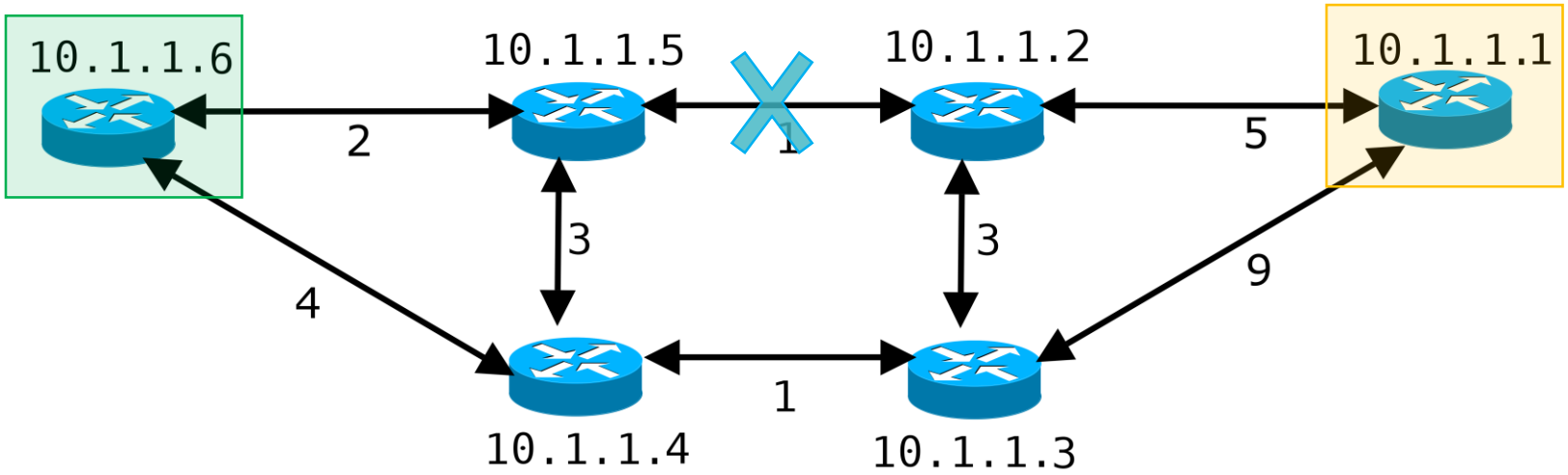
Use the notation from slide 15 of the RIP-OSPF slide set but restrict your illustration to the row that is changing.

Question 2 (b)

Assuming that the link between the router 10.1.1.5 and 10.1.1.2 breaks (infinity= 16), show the evolution of the entry concerning router 10.1.1.1 for the routing tables of both 10.1.1.6 and 10.1.1.5 when using RIP. (Hint: what metrics are being used by RIP?)

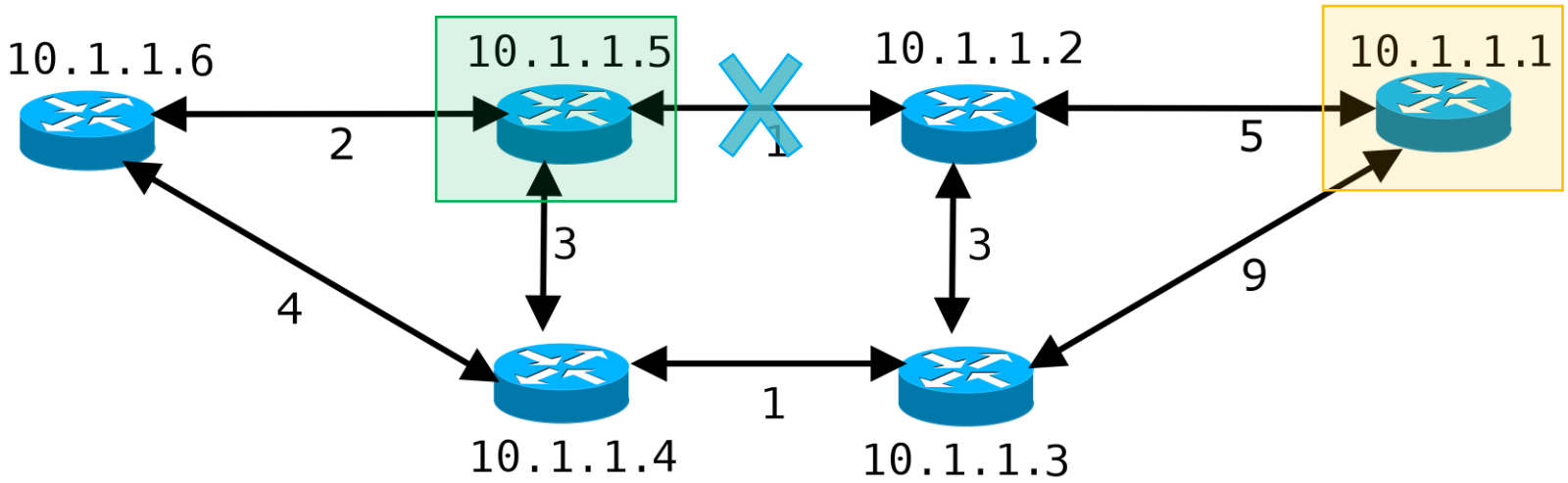
- Distance metric: *Hop count or number of hops (max = 15 hops)*

Question 2 (b)



status	update	destination	next	#hops
before link-failure	0	10.1.1.1	10.1.1.5	3
after link-failure	1	10.1.1.1	10.1.1.4	3

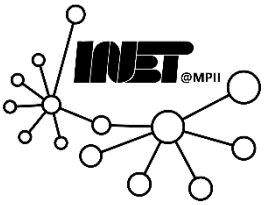
Question 2 (b)



status	update	destination	next	#hops
before link-failure	0	10.1.1.1	10.1.1.2	2
after link-failure	1	10.1.1.1	10.1.1.4	3

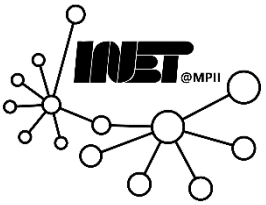
Question 2 (c)

In the final state of the previous part of the exercise: Is 10.1.1.1 reachable from 10.1.1.6? If yes, through which path? If not, can you explain why not?

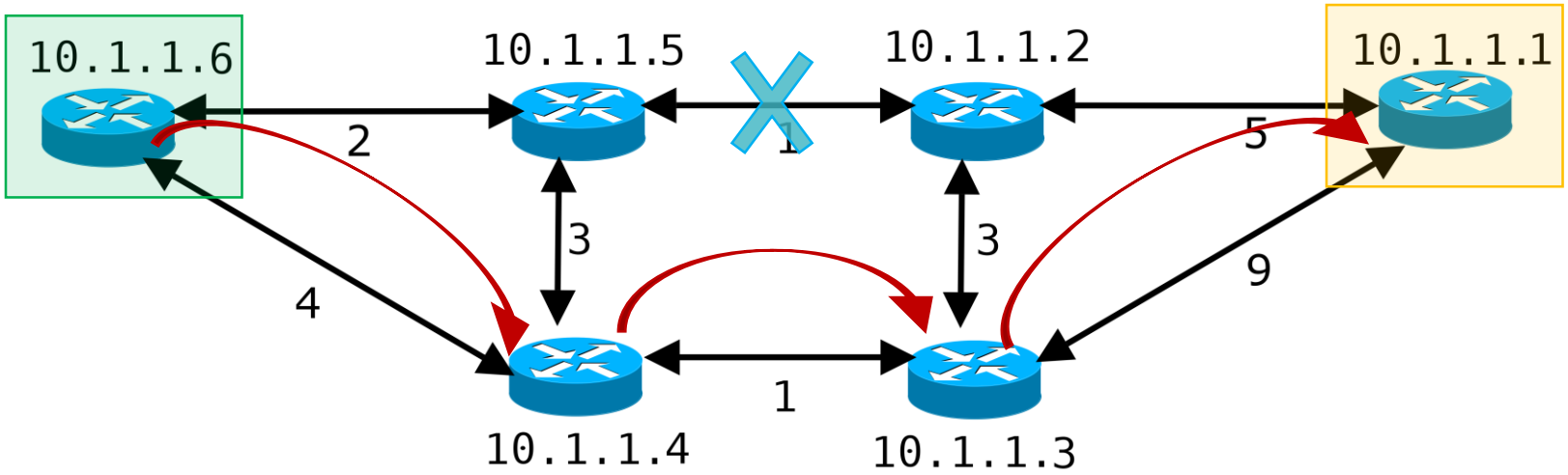


Question 2 (c)

In the final state of the previous part of the exercise: Is 10.1.1.1 reachable from 10.1.1.6? If yes, through which path? If not, can you explain why not?



Question 2 (c)



Yes: 10.1.1.6 –> 10.1.1.4 –> 10.1.1.3 –> 10.1.1.1

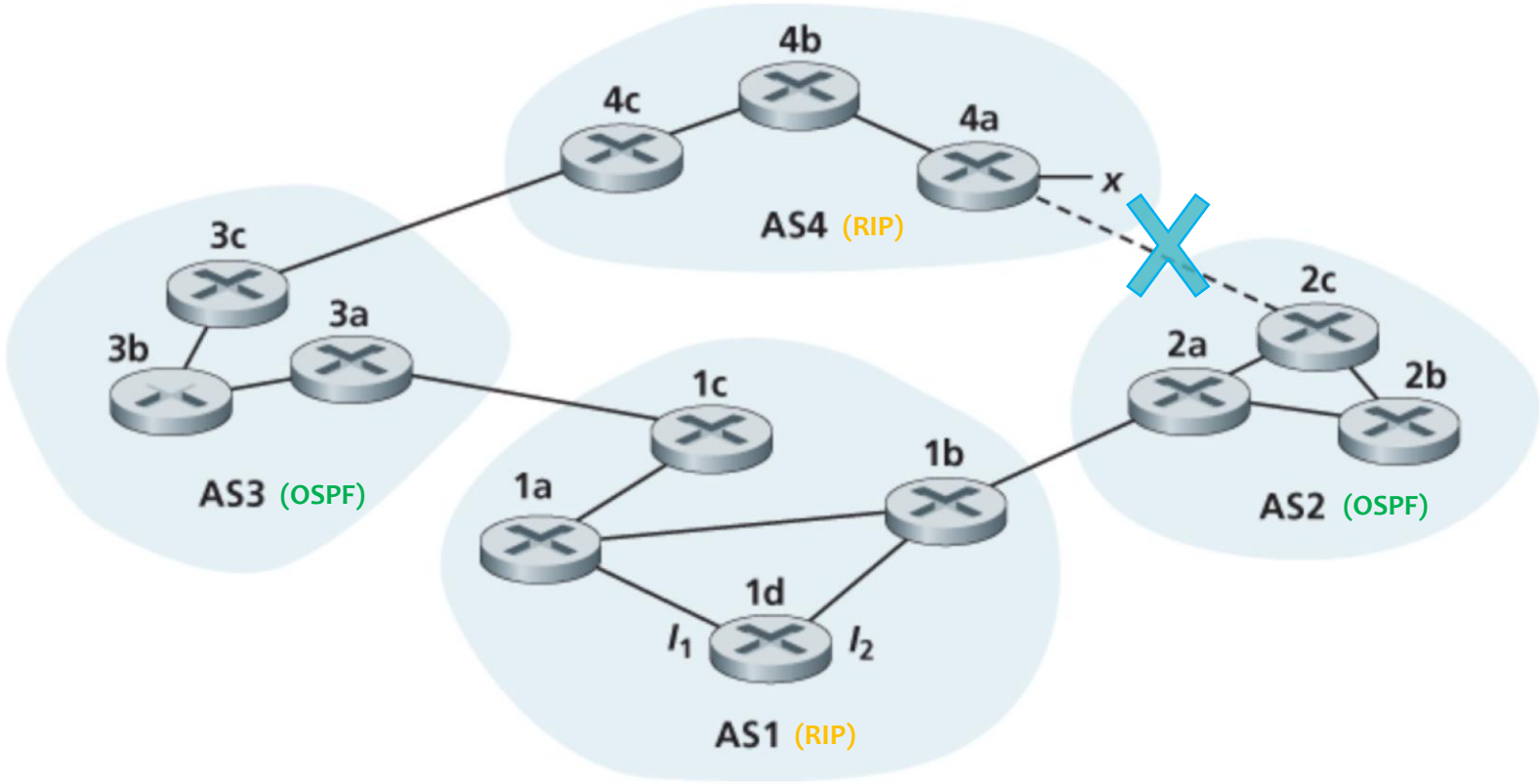
Question 3 (Inter-AS Routing)

Consider the following network topology. AS3 and AS2 are running OSPF for their intra-AS routing protocol. AS1 and AS4 are running RIP for their intra-AS routing protocol. eBGP and iBGP are used for the inter-AS routing protocol. Initially suppose there is no physical link between AS2 and AS4. x is an arbitrary IP prefix.

Question 3 (Inter-AS Routing)

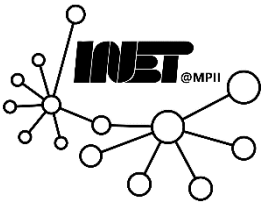
Consider the following network topology. AS3 and AS2 are running OSPF for their intra-AS routing protocol. AS1 and AS4 are running RIP for their intra-AS routing protocol. eBGP and iBGP are used for the inter-AS routing protocol. Initially suppose there is no physical link between AS2 and AS4. x is an arbitrary IP prefix.

Question 3 (Inter-AS Routing)



Question 3 (a)

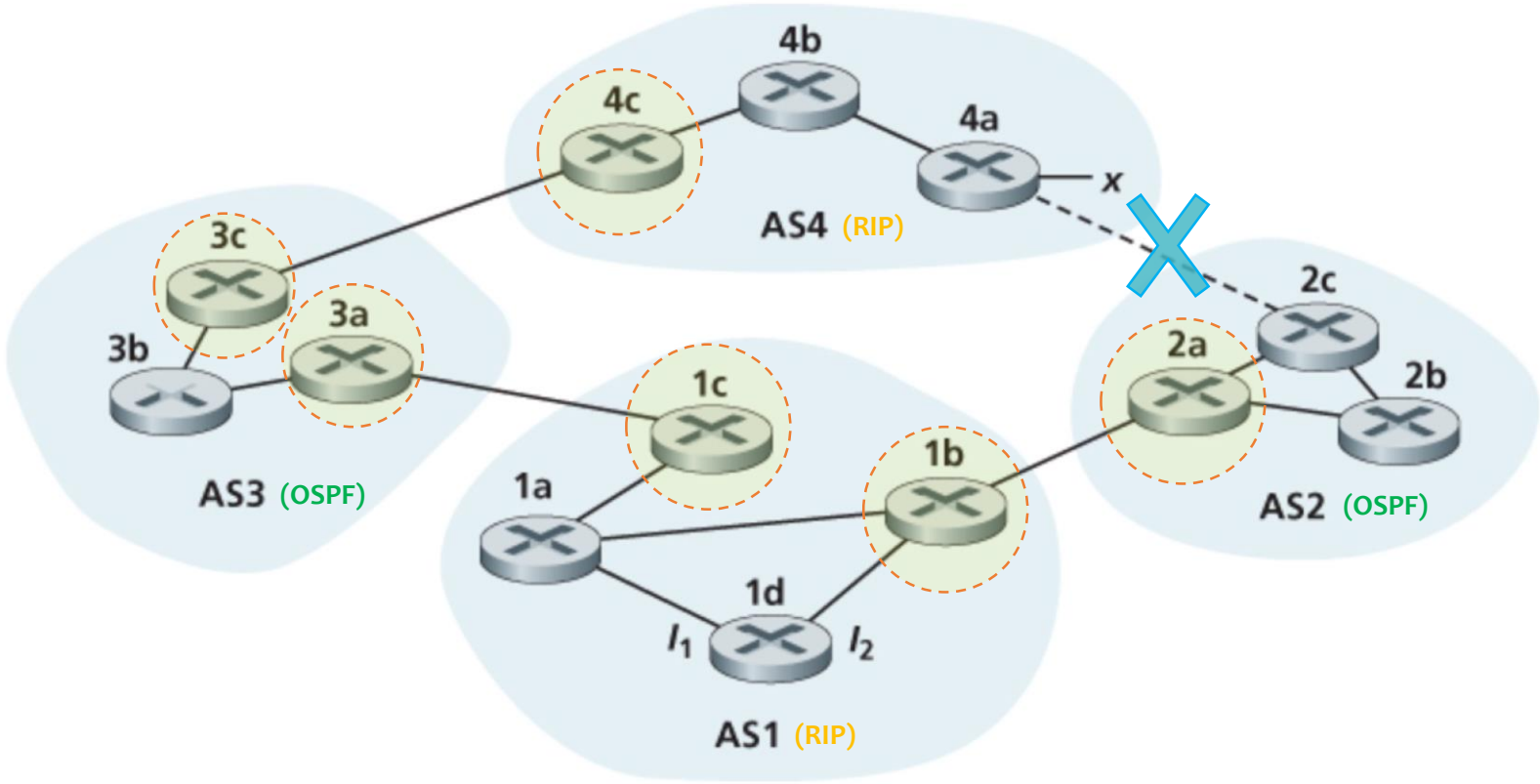
Please list the routers that need to run eBGP? Explain why in one sentence.



Question 3 (a)

Please list the routers that need to run eBGP? Explain why in one sentence.

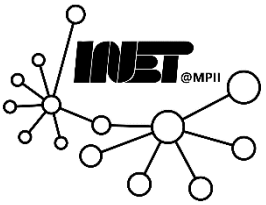
Question 3 (a)



Because these routers are connected to different Autonomous Systems (AS).

Question 3 (b)

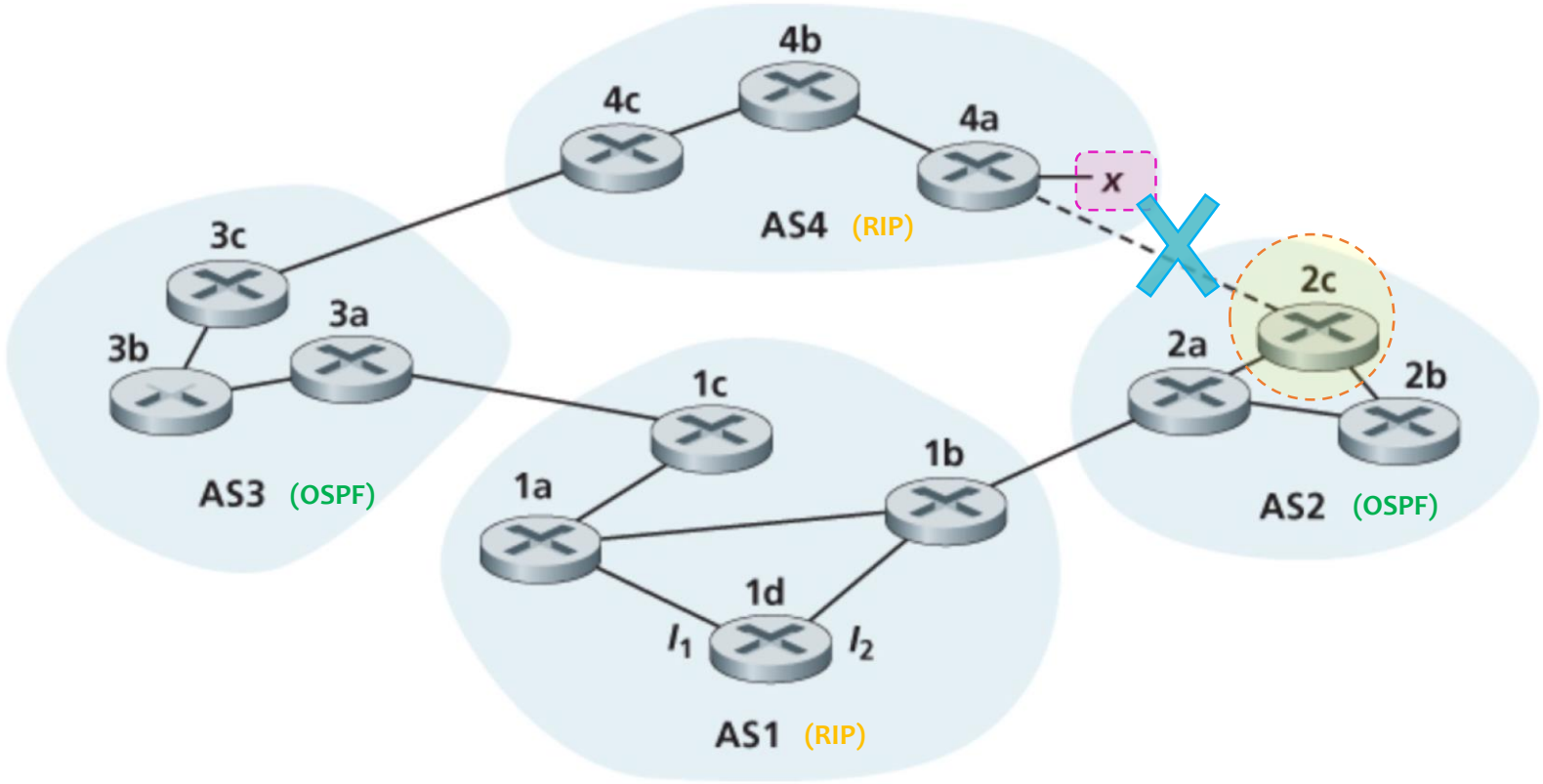
Router 2c will learn about prefix x via which protocol?
Explain why in one sentence.



Question 3 (b)

Router 2c will learn about prefix x via which protocol?
Explain why in one sentence.

Question 3 (b)



Option1: The router 2c learns about x via iBGP

Option2: Redistribution of eBGP routes to OSPF (not recommended)

Question 3 (c)

Now suppose that there is a physical link between AS2 and AS4, shown by the dotted line.

How can AS4 force AS1 to route all traffic destined to prefix x via AS3, and use the AS2 as an alternative path only if the link between AS4-AS3 is down? Explain at least two reasons why would AS4 do that?

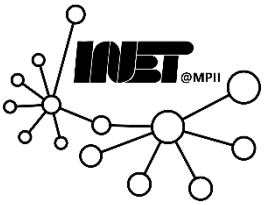
Hint: learn about AS path prepending.

Question 3 (c)

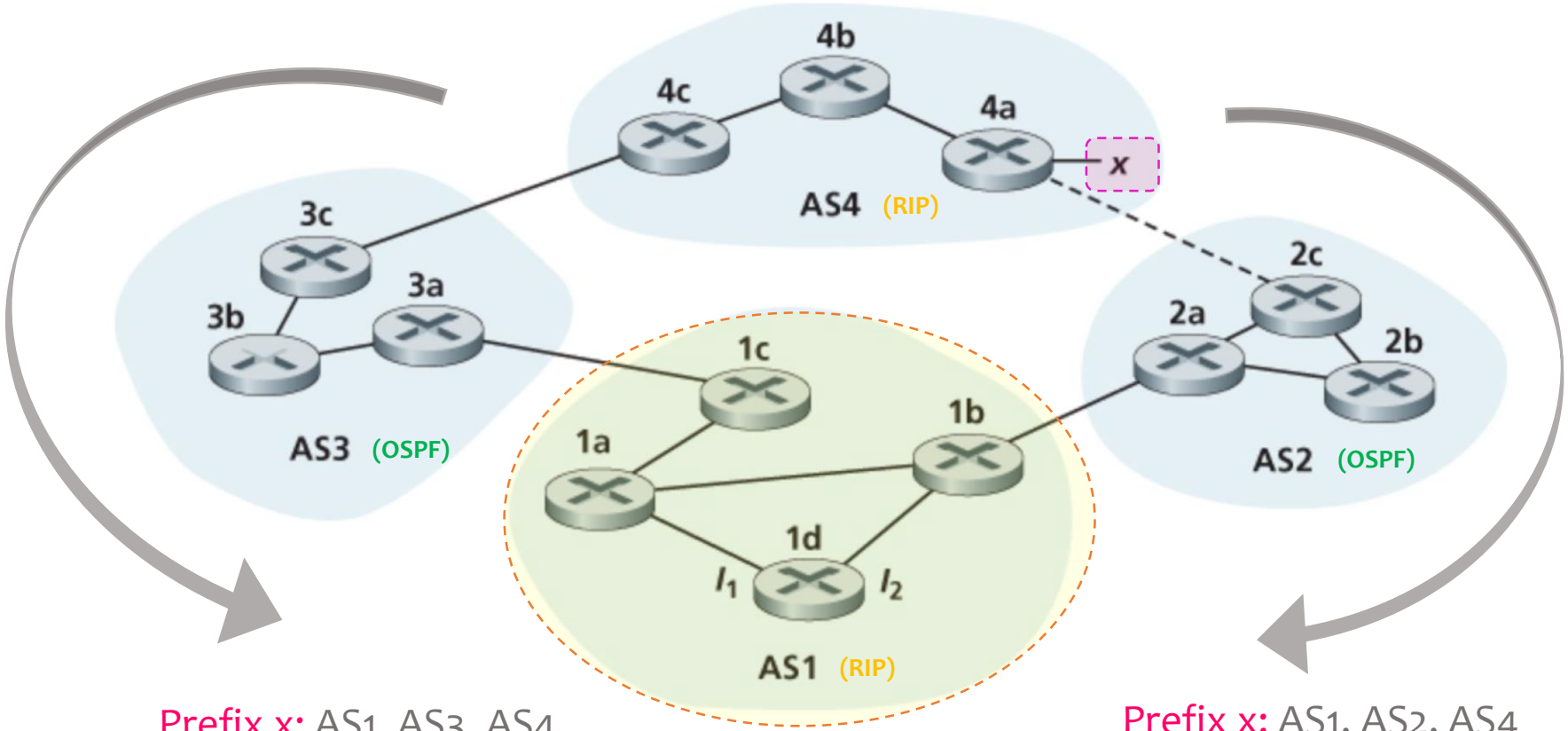
Now suppose that there is a physical link between AS2 and AS4, shown by the dotted line.

How can AS4 force AS1 to route all traffic destined to prefix x via AS3, and use the AS2 as an alternative path only if the link between AS4-AS3 is down? Explain at least two reasons why would AS4 do that?

Hint: learn about AS path prepending.



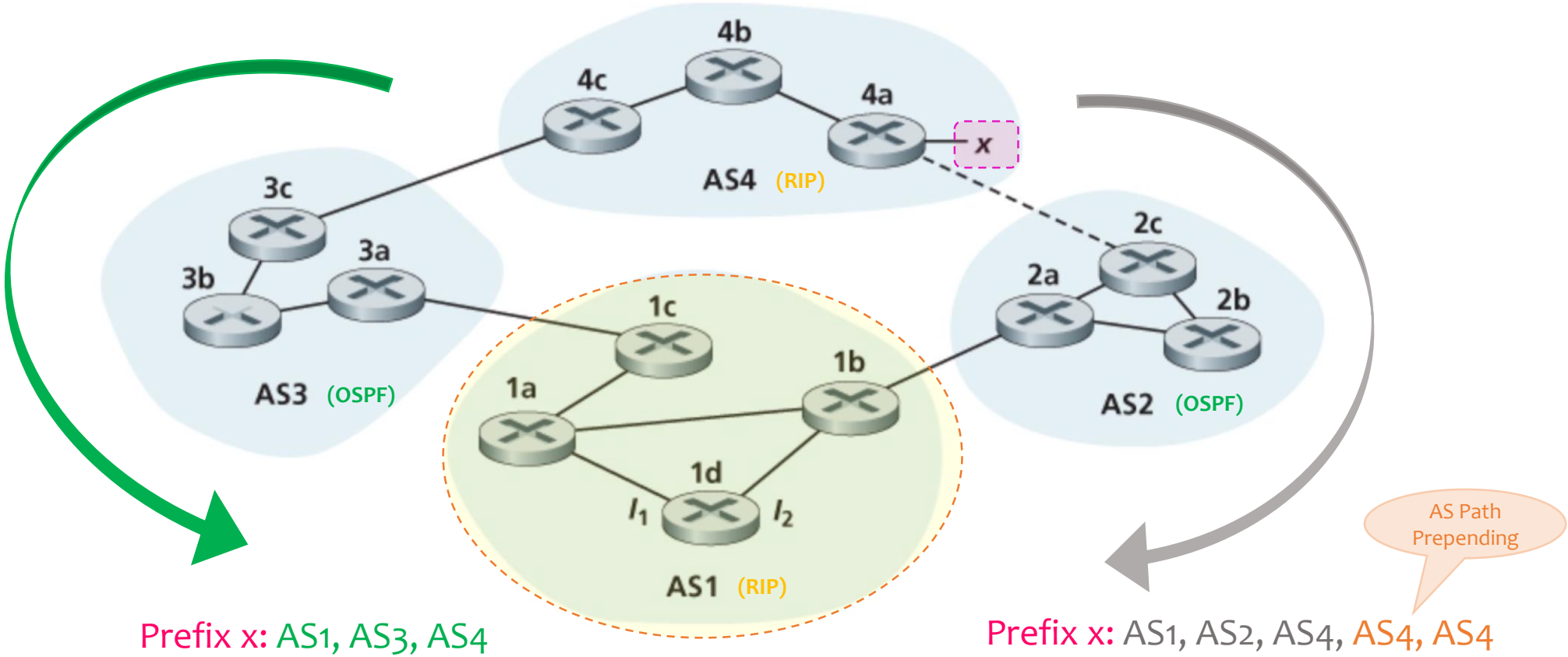
Question 3 (c)



Prefix x: AS1, AS3, AS4

Prefix x: AS1, AS2, AS4

Question 3 (c)



Prefix x: AS1, AS3, AS4

Prefix x: AS1, AS2, AS4, AS4, AS4

Question 3 (c)

Reasons:

- Financial reasons
- Quality reasons
- Traffic Engineering

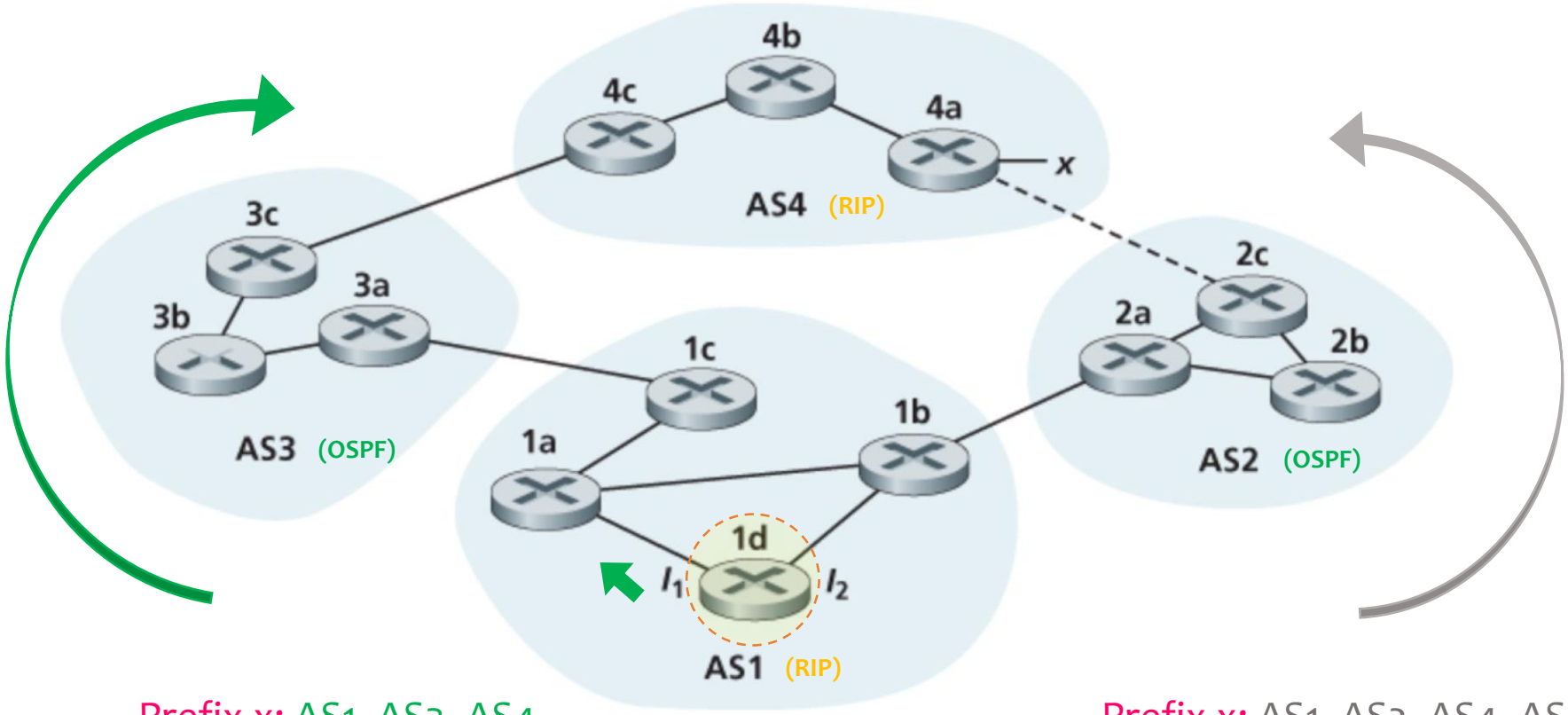
Question 3 (d)

Suppose the policy explained in part (c) is still applied by AS4, and router 1d learns that prefix x is accessible via both AS2 and AS3. Will router 1d use I_1 or I_2 for traffic destined to prefix x? Justify your answer. I is abbreviation for Interface.

Question 3 (d)

Suppose the policy explained in part (c) is still applied by AS4, and router 1d learns that prefix x is accessible via both AS2 and AS3. Will router 1d use I_1 or I_2 for traffic destined to prefix x? Justify your answer. I is abbreviation for Interface.

Question 3 (d)



Prefix x: AS1, AS3, AS4

Prefix x: AS1, AS2, AS4, AS4, AS4

Question 3 (e)

Hereafter, consider the following BGP relations apply between ASes:

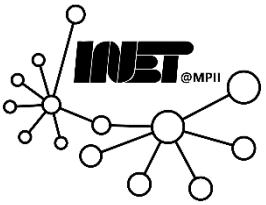
AS4 is the customer of AS2

AS3 is the customer of AS1

AS1 and AS2 are peering partners

AS3 and AS4 are peering partners

Will AS2 advertise prefix x to AS1? Justify your answer.



Question 3 (e)

Hereafter, consider the following BGP relations apply between ASes:

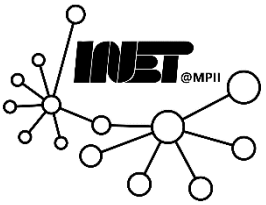
AS4 is the customer of AS2

AS3 is the customer of AS1

AS1 and AS2 are peering partners

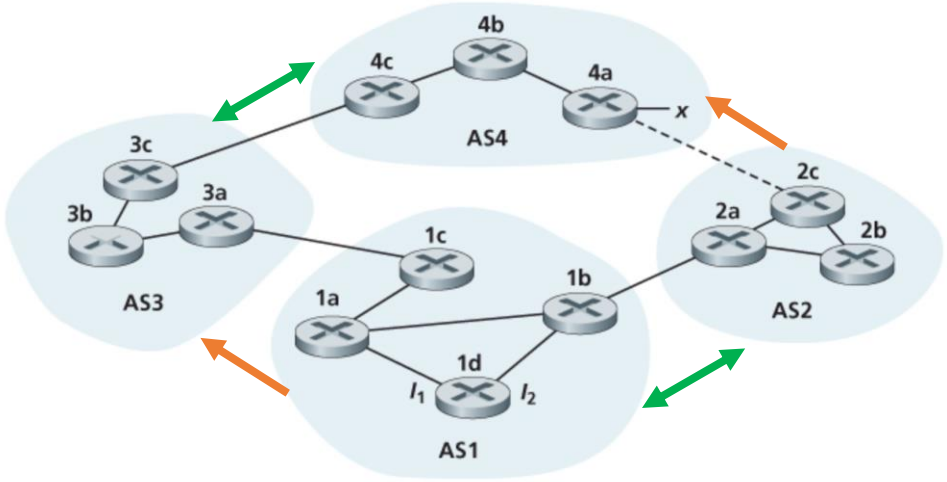
AS3 and AS4 are peering partners

Will AS2 advertise prefix x to AS1? Justify your answer.



Question 3 (e)

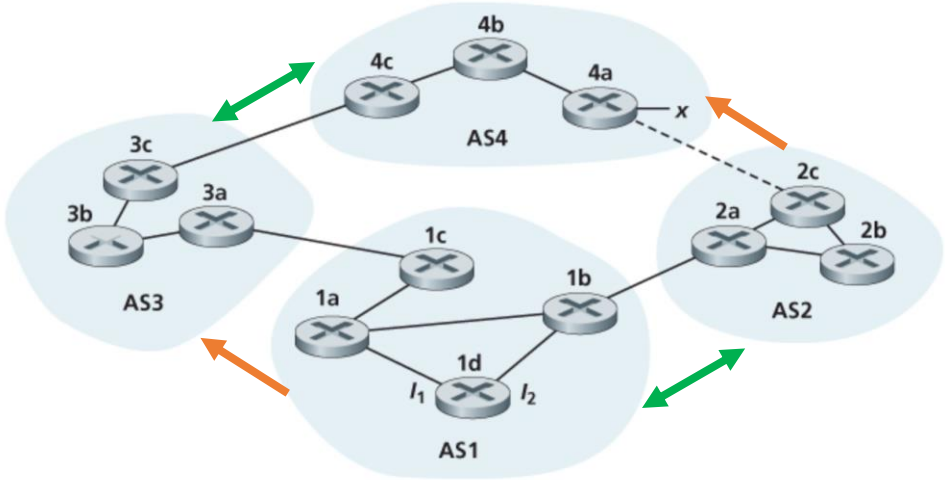
- AS4 is the customer of AS2
- AS3 is the customer of AS1
- AS1 and AS2 are peering partners
- AS3 and AS4 are peering partners



Question 3 (e)

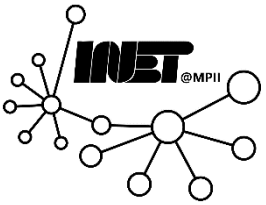
Answer:

Yes, because AS2 is the provider for AS4, which has prefix x, so it will advertise the customer routes to AS1 which is peering partner.



Question 3 (f)

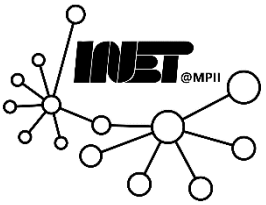
Will AS4 advertise the routers learned from AS2 to AS3? Justify your answer.



Question 3 (f)

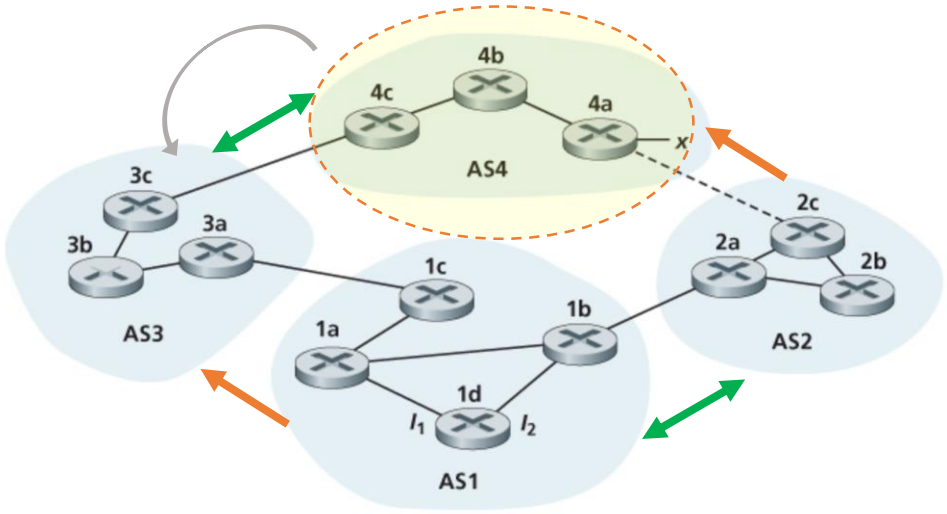
Will AS4 advertise the routes learned from AS2 to AS3?

Justify your answer.



Question 3 (f)

☒ There is no benefit for AS4 advertise the routes learned from AS2 to AS3.



Questions?