# Networks: Tutorial 09

Shinnazar Seytnazarov, PhD

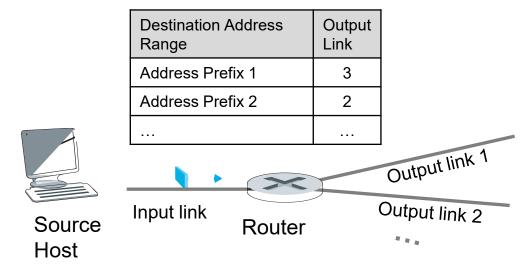
Innopolis University

s.seytnazarov@innopolis.ru

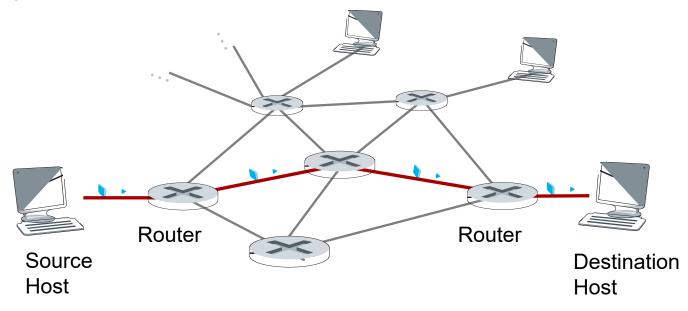
February 14, 2022

### Recap

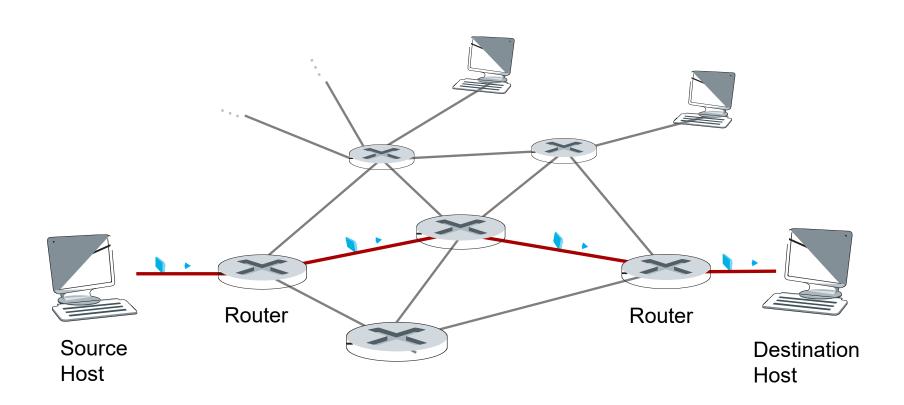
**Forwarding** – a local action at a router level (to forward packets to some output port)



**Routing** – a network-wide action, to compute the least-cost path



Routing problem – to compute the least-cost path between network hosts Routing algorithm – the solution to this problem



Routing algorithms represent a network as a graph:

- graph nodes are routers, and
- edges are the communication links between routers

Characteristic	Description
Global / Decentralized	
Static / Dynamic	
Sensitivity to Network Load	

Characteristic	Description
Global / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>
Static / Dynamic	
Sensitivity to Network Load	

Characteristic	Description		
Clobal / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>		
Global / Decentralized	<ul> <li>Decentralized (or distributed), e.g. the dynamic-vector alg.:</li> <li>A distributed manner of computation, that is each node contributes to the final result;</li> <li>Each node communicates to its neighbors only</li> </ul>		
Static / Dynamic			
Sensitivity to Network Load			

Characteristic	Description		
Global / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>		
	Decentralized (or distributed), e.g. the dynamic-vector alg.:     A distributed manner of computation, that is each node contributes to the final result;     Each node communicates to its neighbors only		
Static / Dynamic	<b>Static</b> – routes are recomputed very rarely over time, and usually by hand (changes in a forwarding table);		
Sensitivity to Network Load			

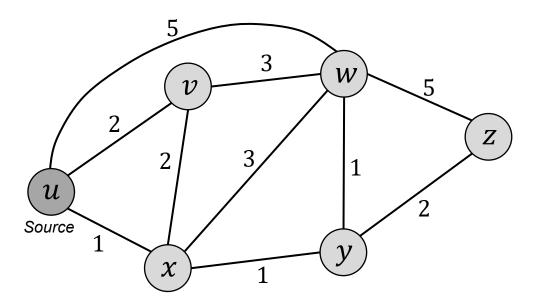
Characteristic	Description		
Clobal / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>		
Global / Decentralized	<ul> <li>Decentralized (or distributed), e.g. the dynamic-vector alg.:</li> <li>A distributed manner of computation, that is each node contributes to the final result;</li> <li>Each node communicates to its neighbors only</li> </ul>		
Static / Dynamic	<b>Static</b> – routes are recomputed very rarely over time, and usually by hand (changes in a forwarding table);		
Static / Dynamic	<b>Dynamic</b> – routes are updated frequently, to reflect changes in a network load (congestion) or topology		
Sensitivity to Network Load			

Characteristic	Description		
Clobal / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>		
Global / Decentralized	<ul> <li>Decentralized (or distributed), e.g. the dynamic-vector alg.:</li> <li>A distributed manner of computation, that is each node contributes to the final result;</li> <li>Each node communicates to its neighbors only</li> </ul>		
Static / Dynamic	<b>Static</b> – routes are recomputed very rarely over time, and usually by hand (changes in a forwarding table);		
Static / Dynamic	<b>Dynamic</b> – routes are updated frequently, to reflect changes in a network load (congestion) or topology		
Sensitivity to Network Load	Load-sensitive – link costs may change dynamically, mainly to reflect network congestion;		

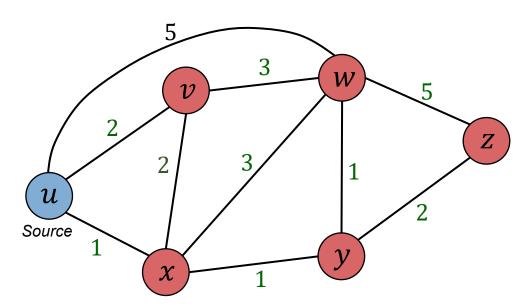
Characteristic	Description		
Global / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>		
	<ul> <li>Decentralized (or distributed), e.g. the dynamic-vector alg.:</li> <li>A distributed manner of computation, that is each node contributes to the final result;</li> <li>Each node communicates to its neighbors only</li> </ul>		
Static / Dynamic	<b>Static</b> – routes are recomputed very rarely over time, and usually by hand (changes in a forwarding table);		
	<b>Dynamic</b> – routes are updated frequently, to reflect changes in a network load (congestion) or topology		
	<b>Load-sensitive</b> – link costs may change dynamically, mainly to reflect network congestion;		
Sensitivity to Network Load	<b>Load-insensitive</b> – no dynamic change of links costs (most of the routing algorithms)		

Characteristic	Description		
Global / Decentralized	<ul> <li>Global (or centralized), e.g. the Link-state alg.:</li> <li>Each node knows the topology of an entire network;</li> <li>Each node communicates to any other network node (via broadcast messages)</li> </ul>		
	<ul> <li>Decentralized (or distributed), e.g. the distance vector alg.:</li> <li>A distributed manner of computation, that is each node contributes to the final result;</li> <li>Each node communicates to its neighbors only</li> </ul>		
Static / Dynamic	<b>Static</b> – routes are recomputed very rarely over time, and usually by hand (changes in a forwarding table);		
	<b>Dynamic</b> – routes are updated frequently, to reflect changes in a network load (congestion) or topology		
	<b>Load-sensitive</b> – link costs may change dynamically, mainly to reflect network congestion;		
Sensitivity to Network Load	<b>Load-insensitive</b> – no dynamic change of links costs (most of the routing algorithms)		

Problem: To find the shortest paths from the source node u to destination nodes v, x, w, y, z



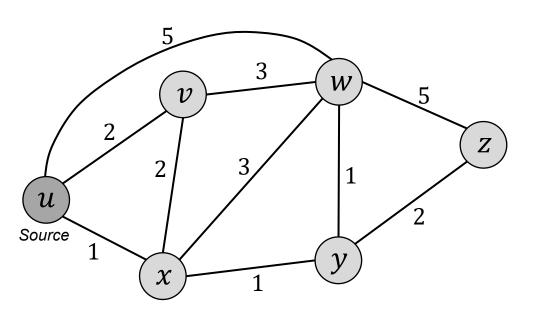
Problem: To find the shortest paths from the source node u to destination nodes v, x, w, y, z



#### Given:

<u> </u>				
u	The source node			
v, x, w, y, z	Destination nodes			
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)			

Problem: To find the shortest paths from the source node u to destination nodes v, x, w, y, z



#### Steps of Dijkstra's algorithm:

N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)

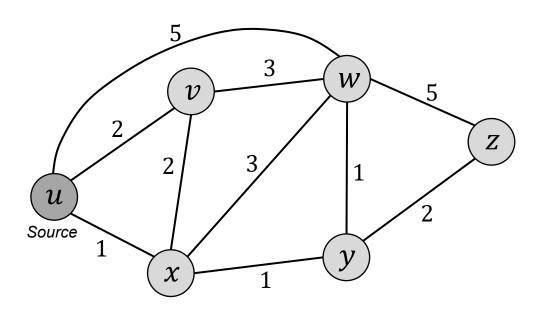
#### Given:

GIVEII.			
u	The source node		
v, x, w, y, z	Destination nodes		
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)		

#### Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0						

Gi

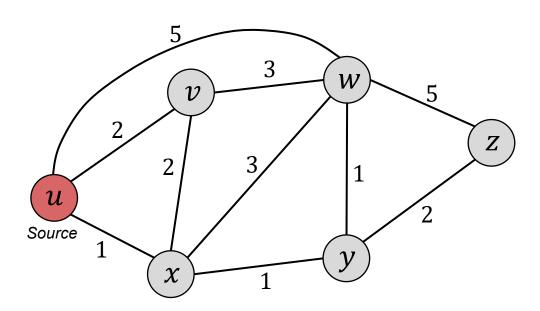
Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

	\ / /
1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z),p(z)
0	{ <b>u</b> }					

Gi

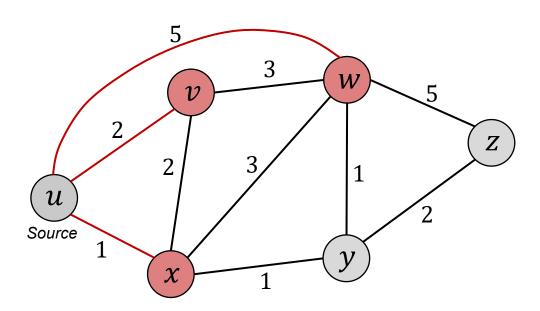
Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	<b>2</b> , <b>u</b>	<b>5</b> , <b>u</b>	<b>1</b> , <b>u</b>		

Gi

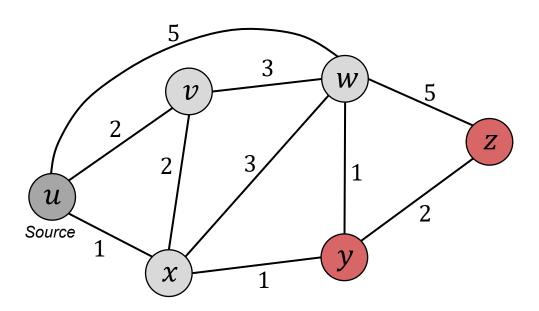
Siven:	
и	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8

Gi

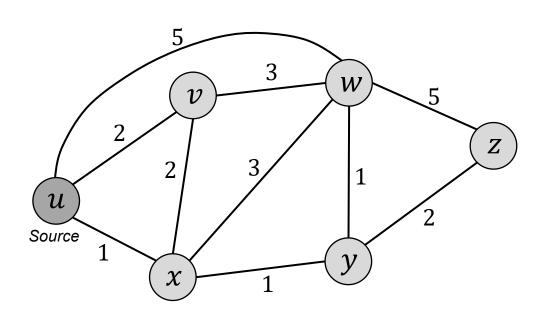
Given:				
u	The source node			
v, x, w, y, z	Destination nodes			
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)			

Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

Problem: To find the shortest paths from the source node *u* to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8

Given:

c(i,k)

и

The source node **Destination nodes** v, x, w, y, z

Link cost between nodes *i* and *k* 

(specified near each graph edge)

Algorithm variables:

A subset of nodes, to which optimal paths N'have been found by a given iteration The cost of the least-cost path to node i, D(i)found at this iteration of the algorithm

The previous node in the currently found p(i)least-cost path from node u to i

Initialization (Step 0):

 $N' \leftarrow \{u\}$ 

If i – the neighbour of u then 3:

 $D(i) \leftarrow c(u, i)$ 

For all nodes  $i \in \{v, x, w, y, z\}$  do

Else  $D(i) \leftarrow \infty$ 5:

Iterations:

4:

5:

Loop

**Find**  $i \notin N'$  such that D(i) is minimal 2:

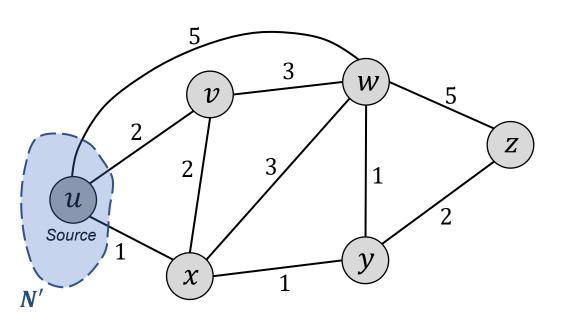
3: add i to N'

**For** each neighbour k of i, with  $k \notin N'$  do 4:

 $D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$ 

Until N' = N6:

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8

Given:	iven:		
u	The source node		
v, x, w, y, z	Destination nodes		
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)		

#### Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found

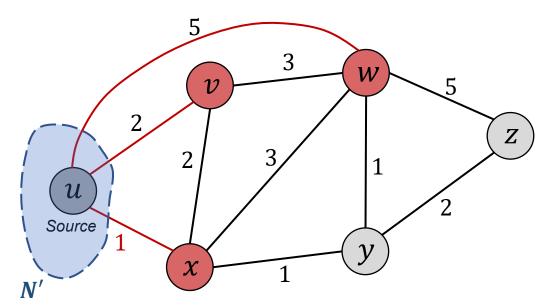
#### Initialization (Step 0):

	\ <i>\ \ \</i>
1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

#### Iterations:

1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Candidate nodes are highlighted in red

Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8

#### Given:

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

#### Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

#### Initialization (Step 0):

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

#### Iterations:

5:

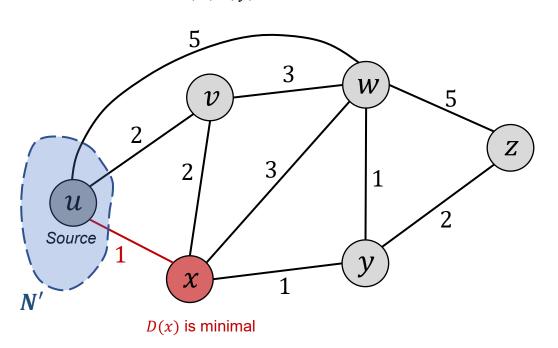
6:

Until N' = N

1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do

 $D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$ 

Problem: To find the shortest paths from the source node u to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(	(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }		2, <i>u</i>	5, u	1, u	8	8

Given: u The source node v, x, w, y, z Destination nodes

Link cost between nodes i and k

#### Algorithm variables:

c(i,k)

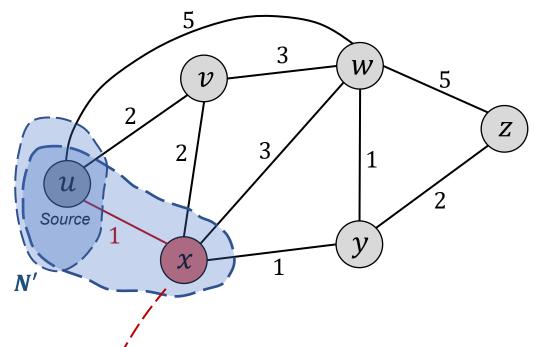
N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found

(specified near each graph edge)

11 11 (1)	
1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$

3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$
Iter	rations:
1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

Problem: To find the shortest paths from the source node u to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'/	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	$\{u\}$	2, <i>u</i>	5, u	1, u	8	8
1	$\{u,x\}$					

Given

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found

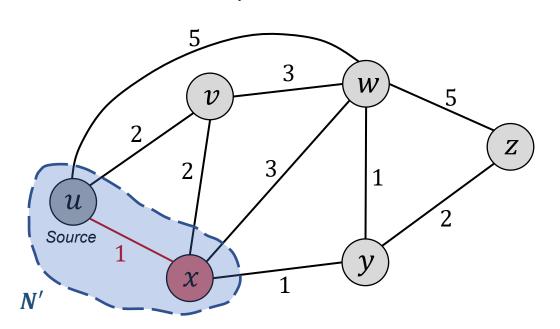
Initialization (Step 0):

$N' \leftarrow \{u\}$
For all nodes $i \in \{v, x, w, y, z\}$ do
If $i$ – the neighbour of $u$ then
$D(i) \leftarrow c(u,i)$
Else $D(i) \leftarrow \infty$

Iterations:

1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8
1	$\{u,x\}$					

Gi۱

Given:			
u	The source node		
v, x, w, y, z	Destination nodes		
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)		

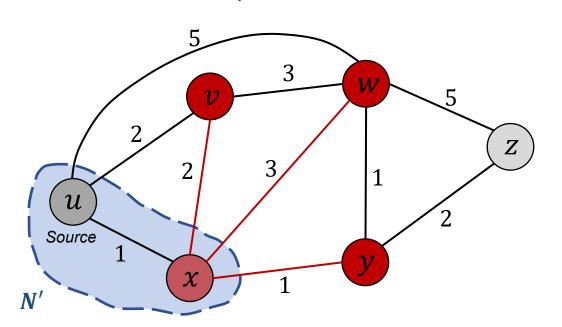
Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$

Iter	Iterations:				
1:	Loop				
2:	Find $i \notin N'$ such that $D(i)$ is minimal				
3:	add $i$ to $N'$				
4:	<b>For</b> each neighbour $k$ of $i$ , with $k \notin N'$ do				
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$				
6:	Until $N' = N$				

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8
1	$\{u,x\}$					

Giv

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

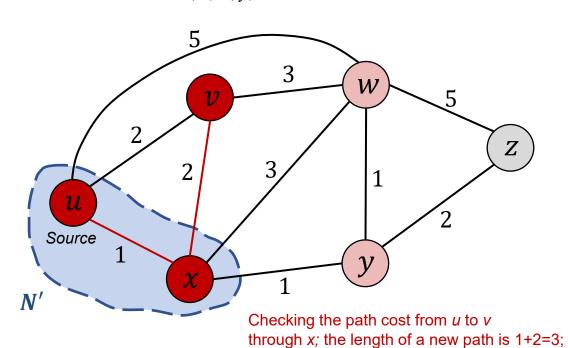
Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
<b>4</b> :	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

	3					
4:	$D(i) \leftarrow c(u,i)$					
5:	Else $D(i) \leftarrow \infty$					
Iter	rations:					
1:	Loop					
2:	Find $i \notin N'$ such that $D(i)$ is minimal					
3:	add $i$ to $N'$					
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do					
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$					
6:	Until $N' = N$					

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	otopo or Dijitoti a o aligoritanini							
	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z), p(z)		
0	{ <i>u</i> }	2, u	5, u	1, u	8	8		
1	$\{u,x\}$	<b>2</b> , <b>u</b>						

keeping the old path as the least-cost

Gi

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

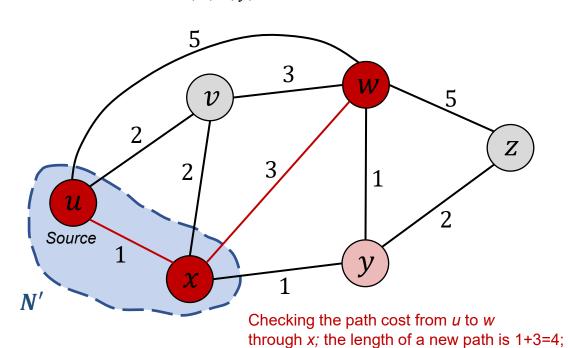
Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

11 11 (1)		
1:	$N' \leftarrow \{u\}$	
2:	For all nodes $i \in \{v, x, w, y, z\}$ do	
3:	If $i$ – the neighbour of $u$ then	
4:	$D(i) \leftarrow c(u,i)$	
5:	Else $D(i) \leftarrow \infty$	

۷.	For all flodes $t \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$
Iter	rations:
1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, <i>u</i>	8	$\infty$
1	$\{u,x\}$	2, u	<b>4</b> , <i>x</i>			

updating the least-cost path

Gi

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

Algorithm variables:

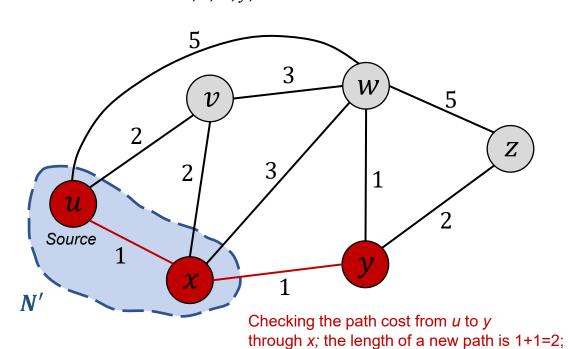
N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1111111	
1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$
Iter	rations:
1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

### Example of Using Link-State Dijkstra's Algorithm

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	$\infty$
1	$\{u,x\}$	2, u	<b>4</b> , <i>x</i>		2, <i>x</i>	

updating the least-cost path

#### Gi۱

Given:				
u	The source node			
v, x, w, y, z	Destination nodes			
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)			

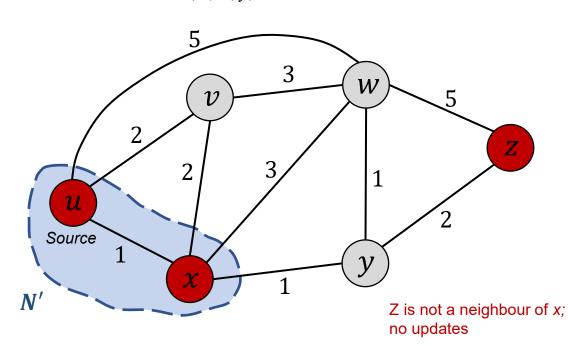
#### Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$

	ii v are neighboar or waren
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$
Iter	rations:
1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8
1	$\{u,x\}$	2, <i>u</i>	<b>4</b> , <i>x</i>		2, <i>x</i>	8

Given:

u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

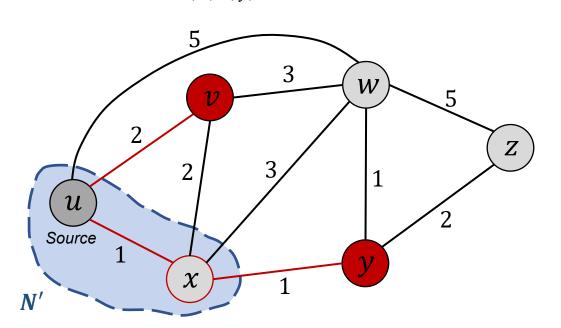
Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node $i$ , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
<i>5:</i>	Else $D(i) \leftarrow \infty$

Iterations:		
1:	Loop	
2:	Find $i \notin N'$ such that $D(i)$ is minimal	
3:	add $i$ to $N'$	
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do	
<b>5</b> :	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$	
6.	Until $N' = N$	

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z),p(z)
0	{ <i>u</i> }	2, <i>u</i>	5, u	1, <i>u</i>	8	8
1	$\{u,x\}$	2, <i>u</i>	4, <i>x</i>		2, x	8
2						
3						
4						
5						

Gi۱

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

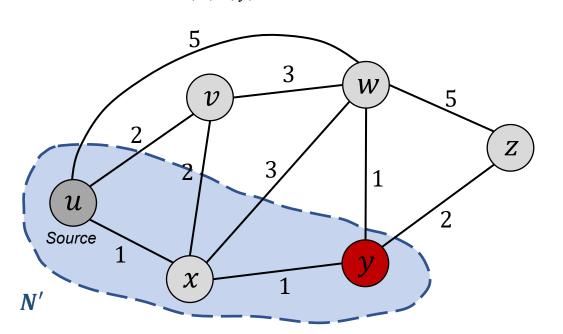
Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

5:	Else $D(i) \leftarrow \infty$
Iter	ations:
1:	Loop
2:	Find $i \notin N'$ such that $D(i)$ is minimal
3:	add $i$ to $N'$
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$
6:	Until $N' = N$

Problem: To find the shortest paths from the source node uto destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	{ <i>u</i> }	2, u	5, u	1, u	8	8
1	$\{u,x\}$	2, <i>u</i>	4, <i>x</i>		2, <i>x</i>	8
2	$\{u,x,y\}$					
ഗ						
4						
5						

Gi۱

Given:	
u	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

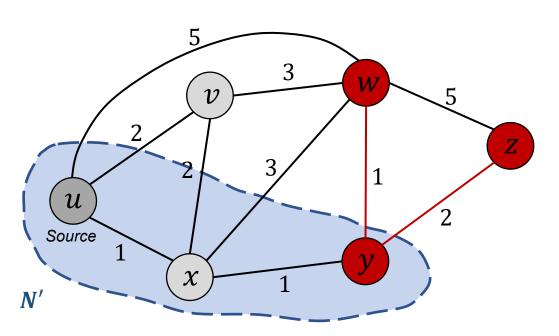
Initialization (Step 0):

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u, i)$
5:	Else $D(i) \leftarrow \infty$

Iterations:

1101	norations.				
1:	Loop				
2:	Find $i \notin N'$ such that $D(i)$ is minimal				
3:	add $i$ to $N'$				
4:	For each neighbour $k$ of $i$ , with $k \notin N'$ do				
5:	$D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$				
6:	Until $N' = N$				

Problem: To find the shortest paths from the source node  $\boldsymbol{u}$ to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y),p(y)	D(z), p(z)
0	{ <i>u</i> }	2, u	5, u	1, <i>u</i>	8	8
1	$\{u,x\}$	2, u	4, <i>x</i>		2, <i>x</i>	8
2	$\{u, x, y\}$	2, <i>y</i>	3, <i>y</i>			4, <i>y</i>
3						
4						
5						

Giv

Given:	
и	The source node
v, x, w, y, z	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each graph edge)

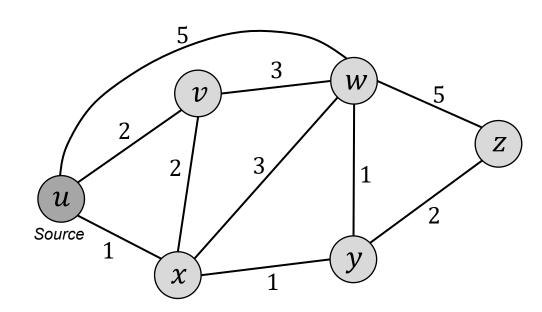
Algorithm variables:

N'	A subset of nodes, to which optimal paths have been found by a given iteration
D(i)	The cost of the least-cost path to node <i>i</i> , found at this iteration of the algorithm
p(i)	The previous node in the currently found least-cost path from node <i>u</i> to <i>i</i>

1:	$N' \leftarrow \{u\}$
2:	For all nodes $i \in \{v, x, w, y, z\}$ do
3:	If $i$ – the neighbour of $u$ then
4:	$D(i) \leftarrow c(u,i)$
5:	Else $D(i) \leftarrow \infty$

3:	If $i$ – the neighbour of $u$ then			
4:	$D(i) \leftarrow c(u,i)$			
5:	Else $D(i) \leftarrow \infty$			
Iter	rations:			
1:	Loop			
2:	Find $i \notin N'$ such that $D(i)$ is minimal			
3:	add $i$ to $N'$			
<b>4</b> :	For each neighbour $k$ of $i$ , with $k \notin N'$ do $D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$			
5:				
6:	Until $N' = N$			

Problem: To find the shortest paths from the source node *u* to destination nodes v, x, w, y, z



Steps of Dijkstra's algorithm:

	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	{ <i>u</i> }	2, u	5, u	1, <i>u</i>	$\infty$	8
1	$\{u,x\}$	2, u	4, <i>x</i>		2, <i>x</i>	∞
2	$\{u,x,y\}$	2, <i>u</i>	3, <i>y</i>			4, <i>y</i>
3	$\{u, x, y, v\}$		3, <i>y</i>			4, <i>y</i>
4	$\{u, x, y, v, z\}$		3, <i>y</i>			
5	$\{u, x, y, v, z, w\}$					

Given:

u	
v, x, w, y, z	

**Destination nodes** ink cost between nodes i and k

c(i,k) Link cost between nodes $i$ and (specified near each graph ed
--

Algorithm variables:

	1	١
1	)	(

A subset of nodes, to which optimal paths have been found by a given iteration The cost of the least-cost path to node *i*,

(i) found at this iteration of the algorithm The previous node in the currently found p(i)least-cost path from node u to i

The source node

Initialization (Step 0):

 $N' \leftarrow \{u\}$ 

For all nodes  $i \in \{v, x, w, y, z\}$  do

If i – the neighbour of u then 3:

 $D(i) \leftarrow c(u, i)$ 

Else  $D(i) \leftarrow \infty$ 5:

Iterations:

4:

5:

Loop 1:

**Find**  $i \notin N'$  such that D(i) is minimal 2:

3: add i to N'

**For** each neighbour k of i, with  $k \notin N'$  do 4:

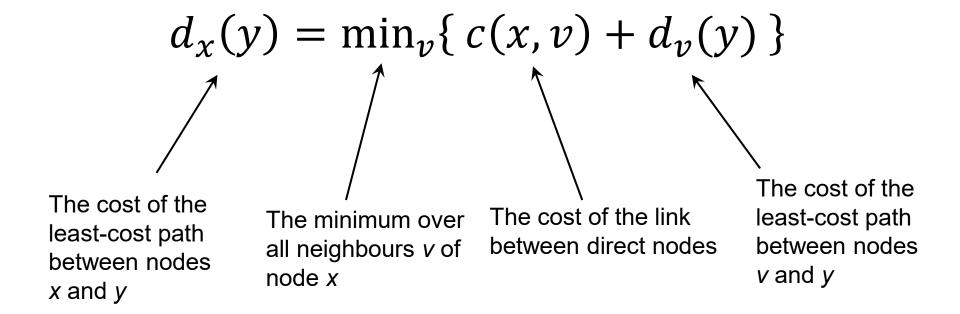
 $D(k) \leftarrow \min\{D(k), D(i) + c(i, k)\}$ 

6:

Until N' = N

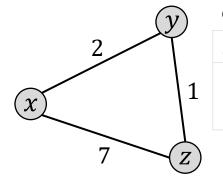
Bellman-Ford Equation for the Distance-Vector Algorithm:

- Specifies the relation between costs of the least-cost paths of neighboring nodes
- The basis to distance-vector algorithms



## Example of Using the Distance-Vector Algorithm

Problem: To find the shortest paths between any two nodes



#### Given:

$N = \{x, y, z\}$	Destination nodes
c(i, k)	Link cost between nodes $i$ and $k$ (specified near each edge)

Algorithm variables:

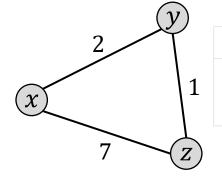
$D_i(k)$	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), \ D_i(y), \ D_i(z)]$

Initialization at node  $i \in \{x, y, z\}$  (Step 0):

initialization at node $i \in \{x, y, z\}$ (Step 0).		
1:	For all nodes $k \in N$ do	
2:	If $k$ – the neighbour of $i$ then	
3:	$D_i(k) \leftarrow c(i,k)$	
4:	Else $D_i(k) \leftarrow \infty$	
<b>5</b> :	<b>For</b> each neighbour $v$ of $i$ do	
<b>6</b> :	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$	
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$	

### Example of Using the Distance-Vector Algorithm

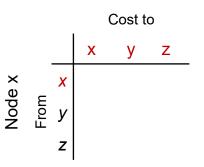
Problem: To find the shortest paths between any two nodes



#### Given:

$N = \{x, y, z\}$	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each edge)

Steps of the Distance-Vector algorithm:



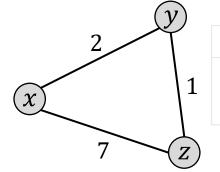
Algorithm variables:

7 agentanii vanasiesi	
$D_i(k)$	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$

Initialization at node  $i \in \{x, y, z\}$  (Step 0):

initialization at node $t \in \{x, y, z\}$ (Step 0).		
1:	For all nodes $k \in N$ do	
2:	If $k$ – the neighbour of $i$ then	
3:	$D_i(k) \leftarrow c(i,k)$	
4:	Else $D_i(k) \leftarrow \infty$	
<i>5:</i>	For each neighbour $v$ of $i$ do	
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$	
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$	

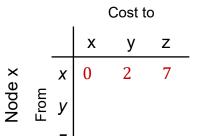
Problem: To find the shortest paths between any two nodes



#### Given:

$N = \{x, y, z\}$	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each edge)

Steps of the Distance-Vector algorithm:

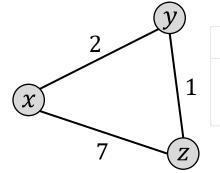


Algorithm variables:

, "90"	T Variables.
$D_i(k)$	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$

mua	alization at node $t \in \{x, y, z\}$ (Step 0).
1:	For all nodes $k \in N$ do
2:	If $k$ – the neighbour of $i$ then
3:	$D_i(k) \leftarrow c(i,k)$
4:	Else $D_i(k) \leftarrow \infty$
5:	<b>For</b> each neighbour $v$ of $i$ do
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$

Problem: To find the shortest paths between any two nodes



#### Given:

$N = \{x, y, z\}$	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each edge)

Steps of the Distance-Vector algorithm:



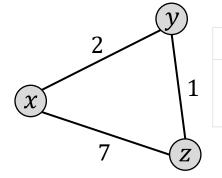
				Cost to	)	
			X	у	Z	
<b>X</b>		X	0	2	7	
Node x	From	У				
		Z				

Algorithm variables:

Augorianni variabioo.		
$D_i(k)$	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm	
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$	

muua	alization at node $t \in \{x, y, z\}$ (Step 0).
1:	For all nodes $k \in N$ do
2:	If $k$ – the neighbour of $i$ then
3:	$D_i(k) \leftarrow c(i,k)$
<b>4</b> :	Else $D_i(k) \leftarrow \infty$
5:	<b>For</b> each neighbour $v$ of $i$ do
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$

Problem: To find the shortest paths between any two nodes



#### Given:

$N = \{x, y, z\}$	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each edge)

Steps of the Distance-Vector algorithm:



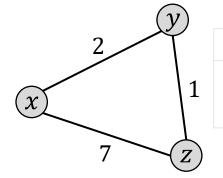
			Х	у	z	
×	_	X	0	2	7	_
Node x	From	У	$\infty$	$\infty$	$\infty$	
			<b>∞</b>	$\infty$	$\infty$	

Algorithm variables:

,g =	variableer
$D_i(k)$	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$

ırııua	alization at node $i \in \{x, y, z\}$ (Step 0):
1:	For all nodes $k \in N$ do
2:	If $k$ – the neighbour of $i$ then
3:	$D_i(k) \leftarrow c(i,k)$
4:	Else $D_i(k) \leftarrow \infty$
<i>5:</i>	For each neighbour $v$ of $i$ do
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$

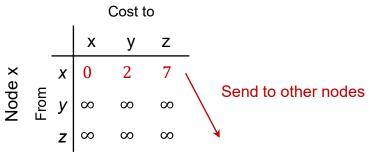
Problem: To find the shortest paths between any two nodes



#### Given:

$N = \{x, y, z\}$	Destination nodes
c(i,k)	Link cost between nodes $i$ and $k$ (specified near each edge)

Steps of the Distance-Vector algorithm:

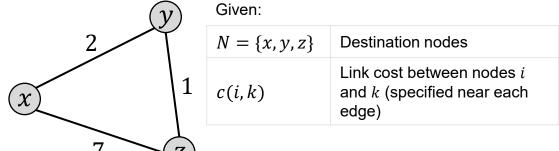


Algorithm variables:

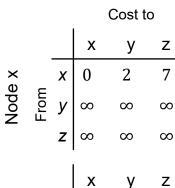
,	T Variables.
$D_i(k)$	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$

muu	alization at node $t \in \{x, y, z\}$ (Step 0).
1:	For all nodes $k \in N$ do
2:	If $k$ – the neighbour of $i$ then
3:	$D_i(k) \leftarrow c(i,k)$
4:	Else $D_i(k) \leftarrow \infty$
<i>5:</i>	For each neighbour $v$ of $i$ do
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:



 $\infty$ 

Χ

 $\infty$ 

Node y

From

From

 $z \mid \infty$ 

 $\chi \mid \infty$ 

Ζ

0		1
œ	)	$\infty$
Ŋ	<b>y</b>	z
œ		∞
œ	)	$\infty$
1		0

 $\infty$ 

 $\infty$ 

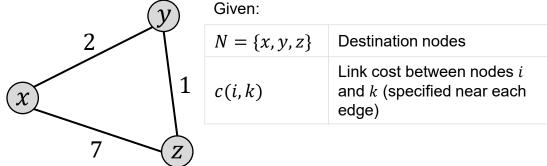
Algorithm variables:							
$D_i(k)$		An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm					
$\mathbb{D}_i$		The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$					
Initia		ion at node $i \in \{x, y, z\}$ (Step 0):					
2:	2: If $k$ – the neighbour of $i$ then 3: $D_i(k) \leftarrow c(i,k)$ 4: Else $D_i(k) \leftarrow \infty$						
3:							
4:							
5:	For	For each neighbour $v$ of $i$ do					

Wait  $\mathbb{D}_v \leftarrow ?$ 

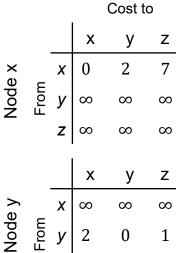
**Send**  $\mathbb{D}_i$  to each neighbour v of i

6:

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:



From

From

Ζ  $\infty$ 

Χ  $\infty$ 

Z

Χ

 $\infty$ 

0	1
$\infty$	$\infty$
у	Z
$\infty$	$\infty$
$\infty$	$\infty$
1	0

ilveii.		
$I = \{x, y, z\}$	Destination nodes	
(i, k)	Link cost between nodes $i$ and $k$ (specified near each edge)	li
aorithm:		,

Algorithm variables: An estimated cost of the least-cost path

$D_i(k)$	from node $i$ to $K$ , found at this iteration of the algorithm	
$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$	

1:

5:

Initi

İ	alization at node $i \in \{x, y, z\}$ (Step 0):
	For all nodes $k \in N$ do

2: If 
$$k$$
 – the neighbour of  $i$  then  
3:  $D_i(k) \leftarrow c(i,k)$ 

4: Else 
$$D_i(k) \leftarrow \infty$$

6: Wait 
$$\mathbb{D}_v \leftarrow ?$$

**For** each neighbour v of i do

**Send**  $\mathbb{D}_i$  to each neighbour v of i

Iterations at node  $i \in \{x, y, z\}$ :

3:

7:

100	ationo at nouc	, ,	_	(x, y)
1:	Loop			

L	וטי
	V

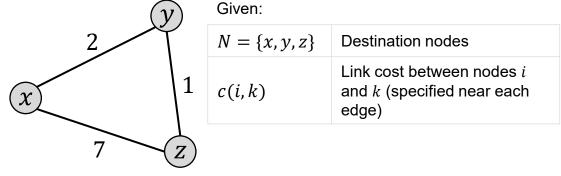
Vait for change in local link cost or msg from neighbor

For each node  $k \in N$  and neighbour v do

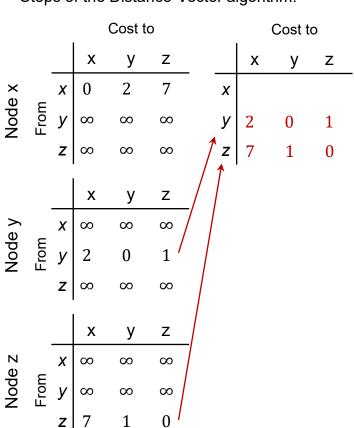
4: 
$$D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$$
  
5: **If**  $D_i(k)$  changed

6: **Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:



Algorithm variables:

An estimated cost of the least-cost path  $D_i(k)$ from node *i* to *k*, found at this iteration of the algorithm

The distance vector of node i:

 $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$ 

4:

5:

6:

 $\mathbb{D}_i$ 

Initialization at node  $i \in \{x, y, z\}$  (Step 0):

1:

For all nodes  $k \in N$  do

2: 3:

If k – the neighbour of i then  $D_i(k) \leftarrow c(i,k)$ 

Else  $D_i(k) \leftarrow \infty$ 

**For** each neighbour v of i do

Wait  $\mathbb{D}_v \leftarrow ?$ 

from neighbor

**Send**  $\mathbb{D}_i$  to each neighbour v of i

Iterations at node  $i \in \{x, y, z\}$ : 1:

2:

3:

4:

Loop

Wait for change in local link cost or msg

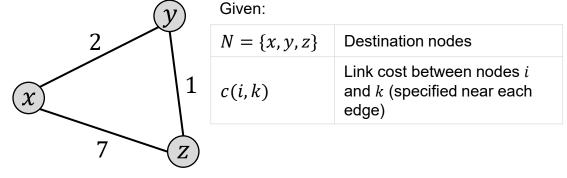
For each node  $k \in N$  and neighbour v do

 $D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$ 

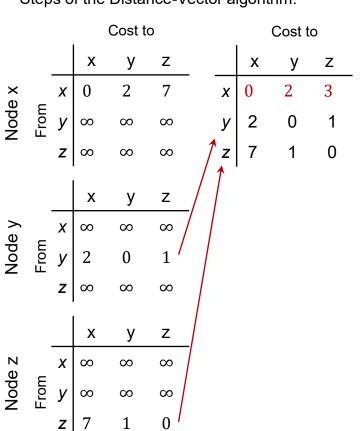
If  $D_i(k)$  changed 5: 6: **Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 

7:

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:



Algorithm variables:

 $D_i(k)$ from node *i* to *k*, found at this iteration of the algorithm The distance vector of node i:  $\mathbb{D}_i$ 

An estimated cost of the least-cost path

1:

2:

3:

6:

 $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$ 

Initialization at node  $i \in \{x, y, z\}$  (Step 0):

For all nodes  $k \in N$  do

If k – the neighbour of i then

 $D_i(k) \leftarrow c(i,k)$ 

4: Else  $D_i(k) \leftarrow \infty$ 5: **For** each neighbour v of i do

Wait  $\mathbb{D}_v \leftarrow ?$ 

from neighbor

**Send**  $\mathbb{D}_i$  to each neighbour v of i

Iterations at node  $i \in \{x, y, z\}$ :

2:

3:

4:

1: Loop

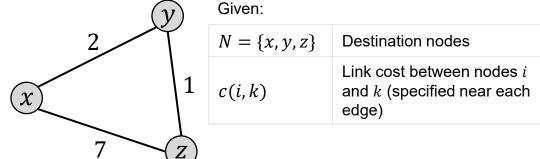
Wait for change in local link cost or msg

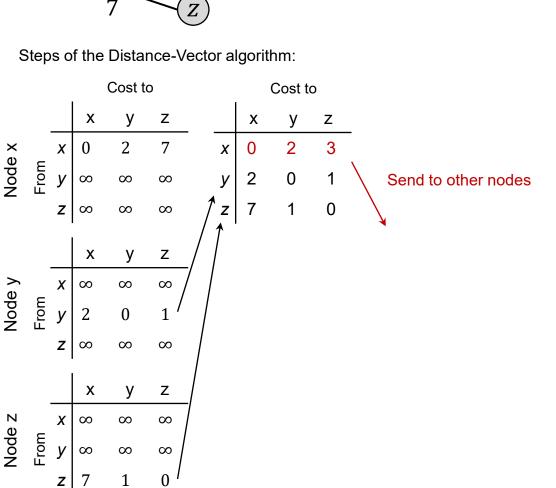
For each node  $k \in N$  and neighbour v do  $D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$ 

If  $D_i(k)$  changed 5:

6: **Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 7:

Problem: To find the shortest paths between any two nodes





Algorithm variables:  $D_i(k) \quad \text{An estimated cost of the least-cost path from node } i \text{ to } k, \text{ found at this iteration of the algorithm}$   $\mathbb{D}_i \quad \text{The distance vector of node } i: \\ \mathbb{D}_i = [D_i(x), \ D_i(y), \ D_i(z)]$ Initialization at node  $i \in \{x, y, z\}$  (Step 0):

1: For all nodes  $k \in N$  do

2: If k – the neighbour of i then

3:  $D_i(k) \leftarrow c(i, k)$ 

Else  $D_i(k) \leftarrow \infty$ 

Wait  $\mathbb{D}_{v} \leftarrow ?$ 

**For** each neighbour v of i do

**Send**  $\mathbb{D}_i$  to each neighbour v of i

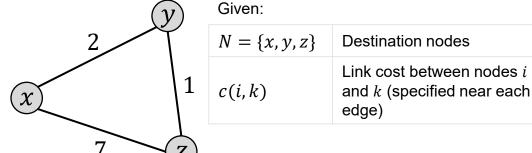
4:

5:

6:

ıtei	rations at node $i \in \{x, y, z\}$ :
1:	Loop
2:	Wait for change in local link cost or msg from neighbor
3:	For each node $k \in N$ and neighbour $v$ do
<b>4</b> :	$D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$
5:	If $D_i(k)$ changed
6:	<b>Send</b> $\mathbb{D}_i$ to all neighbours $v \in N$
7:	Forever

Problem: To find the shortest paths between any two nodes



From

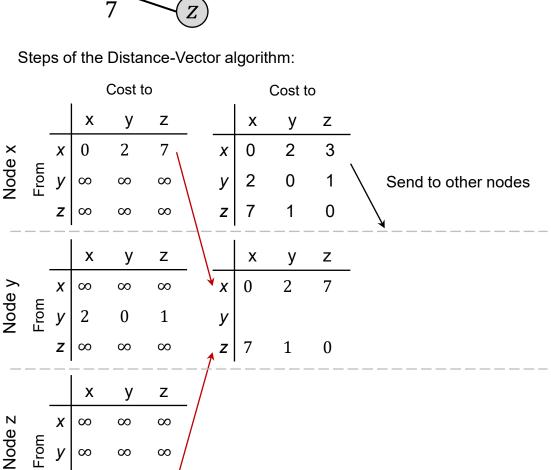
Z

 $\infty$ 

 $\infty$ 

 $\infty$ 

0



Algorithm variables:

 $D_i(k)$ 

from node *i* to *k*, found at this iteration of the algorithm The distance vector of node i:

An estimated cost of the least-cost path

 $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$ 

1:

2:

3:

4:

5:

6:

 $\mathbb{D}_i$ 

Initialization at node  $i \in \{x, y, z\}$  (Step 0): For all nodes  $k \in N$  do

If k – the neighbour of i then

 $D_i(k) \leftarrow c(i,k)$ **Else**  $D_i(k) \leftarrow \infty$ 

**For** each neighbour v of i do

Wait  $\mathbb{D}_v \leftarrow ?$ 

**Send**  $\mathbb{D}_i$  to each neighbour v of i

Iterations at node  $i \in \{x, y, z\}$ : 1:

2:

3:

4:

5:

6:

7:

Loop

Wait for change in local link cost or msg

from neighbor

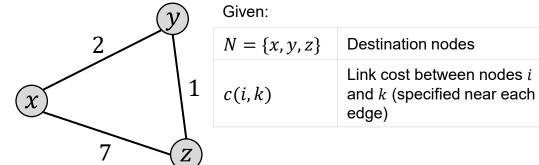
For each node  $k \in N$  and neighbour v do

 $D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$ 

If  $D_i(k)$  changed

**Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 

Problem: To find the shortest paths between any two nodes



From

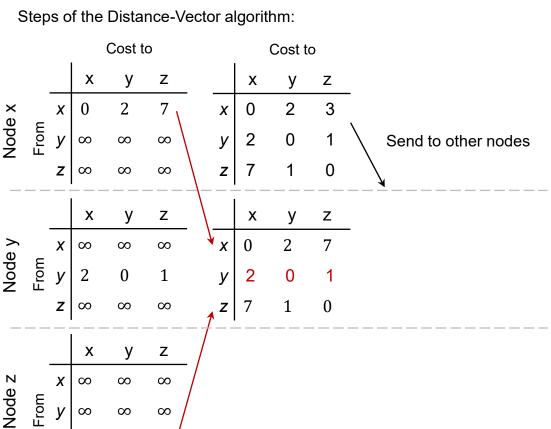
Z

 $\infty$ 

 $\infty$ 

 $\infty$ 

0



Algorithm variables:

 $\mathbb{D}_i$ 

An estimated cost of the least-cost path  $D_i(k)$ from node *i* to *k*, found at this iteration of the algorithm

The distance vector of node i:

 $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$ 

Initialization at node  $i \in \{x, y, z\}$  (Step 0):

1:

3:

4:

5:

6:

For all nodes  $k \in N$  do

2:

If k – the neighbour of i then  $D_i(k) \leftarrow c(i,k)$ 

**Else**  $D_i(k) \leftarrow \infty$ 

**For** each neighbour v of i do

Wait  $\mathbb{D}_{v} \leftarrow ?$ 

**Send**  $\mathbb{D}_i$  to each neighbour v of i

Iterations at node  $i \in \{x, y, z\}$ : 1:

2:

3:

4:

5:

6:

7:

Loop

Wait for change in local link cost or msg

from neighbor

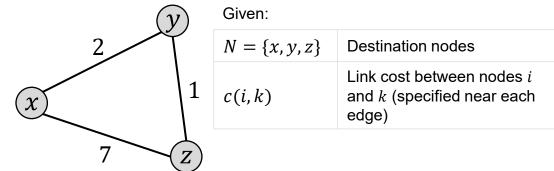
For each node  $k \in N$  and neighbour v do

 $D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$ 

If  $D_i(k)$  changed

**Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:

 $\infty$ 

 $\infty$ 

0

Node z

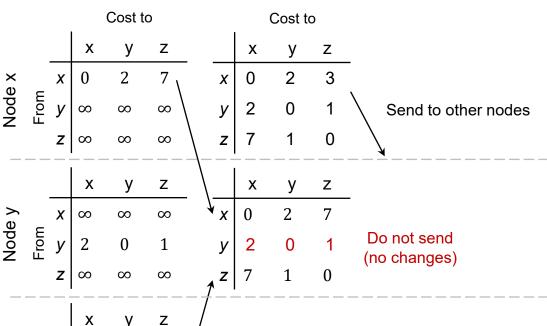
Χ  $\infty$ 

Z

 $\infty$ 

 $\infty$ 

From



Algorithm variables: An estimated cost of the least-cost path  $D_i(k)$ from node *i* to *k*, found at this iteration of

The distance vector of node i:  $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$ 

 $\mathbb{D}_i$ 

Initialization at node  $i \in \{x, y, z\}$  (Step 0):

1:

For all nodes  $k \in N$  do

**Else**  $D_i(k) \leftarrow \infty$ 

the algorithm

2:

4:

5:

6:

If k – the neighbour of i then

3:

 $D_i(k) \leftarrow c(i,k)$ 

**For** each neighbour v of i do

Wait  $\mathbb{D}_v \leftarrow ?$ 

**Send**  $\mathbb{D}_i$  to each neighbour v of i

Iterations at node  $i \in \{x, y, z\}$ : 1:

2:

3:

4:

5:

6:

Loop

Wait for change in local link cost or msg

from neighbor

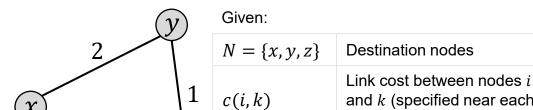
For each node  $k \in N$  and neighbour v do

 $D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$ 

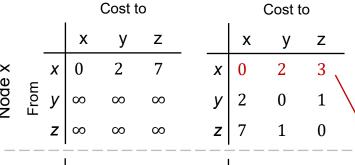
If  $D_i(k)$  changed **Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 

7:

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:



_			I	∞			- 1		1		
			Х	у	Z	_		X	у	Z	_
e <	_	X	8	$\infty$	$\infty$		X	0	2	7	_
Nod	Fron	У	2 ∞	0	1		У	2	0	1	
_		Z	$\infty$	$\infty$	$\infty$		Z	7	1	0	
_		_									

Χ

Χ  $\infty$ 

Z

Algo	orithn	n variables:								
$D_i$	(k)	An estimated cost of the least from node $i$ to $k$ , found at this the algorithm								
	$\mathbb{D}_i$	The distance vector of node $\mathbb{D}_i = [D_i(x), D_i(y), D_i(y)]$								
Initia	alizat	ion at node $i \in \{x, y, z\}$ (Step 0)								
1:	For all nodes $k \in N$ do									
2:		If $k$ – the neighbour of $i$ then								
3:		$D_i(k) \leftarrow c(i,k)$								
4:		Else $D_i(k) \leftarrow \infty$								
5:	For	each neighbour $v$ of $i$ do								
6:	ı	Wait $\mathbb{D}_v \leftarrow ?$								
7:	<b>Send</b> $\mathbb{D}_i$ to each neighbour $v$									

and k (specified near each

edge)

An estimated cost of the least-cost path from node *i* to *k*, found at this iteration of the algorithm

 $\mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]$ 

1:	For all nodes $k \in N$ do
2:	If $k$ – the neighbour of $i$ then
3:	$D_i(k) \leftarrow c(i,k)$
4:	Else $D_i(k) \leftarrow \infty$
5:	For each neighbour $v$ of $i$ do
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$

1:

3:

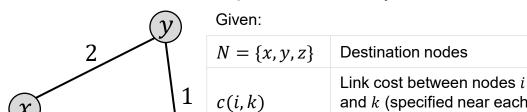
Iterations at node  $i \in \{x, y, z\}$ : Loop Wait for change in local link cost or msg 2:

from neighbor For each node  $k \in N$  and neighbour v do

4:  $D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$ If  $D_i(k)$  changed 5:

6: **Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 7:

Problem: To find the shortest paths between any two nodes



and k (specified near each

edge)

Steps of the Distance-Vector algorithm:

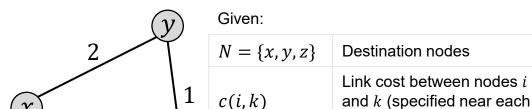
			_	Cost to	0	_		Cost to	0		Cost to				
			X	у	Z		X	у	Z			X	у	Z	_
<b>&lt;</b>		X	0	2	7	х	0	2	3		Χ	0	2	3	_
ב ב ב	From	У	8	$\infty$	$\infty$	У	2	0	1		у	2	0	1	
_		Z	$\infty$	$\infty$	$\infty$	z	7	1	0		Z	3	1	0	

X O	_	X	0	2	7	X	0	2	3	X	0	2	3	
Node x	From	У	$\infty$	$\infty$	$\infty$	У	2	0	1	У	2	0	1	
_		Z	∞	$\infty$	$\infty$	Z	7	1	0	Z	3	1	0	
			x	у	z		X	у	z		Х	у	z	
e >	_	X	∞	∞	∞	X	0	2	7	X	0	2	3	
Node y	From	У	2	0	1	У	2	0	1	У	2	0	1	
		Z	$\infty$	$\infty$	$\infty$	Z	7	1	0	Z	3	1	0	
_			x	у	z		x	у	z		X	у	z	_
N G	_	Χ	8	∞	∞	X	0	2	7	X	0	2	3	
Node	From	У	$\infty$	$\infty$	$\infty$	У	2	0	1	У	2	0	1	
_	_	7	7	1	0	7	3	1	0	7	3	1	0	

Aigo	וווווווו	ii valiabies.									
$D_i$	(k)	An estimated cost of the least-cost path from node $i$ to $k$ , found at this iteration of the algorithm									
	$\mathbb{D}_i$	The distance vector of node $i$ : $\mathbb{D}_i = [D_i(x), \ D_i(y), \ D_i(z)]$									
Initia	alizat	ion at node $i \in \{x, y, z\}$ (Step 0):									
1:	For	all nodes $k \in N$ do									
2:		If $k$ – the neighbour of $i$ then									
3:		$D_i(k) \leftarrow c(i,k)$									
4:	Else $D_i(k) \leftarrow \infty$										
5:	For each neighbour $v$ of $i$ do										
6:	<b>Wait</b> $\mathbb{D}_v \leftarrow ?$										
7:	Ser	<b>nd</b> $\mathbb{D}_i$ to each neighbour $v$ of $i$									
Iter	atior	s at node $i \in \{x, y, z\}$ :									
1:	Loc	ор									
2:	,	Wait for change in local link cost or msg from neighbor									
3:		<b>For</b> each node $k \in N$ and neighbour $v$ do									
4:		$D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}$									
5:		<b>If</b> $D_i(k)$ changed									
6:		<b>Send</b> $\mathbb{D}_i$ to all neighbours $v \in N$									
7:	Fo	rever									

Algorithm variables:

Problem: To find the shortest paths between any two nodes



Steps of the Distance-Vector algorithm:

 $\chi$ 

			_	Cost to	0			Cost to	0	_	Cost to					
	_		х	у	Z		х	у	Z			Х	у	Z	_	
<b>&lt;</b>	_	X	0	2	7	X	0	2	3		X	0	2	3		
ב ב ב	From	У	∞	$\infty$	$\infty$	У	2	0	1		У	2	0	1		
_	_	Z	∞	$\infty$	$\infty$	z	7	1	0		z	3	1	0		
_		_														

edge)

Nod	Fron	У	$\infty$	$\infty$	$\infty$		У	2	0	1		У	2	0	1	
		Z	∞	$\infty$	$\infty$		Z	7	1	0		Z	3	1	0	
				у	z			x	у	z			x	у	z	
Node y	_	Х	∞	∞	∞	-	χ	0	2	7		X	0	2	3	_
	From	У	2	0	1		У	2	0	1		У	2	0	1	
		Z	$\infty$	$\infty$	$\infty$		Z	7	1	0		Z	3	1	0	
			x	у	z				у	z			X	у	z	
Node z	_	Χ	∞	∞	∞	-	Х	0	2	7	•	X	0	2	3	_
	From	У	$\infty$	$\infty$	$\infty$		у	2	0	1		У	2	0	1	
_	_	7	7	1	Ω		7	2	1	0		7	2	1	0	

```
Algorithm variables:
            An estimated cost of the least-cost path
 D_i(k)
            from node i to k, found at this iteration of
            the algorithm
            The distance vector of node i:
   \mathbb{D}_i
                    \mathbb{D}_i = [D_i(x), D_i(y), D_i(z)]
Initialization at node i \in \{x, y, z\} (Step 0):
      For all nodes k \in N do
1:
          If k – the neighbour of i then
2:
                D_i(k) \leftarrow c(i,k)
3:
4:
          Else D_i(k) \leftarrow \infty
      For each neighbour v of i do
5:
         Wait \mathbb{D}_{v} \leftarrow ?
6:
      Send \mathbb{D}_i to each neighbour v of i
Iterations at node i \in \{x, y, z\}:
 1:
      Loop
         Wait for change in local link cost or msg
 2:
                from neighbor
         For each node k \in N and neighbour v do
 3:
              D_i(k) \leftarrow \min_{v} \{ c(i, v) + D_v(k) \}
 4:
         If D_i(k) changed
 5:
```

**Send**  $\mathbb{D}_i$  to all neighbours  $v \in N$ 

6:

7:

#### Acknowledgment

These slides are prepared with the help of Artem Burmyakov and Muhammad Fahim