

Distance vector routing algorithm

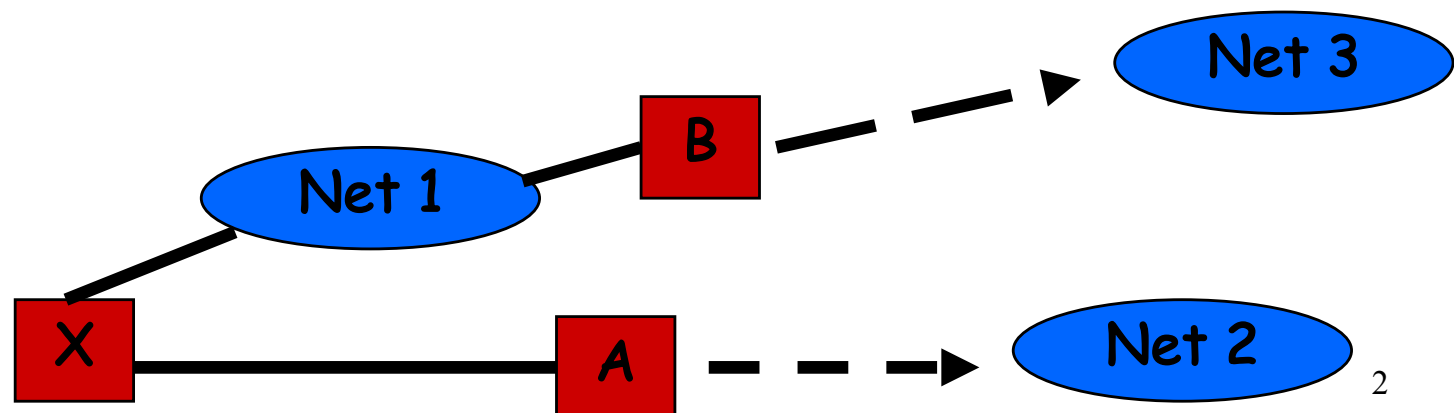


- Messages exchanged between routers have two components:
 - destination network (*vector*)
 - cost to destination (*distance*)
- Also known as *Bellman-Ford* Algorithm
 - Invented in the 1960's

Distance vector: routing table

- Each router maintains a routing table
 - Example: The routing table for a router X

Destination	Cost	Next Hop
Net 1	0	Direct
Net 2	5	Router A
Net 3	3	Router B

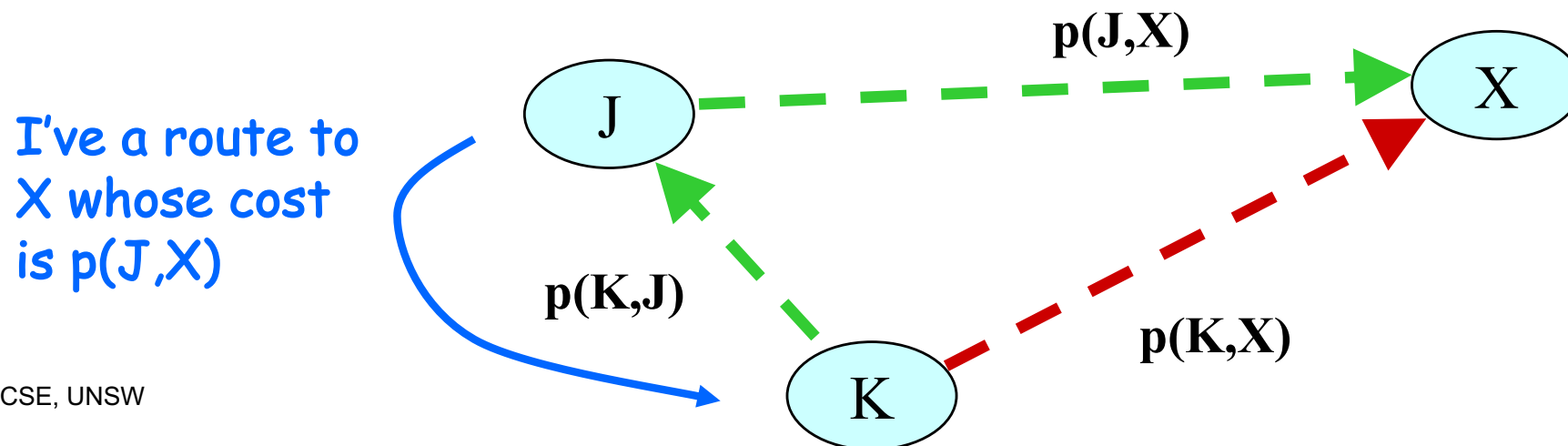


The secret behind distance vector (1)

■ Given:

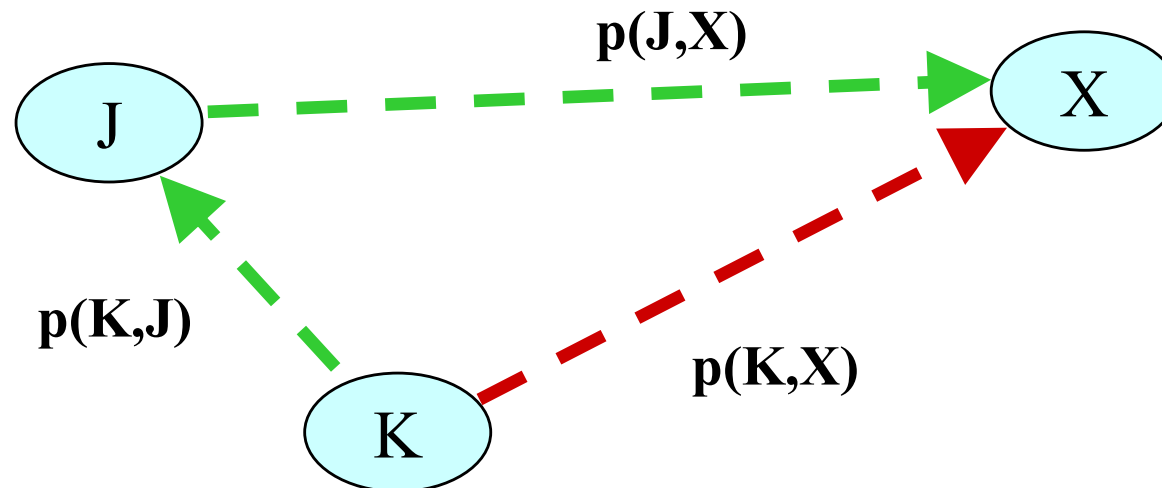
- Router K has an existing route to network X with cost $p(K,X)$
- A neighbouring router J tells K that it has a route to X with cost $p(J,X)$
- Cost between J and K is $p(K,J)$

■ Q: Should K use the existing route or use the one via J?



The secret behind distance vector (2)

- K has two choices
 - Existing route with cost $p(K,X)$
 - Alternative route (via J) with cost $p(K,J) + p(J,X)$
- Distance vector routing algorithm
 - If $p(K,J) + p(J,X) < p(K,X)$: use the route via J
 - Otherwise, use the existing route



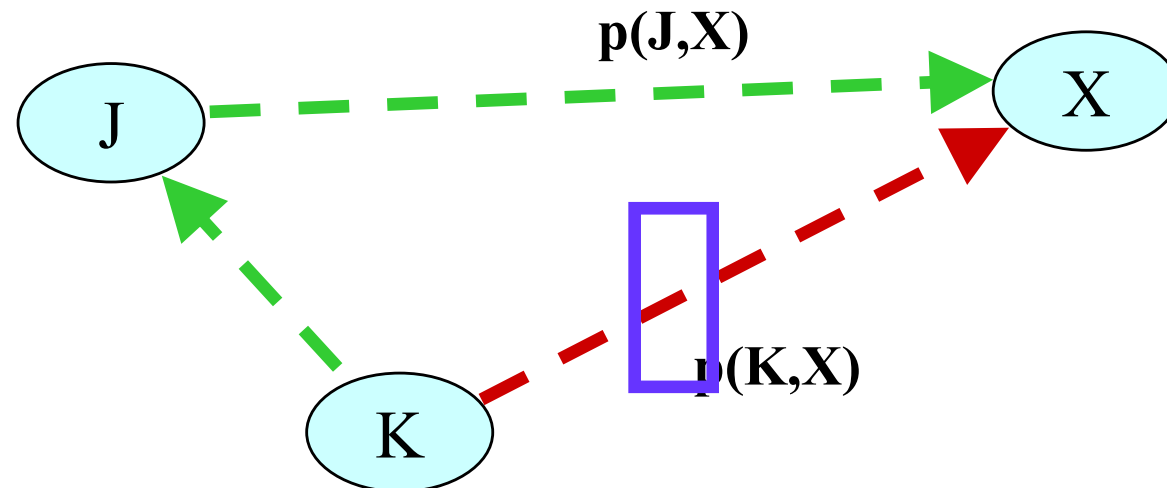
Distance vector: route update (1)



- Periodically, each router sends a copy of its table (destination, cost columns only) to *directly connected* routers
- When router *K* receives table from a neighbouring router *J*, *K* updates its table if:
 - *J* knows a shorter route for a given destination
 - *J* knows a destination *K* didn't know about
 - *K* currently routes to a destination through *J* and *J*'s cost to that destination has changed

Distance vector: route update (2)

- If K updates or adds an entry in response to J 's message,
 - It assigns the Next Hop as Router J
 - It updates the cost. If X is the destination, then cost to $X = p(K,J) + p(J,X)$



Exercise: Route update in distance vector

Existing routing table for router K


Destination	Cost	Next Hop
Net 1	0	Direct
Net 2	0	Direct
Net 4	8	Router L
Net 17	5	Router M
Net 24	6	Router J
Net 30	2	Router Q
Net 42	2	Router J

Routing table from
neighbouring router J:

Destination	Cost
Net 1	2
Net 4	3
Net 17	6
Net 21	4
Net 24	5
Net 30	10
Net 42	3

- Router K receives the routing table from neighbouring router J (showed on the right). If the cost between them is 1, what is the routing table for K after the update?

Solution



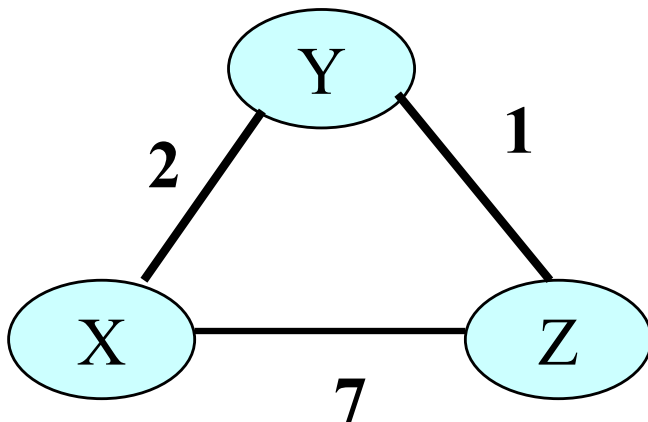
Destination	Cost	Next Hop
Net 1	0	Direct
Net 2	0	Direct
Net 4 (*)	4	Router J
Net 17	5	Router M
Net 21 (*)	5	Router J
Net 24	6	Router J
Net 30	2	Router Q
Net 42 (*)	4	Router J

The entries marked with a red asterisk (*) are updated due to the information from router J

Distance Vector: Start up (1)



- At bootup each router initialises its routing table with directly connected network information
- Example: The following shows a network with 3 nodes. The cost of an edge is showed next to the edge.



The initial routing table for X is:

Destination	Cost	Next Hop
Y	2	Direct
Z	7	Direct

Distance Vector: Start up (2)



- After initialising their routing tables, the nodes start exchanging their routing tables with their neighbouring nodes
- When a node receives a routing table, it updates its routing table according to the distance vector update rules
- Exercise: For the 3-node network on the previous slide, fill in the routing tables on the next page
 1. What are the initial routing tables for Y and Z?
 2. Assuming Y sends its routing table to X and Z, what are their routing tables after the update?

The initial routing tables:

Routing table for X		
Destination	Cost	Next Hop
Y	2	Direct
Z	7	Direct

Routing table for Y		
Destination	Cost	Next Hop

Routing table for Z		
Destination	Cost	Next Hop

Y sends its routing table to its neighbouring nodes and they update their routing tables. After update, the routing tables are:

Routing table for X		
Destination	Cost	Next Hop

Routing table for Y		
Destination	Cost	Next Hop

Routing table for Z		
Destination	Cost	Next Hop

Solution:

The initial routing tables:

Routing table for X		
Destination	Cost	Next Hop
Y	2	Direct
Z	7	Direct

Routing table for Y		
Destination	Cost	Next Hop
X	2	Direct
Z	1	Direct

Routing table for Z		
Destination	Cost	Next Hop
X	7	Direct
Y	1	Direct

Solution: The routing tables after the update are:

Note: * indicates an entry has been updated.

Routing table for X		
Destination	Cost	Next Hop
Y	2	Direct
Z	3 *	Y *

Routing table for Y		
Destination	Cost	Next Hop
X	2	Direct
Z	1	Direct

Routing table for Z		
Destination	Cost	Next Hop
X	3 *	Y *
Y	1	Direct

Distance Vector: Start up (3)



- After this update, X sends its routing table to its neighbouring nodes
 - The routing tables before and after this update is sent is showed on the next slide
- The aim of distance vector routing algorithm is to find the least path route, the update is not causing any changes because the least cost routes have been found

The routing tables before X sends its update:

Routing table for X		
Destination	Cost	Next Hop
Y	2	Direct
Z	3	Y

Routing table for Y		
Destination	Cost	Next Hop
X	2	Direct
Z	1	Direct

Routing table for Z		
Destination	Cost	Next Hop
X	3	Y
Y	1	Direct

The routing tables after X sends its update:

In fact, the tables remain the same.

Routing table for X		
Destination	Cost	Next Hop
Y	2	Direct
Z	3	Y

Routing table for Y		
Destination	Cost	Next Hop
X	2	Direct
Z	1	Direct

Routing table for Z		
Destination	Cost	Next Hop
X	3	Y
Y	1	Direct

Distance Vector: Start up (4)



- Finally, Z sends its routing tables to its neighbouring nodes
 - Their neighbouring nodes updates their routing table
 - The routing table remains the same as before
 - » Because the routing tables already consist of the least cost routes

Operation of Distance Vector



- Distributed - no global view
- Asynchronous - no lock-step updates
- Iterative - several updates until converged

Changes in Distance Vector Tables



The following events may cause a change in the table

- Change of cost of an attached link
- Receipt of an update message from a neighbour