Network Protocol

Network Layer Part:1

ISLAMIC UNIVERSITY COLLEGE
Computer Technical Engineering Department

Computer Networks Protocols

Lecture No.3: Network Layer Part 1

Prepared By: Mr. Karar Al-jawaheri

Routing Concept

ROUTING is **forwarding** of packets from one network to another network choosing the best path from the routing table.

Routing table consist of only the best routes for every destinations.

Types of Routing:

Static	• It is configure by Administrator **manually** • Need for **destination network ID** • It is **secure and fast** • Used for **small organization** which have network of **10-15** routers
Dynamic	• Means **automatically** routing • Dynamic routes means that the router **learns** of paths of destinations by receiving **periodic updates** from other routers • Is **automatically** choose the best shortest path • Can be done by using **routing protocol**
Default	• Is configured for **unknown** destination • When there is **no entry** for the destination network in a routing table, the router will forward the packet to its **default router**. • It is **last preferred routing**

Q/ list the Advantages of dynamic over static?

- There is **no need** to know the destination networks
- **Updates** the topology changes **dynamically**
- Administrative work **reduced**
- Used for **large organizations**

Routing Algorithm

Routing Algorithms

- The main function of the network layer is **routing packets from source to destination**.
- The **routing algorithm** is that part of the network layer software responsible for **deciding** which output line an incoming packet should be transmitted on.
- Routing algorithms can be grouped into two major classes: non adaptive and adaptive.

Non adaptive algorithms (static routing)	Adaptive algorithms (dynamic routing)
• **Do not** base their routing decisions on measurements or estimates of the current traffic and topology. • It is called **static** algorithm.	• **Change** their routing decisions to reflect changes in the topology , and usually the traffic as well. • It is called **dynamic**.

Static Algorithm

1. Flooding routing.

2.Shortest path routing.

Flooding	• A simple local technique, where each router must make decisions based on local knowledge, not the complete picture of the network. • is a simple algorithm to send a packet **along all paths**(Every incoming packet is sent out **on every outgoing** line **except** the one it arrived on). • generates **infinite number of duplicate packets** unless some measures are taken to damp the process. • One such measure is to have a **hop counter** in the header of each packet, which is **decremented** at each hop, with the packet being discarded when the counter reaches zero.
Shortest Path Routing	• Shortest path routing first developed by **Dijkstra**algorithm. • Find the shortest path from a **specified source to all other destinations** in the network. • In the general case, the labels on the lines could be computed **as a function** of the distance, bandwidth, average traffic, communication cost, measured delay, and other factors.

Q/ Find the shortest path from router A to router H?

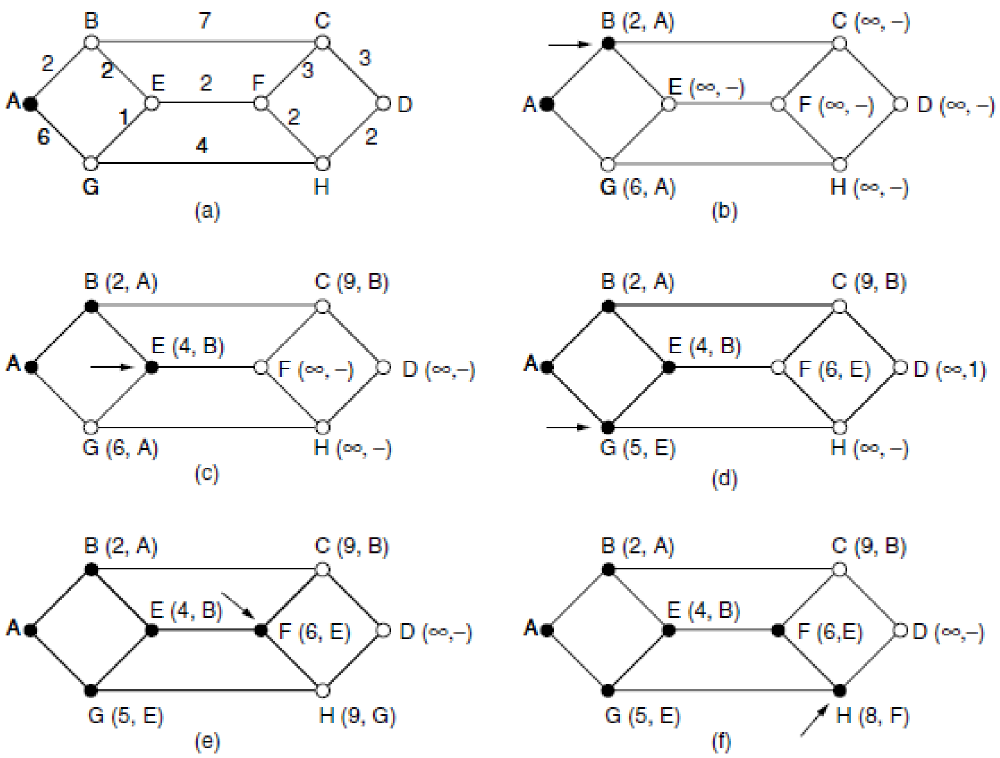


Figure The first six steps used in computing the shortest path from *A* to *D*. The arrows indicate the working node.

Dynamic Routing Algorithm

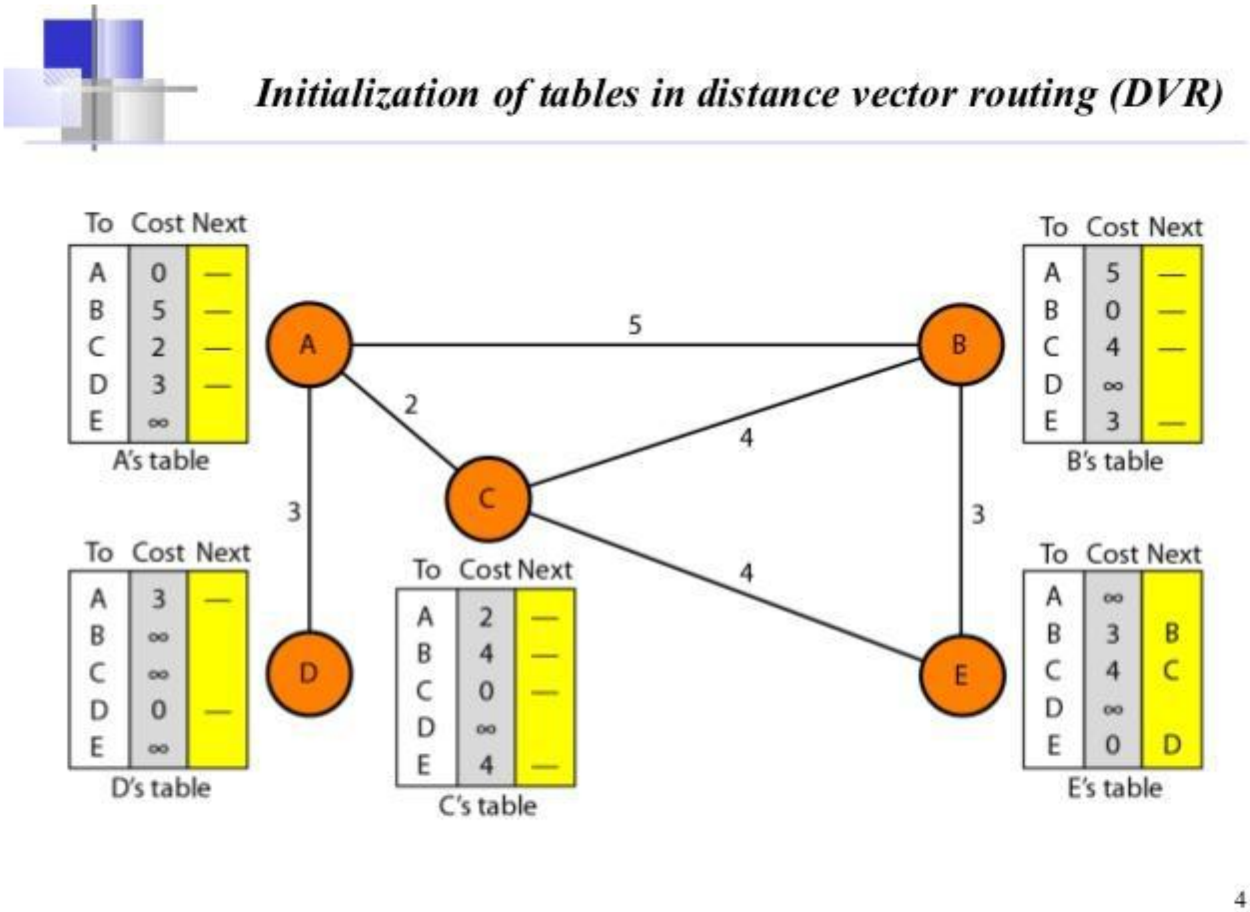
1. Distance Vector Routing.

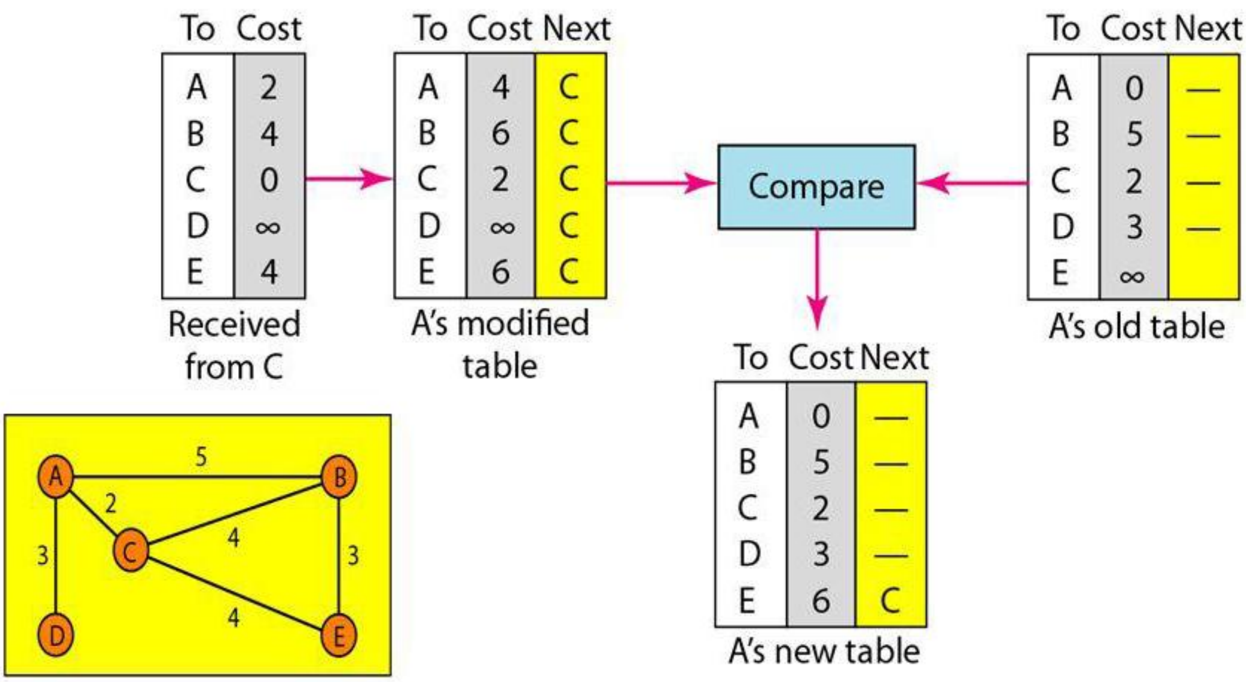
2. Link state routing.

1. Distance Vector Routing.

- Distance Vector routing is **intra-domain** protocols, **inside** Autonomous system, but not between Autonomous system.
- distance-vector routing are based on the **least-cost** goal.
- Distance Vector developed by **Bellman-Ford** algorithm.
- Bellman **equation** is used to find the **least** cost (**shortest distance**) between a source to destination.
- A **distance vector routing algorithm** operates by having each router **maintain a table** (i.e., a **vector**) giving the best known distance to each destination.
- These tables are **updated by exchanging** information with the neighbors router. Every router knows the **best link** to reach each destination.
- **RIP** based on distance vector routing, each router **shares**, **at regular intervals**, its knowledge about entire AS with its neighbor.
- It is so **slow** and does not take **Bandwidth** into consideration when choose the root.

Q/Update the Router A using Distance vector algorithm?





Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

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

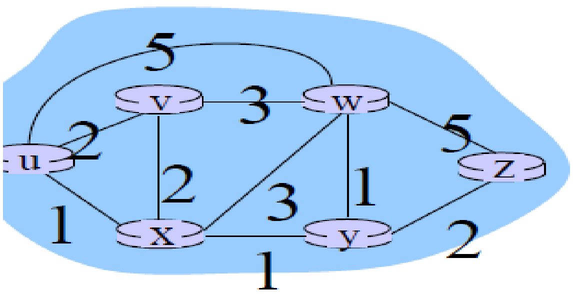
Then

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

- cost from neighbor v to destination y
- cost to neighbor v
- *min* taken over all neighbors v of x

- $D_x(y)$ = estimate of least cost from x to y
  x maintains distance vector $\mathbf{D}_x = [D_x(y): y \in N]$
- node x:
  - knows cost to each neighbor v: $c(x,v)$
  - maintains its neighbors' distance vectors. For each neighbor v, x maintains $\mathbf{D}_v = [D_v(y): y \in N]$

Bellman-Ford example



clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$\begin{aligned} d_u(z) &= \min \{ c(u,v) + d_v(z), \\ &\qquad c(u,x) + d_x(z), \\ &\qquad c(u,w) + d_w(z) \} \\ &= \min \{2 + 5, \\ &\qquad 1 + 3, \\ &\qquad 5 + 3\} = 4 \end{aligned}$$

node achieving minimum is next hop in shortest path, used in forwarding table

2. Link State Routing

The **primary problem in distance vector** that the algorithm often took **too long to converge** after the network topology changed (due to the count-to-infinity **problem**). Consequently, it was **replaced by** a new algorithm, now called link state routing.

The idea behind link state routing is simple and can be stated as five parts. Each router must:

1) **Discove**r its neighbors and learn their network addresses.
2) **Measure** the delay or cost to each of its neighbors.
3) **Construct** a packet telling to all it has just learned.
4) **Send** the packet to all other routers.
5) **Compute** the shortest path to every other router.

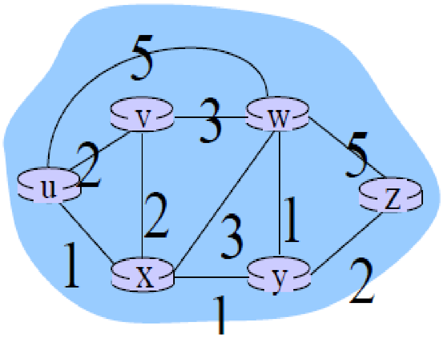
Compared to distance vector routing, link state routing requires **more memory and computation**. Also, the **computation time** grows faster

Nevertheless, in many practical situations, link state routing works **well because it does not suffer from slow convergence problems**.

Q/Consider the following network, define OSPF protocol & construct shortest path tree using dijkstra algorithms :

Dijkstra's algorithm: example

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



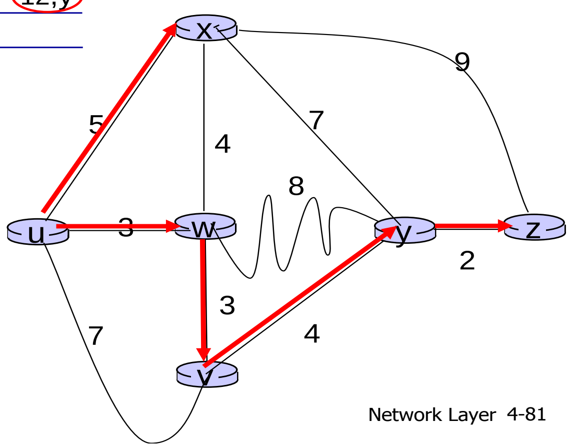
Q/Consider the following network, define OSPF protocol & construct shortest path tree using dijkstra algorithms :

Dijkstra's algorithm: example

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

notes:

- construct shortest path tree by tracing predecessor nodes
- ties can exist (can be broken arbitrarily)



Network Layer 4-81

Comparison of LS and DV algorithms

message complexity

- ***LS:*** with n nodes, E links, O(nE) msgs sent
- ***DV:*** exchange between neighbors only
  - convergence time varies

speed of convergence

- ***LS:*** $O(n^2)$ algorithm requires O(nE) msgs
  - may have oscillations
- ***DV:*** convergence time varies
  - may be routing loops
  - count-to-infinity problem

robustness: what happens if router malfunctions?

LS:

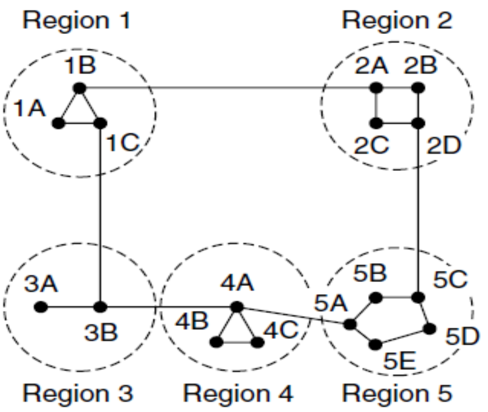
- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:

- DV node can advertise incorrect *path* cost
- each node's table used by others
  - error propagate thru network

<u>Hierarchical Routing</u>

- As networks **grow in size**, the **router routing tables grow proportionally**.
- Not only is router **memory** consumed by ever-increasing tables, but more **CPU time** is needed to scan them and **more bandwidth** is needed to send status reports about them.
- So router can not have table about the entire network.
- **<u>When hierarchical routing is used</u>**, the routers are divided into what we will call **regions.**
- Each router knows all the details about how to route packets to destinations within its **own region** but knows nothing about the internal structure of other regions.



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

<u>Broadcast Routing</u>

- For some applications ,hosts need to send messages to **many or all other hosts**. **Broadcast routing** is used for that purpose.
- The source should send the packet to **all** the necessary destinations. **<u>One of the problems of this method</u>** is that the source has to have the complete list of destinations.
- We have already seen a better broadcast routing technique: **flooding.**

<u>Multicast Routing</u>

- Sending a message to such a **group** is called **multicasting**, and the routing algorithm used is called **multicast routing**.

- All multicasting schemes require some way to **create and destroy groups and to identify which routers are members of group**.

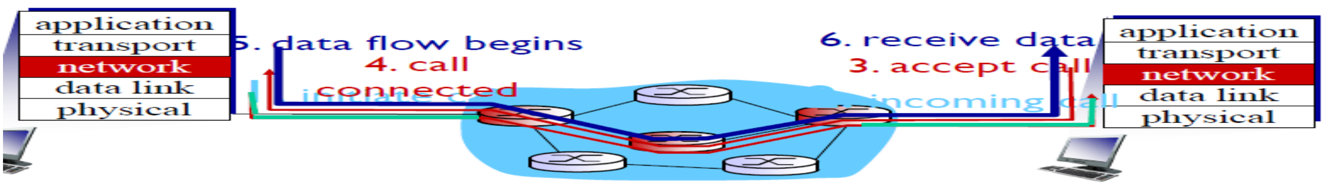
<u>Network service model</u>

<u>Q:</u>What service model for "channel" transporting datagrams from sender to receiver?

individual datagrams	*flow of datagrams*
• guaranteed delivery • guaranteed delivery with less than 40 msec delay	• in-order datagram delivery • guaranteed minimum bandwidth to flow.

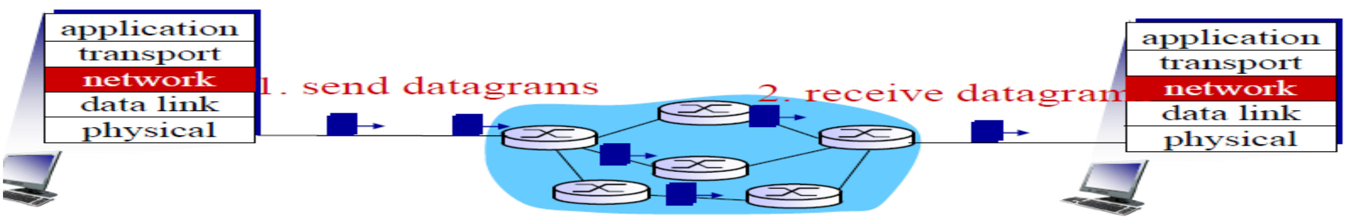
Virtual circuits: **signaling** protocols

- used to setup, maintain, teardown VC
- used in ATM, frame-relay, X.25
- not used in today's Internet



Datagram networks

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets forwarded using destination host address



Q/ Datagram or VC network: why?

Internet (datagram)	*ATM (VC) virtual circuit*
• data exchange among computers –"elastic" service, no strict timing req • many link types –different characteristics –uniform service difficult • "smart" end systems (computers) –can adapt, perform control, error recovery –***simple inside network, complexity at "edge"***	• evolved from telephony • human conversation: –strict timing, reliability requirements –need for guaranteed service • "dumb" end systems –telephones –***complexity inside network***

IP datagram format