UNIT I: - Packet Switching Networks

What is a Computer Network?

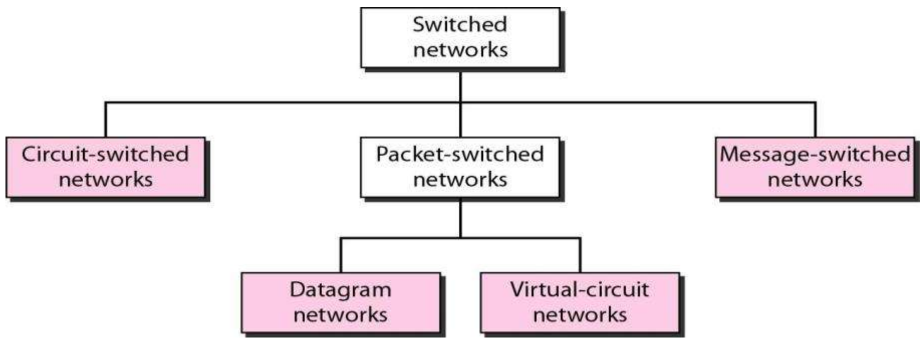
- **Communication Networks:** "Sets of nodes that are interconnected to allow the exchange of information such as voice, sound, graphics, pictures, video, text, data, etc…"
- **Telephone Networks:** " The first well established and most widely used communication networks which are used for voice transmission"
  – Telephone networks originally used analog transmission as a transmission technology for the information. However, digital transmission started to evolve replacing a lot of the analog transmission techniques used in telephone networks.
- **Computer Networks:** "Collection of autonomous computers interconnected by a technology to allow exchange of information"

A network is a series of connected devices. Whenever we have many devices, the interconnection between them becomes more difficult as the number of devices increases . Some of the conventional ways of interconnecting devices are

a. Point to point connection between devices as in mesh topology.
b. Connection between central device and every other device – as in star topology
c. Bus topology-not practical if the devices are at greater distances.

The solution to this interconnectivity problem is switching. A switched network consists of a series of interlinked nodes called switches. A switch is a device that creates temporary connections between two or more systems. Some of the switches are connected to end systems (computers and telephones) and others are used only for routing.

Taxonomy of switched networks



Circuit switching

- Traditional telephone networks operate on the basis of circuit switching
- In conventional telephone networks, a circuit between two users must be established for a communication to occur
- Circuit switched networks requires resources to be reserved for each pair of end users
- The resources allocated to a call cannot be used by others for the duration of the call

- The reservation of the network resources for each user results in an inefficient use of bandwidth for applications in which information transfer is bursty or if the information is small

Packet Switching

- Packet switched networks are the building blocks of computer communication systems in which data units known as packets flow across the networks.
- It provides flexible communication in handling all kinds of connections for a wide range of applications e.g. telephone calls, video conferencing, distributed data processing etc...
- Packet switched networks with a unified, integrated data infrastructure known as the Internet can provide a variety of communication services requiring different bandwidths.
- To make efficient use of available resources, packet switched networks dynamically allocate resources only when required.
- The form of information in packet switched networks is always digital bits.

Differences between Circuit Switching and Packet Switching

Circuit switching	Packet switching
1. Call set up is required.	1. Call setup is not required.
2. Dedicated connection between two Hosts.	2. No dedicated connection between two Hosts.
3. Connection/Communication is lost, if any link in the path between the Hosts is broken.	3. Connection/Communication could continue between the Hosts since data have many routes between the Hosts.
4. Information take the same route between the connected Hosts	4. Information could take different routes to reach the destination Host.
5. Information always arrives in order.	5. Information could arrive out of order to the destination
6. Bandwidth available is fixed.	6. Bandwidth available is variable.
7. Congestion is call based.	7. Congestion is packet based.
8. Bandwidth utilization is partial.	8. Bandwidth utilization is full.
9. It does not uses store-and-forward transmission.	9. It uses store-and forward transmission.
10. It is Transparent.	10. Not transparent.
11. Charging is time based.	11. Charging is packet based.

Packet networks can be viewed from two perspectives:

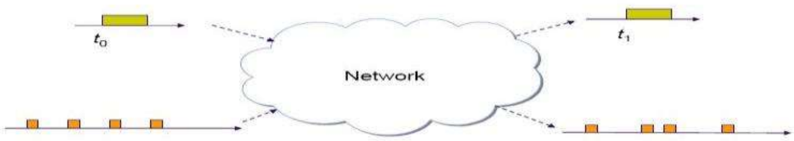
- External view of network :- It is Concerned with the services that the network provides to the transport layer
- Internal operation of the network.

Network services and internal network operation

Essential function of network:

- The essential function of network is to transfer information among the users that are attached to the network.
- Transfer of information may be single block of information or sequence of blocks as shown in below figure.

- In case of single block of information, we are interested in having the block delivered correctly to destination and also interested in delay experienced in traversing the network.
- In case of sequence of blocks, we are interested not only in receiving the blocks correctly and in right sequence.



Network service can be **Connection-oriented service** or **connectionless service**

Connectionless service:

- Connectionless service is simple with two basic interactions (1) a request to network layer that it send a packet (2) an indication from the network layer that a packet has arrived
- It puts total responsibility of error control, sequencing and flow control on the end system transport layer

Connection-oriented service

- The Transport layer can not request transmission of information until a connection is established between end systems
- Network layer must be informed about the new *flow*
- Network layer maintains state information about the flows it is handling
- During connection set up, parameters related to usage and quality of services may be negotiated and network resources may be allocated
- Connection release procedure may be required to terminate the connection

*It is also possible for a network layer to provide a **choice of services** to the user of network like:*

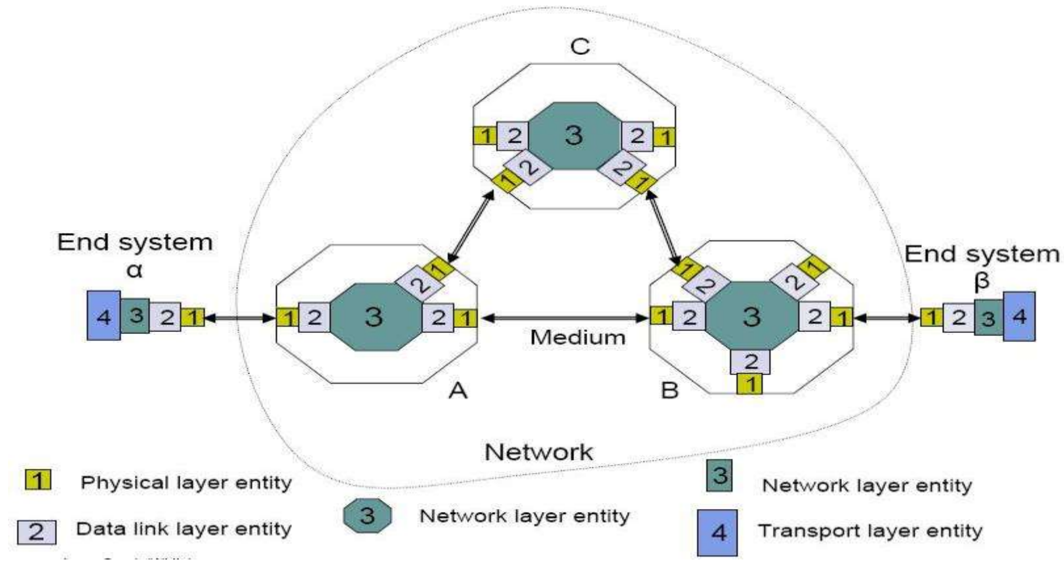
- *best-effort connectionless services*
- *Low delay connectionless services*
- *Connection oriented reliable stream services*
- *Connection oriented transfer of packets with guaranteed delay and bandwidth*

End To End argument for system design

- The end to end argument in system design state that an end-to-end function is best implemented at higher level than at a lower level.
- The reason is that the correct end-to-end implementation requires all intermediate low-level components to operate correctly.
- The higher-level components at the ends are in better position to determine that a function has been carried out correctly and in better position to take corrective action if they have not.

- Keeping the core of the network simple and adding the necessary complexity at the edge enhances the scalability of the network to larger size and scope

Internal network operation



The fig above shows the relation between the service offered by the network and the internal network operation

- The internal operation of the network is *connectionless* if packets are transferred within the network as datagrams
- Each packets are routed independently
- Packets follow different paths from end to end and arrive out of order

- The internal operation of the network is *connection-oriented* if packets follow a virtual circuit along a path that has been established from source to destination.
- Virtual circuit setup is done once, then packets are simply forwarded
- If resources are reserved then bandwidth, delay and loss guarantees are provided.

Network layer essentials

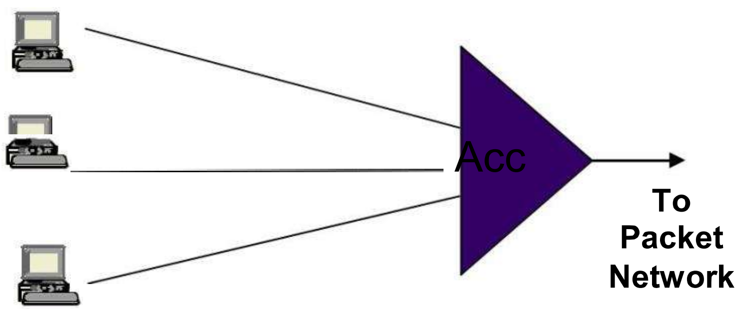
The Functions that need to be carried out at every node in the Network Layer are:-

- **Routing**: mechanisms for determining the set of best paths for routing packets requires the collaboration of network elements
- **Forwarding**: transfer of packets from NE inputs to outputs
- **Priority & Scheduling**: determining order of packet transmission in each NE

Optional: congestion control, segmentation & reassembly, security

Packet Network Topology

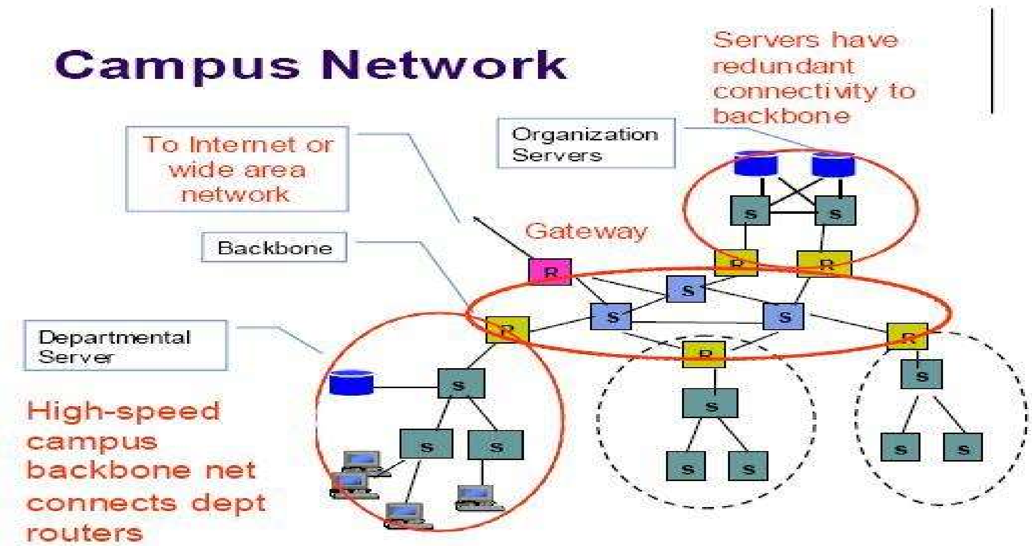
How users access packet networks?



Example -1 – Access Multiplexing

- The diagram above shows an access network with a point to point topology where computer in homes are connected to an access multiplexer located in service provider network.
- The main purpose of the access multiplexer is to combine the bursty traffic flows from individual computers into aggregated flows.
- Eg1. DSL traffic multiplexed at DSL Access Mux
- Eg2. Cable modem traffic multiplexed at Cable Modem Termination System
- Private IP addresses in Home is done using Network Address Translation (NAT).

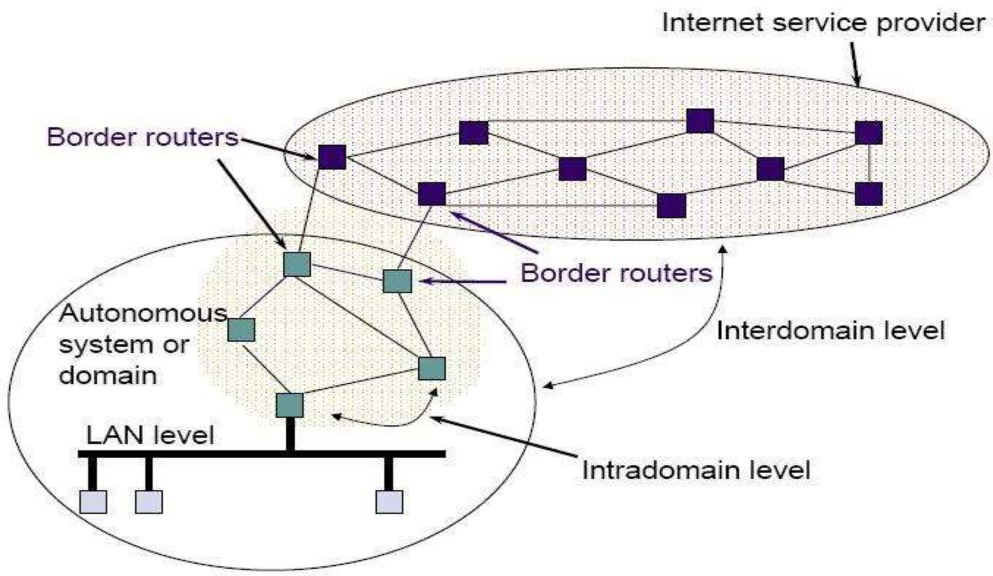
Example -2- Campus Network



- LANs are interconnected through use of LAN switches identified by letter „S" in the figure.
- Resources such as servers and databases that are primarily use are kept within the subnet. This reduces delay in accessing resources.
- Subnet has access to rest of organization through router R that access campus backbone network.
- Subnet uses campus backbone to reach outside world such as Internet through a border router.

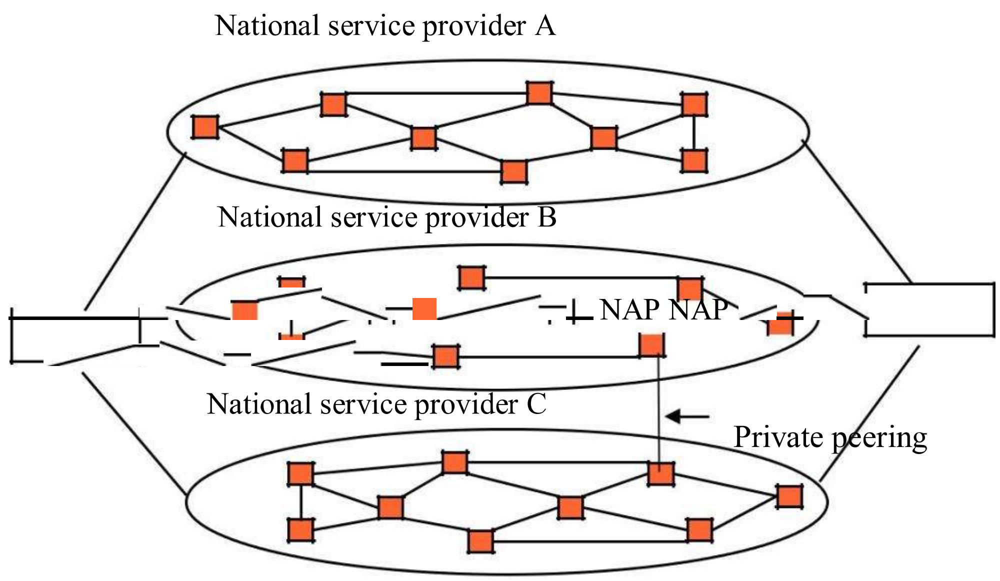
- Servers containing critical resources that are required by entire organization are located in a data center where it can be easily maintained and security can be enforced
- Critical Servers maybe provided with redundant paths to campus backbone network
- The routers in the campus network are interconnected to form the campus backbone network.

Example -3 Connecting to Internet Service Provider



- **<u>Domain:</u>** the routers running the same routing protocol
- **<u>Autonomous System:</u>** one or more domains under the single administration.
- The campus network maybe connected to internet service provider (ISP) through one or more border routers.
- To communicate with other networks, the autonomous system must provide information about its network routes in border routers.
- The border router communicates on an interdomain level, whereas other routers operate at an intradomain level.

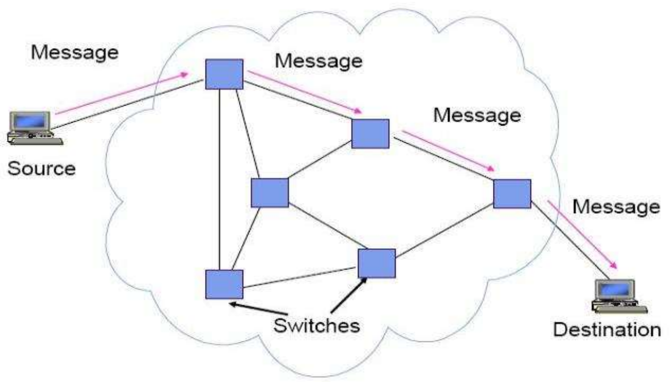
Example -4:- Internet Backbone



- National ISP provides points of presence (POPs) where customer can connect to their network
- The ISP has its own national backbone network for interconnecting its POPs
- The ISPs exchange traffic at public peering points called network access points (NAPs)
- NAP is a collocated set of high-speed routers through which the routers from different ISPs exchange traffic.
- Private peering points can be used to connect ISPs to exchange traffic directly with agreement routing polices.

Datagram and Virtual Circuits

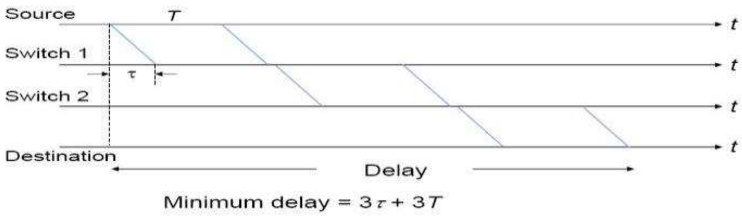
Message Switching



Message Switching

- In message switching, a message is relayed from one switch to another until the message arrives at the destination
- A message switch operates in *store and forward* fashion (a message has to be completely received by the switch before it can be forwarded to next switch)
- At the source each message has header attached to it to provide source and destination address.
- CRC check bits are attached to detect errors
- Each switch performs an error check, and if no errors are detected, the switch examines the header to determine the next hop in the path to the destination.

- Loss of messages may occur when a switch has insufficient buffering to store the arriving message.

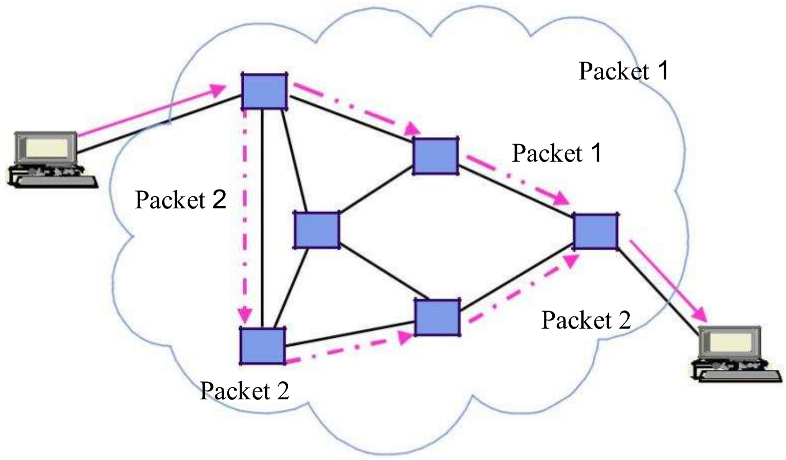


- The above figure shows the minimum delay that is incurred when a message is transmitted over a path that involves two intermediate switches.
- The message has to traverse the link to the first switch
- We assume
  τ – the link has propagation delay in seconds, T – the transmission time
- The message must traverse the link that connect two switches and from second switch to the destination.
- It follows that the minimum delay is $3\tau + 3T$
- In general, the delay incurred in message switching involving L hops is $L\tau + LT$

Disadvantages of message switching

- The probability of error increases with the length of the block. Thus long messages are not desirable
- Not suitable for interactive applications

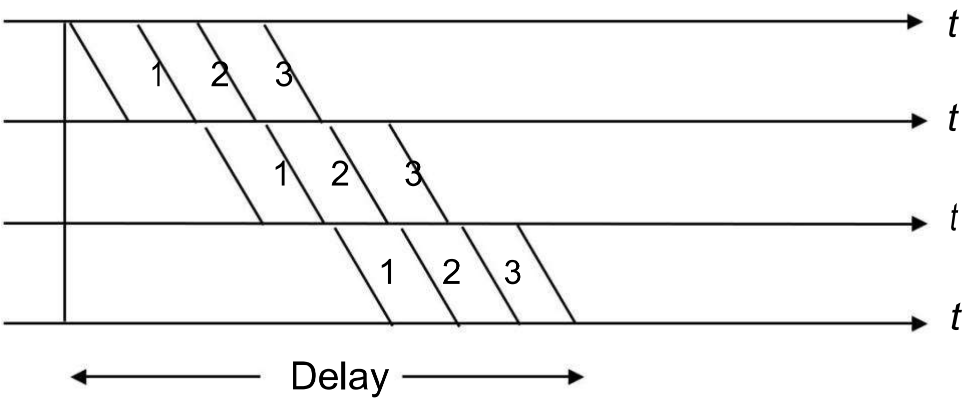
Datagram or Connectionless Packet Switching



- Messages broken into smaller units (packets)
- Source & destination addresses in packet header
- Connectionless, packets routed independently (datagram)
- When a message arrives at the packet switch, the destination address is examined to determine the next hop.
- Packet may arrive out of order
- Re-sequencing maybe required at destination.
- Pipelining of packets across network can reduce delay, increase throughput
- Lower delay that message switching, suitable for interactive traffic

Packet Switching Delay

Assume three packets corresponding to one message traverse same path

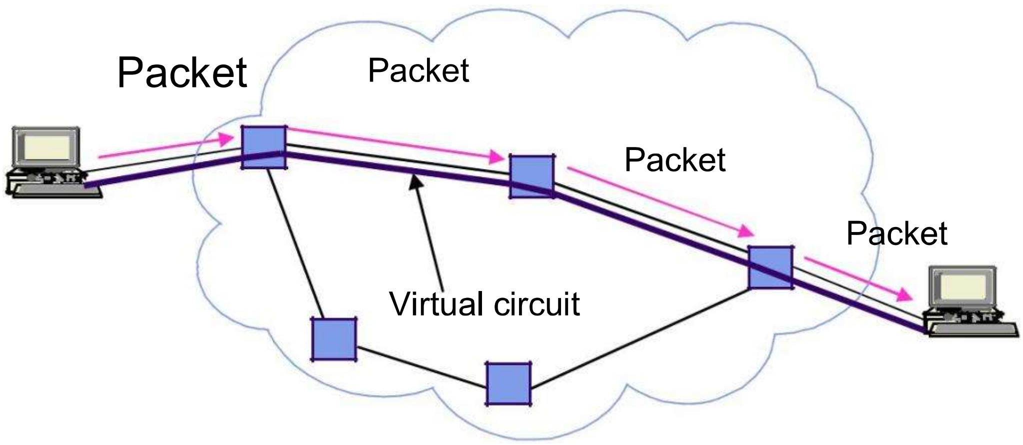


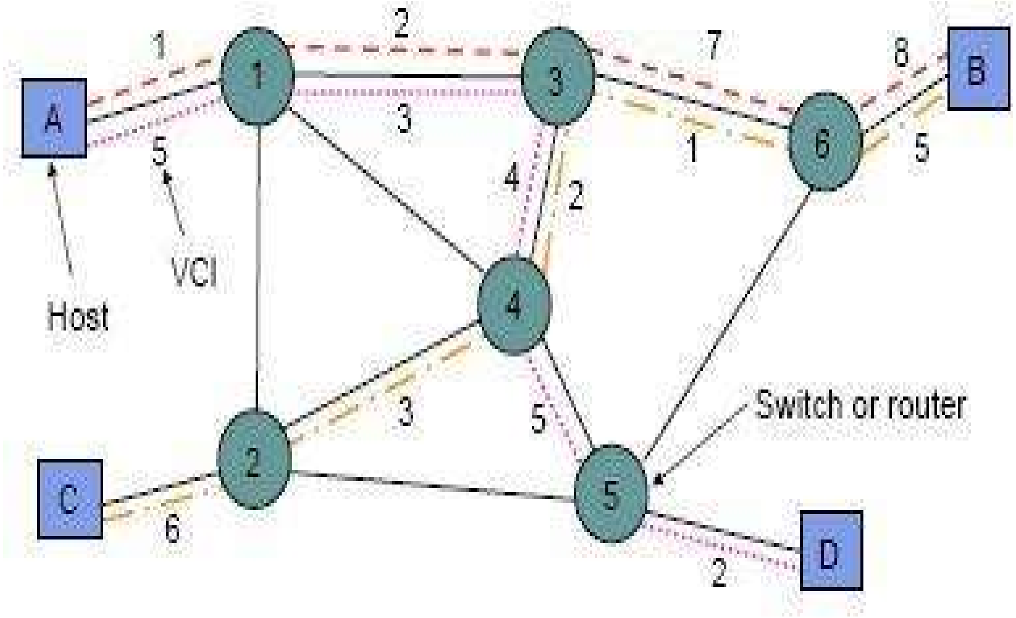
Minimum Delay = $3\tau + 5(T/3)$ (single path assumed)

- Additional queueing delays possible at each link
- Packet pipelining enables message to arrive sooner

In general the delay incurred using a datagram switch involving L hops and consisting of k packets is $L\tau + LP+(k-1)P$

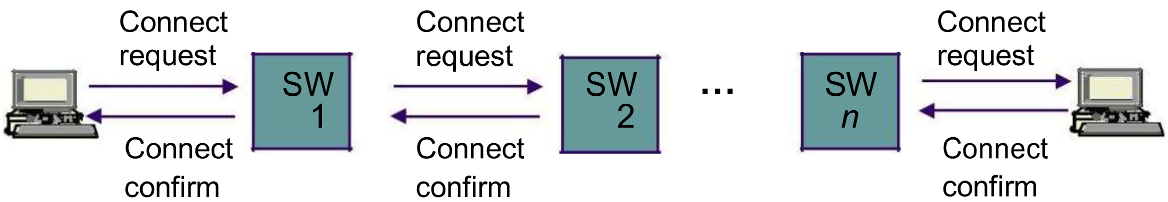
Virtual – Circuit Packet Switching





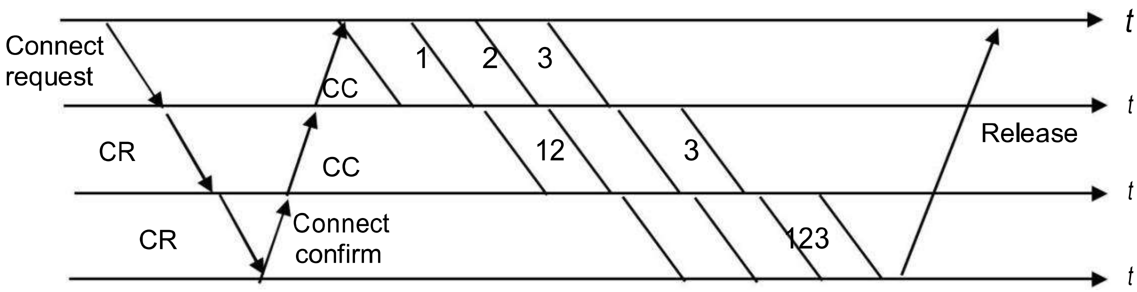
- Before the transmission of packets , it involves establishment of **virtual circuit** between source and destination
- All packets follow the **same path**
- **Abbreviated header** identifies connection on each link
- Packets queue for transmission
- **Variable bit rates possible**

Connection Setup



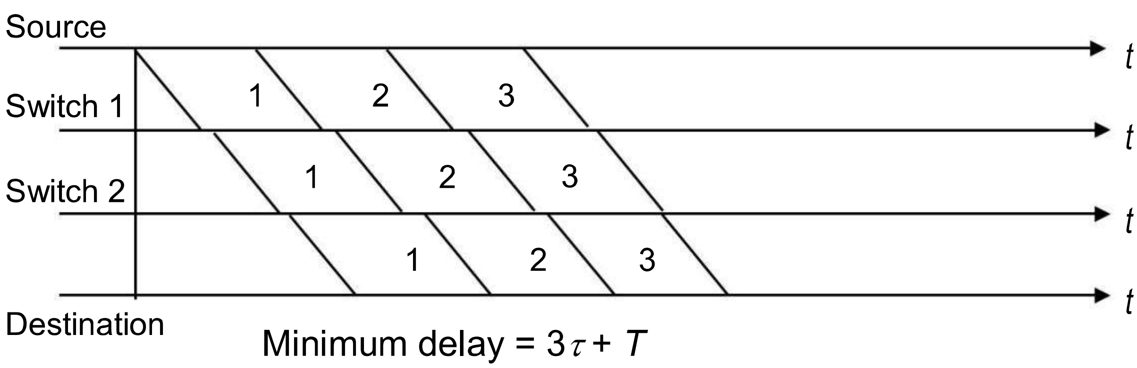
- Signaling messages propagate as route is selected
- Signaling messages identify connection and setup tables in switches
- Typically a connection is identified by a local tag, **Virtual Circuit Identifier (VCI)**
- Each switch only needs to know how to relate an **incoming tag in one input to an outgoing tag** in the corresponding output
- Once tables are setup, packets can flow along path

Connection Setup Delay



- Connection setup delay is incurred before any packet can be transferred
- Delay is acceptable for sustained transfer of large number of packets
- This delay may be unacceptably high if only a few packets are being transferred
- The minimum delay in virtual circuit packet switching is similar to that in datagram packet switching, except for an additional delay required to setup the virtual circuit.

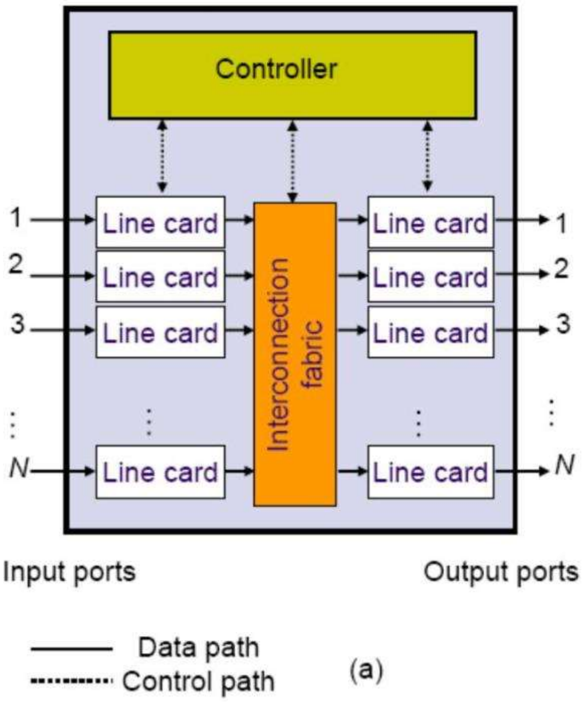
Cut-Through switching



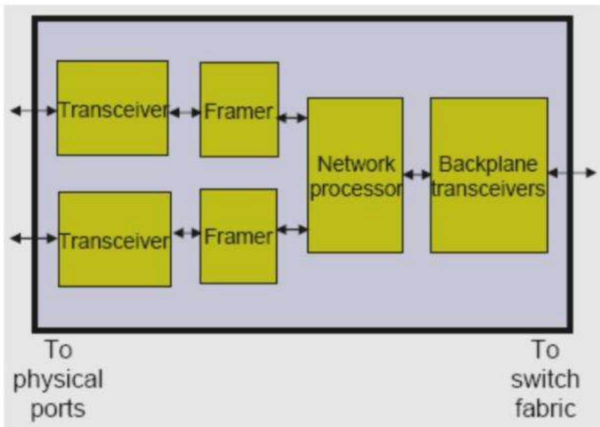
- It is the modified form of virtual circuit packet switching
- Some networks perform error checking on header only, so packet can be forwarded as soon as header is received & processed
- Delays reduced further with cut-through switching

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Structure of a Packet Switch



(a)



(b) Line card

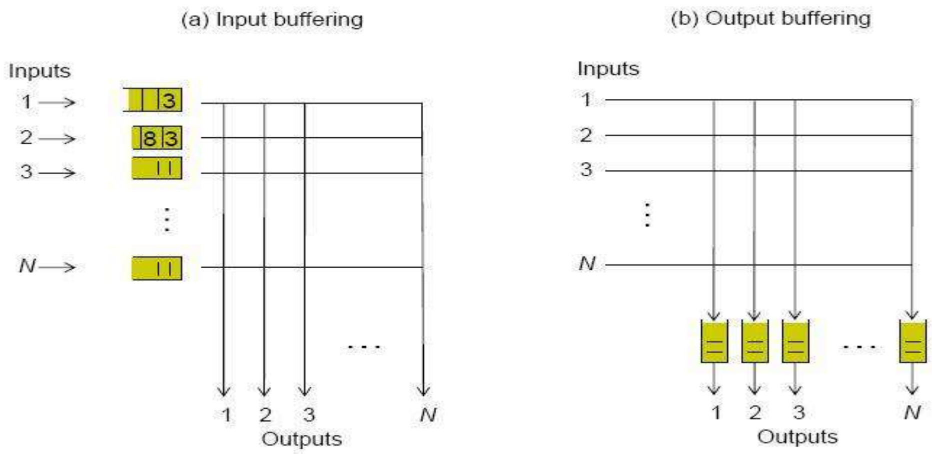
- A packet switch performs two functions 1) routing 2) forwarding.

- ♦ **Routing** functions uses algorithm to find a path to each destination and store the result in routing table.
- ♦ **Forwarding** function processes each incoming packet from an input port and forwards the packet to appropriate output port based on the information stored in routing table.

- The above fig (a) shows a generic packet switch consisting of input ports, output ports, an interconnection fabric and switch controller.

- **Input ports** and **output ports** are normally paired.

- **Line card** contains several input, output ports so that the capacity of the link connecting the line card to the interconnection fabric , which is typically of high speed, and is fully utilized.
- Line card is concerned with symbol timing, line coding, framing, physical addressing and error checking.
- The line card is made up of various chipsets as shown in fig (b) .
- The programmable network processor performs packet-related tasks such as table lookup and packet scheduling.
- The **Controller** in a packet switch contains a general-purpose processor to carry out a number of control and management functions.
- The controller also communicates with line card and the interconnection fabric
- The function of the **Interconnection fabric** is to transfer packets between the line cards.
- If there are high speed line cards the interconnection fabric is likely to be the bottleneck, since all traffic go through it.
- A **bus type interconnection structure** (whereby packets are transmitted serially) does not scale to large size, since the speed of the bus has to be about N times faster than the port speed.
- A **cross bar interconnection fabric** can transfer packets in parallel between input ports and output ports.
- The buffers need to be added to the crossbar to accommodate packet contention.
- The buffers can be located at input ports or output ports as shown in figure below



- Only one packet is allowed to proceed to a particular output in case of input buffering.
- Input buffering causes a problem **head-of-line (HOL) blocking**.
- Consider a situation where there are two packets at input buffer 2 as shown in fig. above.
- The first packet would like to go to output 3 and the second packet to output 8.

- Suppose that the packet from input buffer 1 would like to go output 3 at the same time.
- Suppose that the fabric arbiter decides to transmit the packet from input buffer 1. Then the first packet from input buffer 2 needs to wait until output 3 has transferred the packet from input buffer1. Meanwhile, the second packet has to wait behind the first packet even though output 8 is idle.
- This results in performance degradation of crossbar with input buffering.
- The problem of the first packet holding back other subsequent packets behind it is called **head-of-line (HOL) blocking**.
- One way to eliminate HOL blocking is to provide N separate input buffers at each input port. Such an input buffer is called virtual output buffer.

Routing in Packet Networks

- **Routing:** it is concerned with determining feasible path for packets to follow from each source to destination.
- Which path is **"best**"? The term **best** depends on the objective function that network operator tries to optimize which maybe Min delay, Min hop, Max bandwidth, Min cost, Max reliability

Goals of routing algorithm

1. Rapid and accurate delivery of packets
2. Adaptability to changes in network topology resulting from node or link failures.
3. Adaptability to varying source-destination traffic loads
4. Ability to route packets away from congested links
5. Ability to determine the connectivity of the network
6. Ability to avoid routing loops.
7. Low overhead

Classification of routing algorithms

Static vs Dynamic Routing

Static	Dynamic
1. The paths are pre-computed by a host and are loaded into routing table. 2. The paths are fixed for long duration of time. 3. static routing is good when • network size is small, • traffic load does not change variably • network topology is fixed 4. Does not scale to large network 5. Disadvantage is its inability to react rapidly to network failures.	1. Each node computes the best path by communicating with its neighbors. 2. each node continuously learn the state of network by communicating with its neighbors 3. Adapt to changes in network conditions 4. Scales well 5. Disadvantage is added complexity in the node.

Centralized vs Distributed Routing

Centralized Routing	Distributed Routing
1. Network control center computes the path and uploads the information 2. does not scale well 3. consistent results 4. Problems adapting to frequent topology changes	1. Nodes cooperate by means of message exchanges and perform their own computations. 2. scale well 3. Inconsistent results due to loops. 4. Adapts to topology and other changes

Hierarchical Routing

- The hierarchical approach reduces the size of the routing tables at the routers in assigning the addresses.
- Hosts that are near each other (i.e. a group) should have addresses that have common prefixes. The routers examine only part of the address (i.e.. the prefix) to decide how a packet should be routed.
- Figure below gives an example of hierarchical address assignment and a flat address assignment.

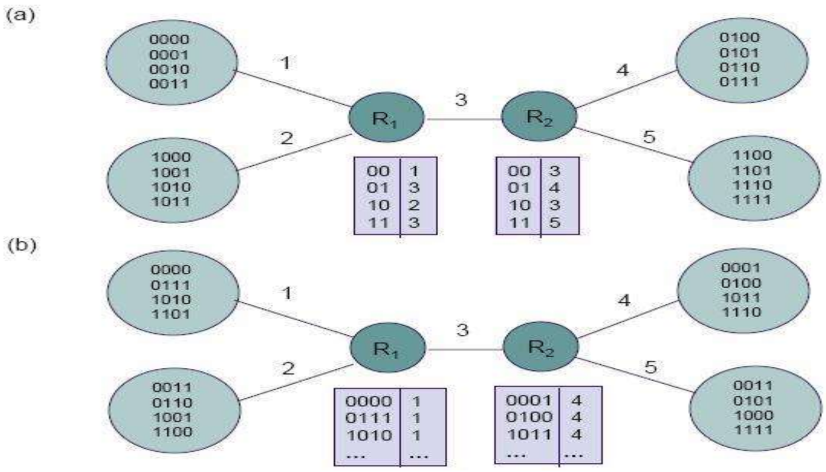


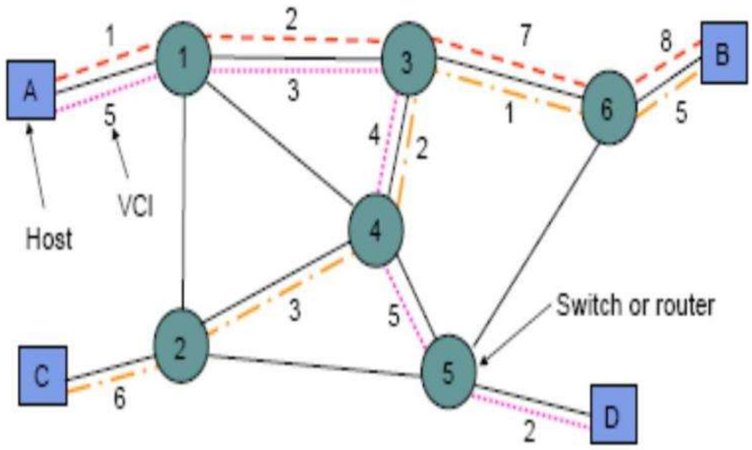
FIG (a) Hierarchical Routing and (b) Flat Routing

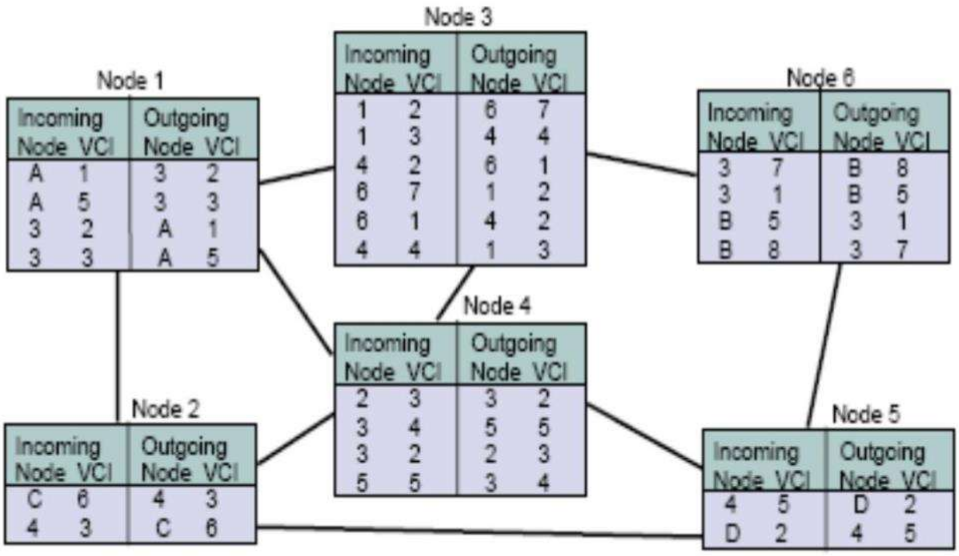
- In figure (a) the hosts at each of the four sites have the same prefix. Thus the two routers need only maintain tables with four entries as shown.
- On the other hand, if the addresses are not hierarchical (Figure 7.27b), then the routers need to maintain 16 entries in their routing tables.

ROUTING TABLES

Routing Table for Virtual circuit networks

- virtual circuit identifier determines the destination





Node 1

Incoming Node	Incoming VCI	Outgoing Node	Outgoing VCI
A	1	3	2
A	5	3	3
3	2	A	1
3	3	A	5

Node 3

Incoming Node	Incoming VCI	Outgoing Node	Outgoing VCI
1	2	6	7
1	3	4	4
4	2	6	1
6	7	1	2
6	1	4	2
4	4	1	3

Node 6

Incoming Node	Incoming VCI	Outgoing Node	Outgoing VCI
3	7	B	8
3	1	B	5
B	5	3	1
B	8	3	7

Node 4

Incoming Node	Incoming VCI	Outgoing Node	Outgoing VCI
2	3	3	2
3	4	5	5
3	2	2	3
5	5	3	4

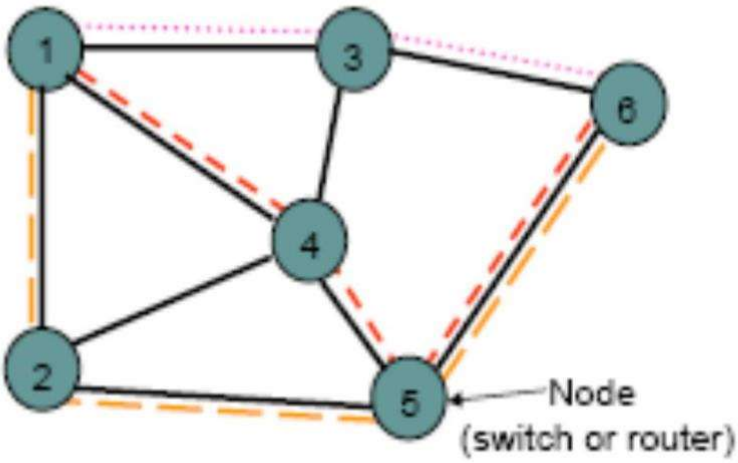
Node 2

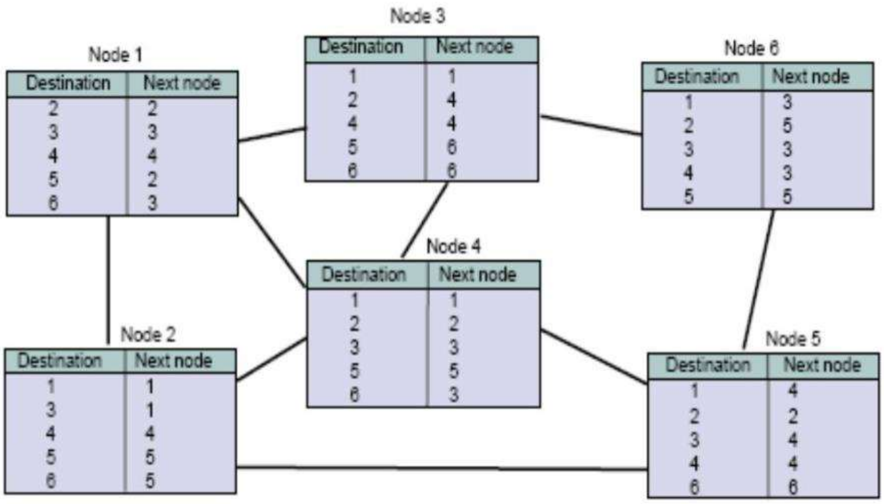
Incoming Node	Incoming VCI	Outgoing Node	Outgoing VCI
C	6	4	3
4	3	C	6

Node 5

Incoming Node	Incoming VCI	Outgoing Node	Outgoing VCI
4	5	D	2
D	2	4	5

Routing table for datagram networks





Specialized Routing

Flooding

- **Principle of flooding:** a node (or a packet switch) forwards an incoming packet to all ports except to the one it arrived on.
- Each node (a switch) performs the flooding process such that the packet will reach the destination as long as at least one path exists between the source and the destination.
- Flooding is a useful
  - when the information in the routing tables is not available, such as during system startup,
  - when survivability is required, such as in military networks.
  - when the source needs to send a packet to all hosts connected to the network (i.e., broadcast delivery)

- Flooding generates vast numbers of duplicate packets

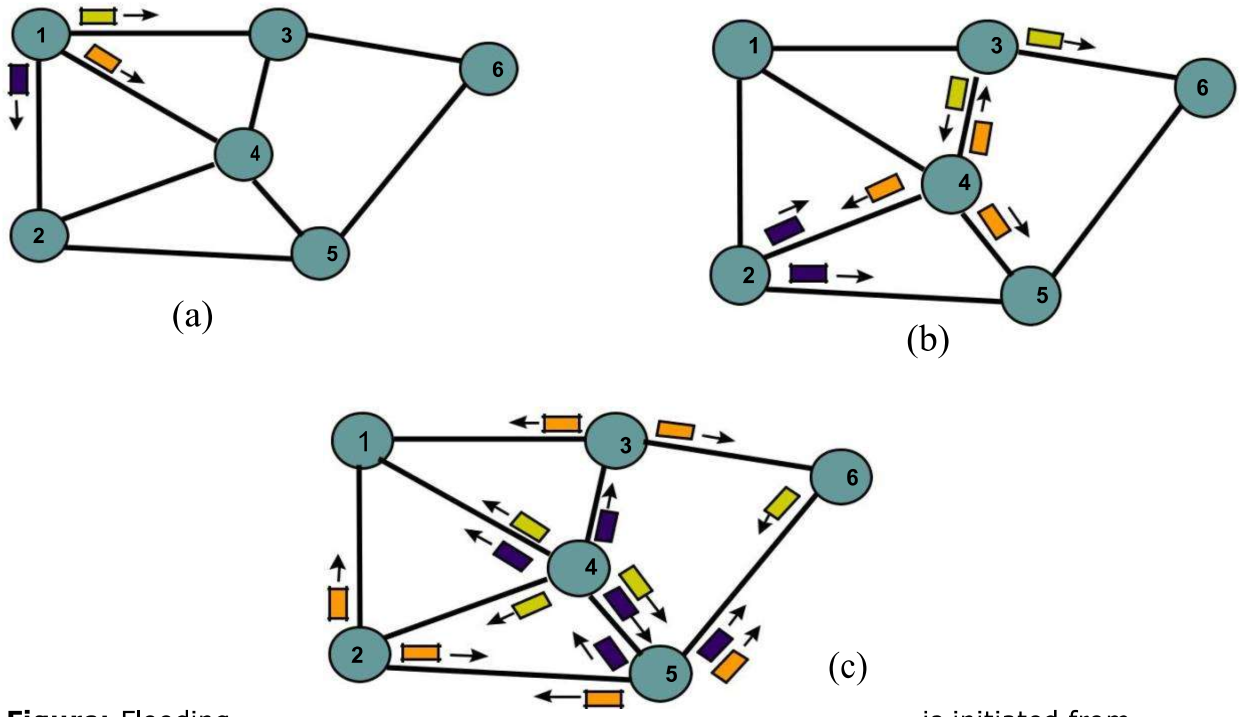


Figure: Flooding is initiated from Node 1: (a) Hop 1 transmissions (b) Hop 2 transmissions (c) Hop 3 transmissions

In figure, initially one packet arriving at node 1 triggers three packets to nodes 2, 3, and 4. In the second phase nodes 2, 3, and 4 send two, two, and three packets respectively. These packets arrive at nodes 2 through 6. In the third phase 15 more packets are generated giving a total of 25 packets after three phases.

The flooding needs to be controlled so that packets are not generated excessively.

How to control this?

There are three methods to reduce the resource consumption in the network

1) **Use a time-to-live (TTL) field in each packet**.
   - When the source sends a packet, the time-to-live field is initially set to some small number.
   - Each node decrements the field by one before flooding the packet. If the value reaches zero, the switch discards the packet.
   - To avoid unnecessary waste of bandwidth, the time-to-live should ideally be set to the minimum hop number between two furthest nodes (called the diameter of the network).
2) **Add an identifier before flooding**
   - Every node adds an identifier before flooding
   - When a node identifies a packet that contains the identifier of the switch, it discards the packet.
   - This method effectively prevents a packet from going around a loop.
3) **Have a unique sequence number**
   •Each packet from the given source is uniquely identified with a sequence number

- When a node receives a packet, it records the source address and the sequence number
- If node discovers that packet has already visited the node, it will discard the packet

Deflection Routing

- Deflection routing was first called as Hot-potato routing.
- It requires the network to provide multiple paths for each source-destination pair.
- Each switch first tries to forward a packet to the preferred port. If the preferred port is busy or congested, the packet is deflected to another port.
- Deflection routing works well in a regular topology.

Example :- <u>Manhattan Street network:</u>

- Each column represents an avenue, and each row represents a Street.
- Each switch is labeled (i,j) where i denotes the row number and j denotes the column number.
- The links have directions that alternate for each column or row.
- If switch (0,2) would like to send a packet to switch (1,0), the packet could go two left and one down. However, if the left port of switch (0,1) is busy (see Figure), the packet will be deflected to switch (3,1). Then it can go through switches (2,1), (1,1), (1,2), (1,3) and eventually reach the destination switch (1,0).
- One advantage of deflection routing is that the switch can be bufferless, since packets do not have to wait for a specific port to become available. Since packets can take alternative paths, deflection routing cannot guarantee in-sequence delivery of packets.
- Deflection routing is used to implement many high-speed packet switches where the topology is very regular and high-speed buffers are relatively expensive compared to deflection routing logic.

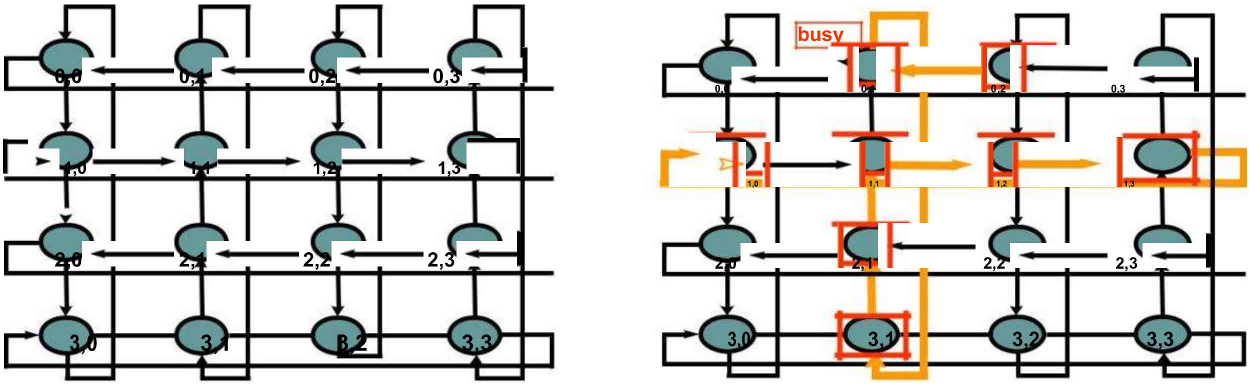


Figure: Manhattan street network

Shortest Path Routing

Shortest path algorithms are used to determine the shortest path based on following metrics.
The shortest path routing are Bellman-Ford algorithm and Dijkstra"s algorithm

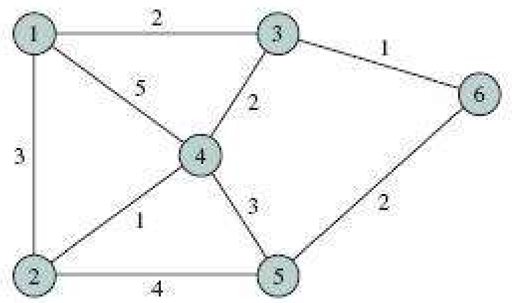
- Possible metrics
  - Hop count
  - Reliability:
  - Delay: sum of delays along path
  - Bandwidth: "available capacity" in a path
  - Load: Link & router utilization along path
  - Cost: $$$

Bellman-Ford Algorithm or Ford-Fulkerson algorithm

Principle: - If each neighbor of node A knows the shortest path to node Z, then node A can determine its shortest path to node Z by calculating the cost/distance to node Z through each of its neighbors and picking the minimum.

Bellman-Ford algorithm

- Consider computations <u>for one destination</u> d
- Initialization
  - Each node table has 1 row for destination d
  - Distance of node d to itself is zero: Dd=0
  - Distance of other node j to d is infinite: $Dj=\propto$, for $j\neq$ d
  - Next hop node nj = -1 to indicate not yet defined for $j \neq$ d
- Send Step
  - Send new distance vector to immediate neighbors across local link
- Receive Step
  - <u>At node j, find the next hop that gives the minimum distance to d</u>,
    - Minj { Cij + Dj }
    - Replace old (nj, Dj(d)) by new (nj*, Dj*(d)) if new next node or distance
  - Go to send step



For the above figure **Apply Bellman-ford algorithm** to find both minimum cost from each node to the destination node 6 and the next node along the shortest path.

Each node I maintains an entry (n,D_i), where n is the next node along the current shortest path and D_i is the current minimum cost from node i to destination.

Initially all nodes, other than the destination node 6, are at infinite cost (distance) to node 6. Node 6 informs its neighbors it is distance 0 from itself.
Iteration 1:- Node 3 finds that it is connected to node 6 with cost of 1. Node 5 finds it is connected to node 6 at a cost of 2. nodes 3 and 5 update their entries and inform their neighbors.
Iteration 2:- Node1 finds it can reach node 6, via node 3 with cost 3. Node 2 finds it can reach node 6, via node 5 with cost 6. Node 4 finds it has paths via nodes 3and 5,

with costs 3 and 7 respectively. Node 4 selects the path via node 3. Nodes 1, 2, and 4 update their entries and inform their neighbors.

Iteration 3:- Node 2 finds that it can reach node 6 via node 4 with distance 4. Node 2 changes its entry to (4,4) and informs its neighbors.

Iteration 4:- nodes 1,4, and 5 process the new entry from node 2 but do not find any new shortest paths. The algorithm has converged.

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1, ∞)	(-1, ∞)	(-1, ∞)	(-1, ∞)	(-1, ∞)
1	(-1, ∞)	(-1, ∞)	(6, 1)	(-1, ∞)	(6,2)
2	(3,3)	(5,6)	(6, 1)	(3,3)	(6,2)
3	(3,3)	(4,4)	(6, 1)	(3,3)	(6,2)

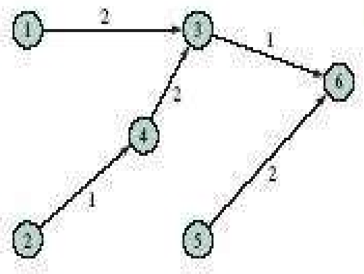


FIGURE 7.29 Shortest-path tree to node 6

What happens when a link fails?

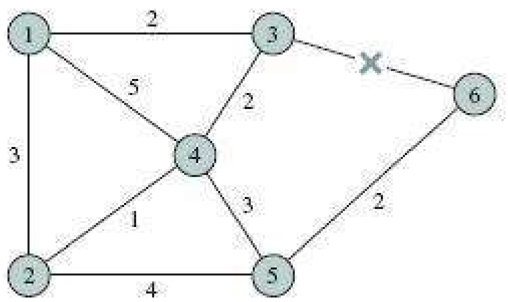


FIGURE 7.30 New network topology following break from node 3 to 6

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
1	(3,3)	(4,4)	(4, 5)	(3,3)	(6,2)
2	(3,7)	(4,4)	(4, 5)	(2,5)	(6,2)
3	(3,7)	(4,6)	(4, 7)	(5,5)	(6,2)
4	(2,9)	(4,6)	(4, 7)	(5,5)	(6,2)

Counting to Infinity Problem

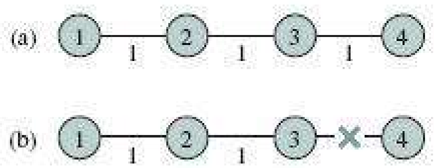


FIGURE 7.31 Topology before and after link failure

Nodes believe best path is through each other (Destination is node 4)

Update	Node 1	Node 2	Node 3
Before break	(2,3)	(3,2)	(4, 1)
After break	(2,3)	(3,2)	(2,3)
1	(2,3)	(3,4)	(2,3)
2	(2,5)	(3,4)	(2,5)
3	(2,5)	(3,6)	(2,5)
4	(2,7)	(3,6)	(2,7)
5	(2,7)	(3,8)	(2,7)
...	...	...	...

Problem: Bad News Travels Slowly

Remedies

- Split Horizon
  - Do not report route to a destination to the neighbor from which route was learned
- Poisoned Reverse
  - Report route to a destination to the neighbor from which route was learned, but with infinite distance
  - Breaks erroneous direct loops immediately
  - Does not work on some indirect loops

Split Horizon with Poison Reverse

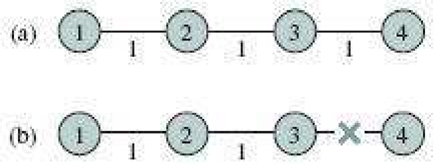


FIGURE 7.31 Topology before and after link failure

Nodes believe best path is through each other

Update	Node 1	Node 2	Node 3	
Before break	(2, 3)	(3, 2)	(4, 1)	
After break	(2, 3)	(3, 2)	(-1, ∞)	Node 2 advertizes its route to 4 to node 3 as having distance infinity; node 3 finds there is no route to 4
1	(2, 3)	(-1, ∞)	(-1, ∞)	Node 1 advertizes its route to 4 to node 2 as having distance infinity; node 2 finds there is no route to 4
2	(-1, ∞)	(-1, ∞)	(-1, ∞)	Node 1 finds there is no route to 4

Source Routing

- In source routing, path to the destination is determined by the source.
- Source routing works in either datagram or virtual-circuit packet switching.
- Before sending packet, the source has to know the path to the destination in order to include the path information in the packet header.
- The path information contains the sequence of nodes to traverse and should give sufficient information to intermediate node to forward the packet.
- The below fig. shows how source routing works in datagram network.
- Intermediate switch reads header and removes the address identifying the node and forwards the packets to next node.
- Source host selects path that is to be followed by a packet
    - Strict: Here, the source specifies the address of each node along the path to the destination.
    - Loose: When source knows partial information of network topology, the source can use loose source routing.

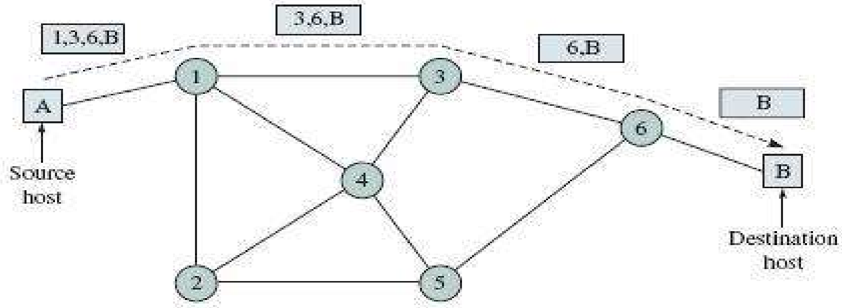


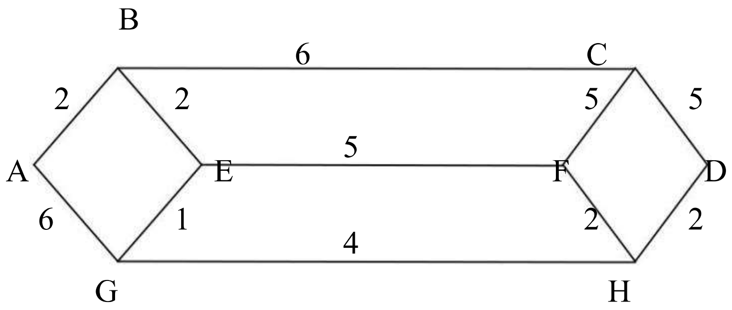
FIGURE 7.36 Example of source routing

1. What are the different network services provided by network layer of the packet switching networks? **(10M)**
2. Why is packet switching more suitable than message switching for interactive applications? Compare the delays in datagram packet switching and message switching. **(10 M)**
3. Compare the Bellman –Ford algorithm and Dijkstra's algorithm for finding the shortest paths from a source node to all other nodes in a network.

   (6M)
4. Differentiate between virtual circuits and datagram subnets.

   (5M)
5. Define routing and forwarding. What are the goals of routing algorithm?

   (6M)
6. Using Dijkstra's algorithm find the shortest path between A and D



   (5M)
7. Explain hierarchical routing. **(5M)**
8. What are the drawbacks of flooding? **(4M)**

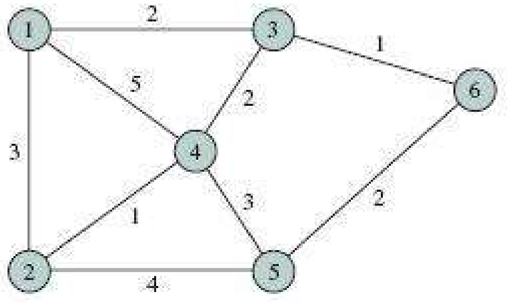
   Define routing. Explain Bellman-Ford routing algorithm with necessary illustration. What are the drawbacks of this algorithm?

   (10 M)
9. Good news spreads fast; bad news propagates slowly in bellman ford algorithm. Explain with an example. **(5M)**
10. Explain Count to infinity problem. **(6M)**
11. Consider the network in fig

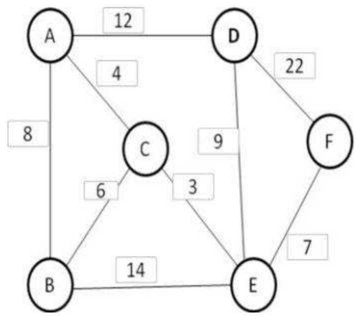
    (i) Use the Dijkstra's algorithm to find the set of shortest path from node 4 to other node.

    (ii)Find the set of associated routing table entries

(iii) find the shortest path from node 5 to other destination node and Find the shortest path tree from 5 to other nodes. **(10 M)**

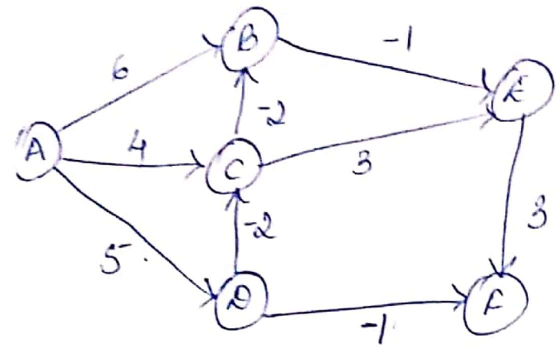


13. Differentiate between virtual circuit and datagram’s. Explain routing table for both. **(10 M)**
14. Differentiate between connectionless packet switching and virtual packet switching. **(8 M)**
15. Explain briefly the structure of a generic packet switch, with the help of the diagram **(7 M)**
16. With examples, Differentiate between datagram, virtual circuits and packet switching. **(6 M)**
17.
18. Explain Dijikstras algorithm. Solve the following problem **(10 M)**



Bellman Ford Algorithm

1



$n = 6$

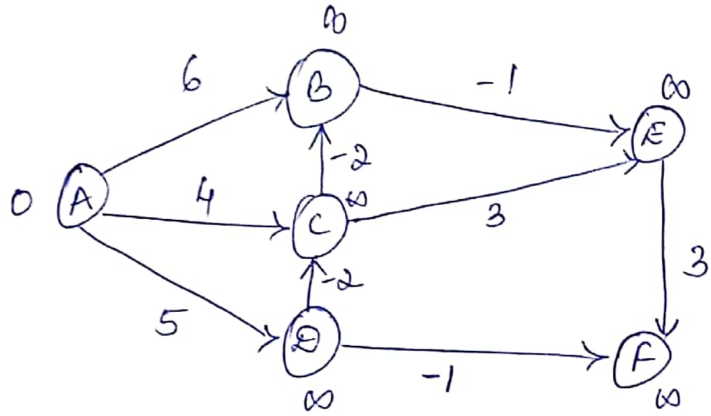
$\therefore (n-1)$ iterations

Edges present : (A,B) (A,C) (A,D) (B,E) (C,E) (D,C) (D,F) (E,F) (C,B)

Step 1:

Initialize the cost from source (node A) to A = 0.

Initialize the cost from source (node A) to all other nodes $= \infty$



using the formula:

if $d[u] + c[u,v] < d[v]$

then $d[v] = d[u] + c[u,v]$

Iteration 1:

Starting from node A to B
(u = A, v = B)

$d[u] + c[u,v] < d[v]$

$0 + 6 < \infty$

then

$d[v] = 6$

for (A, C)

$(0+4) < \infty$

$\underline{\underline{d[v] = 4}}$

for (A, D)

$(0+5) < \infty$

$d[v] = 5$

for (B, E)

$(6-1) < \infty$

$d[v] = 5$

for (C, E)

$(4+3) < \cancel{\infty}\ 5$

retain $d[v] = 5$

for (D, C)

$(5-2) < 4$

$d[v] = 3$

for (D, F)

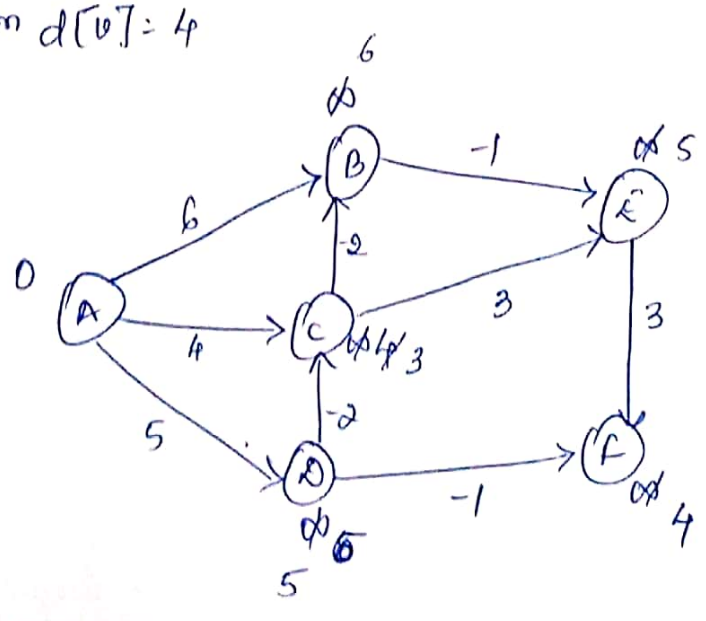
$(5-1) < \infty$

$d[v] = 4$

for (E, F)

$(5+3) < 4$

retain $d[v] = 4$



II iteration :-

for (A,B) → no updation

for (A,C) → no —"—

for (A,D) → no —"—

for (B,E) →

$(1-1) < 5$

~~so dd~~

So $d[v] = 0$.

for (C,E) → no updation.

for (D,C) → —"—

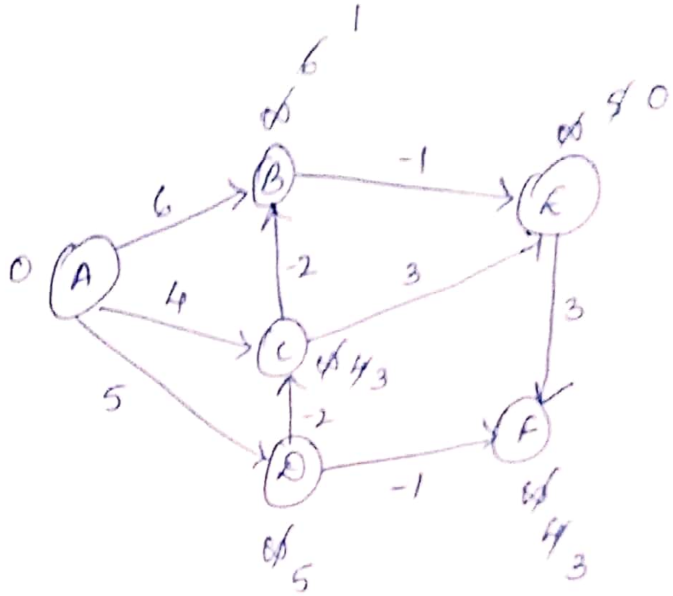
for (D,F) → —"—

for (E,F) →

$(0+3) < 4$

So $d[v] = 3$

for (C,B) → no updation

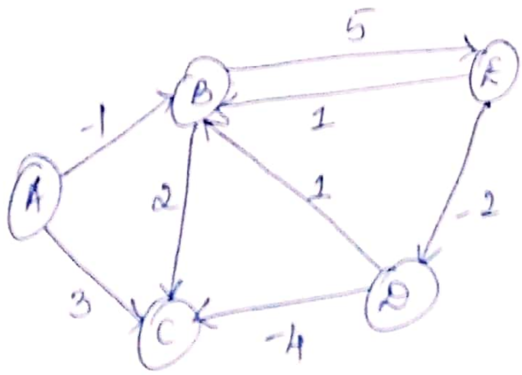


III iteration :-

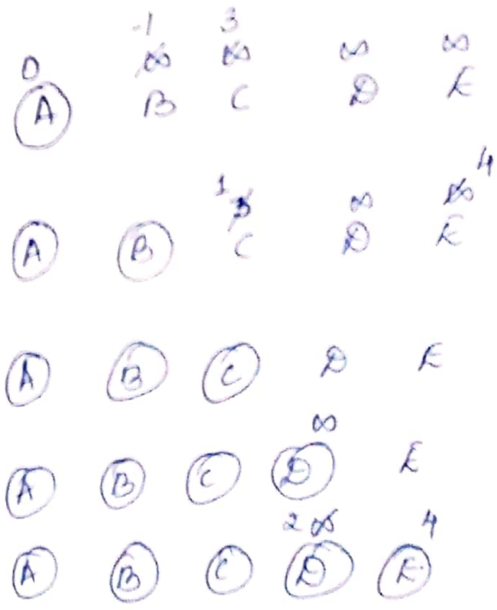
no changes

$\therefore$ $\boxed{A \to 0,\ B \to 1,\ C \to 3,\ D \to 5,\ E \to 0,\ F \to 3.}$

(2)



I



II

$\overset{0}{A} \quad \overset{-1}{B} \quad \overset{1}{C} \quad \overset{2}{D} \quad \overset{4}{E}$

$\overset{0}{A} \quad \overset{-1}{B} \quad \overset{1}{C} \quad \overset{2}{D} \quad \overset{4}{E}$

$A \quad \overset{-1}{B} \quad \overset{1}{C} \quad \overset{2}{D} \quad \overset{4}{E}$

$\overset{0}{A} \quad \overset{-1}{B} \quad \overset{1}{C} \quad \overset{2}{D} \quad \overset{4}{E}$

$\overset{0}{A} \quad \overset{-1}{B} \quad \overset{-2\cancel{1}}{C} \quad \overset{2}{D} \quad \overset{4}{E}$

III

$\overset{0}{A} \quad \overset{-1}{B} \quad C^{-2} \quad D^{2} \quad E^{4}$

$\overset{0}{A} \quad \overset{-1}{B} \quad C^{-2} \quad \overset{2}{D} \quad E^{4}$

$\overset{0}{A} \quad \overset{-1}{B} \quad C^{-2} \quad \overset{2}{D} \quad E^{4}$

$\overset{0}{A} \quad \overset{-1}{B} \quad \overset{-2}{C} \quad \overset{2}{D} \quad \overset{4}{E}$

$\overset{0}{A} \quad \overset{-1 \checkmark}{B} \quad C^{-2} \quad \overset{2}{D} \quad E^{4}$

No changes – reached solution

$$\boxed{\begin{array}{l} A \to 0 \\ B \to -1 \\ C \to -2 \\ D \to 2 \\ E \to 4 \end{array}}$$

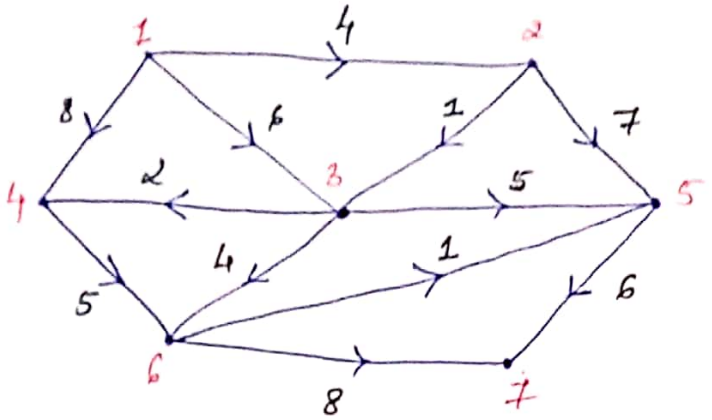
Dijkstra's Algorithm:-

- shortest path b/w the node in a graph.
  - used to find the shortest path b/w the source node and every other node; b/w a single node to a single destination node

Algorithm:- for a given graph g, with single source vertex

- Create a set ie shortest path tree set that keeps track of vertices included in the shortest path tree - ie whose min distance from source is calculated & finalized.
  Initially this set is empty.
- Assign a distance value to all vertices in the i/p graph. Initialize all distance values as infinite.
  Assign distance values as 0 for the source vertex so that it is picked first.
- While sptSet doesn't include all vertices
  - pick a vertex u which is not ~~there~~ there in spt set & has a minimum distance value.
  - Include u to sptSet
  - Update distance value of all adjacent vertices of u.
    To update the distance values, iterate through all adjacent vertices.
  - for every adjacent vertex v, if the sum of distance value of u & weight of edge u-v is ~~the~~ less than the distance value of v, then update the distance value of v.

Find the shortest path using Dijkstra's Algorithm:-



First iteration :-

$P = \{1\}$ and $t_j = q_{1j}$ for $j = 2,3,4,5,6,7$

$t_2 = 4,\ t_3 = 6,\ t_4 = 8,\ t_5 = \infty,\ t_6 = \infty,\ t_7 = \infty$

Step 1
here t_j is minimum for $j = 2$, where $t_2 = 4$

- label the arc$(1,2)$ as p_1
- $P = \{1,2\}$

Step 2 $P = \{1,2\}$ and $t_2 = 4$

$$\text{new } t_3 = \min\{t_3,\ t_2 + q_{23}\}$$
$$= \min\{6,\ 4+1\} = 5$$

$$\text{new } t_4 = \min\{t_4,\ t_2 + q_{24}\}$$
$$= \min\{8,\ 4+\infty\} = 8$$

$$\text{new } t_5 = \min\{t_5,\ t_2 + q_{25}\}$$
$$= \min\{\infty,\ 4+7\} = 11$$

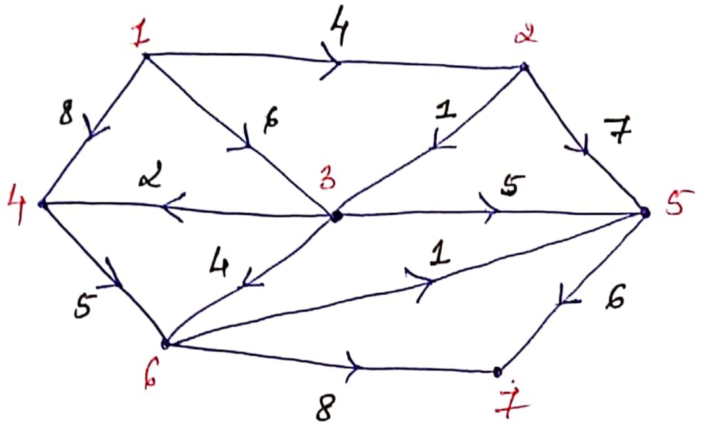
$$\text{new } t_6 = \min\{t_6 ,\ t_2 + q_{26}\} = \infty$$

$$\text{new } t_7 = \min\{t_7,\ t_2 + q_{27}\} = \infty$$

Second iteration :-

$P = \{1,2\}$ and $t_2 = 4$

Find the shortest path using Dijkstra's Algorithm:-



First iteration :-

$P = \{1\}$ and $t_j = q_{1j}$ for $j = 2,3,4,5,6,7$

$t_2 = 4, t_3 = 6, t_4 = 8, t_5 = \infty, t_6 = \infty, t_7 = \infty$

Step 1

here t_j is minimum for $j = 2$, where $t_2 = 4$

- label the arc $(1,2)$ as p_1
- $P = \{1,2\}$

Step 2 $P = \{1,2\}$ and $t_2 = 4$

new $t_3 = \min\{t_3, t_2 + q_{23}\}$

$= \min\{6, 4+1\} = 5$

new $t_4 = \min\{t_4, t_2 + q_{24}\}$

$= \min\{8, 4+\infty\} = 8$

new $t_5 = \min\{t_5, t_2 + q_{25}\}$

$= \min\{\infty, 4+7\} = 11$

new $t_6 = \min\{t_6, t_2 + q_{26}\} = \infty$

new $t_7 = \min\{t_7, t_2 + q_{27}\} = \infty$

Second iteration :-

$P = \{1,2\}$ and $t_2 = 4$

Scanned with CamScanner

$t_3 = 5,\ t_4 = 8,\ t_5 = 11,\ t_6 = \infty,\ t_7 = \infty$

Step1 : min being $t_3 = 5$, the arc (2,3) has the least weight

- label the arc (2,3) as p_2.
- $P = \{1,2,3\}$

Step2 : $P = \{1,2,3\}$, $t_2 = 4$, $t_3 = 5$.

$$\text{new } t_4 = \min\{t_4, t_3 + q_{34}\}$$
$$= \min\{8, 5+2\} = 7$$

$$\text{new } t_5 = \min\{t_5, t_3 + q_{35}\}$$
$$= \min\{11, 5+5\} = 10$$

$$\text{new } t_6 = \min\{t_6, t_3 + q_{36}\}.$$
$$= \min\{\infty, 5+4\} = 9$$

$$\text{new } t_7 = \min\{\infty, 5 + q_{37}\}$$
$$= \infty$$

Third iteration:

$P = \{1,2,3\}$, $t_2 = 4$, $t_3 = 5$, Also $t_4 = 7$, $t_5 = 10$, $t_6 = 9$, $t_7 = \infty$

Step1 : min being $t_4 = 7$, the arc (3,4) as p_3, the least weight

- label the arc (3,4) as p_3
- $P = \{1,2,3,4\}$

Step2: $P = \{1,2,3,4\}$, $t_2 = 4$, $t_3 = 5$, $t_4 = 7$

$$\text{new } t_5 = \min\{t_5, t_4 + q_{45}\} = \min\{10, 7+\infty\} = 10$$

$$\text{new } t_6 = \min\{t_6, t_4 + q_{46}\} = \min\{9, 7+5\} = 9$$

$$\text{new } t_7 = \min\{t_7, t_4 + q_{47}\} = \infty$$

Fourth iteration:-

$P = \{1,2,3,4\}, t_2 = 4, t_3 = 5, t_4 = 7$, Also $t_5 = 10, t_6 = 9, t_7 = \infty$

Step 1:- min $t_6 = 9$

- label the arc(3,6) as p_4

$P = \{1,2,3,4,5,6\}$

Step 2:- $P = \{1,2,3,4,6\}, t_2 = 4, t_3 = 5, t_4 = 7, t_6 = 9$

new $t_5 = \min\{t_5, t_6 + a_{65}\} = \min\{10, 9+1\} = 10$

new $t_7 = \min\{t_7, t_6 + a_{67}\} = \min\{\infty, 9+8\} = 17$

Fifth iteration:-

$P = \{1,2,3,4,6\}, t_2 = 4, t_3 = 5, t_4 = 7, t_6 = 9$ Also $t_5 = 10, t_7 = 17$

Step 1: min $t_5 = 10$

- label the arc (6,5) as p_5
- $P = \{1,2,3,4,6,5\}$

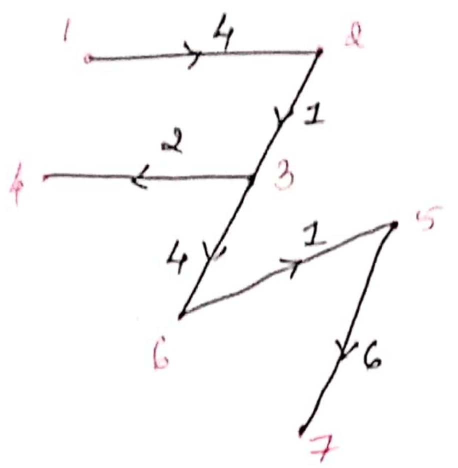
At this stage, only t_7 is left. Among all the arcs from the vertices in P to vertex 7, arc (5,7) has the least weight.

- label the arc(5,7) as p_6.
- join 7 to P

$\therefore P = \{1,2,3,4,6,5,7\}$

Arcs labelled as p_1 to p_6 are

(1,2) (2,3) (3,4) (3,6) (6,5) (5,7)



Path from 1 to 2 : weight 4

Path from 1 to 3 :

$1 \to 2 \to 3$, $w = 5$

Path from 1 to 4 :

$1 \to 2 \to 3 \to 4$, $w = 7$

Path from 1 to 5 :

$1 \to 2 \to 3 \to 6 \to 5$, $w = 10$

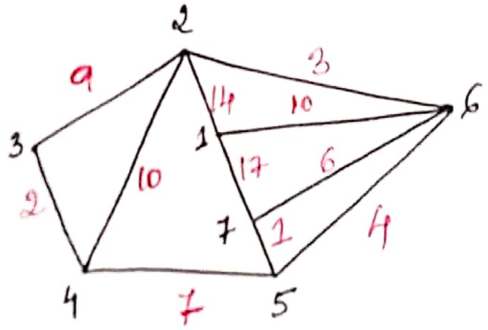
Path from 1 to 6 :

$1 \to 2 \to 3 \to 6$; $w = 9$

Path from 1 to 7 :

$1 \to 2 \to 3 \to 6 \to 5 \to 7$; $w = 16$.

(2)



<u>First iteration</u> :-

$P = \{1\}$, for $j = 2,3,4,5,6,7$

where $t_2 = 14$, $t_3 = \infty$, $t_4 = \infty$, $t_5 = \infty$, $t_6 = 10$, $t_7 = 17$

<u>Step 1</u> $\min t_6 = 10$,

label the arc $(1,6)$ as $p1$

$P\{1,6\}$

<u>Step 2</u> $P = \{1,6\}$, $t_6 = 10$

new $t_2 = \min\{t_2, t_6 + q_{62}\} = \min\{14, 10+3\} = 13$

new $t_3 = \min\{t_3, t_6 + q_{63}\} = \min\{\infty, 10+\infty\} = \infty$

$next_4 = \infty$

$new\ t_5 = 14$

$new\ t_7 = 16$

Second Iteration

for $P = \{1, 6\}$, $t_6 = 10$, $t_2 = 13$, $t_3 = \infty$, $t_4 = \infty$, $t_5 = 14$, $t_7 = 16$

Step 1 $t_2 = 13$ is min.

label the arc $(6, 2)$ as $p2$.

$P = \{1, 6, 2\}$

Step 2

$P = \{1, 6, 2\}$, $t_6 = 10$, $t_2 = 13$

new $t_3 = 22$, new $t_4 = \infty$, new $t_5 = 14$, new $t_7 = 16$

Third Iteration

for $P = \{1, 6, 2\}$, $t_6 = 10$, $t_2 = 13$, $t_3 = 22$, $t_4 = \infty$, $t_5 = 14$ &

$t_7 = 16$.

Step 1: $t_5 = 14$ is min.

label the arc $(6, 5)$ as $p3$

$P = \{1, 6, 2, 5\}$.

Step 2: $P = \{1, 6, 2, 5\}$, $t_6 = 10$, $t_2 = 13$, $t_5 = 14$

new $t_3 = 22$, new $t_4 = 21$, new $t_7 = 15$

Fourth Iteration

$P = \{1, 6, 2, 5, 7\}$, $t_6 = 10$, $t_2 = 13$, $t_5 = 14$, $t_7 = 15$

new $t_3 = 22$, new $t_4 = 21$

Fifth iteration

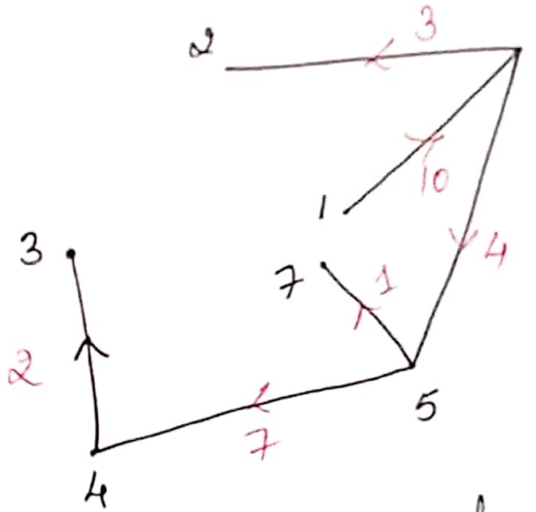
$P = \{1, 6, 2, 5, 7\}$, $t_6 = 10$, $t_2 = 13$, $t_5 = 14$, $t_7 = 15$, $t_3 = 22$, $t_4 = 21$

where min $t_4 = 21$

$P = \{1, 6, 2, 5, 7, 4\}$

So the arcs are:

(1,6) (6,2) (6,5) (5,7) (5,4) (4,3)



Path from a to f : $a \to f$; $w = 10$

a to b : $a \to f \to b$; $w = 13$

a to e : $a \to f \to e$; $w = 14$

a to g : $a \to f \to e \to g$; $w = 15$

a to d : $a \to f \to e \to d$; $w = 21$

a to c : $a \to f \to e \to d \to c$; $w = 23$