Name of Teacher: Miss Radhika M. Patil

Class: B.Sc. Computer Science (Entire)- III **Semester : 6**

Course Title: Computer Network

UNIT II: Network Layer and Transport Layer

- **Routing Algorithms**

- A routing algorithm is a procedure that lays down the route or path to transfer data packets from source to the destination.
- They help in directing Internet traffic efficiently.
- After a data packet leaves its source, it can choose among the many different paths to reach its destination.
- Routing algorithm mathematically computes the best path, i.e. “least – cost path” that the packet can be routed through.

- **Types of Routing Algorithms**

The Routing algorithm is divided into two categories:

1. Adaptive Routing algorithm
2. Non-adaptive Routing algorithm

1. Adaptive Routing algorithm

- An adaptive routing algorithm is also known as dynamic routing algorithm.
- This algorithm makes the routing decisions based on the topology and network traffic.
- The main parameters related to this algorithm are hop count, distance and estimated transit time.

An adaptive routing algorithm can be classified into three parts:

1. Centralized algorithm:

- It is also known as global routing algorithm as it computes the least-cost path between source and destination by using complete and global knowledge about the network.
- This algorithm takes the connectivity between the nodes and link cost as input, and this information is obtained before actually performing any calculation.

Example: Link state algorithm

2. Isolation algorithm:

- It is an algorithm that obtains the routing information by using local information rather than gathering information from other nodes.

3. Distributed algorithm:

- It is also known as decentralized algorithm as it computes the least-cost path between source and destination in an iterative and distributed manner.
- In the decentralized algorithm, no node has the knowledge about the cost of all the network links.
- In the beginning, a node contains the information only about its own directly attached links and through an iterative process of calculation computes the least-cost path to the destination.

Example: Distance vector algorithm.

2. Non-Adaptive Routing algorithm

- Non Adaptive routing algorithm is also known as a static routing algorithm.
- When booting up the network, the routing information stores to the routers.
- Non Adaptive routing algorithms do not take the routing decision based on the network topology or network traffic.

The Non-Adaptive Routing algorithm is of two types:

1. Flooding:

- In case of flooding, every incoming packet is sent to all the outgoing links except the one from it has been reached.
- The disadvantage of flooding is that node may contain several copies of a particular packet.

2. Random walks:

- In case of random walks, a packet sent by the node to one of its neighbours randomly.
- An advantage of using random walks is that it uses the alternative routes very efficiently.

Routing Algorithms:

1. **Shortest Path Algorithm**
2. **Flooding**
3. **Distance Vector Routing**

1. Shortest Path Algorithm

- In computer networks, the shortest path algorithms aim to find the optimal paths between the network nodes so that routing cost is minimized.

- **Common Shortest Path Algorithms**

  Some common shortest path algorithms are −

1. Bellman Ford's Algorithm
2. Dijkstra's Algorithm
3. Floyd Warshall's Algorithm

- **Dijktstra's Algorithm:**

- Dijkstra algorithm’ technique is widely used in many forms because it is simple and easy to understand.
- The idea is to make a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line(often known as a link).
- For selecting a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.
- One method of measuring path length is the number of hops.
- The shortest path is the fastest path with this graph labelling, rather than the path with the fewest arcs or kilometres.
- Generally, the labels on the arcs could be computed as a function of the distance, bandwidth, average traffic, communication cost, mean queue length, measured delay, and other factors.
- In starting, the paths are not known, therefore all nodes are labelled with infinity.
- As the algorithm proceeds and paths are obtained, the labels may change, reflecting better paths.
- All labels are tentative in starting.
- When it is discovered that a label tells the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

- For illustrating the working of labelling algorithm see at the weighted undirected graph of fig (a) where the weights represent distance.
- For finding the shortest path **from A to D.**

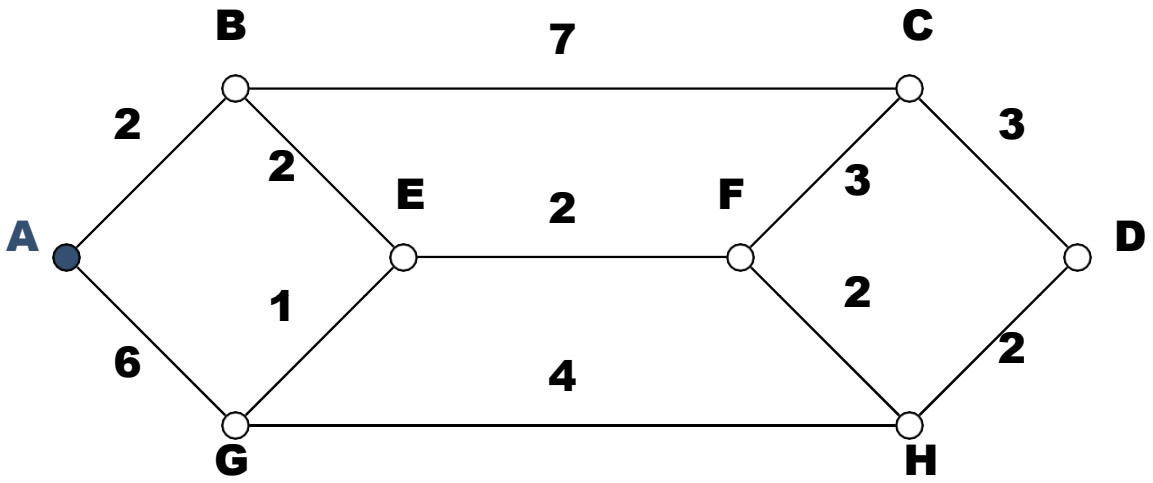
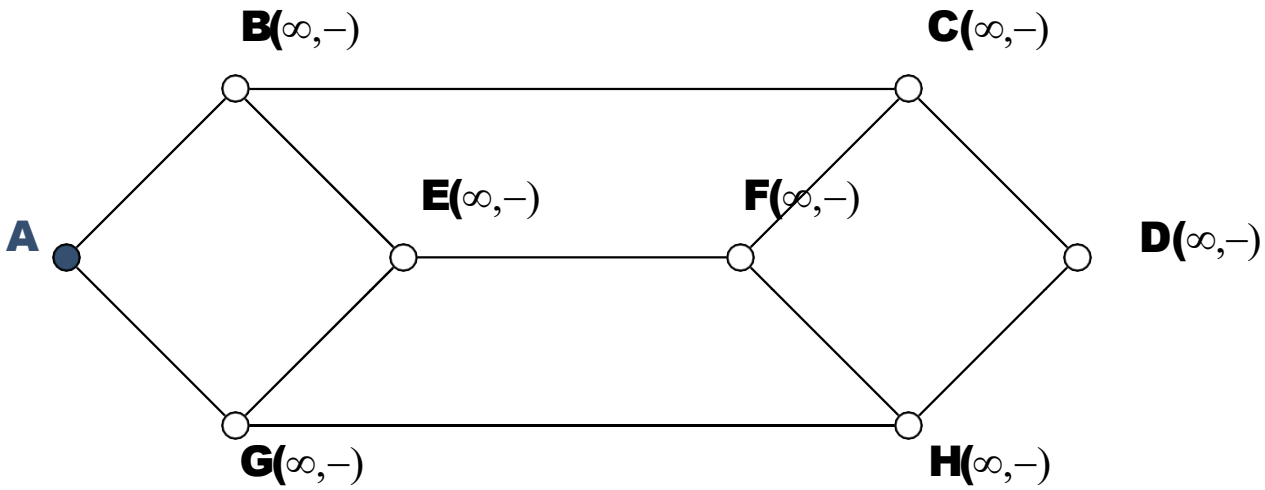


Fig (a)

- We start out by marking node A as permanent, denoted by a filled-in circle.
- Initially no paths are known, so all nodes are labelled with infinity as shown in following fig .



- Then, each of the nodes adjacent to A i.e. node B and node G, relabeling each one with the distance to A.

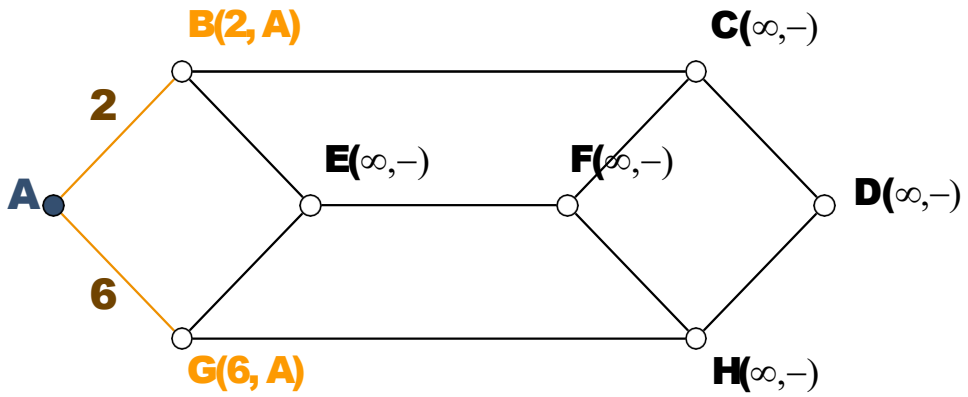


Fig (b)

- We make the node B permanent with the smallest label from node A.
- So now the node B becomes the new Working node.

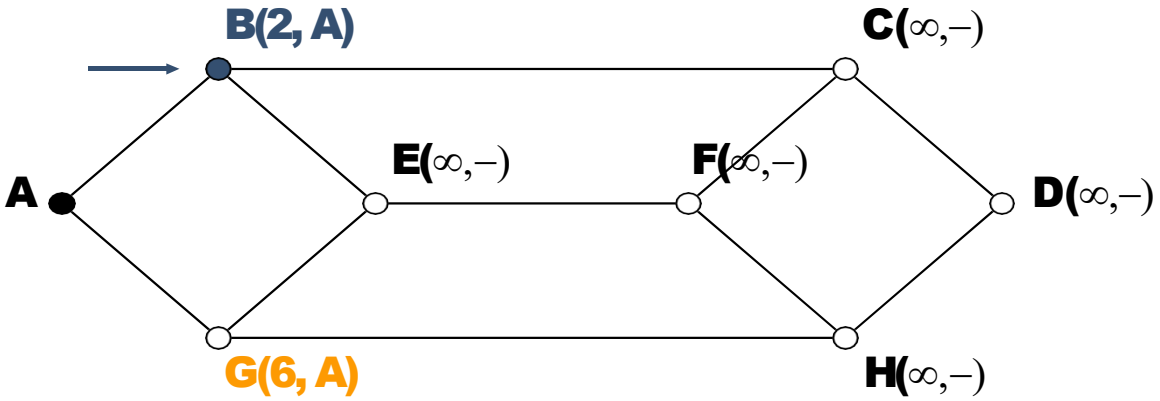


Fig (c)

- We make the node B permanent with the smallest label from node A.
- So now the node B becomes the new Working node.

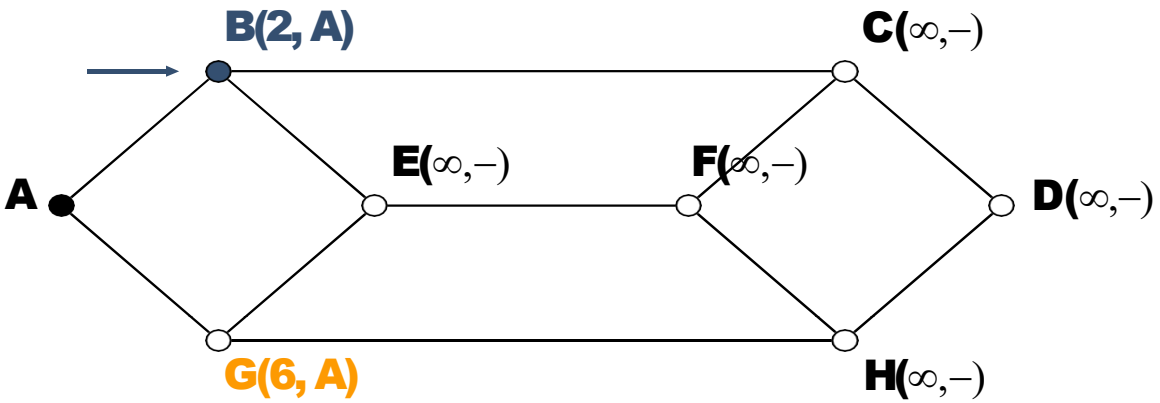


Fig (d)

- We examine each of the nodes adjacent to B (in this example node C and node E) and relabeling each one the distance from B.

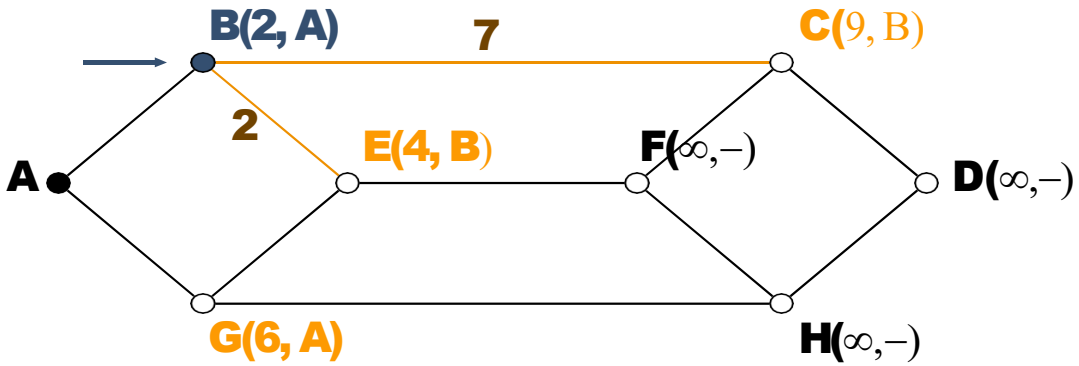


Fig (e)

- We make the node E permanent with the smallest label from node B.
- So now the node E becomes the new Working node.

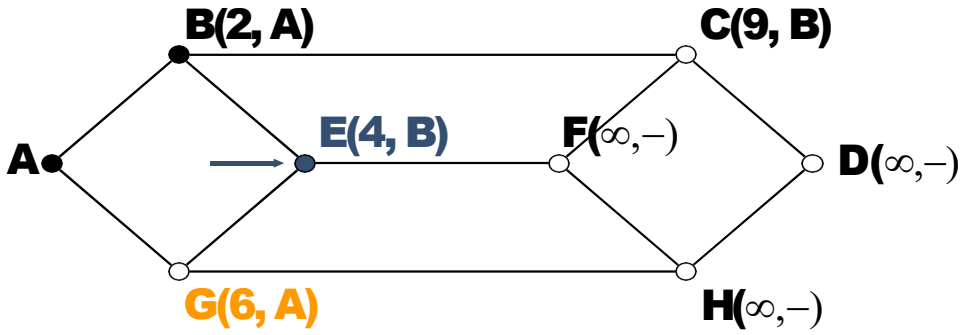


Fig (f)

- We examine each of the nodes adjacent to E (in this example node G and node F) and relabeling each one the distance from E.

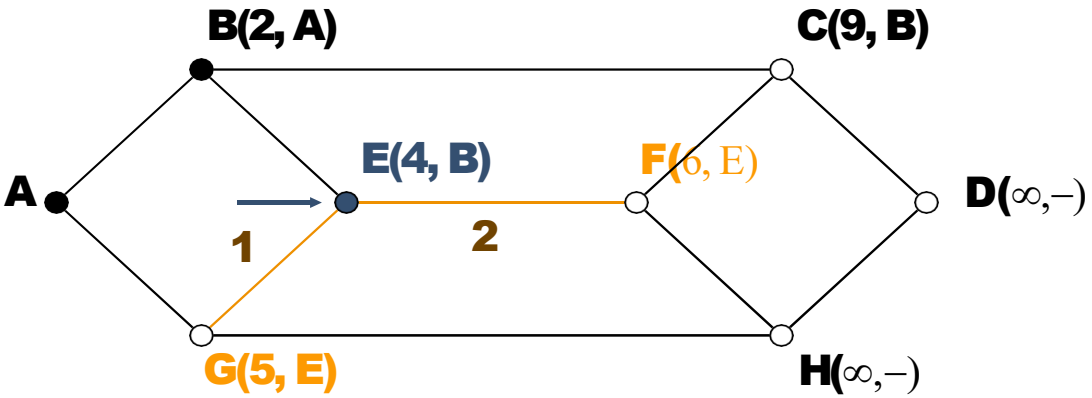


Fig (g)

- We make the node G permanent with the smallest label from node E.
- So now the node G becomes the new Working node.

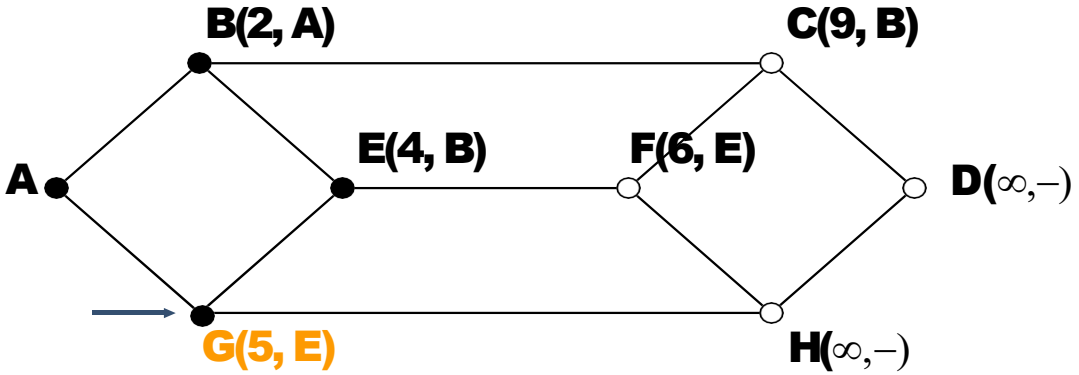


Fig (h)

- We examine each of the nodes adjacent to G (in this example node H and not E because it has already been visited. So, we don't take E) and relabeling each one the distance from G.

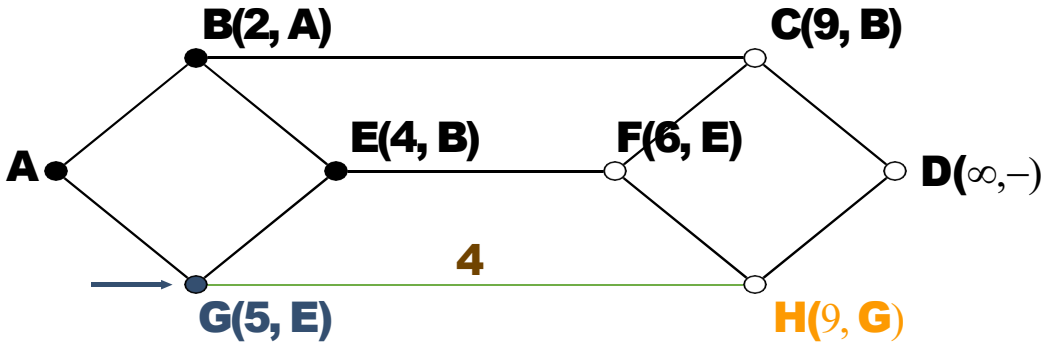


Fig (i)

- We make the node H permanent.
- So now the node H becomes the new Working node.

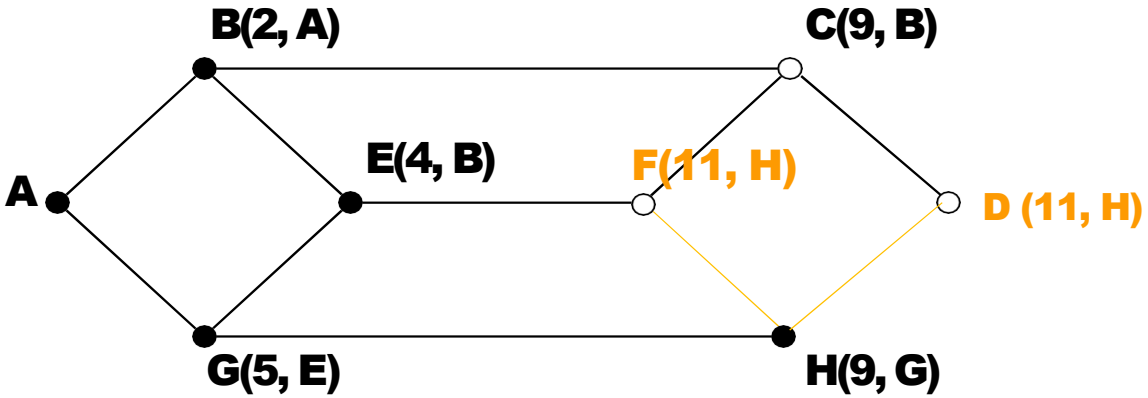


Fig (j)

Here we select only the node D as an adjacent node of H as it is the destination and process stops here.

So the final shortest from A to D is A- B – E – G – H – D.

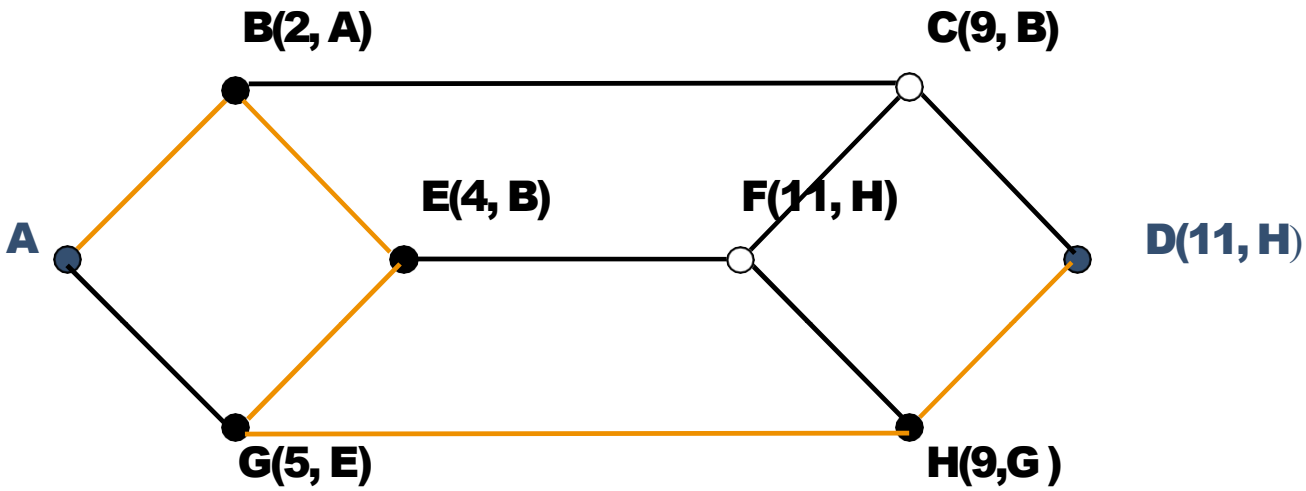
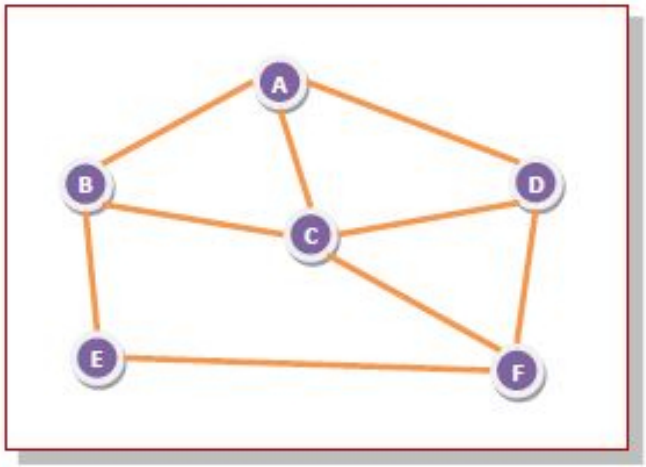


Fig (k) Final Route from Source to Destination

2. Flooding

- Flooding is a non-adaptive routing technique.
- This is simple method in which when a data packet arrives at a router, it is sent to all the outgoing links except the one it has arrived on.
- Requires no network information like topology, load condition ,cost of diff. paths
- All possible routes between Source and Destination are tried.
- All nodes directly or indirectly connected are visited
- Flooding is a way to distribute routing protocols updates quickly to every node in a large network.
- For example, let us consider the network in the figure, having six routers that are connected through transmission lines.



Using flooding technique −

- An incoming packet to A, will be sent to B, C and D.
- B will send the packet to C and E.
- C will send the packet to B, D and F.
- D will send the packet to C and F.
- E will send the packet to F.
- F will send the packet to C and E.

- **Types of Flooding**

Flooding may be of three types :

1. Uncontrolled flooding:

- Here, each router unconditionally transmits the incoming data packets to all its neighbours.

2. Controlled flooding:

- They use some methods to control the transmission of packets to the neighbouring nodes.
- The two popular algorithms for controlled flooding are Sequence Number Controlled Flooding (SNCF) and Reverse Path Forwarding (RPF).

3. Selective flooding :

- Here, the routers don't transmit the incoming packets only along those paths which are heading towards approximately in the right direction, instead of every available paths.

- **Advantages of Flooding**

1. It is very simple to setup and implement, since a router may know only its neighbours.
2. It is extremely robust.
3. Even in case of malfunctioning of a large number routers, the packets find a way to reach the destination.
4. All nodes which are directly or indirectly connected are visited. So, there are no chances for any node to be left out. This is a main criteria in case of broadcast messages.
5. The shortest path is always chosen by flooding.

- **Limitations of Flooding**

1. Flooding tends to create an infinite number of duplicate data packets.
2. It is wasteful if a single destination needs the packet, since it delivers the data packet to all nodes irrespective of the destination.
3. The network may be congested with unwanted and duplicate data packets.
4. This may disturb delivery of other data packets.

3. Distance Vector Routing Algorithm:

- It is distributed in that each node receives information from one or more of its directly attached neighbours, performs calculation and then distributes the result back to its neighbours.
- It is iterative in that its process continues until no more information is available to be exchanged between neighbours.
- The Distance vector algorithm is a dynamic algorithm.
- It is mainly used in ARPANET, and RIP.
- Each router maintains a distance table known as **Vector**.

- **Three Keys to understand the working of Distance Vector Routing Algorithm:**

1. Knowledge about the whole network:

- Each router shares its knowledge through the entire network. The Router sends its collected knowledge about the network to its neighbours.

2. Routing only to neighbours:

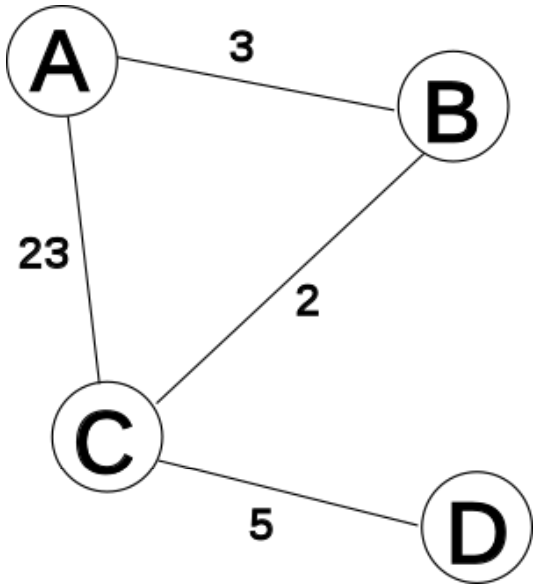
- The router sends its knowledge about the network to only those routers which have direct links.
- The information is received by the router and uses the information to update its own routing table.

3. Information sharing at regular intervals:

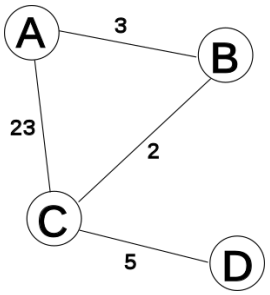
- Within 30 seconds, the router sends the information to the neighbouring routers.

- Routers that use distance-vector protocol determine the distance between themselves and a destination.
- The best route for Internet Protocol packets that carry data across a network is measured in terms of the numbers of routers (hops).
- To establish the best route, routers regularly exchange information with neighbouring routers, usually their routing table, hop count for a destination network and possibly other traffic related information.
- Distance-vector protocols update the routing tables of routers and determine the route on which a packet will be sent by the *next hop* which is the exit interface of the router and the IP address of the interface of the receiving router.
- Distance is a measure of the cost to reach a certain node.
- The least cost route between any two nodes is the route with minimum distance.
- Updates are performed periodically in a distance-vector protocol where all or part of a router's routing table is sent to all its neighbours that are configured to use the same distance-vector routing protocol.
- Once a router has this information it is able to change its own routing table to reflect the changes and then inform its neighbours of the changes.

In this network we have 4 routers A, B, C and D:



- We mark the current time (or iteration) in the algorithm with T, and begin (at time 0, or T=0) by creating distance matrices for each router to its immediate neighbours.



T= 0

from A	via A	via B	via C	via D
to A				
to B		3		
to C			23	
to D				

from B	via A	via B	via C	via D
to A	3			
to B				
to C			2	
to D				

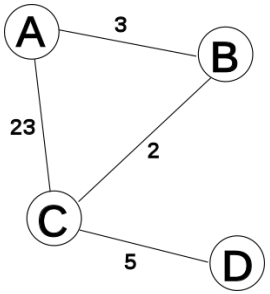
from C	via A	via B	via C	via D
to A	23			
to B		2		
to C				
to D				5

from D	via A	via B	via C	via D
to A				
to B				
to C			5	
to D				

- At this point, all the routers (A,B,C,D) have new "shortest-paths" for their DV (the list of distances that are from them to another router via a neighbour).
- They each broadcast this new DV to all their neighbours: A to B and C, B to C and A, C to A, B, and D, and D to C.
- As each of these neighbours receives this information, they now recalculate the shortest path using it.

For example:

- A receives a DV from C that tells A there is a path via C to D, with a distance (or cost) of 5.
- Since the current "shortest-path" to C is 23, then A knows it has a path to D that costs 23+5=28.
- As there are no other shorter paths that A knows about, it puts this as its current estimate for the shortest-path from itself (A) to D, via C.



T= 1

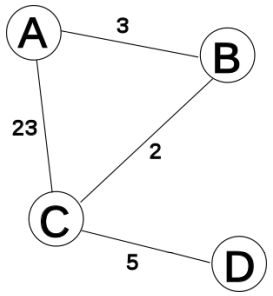
from A	via A	via B	via C	via D
to A				
to B		3	25	
to C		5	23	
to D			28	

from B	via A	via B	via C	via D
to A	3		25	
to B				
to C	26		2	
to D			7	

from C	via A	via B	via C	via D
to A	23	5		
to B	26	2		
to C				
to D				5

from D	via A	via B	via C	via D
to A			28	
to B			7	
to C			5	
to D				

- Again, all the routers have gained in the last iteration (at T=1) new "shortest-paths", so they all broadcast their DVs to their neighbours.
- This prompts each neighbour to re-calculate their shortest distances again.
- For instance: A receives a DV from B that tells A there is a path via B to D, with a distance (or cost) of 7.
- Since the current "shortest-path" to B is 3, then A knows it has a path to D that costs 7+3=10. This path to D of length 10 (via B) is shorter than the existing "shortest-path" to D of length 28 (via C), so it becomes the new "shortest-path" to D.



T= 2

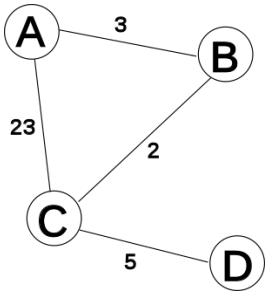
from A	via A	via B	via C	via D
to A				
to B		3	25	
to C		5	23	
to D		10	28	

from B	via A	via B	via C	via D
to A	3		7	
to B				
to C	8		2	
to D	31		7	

from C	via A	via B	via C	via D
to A	23	5		33
to B	26	2		12
to C				
to D	51	9		5

from D	via A	via B	via C	via D
to A			10	
to B			7	
to C			5	
to D				

- This time, only routers A and D have new shortest-paths for their DVs.
- So they broadcast their new DVs to their neighbours
- A broadcasts to B and C, and D broadcasts to C.
- This causes each of the neighbours receiving the new DVs to re-calculate their shortest paths.
- However, since the information from the DVs doesn't yield any shorter paths than they already have in their routing tables, then there are no changes to the routing tables.



T= 3

from A	via A	via B	via C	via D
to A				
to B		3	25	
to C		5	23	
to D		10	28	

from B	via A	via B	via C	via D
to A	3		7	
to B				
to C	8		2	
to D	13		7	

from C	via A	via B	via C	via D
to A	23	5		15
to B	26	2		12
to C				
to D	33	9		5

from D	via A	via B	via C	via D
to A			10	
to B			7	
to C			5	
to D				

- None of the routers have any new shortest-paths to broadcast.
- Therefore, none of the routers *receive* any new information that might change their routing tables.
- The algorithm comes to a stop.

- **Congestion:**

- Congestion is a situation in Communication Networks in which too many packets are present in a part of the subnet.
- Congestion in a network may occur when the load on the network *(i.e.* the number of packets sent to the network) is greater than the capacity of the network *(i.e.* the number of packets a network can handle.).
- Network congestion occurs in case of traffic overloading.
- In other words when too much traffic is offered, congestion sets in and performance degrades sharply.
- **Traffic Shaping** is a mechanism to control the amount and the rate of the traffic sent to the network.
- Approach of congestion management is called Traffic shaping.
- Traffic shaping helps to regulate rate of data transmission and reduces congestion.

- **Congestion control algorithms**

1. Leaky Bucket
2. Token Bucket

1. Leaky Bucket Algorithm (LB):

- It is a traffic shaping mechanism that controls the amount and the rate of the traffic sent to the network.
- A leaky bucket algorithm shapes bursty traffic into fixed rate traffic by averaging the data rate.
- Imagine a bucket with a small hole at the bottom.
- The rate at which the water is poured into the bucket is not fixed and can vary but it leaks from the bucket at a constant rate.
- Thus (as long as water is present in bucket), the rate at which the water leaks does not depend on the rate at which the water is input to the bucket.

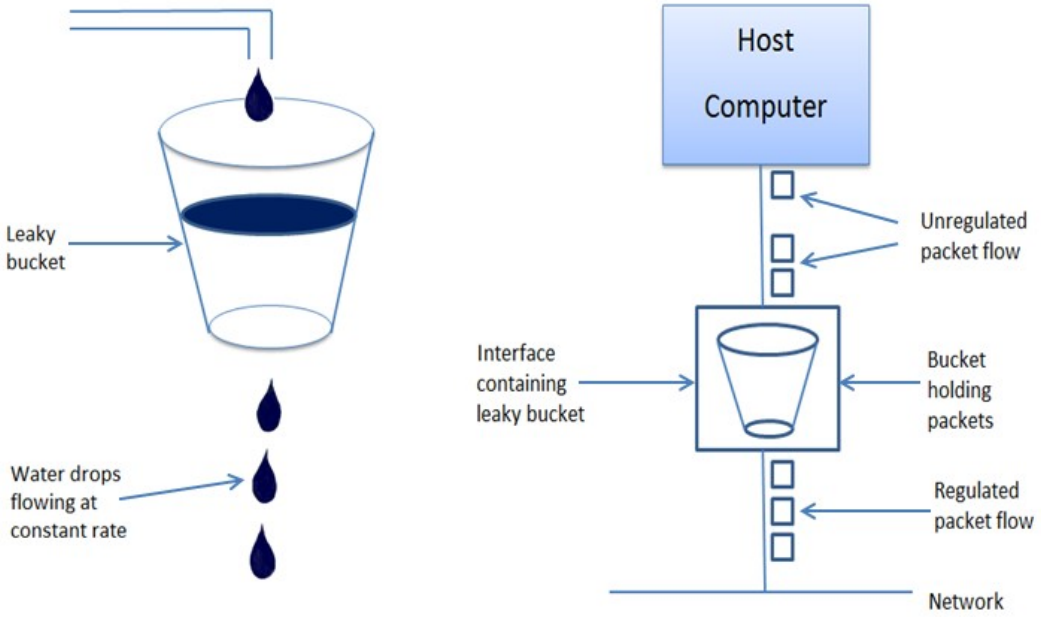


Fig: Leaky Bucket Algorithm

- Also, when the bucket is full, any additional water that enters into the bucket spills over the sides and is lost.
- Each network interface contains a leaky bucket and the following **steps** are involved in leaky bucket algorithm:

1. When host wants to send packet, packet is thrown into the bucket.
2. The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.
3. Bursty traffic is converted to a uniform traffic by the leaky bucket.
4. When bucket is full, the packets are discarded.
5. In practice the bucket is a finite queue that outputs at a finite rate.

- **Advantages:**

1. Token independent
2. Packets are transmitted continuously.
3. It sends the packet at constant rate
4. Input rate can differ but output rate will be constant

- **Disadvantages:**

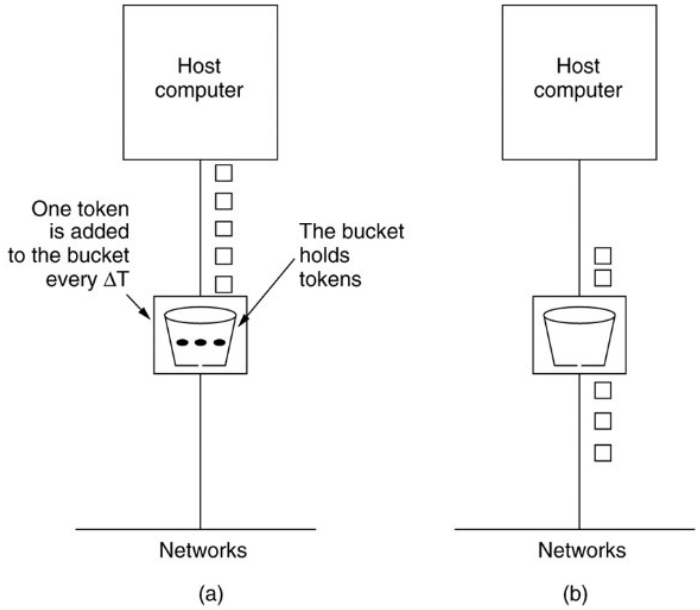
1. It does not save token.
2. If bucket is full packet or data is discarded.
3. LB sends packets at an average rate.

2. Token Bucket (TB) algorithm:

- The leaky bucket algorithm allows only an average (constant) rate of data flow.
- Its major problem is that it cannot deal with bursty data.
- A token bucket algorithm allows bursty data transfers.
- A token bucket algorithm is a modification of leaky bucket in which leaky bucket contains tokens.
- In this algorithm leaky bucket holds token, generated at regular intervals.
- In this algorithm, a token(s) are generated at every clock tick. For a packet to be transmitted, system must remove token(s) from the bucket.
- Thus, a token bucket algorithm allows idle hosts to accumulate credit for the future in form of tokens.
- Main steps of this algorithm can be described as follows:

1. In regular intervals tokens are thrown into the bucket
2. The bucket has a maximum capacity.
3. If there is a ready packet, a token is removed from the bucket, and the packet is send.
4. If there is no token in the bucket, the packet cannot be sent.

- Figure shows the two scenarios before and after the tokens present in the bucket have been consumed.
- In Fig.(a) the bucket holds three tokens, and two packets are waiting to be sent out of the interface
- In Fig.(b) three packets have been sent out by consuming three tokens, and 2 packets are still left.
- The token bucket algorithm is less restrictive than the leaky bucket algorithm, in a sense that it allows bursty traffic.
- However, the limit of burst is restricted by the number of tokens available in the bucket at a particular instant of time.



(a) Token bucket holding three tokens, before packets are send out
(b) Token bucket after three packets are sent, two packets still remain as no tokens are left

- The implementation of basic token bucket algorithm is simple
- A variable is used just to count the tokens.
- This counter is incremented every t seconds and is decremented whenever a packet is sent.
- Whenever this counter reaches zero, no further packet is sent out as shown in Fig.

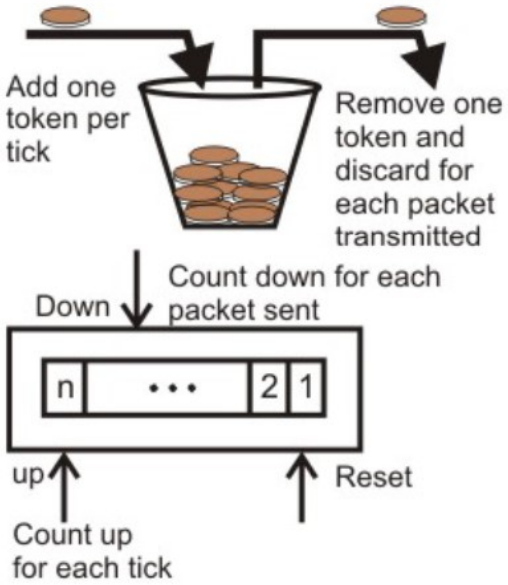


Fig. Implementation of the Token bucket algorithm

- **Transport Layer:**

- The transport layer is a 4th layer from the top.
- The main role of the transport layer is to provide the communication services directly to the application processes running on different hosts.
- The transport layer provides a logical communication between application processes running on different hosts.
- Although the application processes on different hosts are not physically connected, application processes use the logical communication provided by the transport layer to send the messages to each other.
- The transport layer protocols are implemented in the end systems but not in the network routers.
- A computer network provides more than one protocol to the network applications.
- For example, TCP and UDP are two transport layer protocols that provide a different set of services to the network layer.
- All transport layer protocols provide multiplexing/demultiplexing service.
- It also provides other services such as reliable data transfer, bandwidth guarantees, and delay guarantees.
- Each of the applications in the application layer has the ability to send a message by using TCP or UDP.
- The application communicates by using either of these two protocols.
- Both TCP and UDP will then communicate with the internet protocol in the internet layer.
- The applications can read and write to the transport layer.
- Therefore, we can say that communication is a two-way process.

- **Transport Layer Services:**

  There are the two types of services:

1. Connection Oriented Service
2. Connectionless Services

1. Connection Oriented Services

- There is a sequence of operation to be followed by the users of connection oriented service.
- These are:

1. **Connection is established.**

2. Information is sent.

3. Connection is released.

- In connection oriented service we have to establish a connection before starting the communication.
- When connection is established, we send the message or the information and then we release the connection.
- Connection oriented service is more reliable than connectionless service.
- We can send the message in connection oriented service if there is an error at the receivers end.
- Example of connection oriented is TCP (Transmission Control Protocol) protocol.

2. Connection Less Services

- It is similar to the postal services, as it carries the full address where the message (letter) is to be carried.
- Each message is routed independently from source to destination.
- The order of message sent can be different from the order received.
- In connectionless the data is transferred in one direction from source to destination without checking that destination is still there or not or if it prepared to accept the message.
- Authentication is not needed in this.
- Example of Connectionless service is UDP (User Datagram Protocol) protocol.

- **Transport Layer Service Primitives:**

- A service is specified by a set of primitives.
- A primitive means operation.
- To access the service a user process can access these primitives.
- These primitives are different for connection oriented service and connectionless service.

- **Connection Oriented Service Primitives:**

1. LISTEN : When a server is ready to accept an incoming connection it executes the LISTEN primitive. It blocks waiting for an incoming connection.

2. CONNECT : It connects the server by establishing a connection. Response is awaited.

3. RECIEVE: Then the RECIEVE call blocks the server.

4. SEND : Then the client executes SEND primitive to transmit its request followed by the execution of RECIEVE to get the reply. Send the message.

5. DISCONNECT :

- This primitive is used for terminating the connection.
- After this primitive one can't send any message.
- When the client sends DISCONNECT packet then the server also sends the DISCONNECT packet to acknowledge the client.
- When the server package is received by client then the process is terminated.

- **Connectionless Service Primitives:**

  There are 2 types of primitives for Connectionless Service:

UNIDATA	This primitive sends a packet of data
FACILITY, REPORT	Primitive for enquiring about the performance of the network, like delivery statistics.

- **Transport Layer protocols**

The transport layer is represented by two protocols:

1. TCP
2. UDP.

1.TCP:

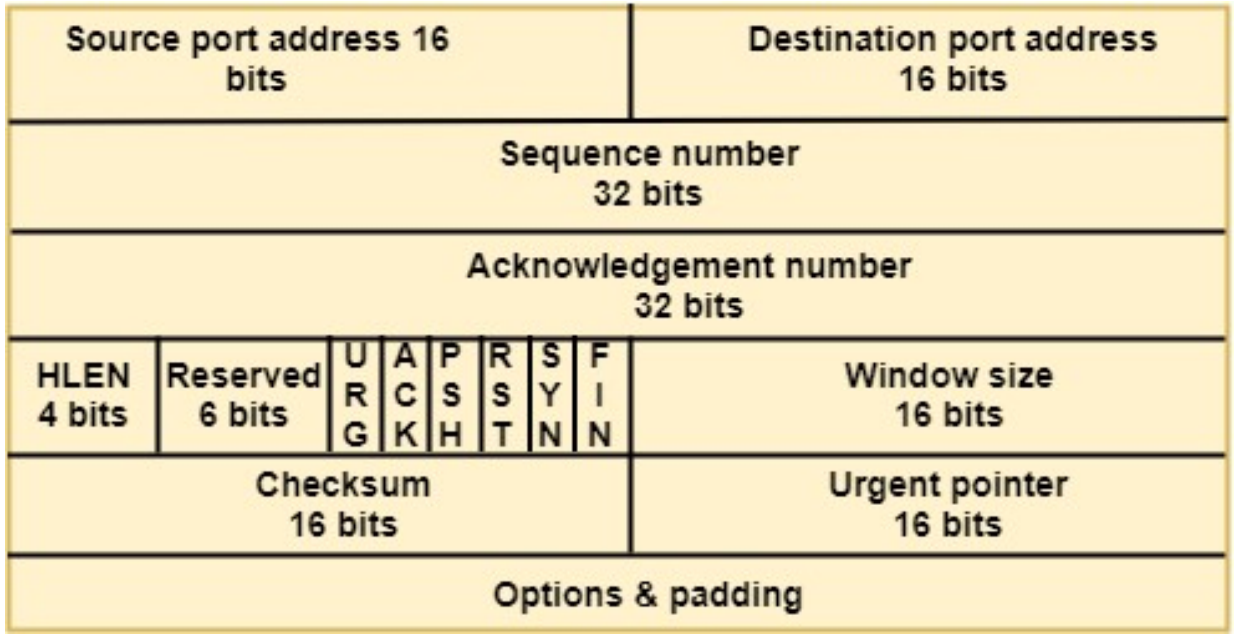
- TCP stands for Transmission Control Protocol.
- It is a connection-oriented protocol means the connection established between both the ends of the transmission. For creating the connection, TCP generates a virtual circuit between sender and receiver for the duration of a transmission.

- **Features of TCP:**

1. TCP is reliable protocol. That is, the receiver always sends either positive or negative acknowledgement about the data packet to the sender, so that the sender always has bright clue about whether the data packet is reached the destination or it needs to resend it.
2. TCP ensures that the data reaches intended destination in the same order it was sent.
3. TCP is connection oriented. TCP requires that connection between two remote points be established before sending actual data.
4. TCP provides error-checking and recovery mechanism.
5. TCP provides end-to-end communication.
6. TCP provides flow control and quality of service.
7. TCP operates in Client/Server point-to-point mode.
8. TCP provides full duplex server, i.e. it can perform roles of both receiver and sender.

- **TCP Segment Format**

The length of TCP header is minimum 20 bytes long and maximum 60 bytes.



Where,

1. **Source Port (16-bits)** - It identifies source port of the application process on the sending device.
2. **Destination Port (16-bits)** - It identifies destination port of the application process on the receiving device.
3. **Sequence Number (32-bits)** - Sequence number of data bytes of a segment in a session.
4. **Acknowledgement Number (32-bits)** - When ACK flag is set, this number contains the next sequence number of the data byte expected and works as acknowledgement of the previous data received.
5. **Data Offset (4-bits)** - This field implies both, the size of TCP header (32-bit words) and the offset of data in current packet in the whole TCP segment.
6. **Reserved (3-bits)** - Reserved for future use and all are set zero by default.
7. **Flags (1-bit each)**

1) **URG** - It indicates that Urgent Pointer field has significant data and should be processed.
2) **ACK** - It indicates that Acknowledgement field has significance. If ACK is cleared to 0, it indicates that packet does not contain any acknowledgement.
3) **PSH** - When set, it is a request to the receiving station to PUSH data (as soon as it comes) to the receiving application without buffering it.

4) RST - Reset flag has the following features:

- It is used to refuse an incoming connection.
- It is used to reject a segment.
- It is used to restart a connection.

5) SYN - This flag is used to set up a connection between hosts.

6) **FIN** - This flag is used to release a connection and no more data is exchanged thereafter. Because packets with SYN and FIN flags have sequence numbers, they are processed in correct order.

8. Windows Size - This field is used for flow control between two stations and indicates the amount of buffer (in bytes) the receiver has allocated for a segment, i.e. how much data is the receiver expecting.

9. Checksum - This field contains the checksum of Header, Data and Pseudo Headers.

10. Urgent Pointer - It points to the urgent data byte if URG flag is set to 1.

11. Options - It facilitates additional options which are not covered by the regular header. Option field is always described in 32-bit words. If this field contains data less than 32-bit, padding is used to cover the remaining bits to reach 32-bit boundary.

2. UDP:

1. UDP stands for **User Datagram Protocol**.
2. UDP is a simple protocol and it provides nonsequenced transport functionality.
3. UDP is a connectionless protocol.
4. This type of protocol is used when reliability and security are less important than speed and size.
5. UDP is an end-to-end transport level protocol that adds transport-level addresses, checksum error control, and length information to the data from the upper layer.
6. The packet produced by the UDP protocol is known as a user datagram.

- **User Datagram Format**

The user datagram has a 16-byte header which is shown below:

Where,

1. **Source port address:** It defines the address of the application process that has delivered a message. The source port address is of 16 bits address.
2. **Destination port address:** It defines the address of the application process that will receive the message. The destination port address is of a 16-bit address.
3. **Total length:** It defines the total length of the user datagram in bytes. It is a 16-bit field.
4. **Checksum:** The checksum is a 16-bit field which is used in error detection.

- **Disadvantages of UDP protocol**

1. UDP provides basic functions needed for the end-to-end delivery of a transmission.
2. It does not provide any sequencing or reordering functions and does not specify the damaged packet when reporting an error.
3. UDP can discover that an error has occurred, but it does not specify which packet has been lost as it does not contain an ID or sequencing number of a particular data segment.

- **Assignment:**

  Differentiate between TCP and UDP.

THANK YOU…