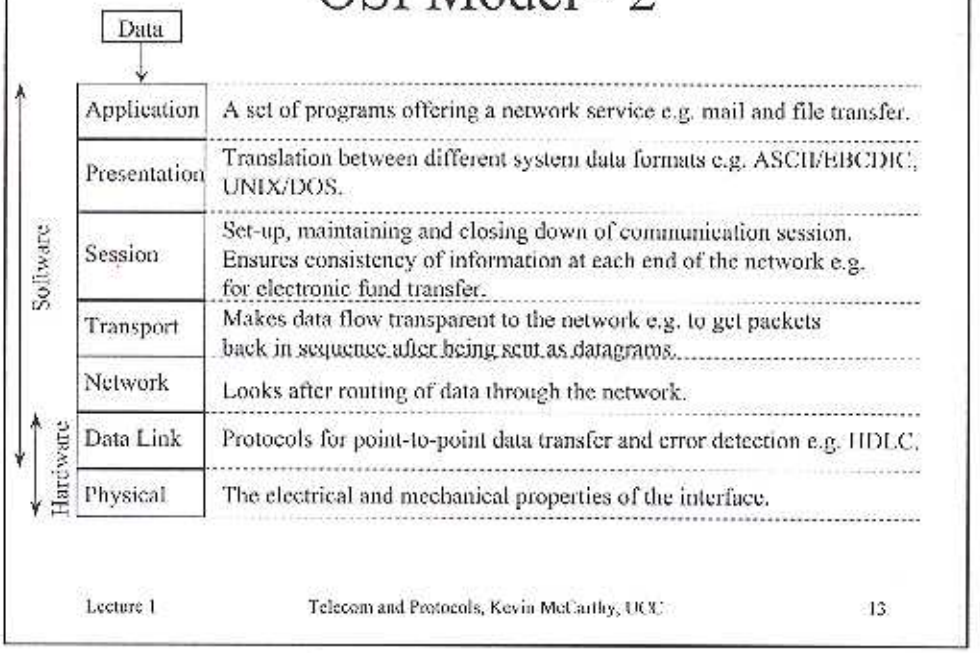
EE4004, Autumn 2002
Q2(b)

9

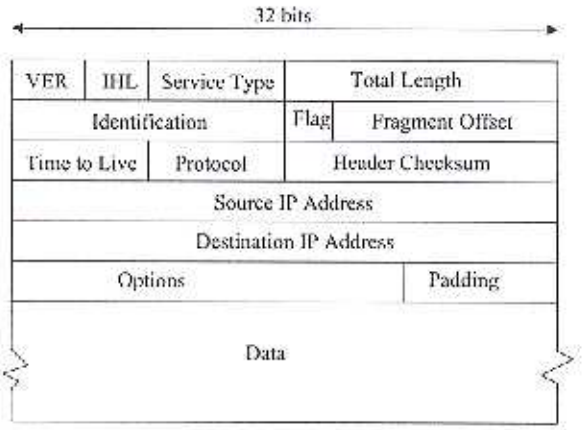
OSI Model - 2



EE4004 Autumn 2002
Q3 (a)

IP Packet Format

The IP data packet has a header of at least 20 bytes followed by optional extra header information and then the data.

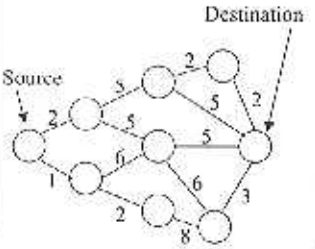


8

Source IP Address: This is the 32 bit address of the computer sending the data
Destination IP Address: The 32 bit address of the destination
Total Length: This gives the length of the packet in bytes. This field has 16 bits so the maximum length of an IP packet is 2^{16} bytes. In practice the IP packets are usually much smaller than this.
Protocol: This specifies the upper level protocol (TCP or UDP) which requested transmission of the message to indicate to the receiver how the message shouldbe handled.
VER: This refers to the version number of the IP protocol being used.
IHL: This is the number of 4-byte words in the header.
Service Type: This indicates the type of service (TOS) requested. TOS refers to the delay, throughput and reliability of the route. This field is ignored in many cases on the assumption that the network is providing a "best effort service anyway.
Identification: This is a two-byte field which holds the packet sequence number i.e. the frame number.
Flags and Fragmentation: Long packets may need to be fragmented into a sequence of smaller packets at the link layer. The flags and fragmentation offset field keep track of this.
Time to Live: This one byte field is set to a value by the source. Every time the IP packet passes through a network node, this value is decremented by one. When it reaches 0 the packet is discarded and an error message is sent to the source. This prevents a packet clogging up the network if it cannot be delivered.
Header Checksum: This is used to perform error detection on the header only.
Options: This is used for some optional services that may have been requested such as recording the route of the packet through the network, following a specific route or time stamping of the packet by each node that it travels through.

EE6004 Autumn 2002
Q3(b)

Bellman-Ford Algorithm - 1



This is a distributed algorithm in which each node has only a partial knowledge of the network. Each node knows the length of the links attached to it. The "length" of the link is usually calculated from a combination of the average transmission time along the link and the recent backlog in the queue of the link's transmitter.

A node i estimates the shortest path to the destination from the following:

$$L(i)=\min\{d(i,j)+L(j)\} \text{ for all nodes j attached to i.}$$

where j is any node directly connected to i, d(i,j) is the length of the link from i to j and L(j) is node j's estimate of its distance to the destination.

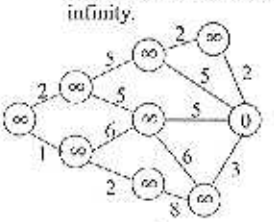
At each step of the algorithm, each node estimates the shortest distance from itself to the destination using the above formula and based on the estimates L(j) received from its neighbours. It remembers the link associated with this shortest path. If, based on the most recent information from its neighbours, a node calculates a smaller shortest path to the destination then it broadcasts this new estimate to its neighbours and it remembers the link associated with the new shortest path.

Initially all nodes estimate their distance to the destination as infinity and the destination sets its distance to 0.

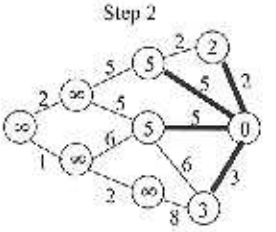
9

Bellman-Ford Algorithm - 2

Step 1: The root sets its distance as zero, all other nodes set to infinity.

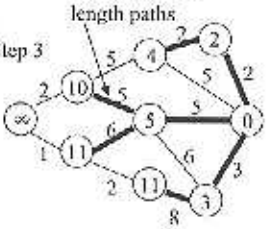


Step 2



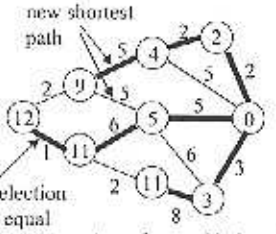
random selection from two equal length paths

Step 3



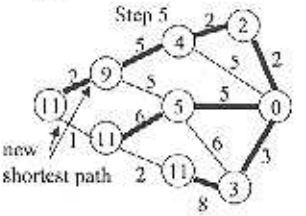
new shortest path

Step 4



random selection from two equal length paths

Step 5



new shortest path

At each step: Update distance estimates based on **link length and previous estimates from neighbouring nodes** and update shortest path links based on new distance estimates.