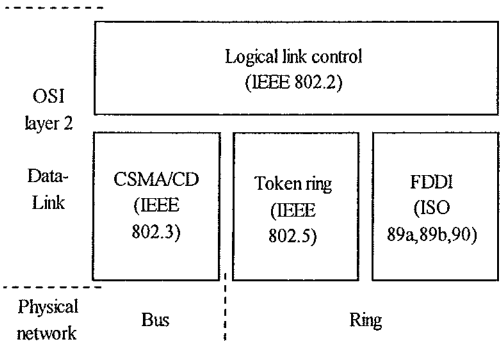
PART IIB 2010 4B15 ADVANCED TELECOMMUNICATIONS NETWORKS DR TD WILKINSON

1. (a) The data link layer forms layer 2 of the OSI reference model. This provides error detection and correction for a link to ensure that the exchange of data is reliable. Provides error free data to the network layer. Adds control information to the data blocks and a frame check sequence (FCS) for error-checking or bits for synchronisation. Retransmits data if errors have occurred (but it does not arrange for retransmissions). The data-link layer is also responsible for defining the length of data segments to be sent to the physical layer. If the data provided by the network layer is too long, then it is broken into suitable sized packets.



Transparency is preserved for the data bits in the layer 2 frames. Two basic services are offered, *connection oriented* and *connectionless,* usually through the use of a LLC sub-layer. In the case of LANs, the data-link layer provides local area address management on top of that provided by the network layer through medium access protocols (MACs). Different types of LAN are characterised by their distinctive topologies. They all comprise a single transmission path interconnecting all the data terminal devices, with a bit speed typically between 1Mbits/s and 1Gbit/s, together with the appropriate protocols (called the logical link control (LLC) and the medium access control (MAC)) to enable data transfer. The LLC and MAC split the data-link layer (Layer 2).

The LLC can multiplex amongst higher layer clients through the use of a service access point (SAP) identifier. Both the source client (SSAP) and the destination client (DSAP) can be identified with this header. This is not strictly an OSI compliant function and the information SAP fields are often ignored by the MAC and LLC.

(b) A bridge is a device which allows the interconnection of LANs at the layer 2 level. A *transparent bridge* is a piece of hardware, which allows frames to be passed between LANs that have different geographical locations and even different LAN protocols. A bridge is a data-link layer device and any interconnection of LANs via bridges is often referred to as a *catenet.* A bridge in its true OSI definition is purely a layer 2 entity, which transparently forwards, discards or floods frames to its ports. This pure bridge function (including the bridge address table) is performed by the relay entity in the MAC layer of the bridge. There is no access to this layer from higher layers such as the LLC or network layer. This is often referred to as an architectural bridge. End stations will be totally oblivious to the bridge at the data link layer. Some basic bridge principles:

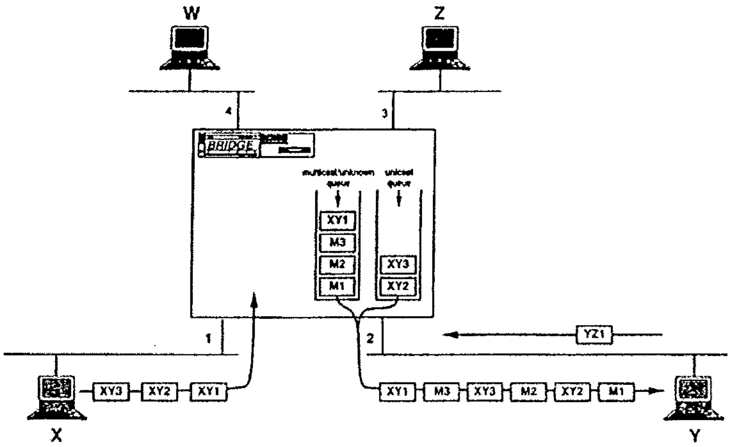
- There are multiple distinct LAN segments interconnected by the bridge.
- Each station has a globally unique 48 bit unicast address.
- The bridge has a *port* or interface on each LAN to which it connects.
- There is a table within the bridge which maps station addresses to bridge ports, hence it knows how each station can be reached.
- The bridge acts in *promiscuous mode*, it receives (or attempts to) every frame on every port regardless of destination address.
- The bridge must apply the MAC protocol of the intended port. Eg detect collisions on Ethernet or use a token on a token ring.
- A frame can incur a bridge transit delay due to traffic or buffering whist awaiting transmission onto the intended port.
- When forwarding a frame, it uses the original source address of the sender rather than inserting its own (if it has one).
- The stations are unaware of the bridge, hence it is transparent.

The transparent nature of bridges means that they can be interconnected with each other to make even more complex LAN topologies, without loss of frames or data. There is no limit to how many bridges can be connected in this way, however there are practical limits due to flooding and multicasts as well as bridge delays.

A bridge tries to make a catenet appear transparent to end stations, as if it were a single LAN. Hence higher layer services will expect a LAN-like performance from the catenet below it. A LAN data-link must exhibit certain properties.

Hard Invariants: Non duplication of frames, Sequential delivery of frames
Soft Invariants: Low error rate, High bandwidth (or utilisation), Low delay (or latency)

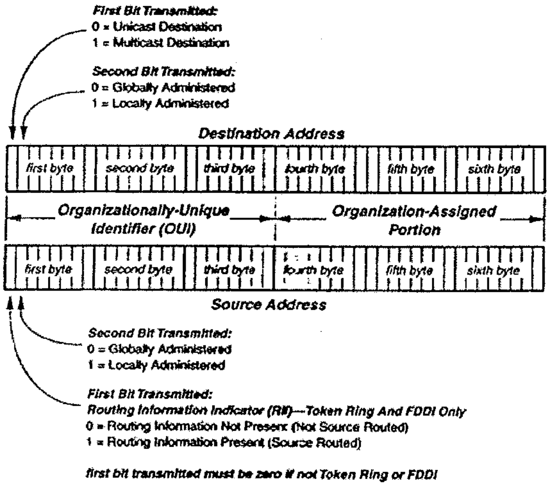
The hard invariants are absolute and cannot be compromised in any way as they are fundamental to the operation of a layer 2 process in the OSI model. The soft invariants are more flexible and can be traded off for more sophistication in certain areas of the LAN performance. Note that there is no protocol mechanism to guarantee these invariants. Bridges can complicate these restrictions and care must be taken, especially when considering the hard invariants.



As shown above, the bridge receives a frame (XY1) for destination Y from source X, however station Y is currently unknown and so the bridge treats it like a multicast. This results in the frame appearing in the same queue as other multicasts which could be awaiting transmission (M1, M2, M3), hence frame XY1 must wait. While XY1 is in this queue, station Y sends a frame to station Z (YZ1) which is received by the bridge. This YZ1 frame means that the bridge now knows which port station Y is on due to its source address, hence it can update its address table. This means that when the next frame XY2 arrives at port 1, it is queued with the unicast traffic for port 2 as station Y is now known. These two queues for port 2 will eventually be emptied and could lead to XY2 being received before XY1, breaking a hard invariant. There is no clear solution:

- ✓ Use a single queue for both unicast and multicast traffic (complicated)
- ✓ Use a time synchronisation mechanism between queues (slow)
- ✓ Discard the later unicast traffic until frames are clear from the multicast queue.
- ✓ Ignore the problem and hope nothing will break.

(c) The MAC address has a standardised 48 bit structure which has been adopted as part of the IEEE 802 set of LAN protocols. The address can identify both the transmitting station (*source address*) and the receiving station (*destination address*).

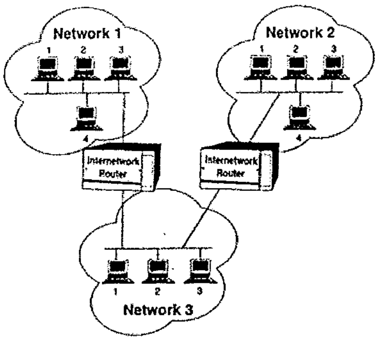

First Bit Transmitted:
0 = Unicast Destination
1 = Multicast Destination
Second Bit Transmitted:
0 = Globally Administered
1 = Locally Administered
Destination Address
first byte
second byte
third byte
fourth byte
fifth byte
sixth byte
Organizationally-Unique Identifier (OUI)
Organization-Assigned Portion
first byte
second byte
third byte
fourth byte
fifth byte
sixth byte
Source Address
Second Bit Transmitted:
0 = Globally Administered
1 = Locally Administered
First Bit Transmitted:
Routing Information Indicator (RII)—Token Ring And FDDI Only
0 = Routing Information Not Present (Not Source Routed)
1 = Routing Information Present (Source Routed)
first bit transmitted must be zero if not Token Ring or FDDI

The 48bitMAC address space is split into two halves:

- A *unicast address* identifies a single device or interface. The source address is always unicast.
- A *multicast address* identifies a group of logically related devices.

The first bit of the destination address defines if it is a unicast (0) or a multicast (1) address. Source addresses are always unicast so the first bit is zero except in token ring or FDDI, where is describes how the packet is routed (equals 1 for source routing).

The second bit of the address defines whether the address is globally unique (0) or locally unique (1) to the LAN. Globally unique addresses are assigned by the manufacturer, whereas locally unique addresses are assigned by the LAN administrator. The 48-bit MAC address is divided into two parts. The first 24 bits constitute the organisationally unique identifier (OUI), which indicates which organisation (typically the manufacturer) is responsible for assigning the remaining 24 bits of the address. A MAC address is usually expressed in canonical byte form *aa-bb-cc-dd-ee-ff* where the first 3 pairs of bytes are the OUI and the last 3 pairs of bytes are set by the manufacturer. An address is read *aa* byte first, however there are two bit conventions, little and big endian.



Any station can be uniquely identified by a catenation of:

Network Identifier (globally-unique)	Station Identifier (locally-unique)

The MAC address for a NIC maker must be globally unique and contain a valid OUI registered to that company. Hence there are 3 bytes left giving $2^{24} = 16777216$ addresses.

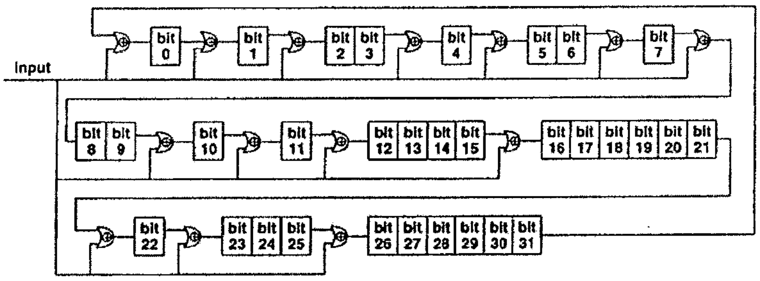
The maker could buy another OUI, or the net software could be adapted to use higher layer address structures as well. Within the data link layer, only stations on the same LAN need to have unique addresses, as more complex network addresses can be made by catenation of a station address with the address of the LAN which contains it. This could be a higher layer address such as an IP address. Eg. [Network 1 | Station 4] can communicate with [Network 3 | Station 4] even though they have the same data-link address.

(d) In order for a bridge to locate a port to which a destination station is attached it must search for it in the bridge address table. This should be as efficient as possible to minimise potential delays in the catenet. There are three main techniques: The hash table, binary search and content addressable memory (CAM).

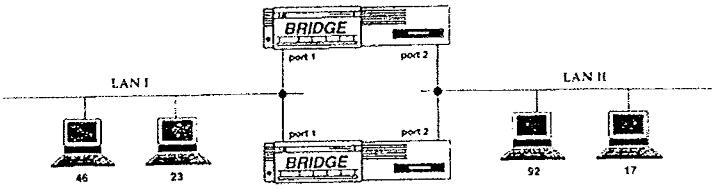
A way to avoid huge search areas is to map the address onto a small pointer space, using what is known as a hash function. Hashing is talking the 48bit address and producing a shorter field, which will then point at a subset of locations in memory. The hash function must:

- ✓ Produce a consistent value for the same address (be repeatable)
- ✓ Produce a relatively uniform distribution of pointers for a given set of inputs.

Hence, a hash function should not cause uneven spreading of addresses to output values. Each output value should have roughly the same number of addresses mapping to it in a random manner. This is important as most MAC addresses on a LAN will be a subset as they will have come from the same manufacturer and have the same 24bit OUI. There are many different hash functions in the literature, however there is an automatic generation algorithm, which will perform the hash function on the data in the form of the FCS generator and checking circuitry. As the frame passes through the FCS circuit, there is a point when the 48bit destination address has passed through it, hence we only need to sample the output from the FCS at this point in time to generate a suitable hashed address. This is usually done with a linear feedback shift register (LFSR). The address table is organised as hash buckets, with the hash function (the bits from the LFSR) pointing to the appropriate bucket.



- ✓ The destination address is simple to calculate, but as the source address follows straight after it, it is not so simple to hash. Either a second LFSR are needed or a software option is required.
- ✓ Depending on the addresses in the catenet, the buckets may not fill uniformly leading to buckets overflowing. This can be altered by changing the bits chosen by the LFSR.



Examiners comment:

The question was well answered, if a little too easy. There were some interesting answers to the expansion of MAC addresses.

Q2 (a) If we consider the above bridge connection, using ordinary transparent store and forward bridges which have up to date address tables, what will happen when frames are sent? If station 46 sends a frame to station 17, each bridge receives the frame and sends it to the appropriate port (port 2) and after queuing, the frames appear at station 17. Station 17 will receive two frames from station 46 which violate the non-duplication invariant. In fact every station on LAN II will receive duplicate unicast frames from LAN I.

What happens if LAN 46 sends a multicast frame? Both bridge A and B will forward the frame, however bridge A will receive the forwarded frame from bridge B (and vice versa) on port 2. Upon receiving this frame, the bridge will look at the source address of the frame and reconfigure station 46, thinking it is now connected to port 2. This process will continue indefinitely with both bridges continually updating their address tables. The same situation will occur in the first scenario, with non-convergence of the address table in the unicast case as well.

This problem can be fixed by careful network planning, however in complex structures, loops are inevitable. The problem above can be fixed by disabling the ports on one of the two bridges, effectively removing it from the network. It is still listening, but no longer routes frames. This disabling process can be done manually for small catenets, however for large complex structures this is impractical and often dangerous. There are several automatic mechanisms which can be implemented in bridges, the best of which is the spanning tree, which allocates ports on bridges and therefore routes by measuring the speed and efficiency of transport. Hence in the above example the slowest bridge is always eliminated.

(b) **Routing type** – This field in the source routed frame specifies the type of routing frame used

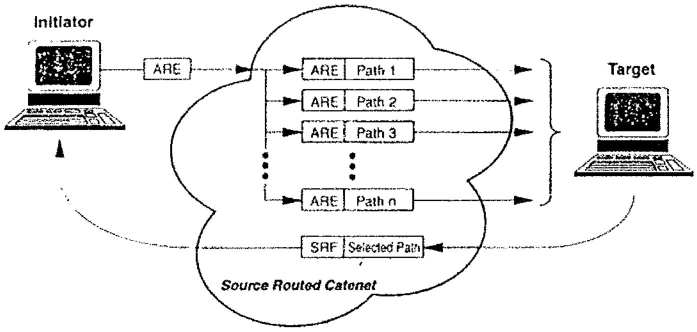
.

1. *Specifically routed frames (SRF)* – Carry routing type 0b0XX. A SRF carries a list of route descriptors and is forwarded along this route. This is used for the bulk of data frames.

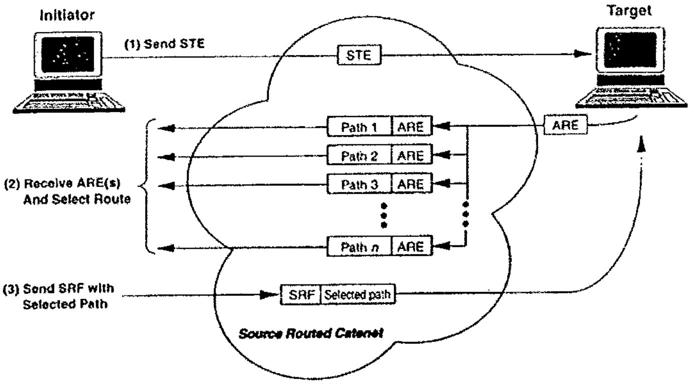
Source frames which are not specifically routed are called *explorer frames* and have the first bit of the routing type set to 1

2. *Spanning tree explorer (STE) frames* – Routing type 0b11X. A SPE frame is only forwarded by bridges that are designed as part of the spanning tree. One copy of these frames will appear on each ring in the catenet (non-duplicated). They can be used for either multicast traffic or route discovery. STE frames are originally sent with no route descriptors. Forwarding bridges insert descriptors to identify the route taken.
3. *All routes explorer (ARE) frames* – Routing type 0b10X. An ARE frame is forwarded by all bridges along every possible path between source and destination. Thus the destination may receive multiple frames for different routes taken. ARE frames are sent with no route descriptors. These are added by each bridge as it forwards the frame.

(c) Before an SRF can be sent, the route discovery process must be done to find a suitable route. There are several possible ways of doing this with STE and ARE frames, but here are two most common.



ARE request, specifically routed response – The sending station send an ARE frame which will arrive at the destination by multiple recorded routes. The receiving station selects a route and sends a SRF back to the sender to indicate the chosen route. This places a processing burden on the receiver, so a variation of this is for the receiver to send an SRF for every ARE it receives and let the sender select the route.



Spanning tree request – The sending frame sends an STE frame which is forwarded by the bridges on the spanning tree on all rings in the catenet and eventually to the target destination. The receiving station responds by sending an ARE frame back to the sender which will indicate to the sender all of the available routes. Hence the processing burden is returned to the sender who must select a route.

The route selection process can be a complex one, as it is unknown how many ARE responses will occur. Is a better route yet to come?? There are two basic mechanisms.

- ✓ *Take the first route* – Just select the first ARE which arrives and discard the rest. The first ARE indicates the route of least delay. This is what the majority of stations will do.
- ✓ *Take the first route which meets a specific requirement* – select the route with a suitable MTU of with a minimum number of bridge hops.

(d) The relative merits of the two types of routing are quite subtle and buried in historical evolution. Source routing was developed by IBM as part of their token ring protocol and is in fact found as part of the IEEE token ring and FDDI standards. The logic behind it was that source routing required more computing power at the stations and IBM were good at making computers... There is no source routing mechanism for Ethernets (Xerox and IBM didn't get on). There are a whole host of different arguments about link efficiency, traffic management, link redundancy, latency control etc which can be used to justify or reject source routing, however a more important lesson is it parallels with the switching protocols used in the synchronous digital hierarchy (SDH) and asynchronous transfer mode (ATM) switching.

Source routing is a more efficient as long as the network can cope well with multicast frames. This may seem contradictory as multicasting is normally associated with clogging up networks. This is not the case in source routed networks, as the bridges are now very simple transfer and record devices with no real intelligence above layer 2, hence the overhead of multicasting is not very large. This is clearly evident in both SDH and ATM, where the excess cost of multicasting is not that considerable.

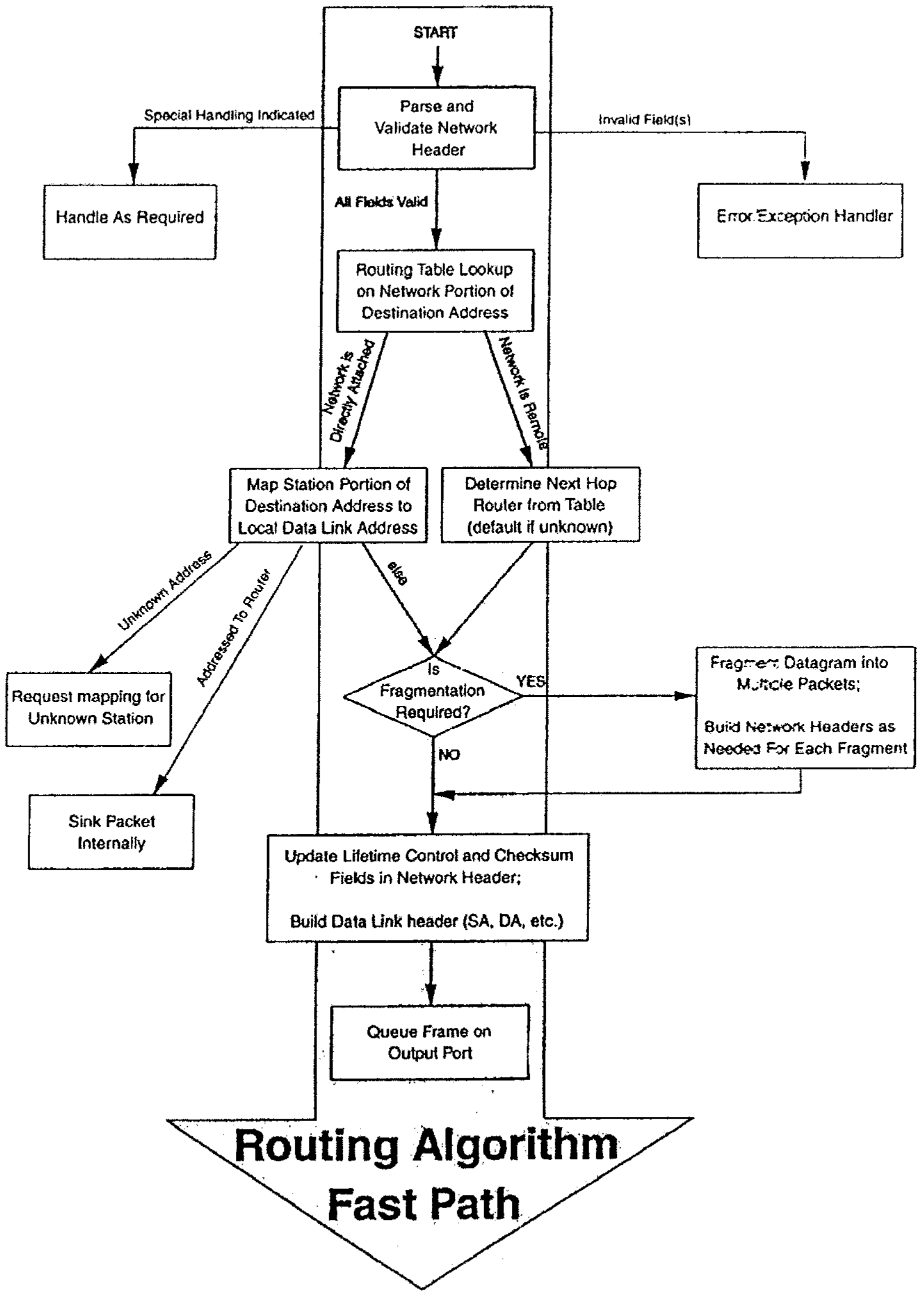
The main reason in favour of destination routing is flexibility. In a network that is always growing or evolving, the ability to 'learn' the rotes can be a big advantage, however this flexibility is at the cost of complexity in both hardware and protocol(s). It is interesting to note that today's destination routed networks all require at least 3 protocols to work efficiently (ie TCP/IP + Ethernet). This is a result of the growing complexity of the requirements of routing frames. This has a huge impact on the hardware as well, with large backbone routers being extremely complex and expensive.

If we consider the internet, then flexibility is more important than complexity, hence destination routing has evolved in the way it has. Source routing would be difficult with so many users and is probably not sufficiently scaleable to be effective. If we consider other issues such as priority services and quality of service across the internet, then source routing could be much more effective.

Examiner's comment:

This question was well answered with most knowing the STP well. The last section was quite speculative and was very well answered considering the wide scope of the question.

3 a) The fast path and IP Packet parsing and validation – The router need to separate certain fields to determine the type of handling required. Check the IP version number. Check the header length field (>20bytes means routing information present). Calculate the header checksum. Validating the source address


START
Parse and Validate Network Header
Special Handling Indicated
Handle As Required
Invalid Field(s)
Error Exception Handler
All Fields Valid
Routing Table Lookup on Network Portion of Destination Address
Network is Directly Attached
Network is Remote
Map Station Portion of Destination Address to Local Data Link Address
Determine Next Hop Router from Table (default if unknown)
Unknown Address
Addressed To Router
else
Request mapping for Unknown Station
Sink Packet Internally
Is Fragmentation Required?
YES
NO
Fragment Datagram into Multiple Packets;
Build Network Headers as Needed For Each Fragment
Update Lifetime Control and Checksum Fields in Network Header;
Build Data Link header (SA, DA, etc.)
Queue Frame on Output Port
Routing Algorithm
Fast Path

Routing table look up – The router performs a table look up to determine the output port to direct the packet to, based on the network identifier of the IP address. The result of this will be that either:

- ✓ The destination network is reachable only by forwarding the packet to another router (remote network). This may occur due to a match with the network identifier or due to the selection of a route for an unknown destination network. Either way the look up will return the address of the next router and the port on which it can be reached.

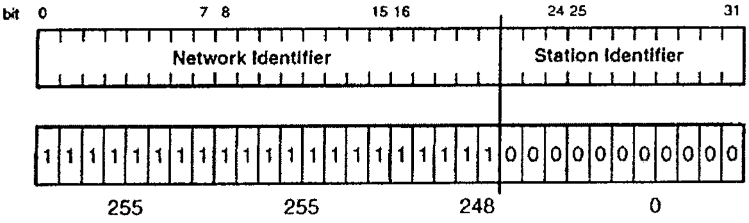
✓ The destination network is known to be directly attached to the router. The router will return the port on which the directly attached network can be found, including a pseudo port for packets addressed to the router. The station identifier portion of the IP address must also be mapped onto the layer 2 address using the address resolution protocol (ARP) cache.

Fragmentation – Each available output port will have an associated maximum transmission unit (MTU), which is the largest frame length permitted. It is generally a function of network technology or MAC used (Ethernet, Token ring, PPP etc). If the packet is larger than the MTU then it must be fragmented.
Update lifetime control and checksum – The router adjusts the time to live (TTL) field in the packet which is used to prevent packets from endlessly bouncing around internetworks. Packets have their TTL decremented when routed and are discarded once the TTL expires. Finally the header checksum is recalculated for the new TTL.

The majority of packets routed will undergo this exact process. A few specialised processes will occur off the fast path:
✓ Fragmentation and assembly
✓ Source routing option
✓ Route recording option
✓ Timestamp option
✓ ICMP message generation
✓ Routing protocols (RIP, OSPF, BGP)
✓ Network management (SNMP)
✓ Configuration (BOOTP, DHCP)

b) It should be noted that the routing look up process is considerably more complex than that for a bridge, as there is a complex address structure to be broken down into a pair of variable fields. Hence the search algorithms are considerably more 'intelligent'.

IP Address



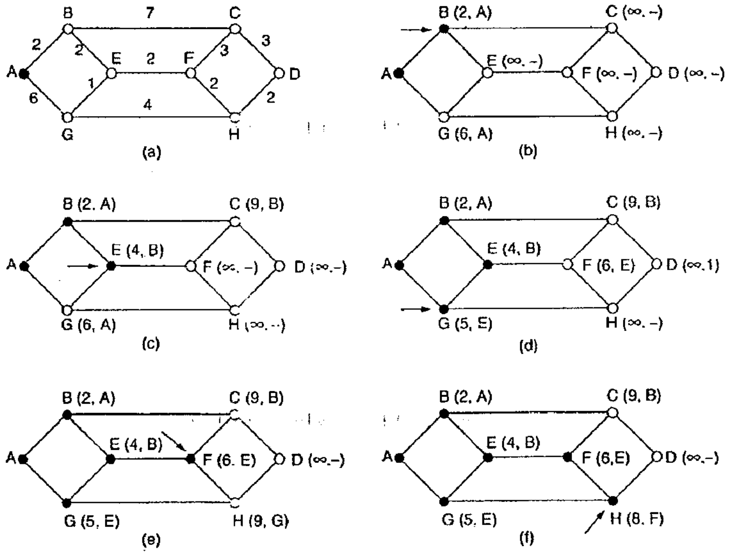
Subnet Mask

Unlike in a bridge, where the search is for a fixed length address field, the IP network identifier has a variable length from 0 (usually specifying a default route) to 32 bits (for host-specific routes). A given destination address may yield multiple matches with entries in the router table corresponding to different network identifier lengths. The IP routing rules specify that the routing result returned should be for the one for the largest number of matching bits. Hence the look up process implies a search for the longest match against a variable length field.

There are appropriate algorithms for such searches including many based on a compressed binary tree (such as a radix or PATRICIA tree) data structure. In addition, a binary tree can be incorporated with the ARP cache mapping process. In some structures, the entire layer 2 addressing table can also be included. Router look ups are often implemented as hardware state machines using complex data structures, hard wired state machines and content addressable memory.

c) The idea is to build a graph of the subnet where each node represents a router and then select the shortest possible path through the subnet based on the chosen link criteria. The choice of route will depend on the chosen criteria. If number of hops is chosen, then it will probably give a different series of routes than if geographical distance were chosen. There are many other criteria such as queue delay, transit times etc. The graph must be updated every time period to keep track of any changes in the subnet's performance or topology. In fact, it is most likely that the weighting factors given to each link on the graph will be a function of many different metrics.

There are several algorithms for choosing the shortest path in a connected graph, but one of the commonest was invented by Dijkstra in 1959. Each node is labelled in brackets with the distance from the source node along the best known path. Initially no paths are known so all nodes are labelled with infinity. As nodes and paths are found then the labels are updated with the current best known path to that point. Initially labels are tentative (open circle) but once it is known that the node is on the shortest path from the source it becomes permanent (full circle). In the example on the next page we start at permanent node A. We then find the paths to B and G with the appropriate distances. We then make B permanent as I is the shortest from A. We then move to B and repeat the process, each time locating the shortest possible distance. After all the tentative nodes have been labelled, then the graph is searched to find the shortest path from any permanent node to the next tentative node. This is then made permanent.



- d) ***Resource reservation protocol (RSVP)***

A good example of a QoS based protocol is RSVP, where resources within a network are reserved on pre-determined basis in a manual fashion. This is a flow based protocol which uses a network configured as a series of spanning trees set up between network nodes which are connected in an optimal fashion. When a station wants a particular QoS, then a request is sent along the spanning tree to reserve the bandwidth at each node. If this is not possible, then a message is returned to inform about the lack of service availability. There are many ways in which this can be done, especially for a multicast service such as TV broadcasts. Different capacities can be reserved for different services and multicasts can be sent across the spanning tree to groups of receivers. The problem is that non-reserved data must wait for the remaining bandwidth (if there is any).

- ***Differentiated services***

One of the main limitations of RSVP is that is does not scale well in large subnets and it also requires extensive modifications to existing routers. A simpler approach is to define differentiated services (DS), where a packet uses a type of service indicator field in the packet to indicate a priority of service within a domain of DS enabled routers. This allows predetermined services to be accessed on the fly as the packets arrive in a domain. Essentially it is like the class based system used in the postal service. The DS system means that no one call or flow gets reserved bandwidth, it is in fact reserved for all who use that class.

Examiner's comment:

. This question was well answered on the whole even though it was the most book based. A lot over explained the algorithm and solved the graph which was not asked for. Most got some form of QoS answer in Ok.