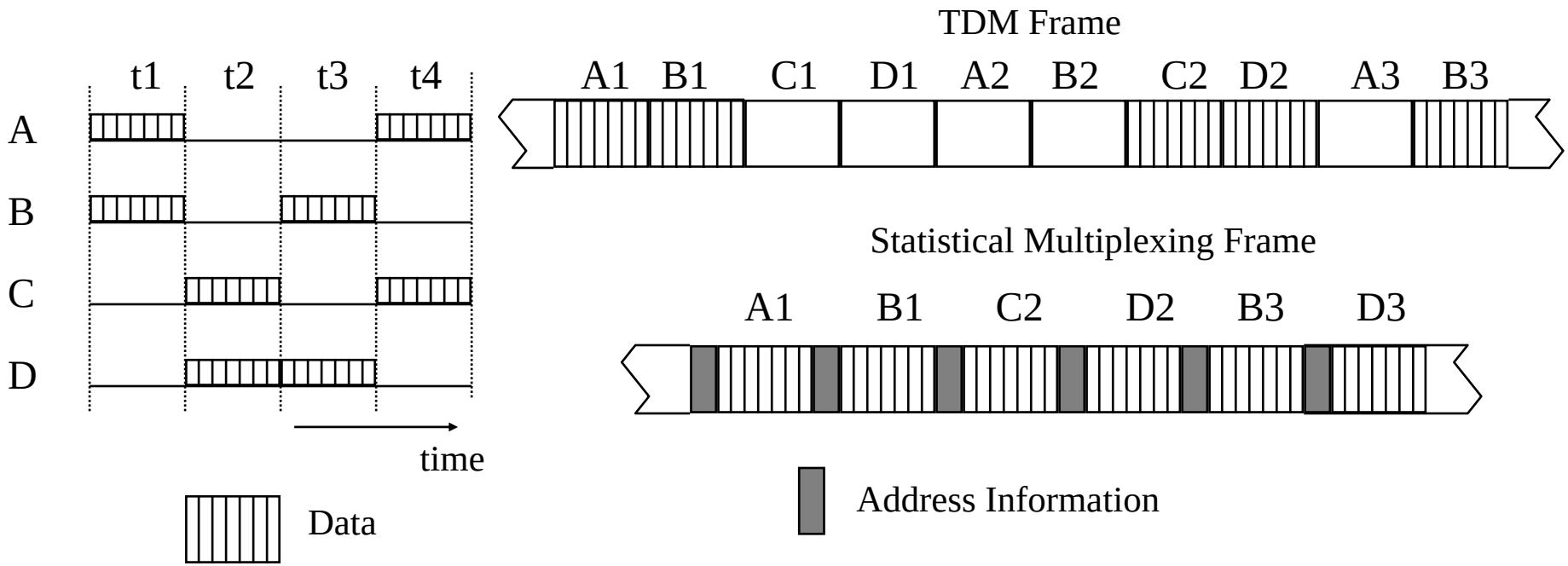
EE4004 Telecommunications Summer 2006

Question 2(b) 10 marks

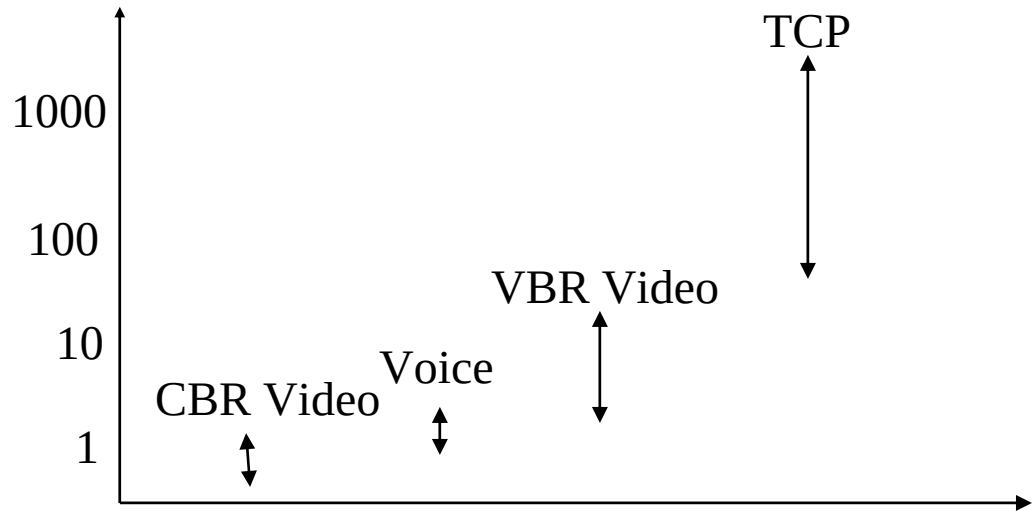
In SM if an input channel has no data during a time slot then this is skipped and the multiplexer moves on to the next channel. The time slot on the output line corresponding to a given input line thus varies. A label must be added to the output data to indicate the input channel it comes from.
In SM the output line is designed to cope with the average data rates of the input lines and can thus cope with a much higher number of input lines for a given output link rate than TDM. This gives a multiplexing gain.



Question 2(b) continued

$$\text{Multiplexing Gain} = \frac{\text{No. of input channels for SM}}{\text{No. of input channels for TDM}} \quad \text{(for a given link rate)}$$

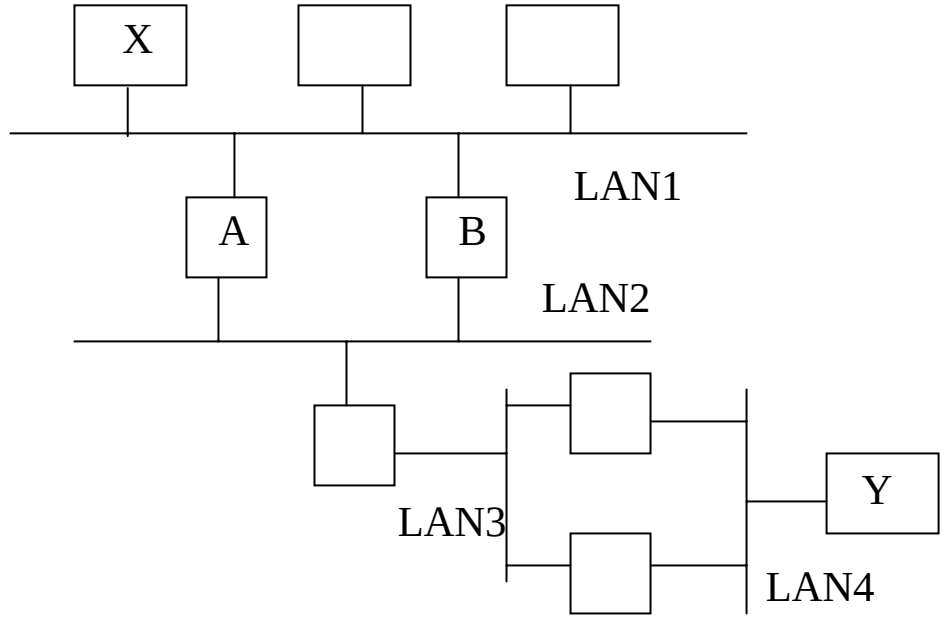
The multiplexing gain depends on the data characteristics on the input line. For very bursty data (e.g. internet) the multiplexing gain is very high. For constant rate data (e.g. voice or video) it is low.



TDM: Time Division Multiplexing
SM: Statistical Multiplexing

EE4004 Telecommunications Summer 2006

Question 3(a) 5 marks



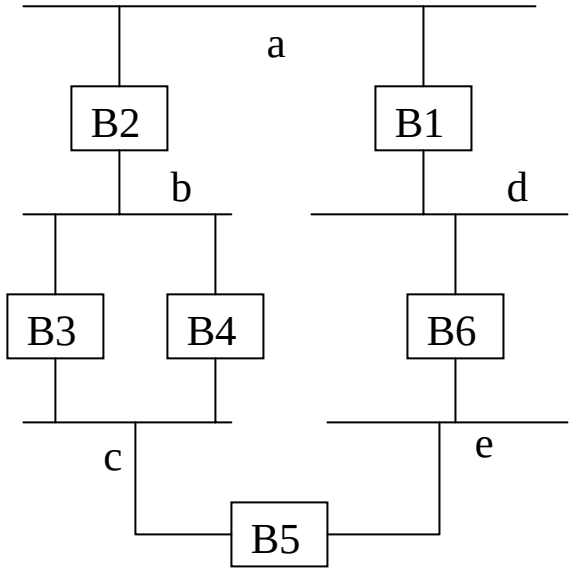
Transparent routing and problems with loops

In a system with multiple local area networks (LANs) interconnected using bridges, the brides implement a transparent routing function which ensures that computers can communicate with any other computer regardless of which LAN they belong to. This is achieved by the bridge monitoring the source address and the destination address of data packets on the networks they are attached to and copying these packets between the networks where necessary.

In the example here, if X sends a message to Y using LAN1, then bridge A detects that Y does not belong to LAN1 and so it re-broadcasts the message on LAN2. B now sees a message on LAN2 but knows that the destination is not on LAN2 and therefore re-transmits the message on LAN1. A detects this and repeats the process, giving a potential for packets to loop continuously between LAN1 and LAN2.

Question 3(b) 15 marks

EE4004 Telecommunications Summer 2006



ID.sender = the ID number of the bridge sending a message

ID.presumed_root = the ID of the bridge which the transmitting bridge thinks is the root.

distance.presumed_root = the number of "hops" between the transmitting bridge and the presumed root.

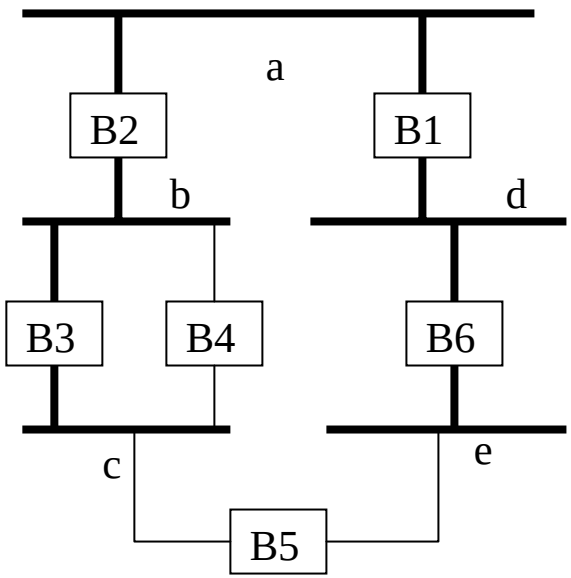
B1 is one hop away from B2 and B6.
B1 is two hops away B3, B4 and B5.

Each bridge has an identification number (e.g. an IP address). The numbers are 1 to 6 here. The bridge with the lowest number is called the "root".

Algorithm:

1. Each bridge sends out a message of the form [ID.sender | ID.presumed_root | distance.presumed_root].
2. Each bridge stores the best message that it has received so far.
   [S | R | D] is "better" than [S' | R' | D'] if
   R < R', or
   R = R' and D < D', or
   R = R' and D =D' and S < S'
3. When a bridge realizes that it is not the root it stops sending messages.
4. When a bridge gets a better message on a port than the ones it has sent so far on that port it stops sending messages on that port and only relays other messages after adding one to their distance.
5. Eventually, only the root (smallest ID) sends messages and the other bridges relay them.
6. This process is repeated regularly to cope with updates in the network topology.

Question 3(b) continued



The algorithm applied to this network:

1. Initially all bridges assume they are “the root” so

   B1 sends message [B1:B1:0] B2 sends message [B2:B2:0]
   B3 sends message [B3:B3:0] B4 sends message [B4:B4:0]
   B5 sends message [B5:B5:0] B6 sends message [B6:B6:0]

2. When B2 gets [B1:B1:0] it knows it is not the root and stops sending messages claiming to be the root. However, it relays the message [B2:B1:1] onto its lower ports.
3. When B6 gets [B1:B1:0] it stops sending its own messages and relays [B6:B1:1] on its lower port.
4. When B4 gets [B3:B3:0] on both ports it knows that a “better” bridge is attached to both its ports and so it is not on the shortest path.
5. When B3 gets [B2:B2:0] and later it gets [B2:B1:1] it knows it is not the root and only relays messages onto its lower port. It eventually only relays the message [B3:B1:2]
6. B5 eventually receives the messages[B3:B1:2] and [B6:B1:1] indicating that B3 and B6 are both closer to the root than itself and therefore that it is not on the shortest path.