

## CS4251 HW#4 Solutions

1. a. See table 1. Note that the routing table will have a single "next hop"; all possible answers have been included in this answer (see part d., also).

From	To	Next Hop
A	B	B
	C	C
	D,E,F	B or C
B	A	A
	C	A or D
	D,E,F	D
C	A	A
	B	A or D
	D,E,F	D
D	A	B or C
	B	B
	C	C
	E	E
	F	F
E	A,B,C,D,F	D
F	A,B,C,D,F	D

Table 1: Routing Table for 1a

- b. Many of the routes have two possible options (for example, A to D may go through B or C). Even if A tries to intelligently choose the correct route (by looking at congestion on its links, round-robin scheduling, etc), it doesn't know the congestion on the links between B and D and between C and D. It may overload one of those links with its choice and leave the other link unused.

Furthermore, sometimes the shortest link isn't the best. If the A-B link is saturated, A can still get to B by going C-D-B.

- c. Yes, VC schemes that take into account the current congestion and allocate an efficient path can get around these issues.
  - d. Use a scheduling algorithm over all of the possible next hops. Even a round-robin schedule between two possible links will probably provide better utilization. A congestion-aware solution would be even better.
2. If the basic scheme is to always send high priority traffic over low priority traffic, then if there is a sufficient amount of high priority traffic to saturate the link, no low priority traffic will ever get sent. It will wait until there is no more high priority traffic to send, which may never happen.

If  $r$  bits are allocated for low priority traffic, then the sender can keep two queues and multiplex the line over time. Low traffic priority would get  $r/R$  units of time, and high priority traffic would get the rest. If a particular queue was empty at a given moment, then its time slice could be used for the other queue.

Another scheme would be to track the bandwidth being allocated to each queue and decide which queue should come next. The high priority queue would always be chosen unless it is empty or has had  $R - r$  of the total  $R$  bandwidth for some sample period. Otherwise, the low priority queue is chosen.

Obviously, the high priority traffic's total available bandwidth will be reduced when there is low priority traffic. Also, the high priority traffic may have increased latency, since it might be queued behind low priority traffic if it arrives during a low-priority timeslice.

3. Allocations are shown below:

**A-B-C** 10

**D-A-B** 75

**A-B-E** 15

**B-E-G** 15

**C-G** 20

**C-G** 20

**D-E-G 15**

4. The bandwidth in A-B-E can be reallocated. The B-E bandwidth is not used, but the A-B bandwidth is can be given to the D-A-B connection. The new allocations are:

**A-B-C 10**

**D-A-B 90**

**B-E-G 15**

**C-G 20**

**C-G 20**

**D-E-G 15**

5. A minimum rate is useful for meeting Quality-of-Service guarantees for application, such as real-time multimedia. To give a bandwidth guarantee, allocate that amount of bandwidth in the fair share scheme, and then proceed with calculations. It is possible in this case since A-B-E's links all have a bandwidth of at least 30. The final allocations are:

**A-B-C 10**

**D-A-B 60**

**A-B-E 30**

**B-E-G 0**

**C-G 20**

**C-G 20**

**D-E-G 30**

6. Obviously, VBR can provide minimum bandwidth QoS, since it allocates bandwidth for each channel. Because of VBR (as opposed to CBR), unused bandwidth can be allocated to another channel that needs it. However, this allocation may not always be perfect because of statistical multiplexing.

IP, on the other hand, does not provide QoS guarantees; it promises only to fill the link if traffic is available, without regard to the contents of the traffic. Therefore, its multiplexing is nearly optimal, at the cost of bad QoS.