

Homework #3

1. Chapter 3, #22 For satellite channel, $\tau = 270ms$ Using the equation:

$$U = \frac{\omega D}{2F + 2\tau C} \quad (1)$$

Assuming the headers are small, $D \approx F$. Hence,

$$U = \frac{\omega 1000}{2 \times 1000 + 2 \times 270 \times 10^{-3} \times 10^6} \quad (2)$$

With 3 bit sequence number

- (a) Stop-and-wait, $\omega = 1, U = 0.18\%$
 (b) Protocol 5, $\omega = 2^3 - 1 = 7, U = 1.29\%$
 (c) Protocol 6, $\omega = (7 + 1)/2 = 4, U = 0.74\%$

With 8 bit sequence number,

- (a) Stop-and-wait, $\omega = 1, U = 0.18\%$
 (b) Protocol 5, $\omega = 2^8 - 1 = 255, U = 47.05\%$
 (c) Protocol 6, $\omega = (255 + 1)/2 = 128, U = 23.62\%$

2. Chapter 3, #24

For satellite channel, $\tau = 270ms$. We also assume $Ack \approx 0$, and $D \approx F$ for small headers and very short acknowledgements.

For $\omega F/C < F/C + 2\tau$

$$\begin{aligned} \text{Throughput} &= \text{utilization} \times \text{data rate} \\ &= \frac{\omega FC}{F + 2\tau C} \\ &= \frac{\omega \times 512 \times 8 \times 64k}{512 \times 8 + 2 \times 270m \times 64k} \end{aligned}$$

For $\omega F/c \geq F/C + 2\tau$, $\text{Throughput} = C$

- (a) $\omega = 1, \text{Throughput} = 6.78kbps$
 (b) $\omega = 7, \text{Throughput} = 47.5kbps$
 (c) $\omega = 15 \text{ or } 127, \text{Throughput} = 64kbps$

3. Chapter 5, #1.

File transfer, remote login, and video on demand need connection-oriented service. On the other hand, credit card verification and other point-of-sale terminals, electronic funds transfer, and many forms of remote database access are inherently connectionless, with a query going one way and the reply coming back the other way.

4. Chapter 5, #5.

Four hops means that five IMPs are involved (If you use 3 or 4, I will not deduct any point) The virtual circuit implementation requires tying up $5 * 8 = 40$ bytes of memory for 1000 sec. The datagram implementation requires transmitting $12 * 4 * 200 = 9600$ bytes of header over and above what the virtual circuit implementation needs. Thus the question comes down to the relative cost of 40,000 byte-sec of memory vs. 9600 byte-hops of circuit capacity. If memory is depreciated over 2 years, which is $2 * 52 * 40 * 3600 = 1.5 * 10^7$ sec (any reasonable days for a year will be accepted), a byte-sec costs $6.7 * 10^{-8}$ cents, and 40,000 of them cost just over 2 millicents. If a byte-hop cost 10^{-6} cents, 9600 of them cost 9.6 millicents. Virtual circuits are cheaper for this set of parameters.

5. Chapter 5, #8

Going via B gives (11, 6, 14, 18, 12, 8).

Going via D gives (19, 15, 9, 3, 9, 10).

Going via E gives (12, 11, 8, 14, 5, 9).

Taking the minimum for each destination except C gives (11, 6, 0, 3, 5, 8).

The outgoing lines are (B, B, -, D, E, B).

6. Chapter 5, #9.

The routing table is 400 bits. Twice a second this table is written onto each line, so 800 bps are needed on each line in each direction. The answer for bandwidth per (full-duplex) line is 1600 bps.

Note: You should write your unit, say 800bps per direction or 1600bps per line.

Otherwise, 1 point will be deducted.

7. Find the shortest paths from each node to Node 1 in the network below using

(a) Dijkstra's Algorithm.

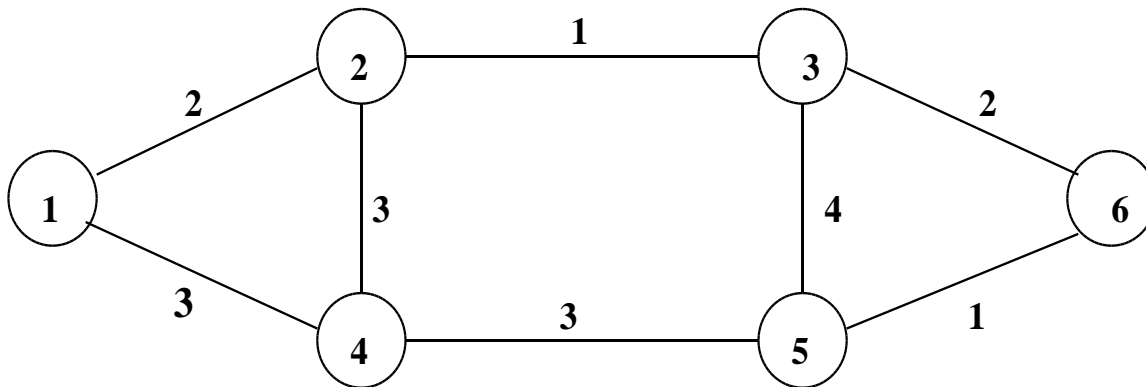
(b) (Centralized) Bellman-Ford.

Step	D(2)	D(3)	D(4)	D(5)	D(6)
0 : { 1 }	2	∞	3	∞	∞
1 : { 1, 2 }		3	3	∞	∞
2 : { 1, 2, 3 }			3	7	5
3 : { 1, 2, 3, 4 }				6	5
4 : { 1, 2, 3, 4, 6 }				6	
5 : { 1, 2, 3, 3, 5, 6 }					

Table 1: Dijkstra's algorithm for problem 3.

Step	n(2)	D(2)	n(3)	D(3)	n(4)	D(4)	n(5)	D(5)	n(6)	D(6)
0	?	∞	?	∞	?	∞	?	∞	?	∞
1	1	2	2	3	1	3	4	6	3	5
2	1	2	2	3	1	3	4	6	3	5

Table 2: Centralized Bellman-Ford algorithm for problem 3.

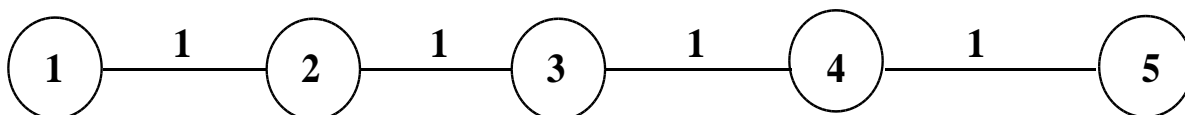


8. Find the shortest paths from each node to Node 1 in the network below

(a) Applying the Bellman-Ford Algorithm in Node order 2, ..., 5.

(b) Applying the Bellman-Ford Algorithm in Node order 5, ..., 2.

Note: If you use Distributed Bellman-Ford algorithm correctly, no points are deducted.



Step	n(2) D(2)	n(3) D(3)	n(4) D(4)	n(4) D(5)	n(4) D(6)
0	? ∞	? ∞	? ∞	? ∞	? ∞
1	1 1	2 2	2 2	3 3	4 4
2	1 1	2 2	2 2	3 3	4 4

Table 3: Centralized Bellman-Ford algorithm for 4(a).

Step	n(5) D(5)	n(4) D(4)	n(3) D(3)	n(2) D(2)
0	? ∞	? ∞	? ∞	? ∞
1				1 1
2			2 2	1 1
3		3 3	2 2	1 1
4	4 4	3 3	2 2	1 1
5	4 4	3 3	2 2	1 1

Table 4: Centralized Bellman-Ford algorithm for 4(b).