

**Midterm Exam SOLUTIONS**  
CMPSCI 591: Computer Networks  
Fall 2000  
Prof. Jim Kurose

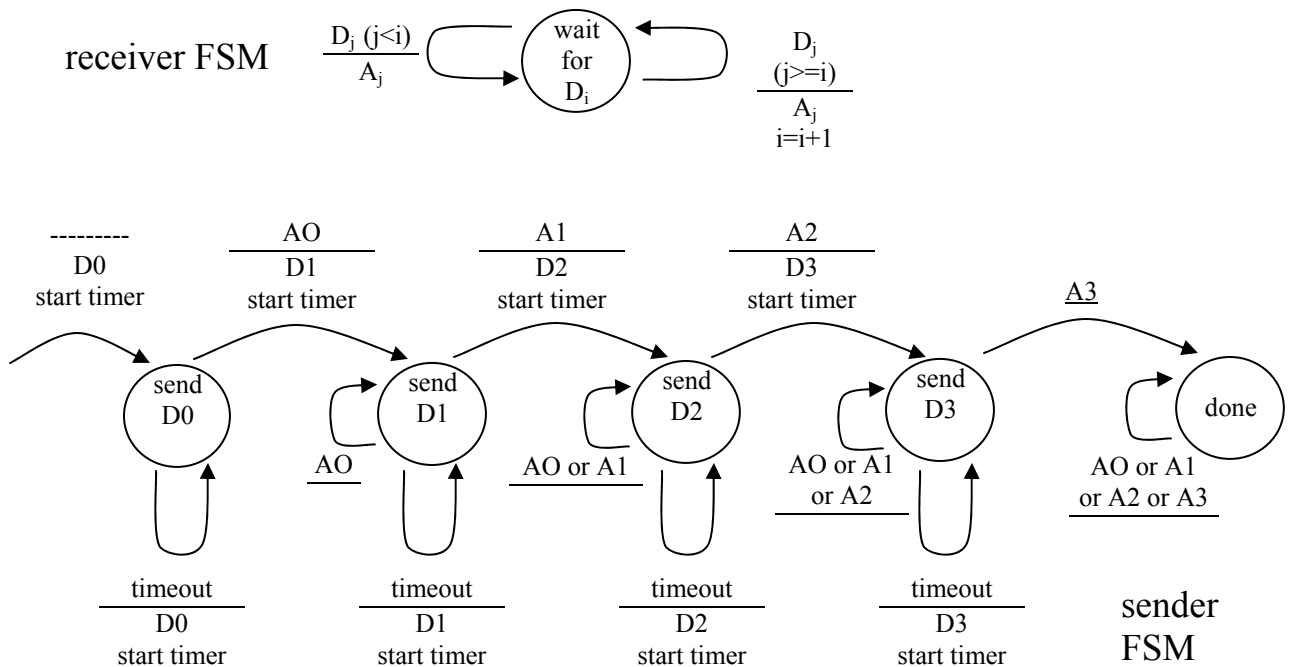
**Question 1: "Quickies" (24 points, 15 minutes)**

- A connection-oriented server has two sockets that are involved with communicating with a client. One socket is the “welcoming” socket – it accepts incoming connections. A new socket is created when a connection is accepted, and is used to communicate with the accepted client.
- A protocol is stateful if it maintains information about an on-going connection with the other side(s) over the course of a number of message exchanges. If no such information is maintained, and each new arriving message is handled completely separately from previous messages, the protocol is stateless. HTTP is stateless, FTP is stateful.
- Control commands (e.g., to retrieve a file or do a directory listing) are performed using a different connection than the connection on which ftp file data is exchanged. In this sense, ftp uses *out-of-band control*.
- In a recursive query, if a DNS server is not able to return a translation to the origin client, it (the server) will take the responsibility for getting that translation done - contacting another DNS server and returning the translation to the origin client. In an iterated query, the server would simply say it can not provide the translation; it would then be the responsibility of the origin client to contact another server.
- Source port, destination port, sequence number, acknowledgement field/bit, checksum bits.
- TCP uses cumulative ACKs and will always return an ACK for the most recent correctly-received in-order segment. If the same segment is being ACKed multiple times, it is likely that the segment following the segment being ACKed has been lost.



**Question 3: A reliable data backup protocol (26 points, 20 minutes)**

- A **receiver FSM** is shown below first. Because the sender need not worry about who received the data, receiver need only send an ACK number back. It need not include its identity. Note that the receiver, when waiting for  $D_i$  will ACK any packet it receives:
  - If it receives  $D_j$  ( $j < i$ ), then even though it has already ACKed  $D_j$  it will do so again, in case the sender has not yet received the ACK.
  - If it receives  $D_j$  ( $j \geq i$ ), this case is more interesting. If the sender is sending  $D_j$ , then the sender must have received ACKs for  $D_i, D_{i+1}, \dots, D_{j-1}$  (presumably from the other receiver). In this case, the receiver should ACK  $D_j$ , and then begin waiting for  $D_{j+1}$ . Note that if the receiver were only to ACK an incoming packet if it equaled the expected sequence number, this would be the same as requiring each sender to have to itself receive all of the data.
- The sender FSM is relatively straightforward. As soon as an ACK arrives for piece of data, the sender moves to the next state. A timer is used to retransmit. because of premature retransmissions, sequence numbers are used. An ACK carries the sequence number of the data it is acknowledging.
- It would not be simple to generalize this solution for an arbitrarily large amount of data. The difficulties relate to sequence number wrap-around, and unbounded channel delays. Let  $S$  be the maximum sequence number. To see the difficulty, suppose A sends  $D_0$ ; it will take a long time for  $D_0$  to get to B but C gets it immediately and ACKS. B then sends  $D_1, D_2$  up through  $D_S$ , with C ACKING immediately. The sequence number space now wraps around and A sends a new piece of data, reusing sequence number 0. C never gets this new piece of data, but finally B gets the very first  $D_0$  and sends  $A_0$ . A sees  $A_0$  and then moves on to reuse  $D_1$  – with the Newer  $D_0$  item never having been received!



**Question 4: Routing Algorithms (26 points, 15 minutes)**

<u>N</u>	<u>B</u>	<u>C</u>	<u>D</u>	<u>E</u>	<u>F</u>
A	1,A	3,A	4,A	infty	infty
AB		2,B	4,A	11,B	5,B
ABC			3,C	11,B	5,B
ABCD				7,D	5,B
ABCDF				6,F	

The shortest path routes are shown in thick lines below.

The distance table in D is shown below. The “tricky” entries is to F via C, and C via E.

		via			
		A	C	E	
dest	A	4	3	10	
	B	5	2	9	
	C	6	1	9	
	E	10	6	4	
	F	9	6	5	

