UCSD CSE CS 118 Computer Networks

Sample Final Exam Solutions

Directions: Write your name on the exam. Write something for every question. You will get some points if you attempt a solution but nothing for a blank sheet of paper. Problems take long to read but can be answered concisely.

Question	Maximum	Score
1	30	
2	10	
3	10	
4	10	
5	15	
6	10	
7	10	

1, Fundamentals, 30 points total, 2.5 points each: Give short 1 line answers for each question. If you are asked to give a reason for something, give the most important reason you can think of.

- **Media:** Why it is still cheaper to run twisted pair to workstations though the cost of fibre wire is fairly cheap.

  Optics are expensive (things like LEDs, lasers, photodiodes)

- **Physical Layer:** If a designer of a physical link finds that there is Intersymbol Interference (ISI) when he tries to send bits over a link, what can the designer do? (Any one reasonable option will do.)

  Send at a slower rate, or use a better quality link that has less capacitance.

- **Addressing:** Why Ethernet addresses are 48 bits in length although most LANs have only 1000 stations.

  To make them globally unique so that stations can use the same Ethernet address wherever they move.

- **Protocol Specifications:** Besides the specification of how the protocol responds to various events and the interfaces to higher and lower layers, what is the other major component of a protocol spec.

  `The message formats, bit and byte order in which message formats are to be transmitted.`

- **Bridges versus Routers:** Peter Protocol is building an application that needs low latency. Peter decides he wants his network to be full of routers although the bridges are slightly faster. Explain why.

  `Routers offer shortest path routing which can be less hops that routing along a spanning tree`

- **Spanning Tree Protocol:** Although we did not tell you this in class, bridges time out learned addresses faster after a spanning tree topology change. Why?

  `Because normally the reason to time out an address is because a station physically moves which can take minutes; however, a spanning tree change can make a large group of stations change sides wrt a bridge in seconds.`

- **Link State Routing:** Why a source S sending a link state packet may get a packet with source S and a higher sequence number than S is currently using.

  `Because S may have been sending a large sequence number before crashing and restarting with 0 which will (by the intelligent flooding rules) cause other routers to send back the old number to S (which causes S to jump.)`

- **Distance Vector Routing:** Hugh Hopeful suggests stopping the count-up of Distance Vector when the distance reaches the diameter of the network. What is the problem with Hugh's suggestion.

  `Diameter is not well-defined if we have node or link failures. A network in the shape of a wheel with a central router connecting every node and where every node is also connected in a ring can have a diameter of two. If the central router fails then the diameter increases to halve the number of nodes.`

- **Congestion Control:** Why can the throughput of a network go to zero (congestion collapse) if too much traffic is allowed to enter the network?

  `Because the network can be filled with traffic, all of which reaches part of the way to the destination and gets dropped because of other traffic that has a similar property.`

- **Transport:** What resources does a transport connection consume at a workstation even when the user of the connection is not sending any data.

  Bandwidth for sending keep alive messages, and memory in connection tables.

2. BGP versus distance vector, 10 points: Explain briefly the main differences between BGP and Distance vector in terms of a) how routes are chosen b) how routes are considered to be unreachable.

Distance vector always chooses shortest path routes. BGP chooses routes based on policies set by managers; since routers only pass routes that fit their policy to other routers, the result is a complex function of the policies specified in every router.

Distance vector considers a destination unreachable when the distance goes beyond some limit (e.g., 16 in RIP). BGP considers a destination unreachable when all the routes to that destination have a path list that includes this routers AS number.

3, Bridging and Learning, 10 points: Hugh Hopeful notices that at very high speeds it is hard for bridges to learn information from the source addresses in every packet. So Hugh suggests that bridges look at source addresses only in multicast packets. Since routing endnode protocols typically ensure that endnodes send multicast packets (e.g., ARPs, OSI hellos), this should ensure that each bridge periodically hears a multicast packet from each endnode. Also, since multicast traffic is so much less than non-multicast traffic, the processing load on bridges to do learning will be considerably reduced. Peter Protocol, who is brought in as a consultant, points out that not all endnodes send multicast periodically.

- As usual, bridges will flood unknown destination frames. What is one disadvantage of using Hugh's scheme of learning from multicast messages only (based on Peter Protocol's comment).

  If a station X does not send multicast, all frames addressed to X will be flooded, causing unnecessary traffic,

- All IEEE 802 LANS are supposed to support the SYSID-REQ message. When a station X on a LAN sends a SYSID-REQ message to the broadcast address all stations are supposed to send a SYSID-RESP message back to X. This can be used, for instance, by a manager to find how many stations there are on a LAN. How can Hugh use the SYSID-REQ message to avoid Peter Protocol's objection.

  Every bridge periodically sends a SYSID-REQ message to the broadcast address on all ports . If station Y sends a SYSID-RESP message back to bridge X that arrives on Port m of bridge X, then bridge X learns that Y is reachable through Port m.

- Would the SYSID scheme work well in a large Extended LAN with 8000 stations? Explain.

  The SYSID-RESP from stations like Y that do not send multicast may be lost in the flood of messages caused by 8000 responses, many of which the bridge already has information about.

4. Modifying Routing to do Load balancing, 15 points: In the figure below, there are two equivalently good routes from $R0$ to $R6$, one through $R1$ and one through $R3$, both with cost 4. Most routing algorithms today will only choose (arbitrarily) one of the two routes. Thus $R0$ will choose to send to either $R1$ or $R3$ but not to both. However, if $R0$ is a high speed router and the links are slow it may be better for $R0$ to send some packets to $R1$ and some packets to $R3$, thus balancing the load on the two paths and getting more throughput. We are going to see what modifications we need to add to routing for load balancing.

- In distance vector routing, a router R computes the neighbor that is closest to a destination $R6$ using the distances sent by its neighbors as follows:

  The closest neighbor for destination D is the neighbor N such that $Distance(D, N) + Distance(R, N)$ is the smallest over all neighbors.

  How would you modify this protocol to *also* compute *all* neighbors that provide *equal cost* routes to destination D.

  The closest neighbors are the SET of neighbors N such that Dist(D,N) + Distance(R,N) is minimal. (i.e., if there is more than 1, we keep track of all of them). This is exactly what is done in Equal Cost Mult Path (ECMP) load balancing today by most IP routers.

- It is also theoretically possible to not limit ourselves to equal cost paths. For example, in the figure above there is a path of cost 5 between $R0$ and $R6$ through $R5$. It seems that we could do better load balancing by having $R0$ send a small fraction of its packets through $R5$ as well. However, this kind of load balancing can lead to packet looping unless care is taken. Explain why.

At each hop on the path, the packet may be routed along a path longer than the shortest cost. But routing along shortest cost paths is the only way to ensure progress and avoid loops

5. Modifying Endnode Routing to do Load Balancing, 10 points: In the preceding page, we saw how to modify the router code to calculate equal cost routes. Now we turn to endnodes. In the figure below, we see that an endnode S on a LAN has two equally good routes (through either $R1$ or $R3$) to get to destination D. (The heavy lines represent LANs e.g., Ethernets). It is typically worth having S sending half its traffic to D to $R1$ and half to $R3$ because the LANs are much faster than the routers and the links between routers. Thus S needs to find out that $R1$ and $R3$ offer equally good paths to D so that it can split traffic among them. Notice that S must not choose $R5$ to split traffic to, because this only causes an extra hop.

The idea is to have a special QUERY message. If an endnode S has no information cached for D, S sends a QUERY to any router it knows about. The router sends back a REPLY with the list of routers that offer equal cost paths to D

- The algorithm used by a router to reply to a QUERY is trickier than you might think. It is obvious that $R5$ already knows that $R1$ and $R3$ are the best ways for $R5$ to get to D. However, S may choose to ask $R1$. How is $R1$ to know that $R3$ is also an equally good way to get to D? Assume the use of distance vector routing.

Since R1 is using distance vector, it knows the set of all neighbors (including R3 to D). Thus router R1 can easily calculate the set of neighbors that are on the same LAN as S and R1 that have the same cost to D as R1. It then sends this info to S.

- Suppose S has a cache entry for D that says the best two routers are $R1$ and $R3$. Then the link from $R1$ to $R2$ crashes. $R1$ quickly calculates that the best route to D is through $R3$ but S may still have an old cache entry. How should $R1$ react when S sends a packet from D to $R1$. How can S use this information to update its cache?

R1 can send a REDIRECT to S saying that its sending packets to D though R3. R1 then removes R1 from its cache of equal cost routers to get to D.

6. Modifying Transport Protocols to Deal with Load Balancing, 5 points: Hugh Hopeful uses a sliding window transport protocol. over a routing protocol and everything works fine. Hugh Hopeful later modifies the routing protocol so that it can do load balancing as shown above. However, he finds that performance actually decreases when he does load balancing.

- Why does performance go down?
- What simple change does Hugh need to make to his transport protocol implementation?

1. Performance goes down because packets are not being buffered out of order and are being dropped (if the Transport does Go-Back-N) or if it is like TCP and buffers packets, the sender may be forced to do Fast Retransmit which then halves his window.

2. He needs to buffer out of order packets at the very least, and switch to selective reject at the very best. To avoid the congestion window dropping because of fast retransmit, he may have to increase the threshold at which fast retransmit halves the window (3 currently) to more than the maximum amount of load balanced paths (say 10) or use more sophisticated techniques to reduce the congestion window based on the number of congestion encountered bits he sees. (This one is open-ended and any reasonable answer will get points).

7. Reverse Path Congestion Control, 10 points: Recall that in congestion control we had two separate problems. A router had to *sense* congestion on a link and then send *feedback* to all sources using the link. In class, we described the DECbit scheme in which the bit is passed to the destination and then back to the source. Here we describe another scheme in which a congestion bit is passed from the router directly back to the source.

In the figure below, assume that S is sending packets to D through $R1$ and all acks from D return on the same path. The *reverse path congestion rule* is as follows: if a packet p is received from a link l that is congested in the outbound direction, then a congested bit is set in the routing header of packet p. Thus in the figure, if the link from $R1$ to $R2$ gets congested, then the outbound queue at $R1$ and going to $R2$ will build up. When any packet from D to S arrives on this link (for example an ack), $R1$ will set the congested bit and this bit will get to S which then can send at a slower rate.

- What is one advantage of reverse path congestion control over the DECbit scheme?

It provides faster feedback (from point of congestion directly back to source, instead of through destination). It also does not require a path to pass bits from the congested router to the destination.

- The correctness of reverse path congestion control depends on an assumption. What is the assumption and why does it not always hold for all routing algorithms?

  It assumes that the route from S to D is the same as that from D to S. This is not guranteed by distance vector and link state because when there are many equal cost routes from S to Dand vice versa, the S to D and D to S calculations can pick different routes. BGP is even worse with even half the routes calculated by BGP being asymmetrical because of things like hot potato routing where an ISP gets rid of a packet as soon as possible. (All these details are not needed, if you just said ‘‘asymmetrical paths’’ you would get full points.)