

Carnegie Mellon
Computer Science Department.
15-744 Spring 2007
Midterm 1 Solution

Name: _____

Andrew ID: _____

INSTRUCTIONS:

There are 14 pages (numbered at the bottom). Make sure you have all of them.

Please write your name on this cover and put your initials at the top of each page in this booklet **except the last**.

If you find a question ambiguous, be sure to write down any assumptions you make.

It is better to partially answer a question than to not attempt it at all.

Be clear and concise. Limit your answers to the space provided.

Question	A	B	C	D	E	F	G
Points	/ 22	/ 12	/ 14	/ 16	/ 15	/ 18	/ 3

A True/False

+2pts for each correct answer, -2pts for each wrong answer, 0 pts if left blank, 0pts minimum per section

1. Which of the following are advantages that RED has over drop-tail queueing?
 - A. Has less burst losses
 - B. Has shorter queueing delays
 - C. Ensures roughly equal throughput for all flows
 - D. Ensures roughly equal throughput for all TCP flows

Solution: A, B

2. Which of the following is true about BGP? (Circle all letters that apply)
 - T F A BGP router always picks the path with the least number of router hops to the destination.
 - T F A BGP router always picks the path with the least number of AS hops to the destination.
 - T F An Autonomous System will announce routes learned from its customers to its peers.
 - T F An Autonomous System will announce routes learned from its peers to other peers.
 - T F If an Autonomous System learns of 5 different routes to a destination prefix, it will announce all 5 routes to its neighbors.

Solution: T - BGP is a path vector protocol
F - area hierarchies prevent hop count optimization
F - policy routing prevents AS hop optimization
F - routing always prefers more specific routes
T - this is how peers learn about their neighbor's routes
F - this will violate valley free routing
F - BGP only advertises routes that it uses

3. Modern Router Architecture

- T F A typical crossbar-based router uses input queueing so that it can implement sophisticated Quality-of-Service (QoS) processing.
- T F The iSlip crossbar scheduling algorithm (a round-robin Parallel Input Matching algorithm) is efficient, but in practice does not fairly allocate time slots.

B Short Answer

4. Modern high-end router architectures now store the entire forwarding information base (FIB) on every line card and can look up the route to *any* destination in a small amount of time. Previous router architectures used a route-cache mechanism in which lookups that were not in cache would get sent to the main CPU. *Briefly* state one major reason for this change:

Solution: a) A core router handles a huge diversity of flows, making caching less effective.
b) Problems such as Denial-of-Service attacks and scanning worms thrash route caches, degrading performance for all flows.

5. Give one reason that DNS lookups are run over UDP rather than TCP: (3 points)

Solution: OK: Connection-setup overhead, short-duration interaction NOT OK: Header overhead

6. Give one reason that streaming multimedia is run over UDP rather than TCP: (3 points)

Solution: OK: Variable delays from reliability, loss tolerant applications, drastic congestion control
NOT OK: connection setup overhead

Andrew is designing a Network Address Translator device to use at home so that he can let multiple computers share a single global IP address.

- (a) In his first iteration, Andrew's NAT maps an outgoing $\{\text{src}, \text{src port}, \text{dst}, \text{dst port}\}$ tuple to $\{\text{nat-IP}, \text{new port}, \text{dst}, \text{dst port}\}$. To his bogglement, he finds that *every* packet through his nat is dropped due to "corruption." What is Andrew doing wrong?

Solution:

The most likely problem is that Andrew has not updated the header checksum after changing the addresses and ports.

- (b) Andrew corrects this problem. Now he finds that he can successfully browse the web through his NAT, but he cannot use FTP. Why?

Solution: FTP embeds the IP address of the host in its protocol. If the NAT is not aware of the FTP protocol, it won't change them. The outside host will then attempt to contact an IP address that exists only inside the NATted area.

- (c) Through some clever hackery, Andrew fixes *that* problem too. Despite all of this, however, one of Andrew's friends claims that the NAT will still cause (at least) two other problems. What are they?

Solution: a) The NAT violates fate sharing. If the NAT is rebooted, Andrew's connections to the outside world will die.
b) The NAT prevents inbound connections. Programs such as peer-to-peer programs are unlikely to work.

C BGP Tomfoolery

Suppose a hacker obtains control of all the BGP-speaking routers in several different Autonomous Systems (ASes). Our hacker has each AS “hijack” several IP blocks. That is, each AS under his or her control announces via BGP that it owns IP blocks for which it does not. For example, our hacker has AS (CMU) announce a one-hop path to the IP block 18.0.0.0/8 (MIT).

7. Assuming that the AS graph still converges to a stable state, can this attack cause routing loops to form? Explain why or why not. (4 points)

Solution: No. BGP is a path vector protocol and it sends the sequence of ASs for a route in route announcements. This mechanism prevents an AS from showing up multiple times in a route and thus the formation of loops.

Since we didn't mention how arbitrarily the compromised ASes could behave, we gave full credit if you explicitly mentioned the possible loop created when packets forwarded into a hacked AS might be routed back out (and then back in, out, etc.). There are no other cases where routing loops could form after convergence.

8. Suppose the ASes under attack are identified. Can other ASes change their routing policies to ensure that their traffic still reaches the hijacked IP blocks? Explain. (4 points)

Solution: Yes, assuming that it still receives advertisements for a correct route, by decreasing its preference for all paths that go through the ASes under attack.

9. In response to this attack, suppose all ASes agree to check a central registry for IP block ownership before a path is considered valid. That is, whenever an AS receives a route to a prefix P , it checks that the last AS in the route actually owns P . For example, upon receiving a path to 18.0.0.0/8 (MIT), an AS will check that the last AS in the route is 3 (MIT). Can a hacker still hijack IP address blocks belonging to ASes he or she does not control? (i.e., can he or she cause traffic destined to those IP blocks to be routed to the ASes he controls?) Explain. (4 points)

Solution: Yes, at least in some cases. This only protects 1 hop paths. For example, a hacker can still advertise a two hop path to 18.0.0.0/8 from CMU that contains 3 (MIT) (such as 9 3) that will be considered valid. If this path is shorter than the correct path when received by an AS, it will be given preference.

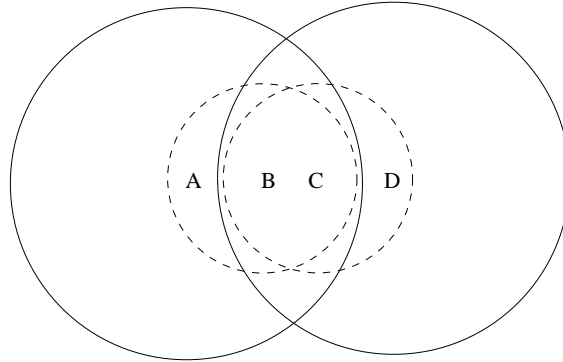
10. Suppose a solution was devised where IANA hosted a server on the Internet that was able to validate *all* AS paths. Assume that this server is always trustworthy and paths are valid if and only if the server says so.

True or false: With this solution an AS can always check the validity of a BGP path advertisement it receives. (2 points)

Solution: False. Contacting the server depends on its reachability, which depends on valid routes to the server existing. A hacker could hijack the IP block in which this server resides or just drop all traffic heading toward it.

D The Power of Wireless

Wireless radios that transmit with higher power will have a larger range. Consider the wireless topology below:



The solid circles represent the transmission radius of nodes A and D, respectively, and the dashed circles represent the transmission range of B and C, respectively. Assume that if two nodes' transmissions will interfere at a location if and only if they transmit at the same time and their transmission areas overlap. In these problems, assume that losses only occur due to collisions.

11. When node **A** transmits to node **B**, list the potential hidden terminals (in either direction - those who might clobber A's transmission or those who A's transmission might clobber) and exposed terminals. (7 points)

Hidden terminals:

Solution:

C and D are both hidden terminals. If they were transmitting, A would not see them and would clobber reception by either B or C. Similarly, if A was transmitting, D would not see the transmission, and might clobber reception at B.

Exposed terminals:

Solution: C. When A is transmitting to B, C could (in the absence of ACKs) transmit to D.

What about when node **B** transmits to node **C**?

Hidden terminals:

Solution: D is a hidden terminal. It might broadcast, not hearing B's transmission, and clobber reception at C.

Exposed terminals:

Solution: None

12. Suppose A is sending data to B and C is sending data to D, both at a constant bit rate equal to the physical capacity (“as fast as they can”).

- (a) Assume that no mechanism is used to detect or avoid collisions. What is the throughput of each transfer as a fraction of its send rate? (2 points)

Solution: $A \rightarrow B \approx 0$, $C \rightarrow D = 1$. All of A’s transmissions to B will collide with C’s transmissions.

- (b) Now suppose that each node uses CSMA/CA. Again, express the throughput of each transfer as a fraction of its send rate. (2 points)

Solution: $A \rightarrow B = 1$, $C \rightarrow D \approx 0$. A’s transmissions will clobber C’s because C will always backoff. A can not hear C so it will never back off.

- (c) Now assume that we also use an RTS/CTS scheme like that in MACAW. An RTS is sent if and only if no other RTS or CTS has been heard recently; ditto for a CTS. Assume that RTS/CTS exchanges are small compared to data packets and have negligible overhead. Again, express the throughput of each transfer as a fraction of its send rate. (2 points)

Solution: $A \rightarrow B \approx 1/2$, $C \rightarrow D \approx 1/2$. The first node-pair to get an RTS and CTS through will obtain the channel. The node pairs are symmetric, so about 1/2 the time, each pair will obtain the channel.

13. Consider the following 3 wireless applications:

- Voice-over-IP, where packets are very small and the send rate is constant.
- MPEG movie streaming, where the packet size is large and send rate is variable.
- Instant messenger chat, where packet size is small and send rate is variable.

For each application, list and explain why you would rather use CSMA/CA, RTS/CTS, or TDMA at the link layer. (3 points)

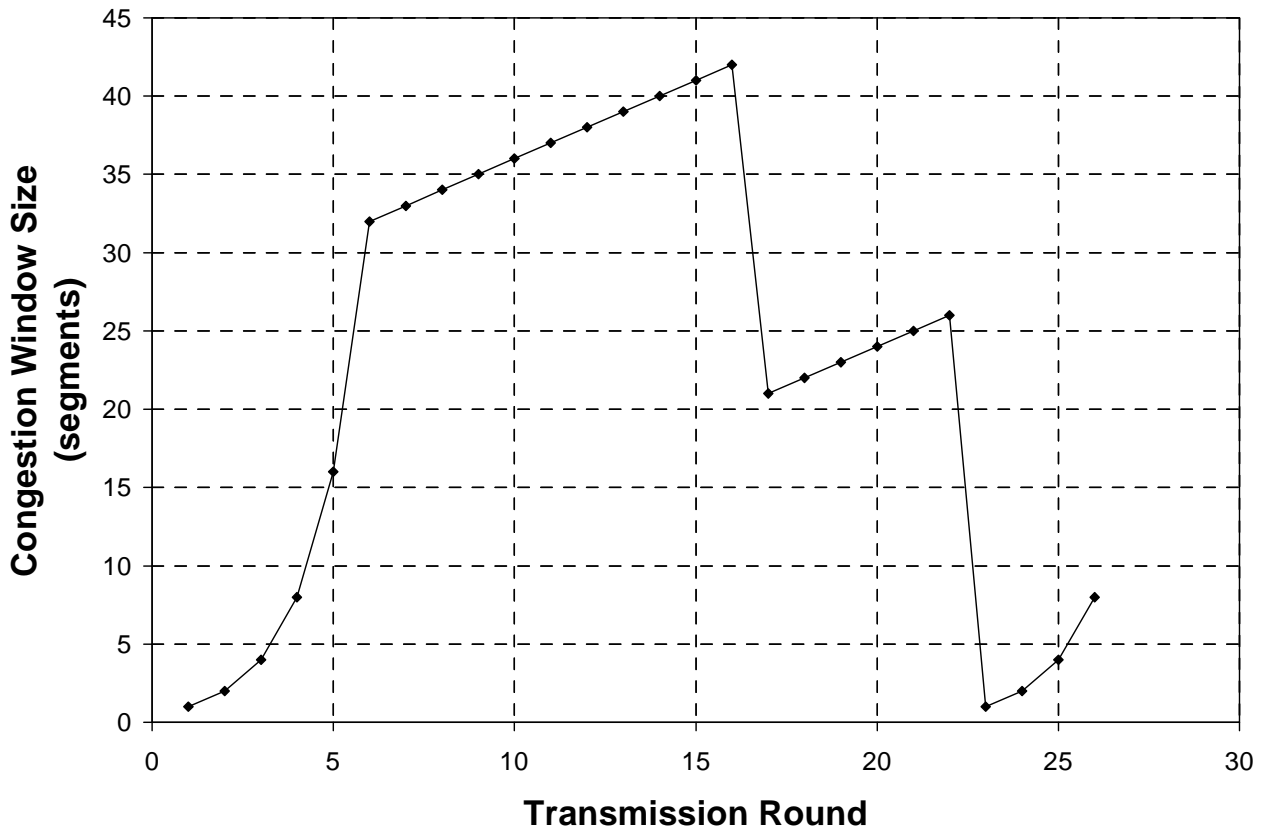
Solution: VoIP: TDMA because we can easily split up each a constant rate flow into constant size slots.

Movies: RTS/CTS because collisions of large packets are expensive and we want to avoid them. TDMA would not allow efficient use of the medium because the send rate is variable.

IM: CSMA/CA: The overhead of RTS/CTS is not worth it for small packets.

E Congestion Window

14. Consider the following plot of TCP window size as a function of time:



Assuming TCP Reno is the protocol experiencing the behavior shown above, answer the following questions.

- (a) Identify the intervals of time when TCP slow start is operating. (2 pts)

Solution: 1-6, 23-26

- (b) Identify the intervals of time when TCP congestion avoidance is operating (AIMD). (1 pt)

Solution: 6-23

- (c) After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout? (2 pts)

Solution: dupack

- (d) What is the initial value of ssthresh at the first transmission round? (2 pts)

Solution: 32

- (e) What is the value of ssthresh at the 18th transmission round? (2 pts)

Solution: 21

- (f) What is the value of ssthreshold at the 24th transmission round? (2 pts)

Solution: 13

- (g) During what transmission round is the 70th segment sent? (2 pts)

Solution: 7

- (h) Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion-window size and of ssthreshold? (2 pts)

Solution: 4,4

F One more bit, wafer thin!

Note: This problem is intended to be the hardest problem on the exam, but is only worth about many points as the other long problems which are hopefully easier. Make sure you balance your time appropriately if you run short on time.

Seeking fame and fortune, you take a small consulting gig to augment your stipend. The company you're consulting for, Rent-A-Puter, sells time on huge clusters of computers to scientists who don't want to set up their own clusters. However, they have a problem: for many of their clients, the most time-consuming step is transferring several terabytes of data to and from Rent-A-Puter. They ask you to help them solve their problem.

You think back fondly to 15-744, recalling that TCP provides congestion control using a single "bit" of information per packet: whether the packet was lost or not. XCP, on the other hand, puts a full indication of the new window size in every packet, providing faster convergence to efficiency. To do so, however, it requires changes to the endhosts, the routers, and even to the IP and TCP protocols themselves. You think that this may be too much to ask your scientists to stomach, but perhaps you could manage it if you could just manage to leave the IP and TCP headers unchanged so that they would continue to work with some legacy equipment.

You look at the IP header and think that you can make better use of the two bits allocated to ECN. Like ECN, you reserve the bit combination "00" for legacy traffic, so we'll ignore legacy traffic from here on. For starters, assume that all flows have the same RTT.

You decide to tell hosts if they're operating in one of three different states based on how busy the router is:

- If outbound link is over 100% utilization over the control interval T_c (say, 200ms), set the bits to "11". This state is **Overload**.
- If $80\% \leq \text{utilization} \leq 100\%$, set the bits to "10". This state is **High load**.
- If the utilization is under 80%, set the bits to "01"; this state is **Low load**.

Hosts respond according to which state the network tells them:

- **Overload: Multiplicative Decrease.** Just like TCP, if the network is overloaded, hosts will multiplicatively decrease their window:

$$cwnd_{next} = cwnd \cdot \beta$$

- **High load: Additive Increase:**

$$cwnd_{next} = cwnd + \alpha$$

- **Low load: Multiplicative Increase:**

$$cwnd_{next} = cwnd \cdot (1 + \gamma)$$

You set the additive increase parameter $\alpha = 1$, just like TCP. You set the multiplicative decrease parameter $\beta = 0.875$, so that if all flows cut down, utilization will go to about 87%, which would take the link from "overload" into "high load", which is just where you want it to be.

15. What is the largest “safe” value of the multiplicative increase factor, γ - the amount by which hosts will increase their windows when the system is operating in “low load”? (By safe, we mean such that the increase improves utilization without overloading the network.) Briefly explain your answer. Note that the hosts increase by $cwnd \cdot (1 + \gamma)$.

Solution: The largest safe value is one that ensures that a client operating at the “high load” threshold ($0.80 - \epsilon$) never goes into overload (100). Thus, $\gamma \leq 0.25$.

16. Using this parameter, how many round-trip times will it take a single flow with $RTT=200ms$ to achieve $\geq 80\%$ utilization of a 1 gigabit/second link, starting from an initial $cwnd$ of one 10,000 **bit** packet? (Size expressed in bits to make the computation easier.) You can leave your answer as an equation if you wish.

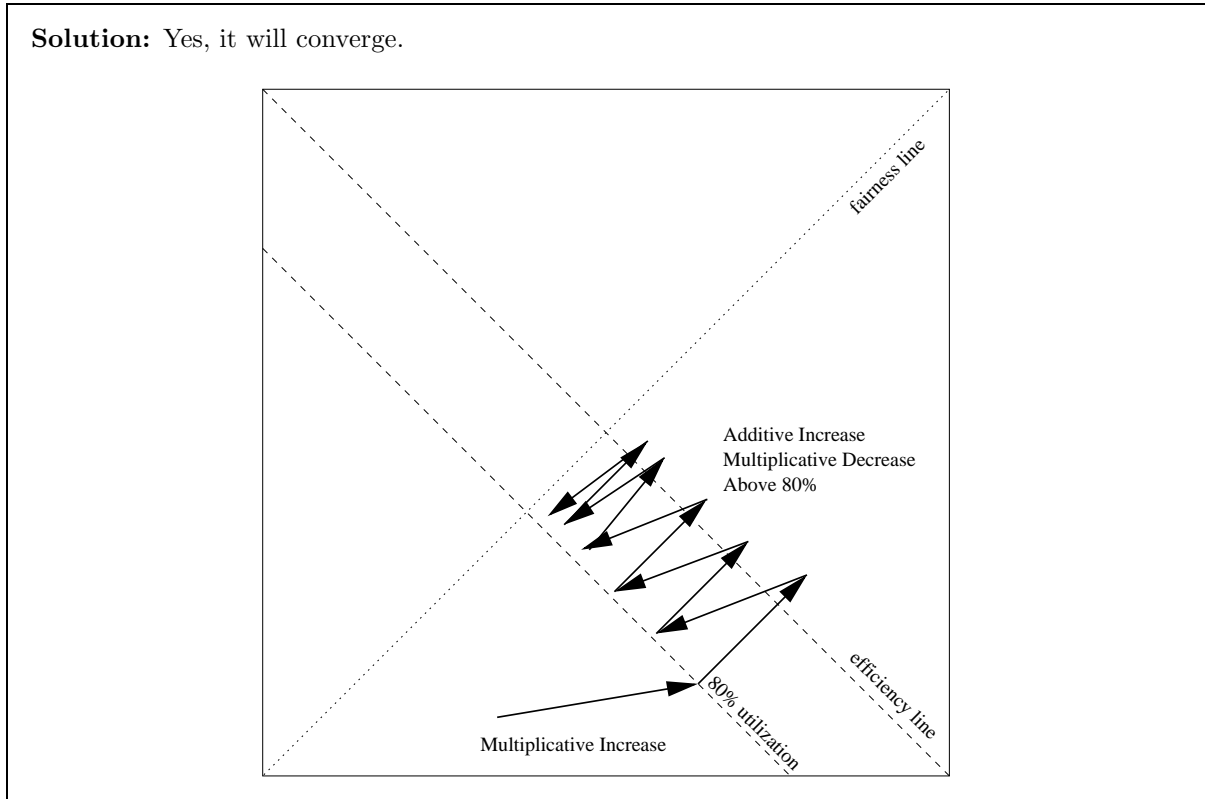
Solution: 80% utilization is 800,000,000 bits/sec, or

$$\begin{aligned} \frac{800,000,000}{10,000} &= 80,000 \text{ packets/sec.} \\ &= 16,000 \text{ packets/RTT} \end{aligned}$$

Thus, it will require $\text{ceil}(\log_{1.25} 16,000) = 44$ RTTs to reach 80% utilization.

17. You're pretty sure that this protocol will converge to efficiency, but will it converge to a fair allocation of bandwidth between TCP flows with the same RTT? Prove your answer using a phase plot. **Hint:** Make sure you show what happens if you start in low, high, and overload.

(lots of space here in case of drawing errors; don't feel compelled to use it all)



18. Regardless of the answer above, you're pretty sure that your protocol is *not* fair to flows with different RTTs. Assume that hosts know the duration of your control period T_c (which is the same order of magnitude as typical RTTs). Each flow has its own actual RTT, RTT_{flow} . The hosts perform the additive increase or multiplicative decrease just like TCP does, once per RTT.

What should the additive increase rule be to fix this basic problem of unfairness with different RTTs, so that all flows increase by the same additive amount over the control period, regardless of their actual RTT? (Note that this solution would also apply to TCP).

Solution:

19. (bit more difficult): What should the multiplicative increase rule be so that all flows increase by the same multiplicative factor over the control period, regardless of the flows' actual RTT?

Solution:

(this page deliberately left blank)

The End – Phew!

G 3 Free Points for Tearing Off Page: Anonymous Feedback

List one thing you liked about the *class* and would like to see more of or see continued (any topic - lectures, homework, projects, discussion site, topics covered or not covered, etc., etc.):

List one thing you would like to have changed or have improved about the class: