Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.033 Computer Systems Engineering: Spring 2020

Quiz I

This PDF is for students who will not have sustained access to Gradescope during the exam. If you can submit the quiz on Gradescope, you should do that.

There are **17 questions** and **17 pages** in this quiz booklet. Answer each question according to the instructions given. All questions are due at noon EDT on April 10th, 2020. Although you have 24 hours to complete this midterm to account for timezone differences, **we do not expect you to take all 24 hours**. This is written to be a **two-hour** exam.

- The questions are organized (roughly) by topic. They are not ordered by difficulty nor by the number of points they are worth.
- **If you find a question ambiguous, post a private question on Piazza.** We will be monitoring Piazza while the midterm is out for clarifications about the question. **All questions while the exam is out must be set to private visibility (only instructors can view them).**
- Some students will be taking a make-up exam at a later date. **Do not** discuss this quiz with anyone who has not already taken it.
- You may fill out this PDF in whatever way is easiest for you (e.g., print it out, write your answers, scan it; fill it in entirely electronically; etc.). **To submit this PDF, email it to Katrina (lacurts@mit.edu) before 12:00pm EDT on April 10th.**

This is an open-book, open-notes, open-laptop quiz, but you may **NOT** use your laptop, or any other device, for communication with any other person, nor should you use the Internet for help during the exam (i.e., no collaborating with peers, no Googling for answers). The only exception to this policy is posting private questions on Piazza.

Name:

I Virtual Memory

1. [4 points]: Alicia is designing a new CPU that relies on a large, single array of page table entries (PTEs) to manage paging translations, instead of using a tree structure (i.e., paging the page table as described in Lecture 3).

Select the description that best captures the trade-offs for this approach:

(a) Alicia's design uses more memory on paging structures but is faster (has less translation overhead).

(b) Alicia's design uses less memory on paging structures but is faster (has less translation overhead).

(c) Alicia's design uses more memory on paging structures but is slower (has more translation overhead).

(d) Alicia's design uses less memory on paging structures but is slower (has more translation overhead).

2. [6 points]: Consider the following segment of the page table from Alicia's CPU:

```
Index   | PTE
0x00000 |  ----
0x00001 | 0x0003
0x00002 | 0x0004
0x00003 | 0x0006
0x00004 | 0x0006
0x00005 |  ----
0x00006 | 0x0004
   ...
```

i. What physical address does the virtual address 0x00003204 translate to? Your answer should be a hexadecimal value starting with '0x'.
Example: `0x1010101`

ii. How many entries are in Alicia's page table?

(a) 2^{20}
(b) 2^{32}
(c) 2^{12}
(d) 2^{16}

iii. Alicia decides she would like to increase the number of virtual addresses that her processes have access to. She proposes switching to a hierarchical page table scheme that uses 8 bits for the outer page number, the next 12 bit for the inner page number, and the last 12 bits to specify the offset into the page. Will this plan accomplish Alicia's goal? Explain.

Continue on the next page...

Initials:

II Unix

3. [4 points]: Answer the following questions about Unix.

i. Which of the following is **not** involved in reading a block of an open file in Unix?

(a) The I-list
(b) The directory the file is in
(c) The I-node
(d) An indirect block

ii. `YourBigFile` is a maximum sized file in the Unix File System. How many disk blocks will you need to read in order to read a block near the end of the file, assuming that you've already read the I-Node?

(a) 1
(b) 2
(c) 3
(d) 4
(e) An unpredictable number

4. [10 points]: Consider the following simple program running on a Unix system

```
int main() {
  int i;

  for (i = 0; i <= 1; i++) {
    fork();
    printf("%d ", i);
  }
}
```

i. How many new processes are created by this code? Do not include the original process that runs `main()` in your count.

Initials:

ii. Recall that the Unix scheduler provides no guarantees on the order in which processes run. Of the options below, **select all** of the ones that are possible outputs for the program above.

(a) 0 1
(b) 0 1 1
(c) 0 1 1 0
(d) 0 0 1 1 1 1
(e) 0 1 0 1 1 1
(f) 0 1 1 0 1 1
(g) 1 1 0 1 1 0
(h) 1 1 1 1 0 0
(i) 0 0 1 1 1 1 1 1
(j) 0 1 1 1 0 1 1 1

Continue on the next page...

III Threads

5. [6 points]: You are implementing a banking system for the bank YourBank. Every time a customer makes an online request, the bank server forks a new process to process the request.

Currently there are three requests (and therefore three threads):

(a) Request 1 is a transfer of $5 from A's account to B's account

(b) Request 2 is a transfer of $10 from B's account to C's account

(c) Request 3 is a transfer of $20 from C's account to A's account

i. One version of the code for each of these threads looks like:

```
define Transfer (from, to, amount)
Acquire YourBank.lock
    from <- from - amount
    to <- to + amount
Release YourBank.lock
```

What kind of problem does this code present?

(a) Deadlock

(b) Race

(c) Performance

(d) None of the above

ii. Another version looks like this:

```
define Transfer (from, to, amount)
    Acquire from.lock
    Acquire to.lock
        from <- from - amount
        to <- to + amount
    Release from.lock
    Release to.lock
```

What kind of problem does this code present?

(a) Deadlock

(b) Race

(c) Performance

(d) None of the above

iii. A third version looks like this:

```
define Transfer (from, to, amount)
    Acquire  from.lock
        from <- from - amount
    Release from.lock
    Acquire to lock
        to <- to + amount
    Release to.lock
```

What kind of problem does this code present?

(a) Deadlock

(b) Race

(c) Performance

(d) None of the above

Continue on the next page...

IV Eraser

6. [6 points]: Suppose Eraser observed the following code on multiple threads during program execution:

```
int v;

lock(&a);
lock(&b);
v := v + 1;
unlock(&b);
unlock(&a);

lock(&b);
lock(&a);
v := v + 1;
unlock(&a);
unlock(&b);
```

Select the most appropriate description of code correctness and Eraser's output. By "correct" we mean that this code is free of race conditions and will run to completion (e.g., it will not deadlock).

(a) The code is correct but Eraser will report it as an error.

(b) The code is incorrect and Eraser will report it as an error.

(c) The code is correct and Eraser will not report it as an error.

(d) The code is incorrect but Eraser will not report it as an error.

V Disks

7. [7 points]: Answer the following questions about hard drive disks (HDDs).

i. On HDDs, reading is generally faster than writing.

(a) True
(b) False

ii. On average, the time it takes to read a piece of data on an HDD is dominated by the time it takes to seek a head to the platter.

(a) True
(b) False

iii. HDDs sometimes read from multiple platters at a single seek position in order to improve read times. This is an example of what basic performance strategy?

(a) Caching
(b) Parallelism
(c) Batching

iv. Which of the following is the best option to improve performance on a workload that consists of many random reads from a few large files?

(a) Lay the files out contiguously on the disk.
(b) Add a cache that stores the value of previous reads. Assume that the cache is large enough to hold 10% of the total data, and that it evicts data using a least-recently-used policy.
(c) Replace the HDD with a solid state drive (SSD).

VI MapReduce

8. [8 points]: Yunyi is running a MapReduce program that counts all occurrences of person names in a large collection of web text. She has one master machine and 100 worker machines. There are no other users on the system.

i. When workers respond to the master during the map phase, they need only tell the master that they have completed the task. The master has all of the other information it needs already.

   (a) True

   (b) False

ii. Suppose exactly one of Yunyi's worker machines fails. The **input data** that was stored on that machine is not available elsewhere in the system.

   (a) True

   (b) False

iii. Suppose exactly one of Yunyi's worker machines fails. The **intermediate data** on that machine is not accessible anywhere else in the system.

   (a) True

   (b) False

iv. Assume all 100 of Yunyi's machines are back online now and operating correctly. All map tasks are completed. For the reduce phase, it turns out that there are only ten intermediate keys on which to reduce. Yunyi decides to run each reduce task on ten separate workers, using all 100 workers at once, to increase parallelism. Assume no failures of any sort occur. What is the most likely downside of this approach, compared to running each task on a single worker?

   (a) Multiple workers might get different results for the same intermediate key.

   (b) It will likely create more network traffic.

   (c) The master now has to wait for all 100 workers to finish before the job is complete.

VII DNS

9. [8 points]: The diagram below shows three nameservers – `a.root.net.`, `names.com.`, and `ns1.google.com.` – and part of their DNS table. Each table gives a list of hostnames (first column), their addresses (second column), and the corresponding record type (third column).

```
a.root.net.             | com. | 2.2.2.2 | NS
                        | org. | 3.3.3.3 | NS
                        | net. | 4.4.4.4 | NS

names.com.              | google.com. | 5.5.5.5 | NS
                        | apple.com.  | 6.6.6.6 | NS


ns1.google.com.         | mail.google.com. | 7.7.7.7 | A
                        | cal.google.com.  | 8.8.8.8 | A
                        | google.com.      | 9.9.9.9 | A
```

Assume that each name server is the only name server for its respective domain (i.e., there are no replicas of any name server; this is not the case in practice). Assume that all name servers are operating correctly and that there are no failures of any type (no name servers fail, no packets are lost, etc.).

i. What is the IP address of `ns1.google.com`?

ii. What will `dig google.com` return?

iii. If a DNS client queries `names.com` for the IP address of `mail.google.com`, it will get a result if only recursion (but not caching) is enabled on `names.com`.

(a) Always
(b) Sometimes
(c) Never

iv. If a DNS client queries `names` for the IP address of `mail.google.com`, it will get a result if only caching (but not recursion) is enabled on `names.com`.

(a) Always
(b) Sometimes
(c) Never

Initials:

10. [4 points]: Answer the following True/False questions about DNS.

i. **TRUE / FALSE** In practice, caching reduces the amount of DNS traffic on the Internet more than recursion does.

ii. **TRUE / FALSE** DNS caches keep hostname-to-IP bindings as long as possible; they only ”forget” a binding when the cache is full and the binding needs to be evicted to make room for a new entry.

Continue on the next page...

VIII Ethernet

11. [6 points]: Answer the following questions about Ethernet

i. Because Ethernet implements carrier detection and interference detection, the only way for two packets to collide is if their respective senders start sending at precisely the same time.

(a) True

(b) False

ii. Ethernet's retransmission algorithm can be approximated by the following code, which is called every time there is a packet p to send.

```
load = 1
until p is sent successfully
   r = random number between 1 and load
   wait r time slots
   send p
   load = 2*load
```

Johnny is implementing this algorithm, and thinks that choosing a random number is too computationally expensive. Instead, he simply uses `r = load`.

Compared to the original algorithm, what will Johnny's network experience?

(a) More collisions, but higher utilization

(b) More collisions, and lower utilization

(c) Fewer collisions, and higher utilization

(d) Fewer collisions, but lower utilization

IX Routing

12. [6 points]: Consider the following network running a distance-vector protocol. Every link in the network has cost 1, except the link A-K, which has cost 100.

```
A -- B -- C -- D -- K -- Z
|                   |
--------------------
        100
```

At time 0, assume that every node in the network knows its neighbors, and its link costs to those neighbors, but nothing else. Advertisements are sent synchronously by all nodes every ten seconds at times 10, 20, 30, etc. Assume the latency on each link is virtually zero and that advertisements are integrated within one second. This means that if a node sends an advertisement at time 10, every node that is meant to receive it will have the advertisement, and have integrated it, by time 11.

There are no failures of any sort.

i. At what time (in seconds) will A know **a** route to Z?

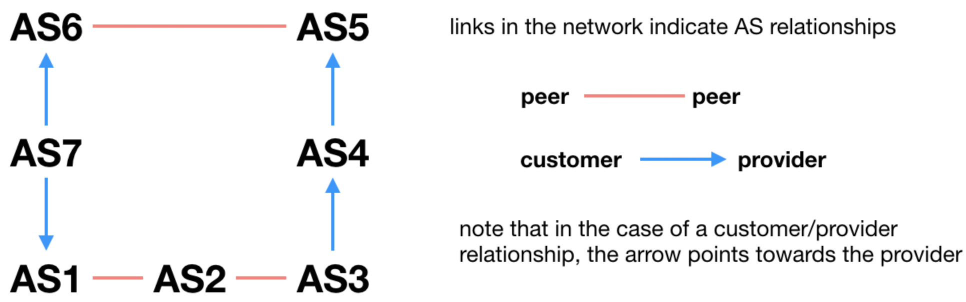
ii. At what time will A know the **correct** route to Z?

iii. At what time will C know **a** route to Z?

iv. At what time will C know the **correct** route to Z?

Initials:

13. [6 points]: The network below runs BGP as its routing protocol. Each node in the network is an Autonomous System (AS).



Answer the following True/False questions based on the above network.

i. **TRUE / FALSE** AS2 pays AS1 and AS3 for access to **all** of their routes.

ii. **TRUE / FALSE** AS4 will learn about a path to AS7 that includes AS6.

iii. **TRUE / FALSE** AS3 receives a route announcement from AS2 that includes AS7.

X 4D

14. [2 points]: In the 4D architecture, the decision plane consists of a set of engineers that determine how and where packets are forwarded every day.

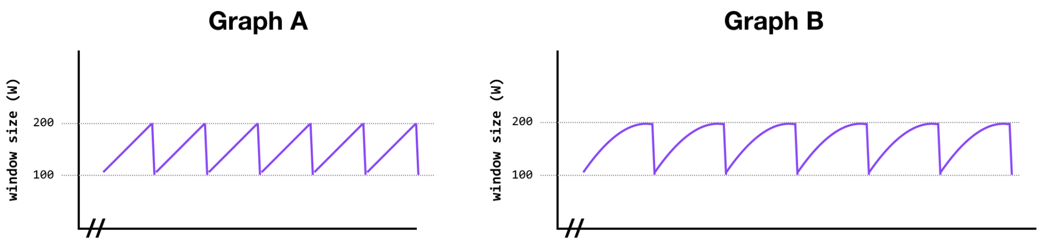
(a) True

(b) False

XI TCP

15. [5 points]: Consider two TCP connections sharing a single bottleneck link. There is no other traffic on the network.

The two graphs below plot the window size of one of the TCP connections over the same period of time. In both graphs, you see the familiar TCP "sawtooth" pattern, which is how TCP behaves in steady-state.



i. What is the x-axis on these graphs? *Hint: think about how the round-trip-time is - or isn't! - affected by traffic in the network*

   (a) The x-axis in Graph A is time as measured in seconds; the x-axis in Graph B is time as measured in number of round-trip-times.

   (b) The x-axis in Graph A is time as measured in number of round-trip-times; the x-axis in Graph B is time as measured in seconds.

ii. What is true about sender for the time period displayed on the graphs?

   (a) The sender experienced no packet losses.

   (b) The sender experienced some packet losses, and they were all handled by timeouts.

   (c) The sender experienced some packet losses, and they were all handled by fast-retransmit/fast-recovery.

   (d) The sender experienced some packet losses; some were handled by timeouts, others by fast-retransmit/fast-recovery.

16. [4 points]: Crystal is running a datacenter. The traffic in her datacenter consists of predominantly large flows; these flows transfer lots of data over time, and are not latency-sensitive. However, every once in awhile there is a **single** small flow. A small flow transfers much less data than the large flows, but is extremely latency-sensitive.

What is the best way for Crystal to manage traffic in her datacenter in order to meet the needs of both the large and (occasional) small flows?

(a) Have all of the traffic in the datacenter use DCTCP.

(b) Have all of the traffic in the datacenter use TCP.

(c) Set up the switches in the datacenter to use priority queueing, giving priority to any small-flow packets.

(d) Set up the switches to have limit memory resources ("shallow buffers") so that latency remains low.

Initials:

XII Queueing

17. [6 points]: Consider the following network topology, where two senders, S1 and S2, are sending packets to a router, which then sends them along to the destination.

```
S1
  \
   \
     Router ------ Destination
   /
  /
S2
```

The router has two queues: one for packets from S1, and another for packets from S2. Each of these queues can hold 1000 packets. The router uses a scheduling algorithm to decide how to send packets from these queues to the destination, and attempts to give each sender an equal share of the bandwidth over time.

S1 sends 1000 packets to the router. Each packet is between 5 and 15 bytes long; the average packet size is 10 bytes. S2 sends 100 packets to the router. Each packet is between 50 and 150 bytes long; the average packet size is 100 bytes.

You are able to observe packets on the link between the router and the destination. For each of the following observations, **decide whether the router could be using Weighted Round Robin (WRR) or Deficit Round Robin (DRR) (or neither)**. Treat each observation independently (i.e., the first observation has nothing to do with the second). Consider that **more than one algorithm may be possible**.

In the case of Weighted Round Robin (WRR), assume that the weight for both queues is 1/2. Although you do not know the exact packet sizes, you can assume that for every queue q, the number of packets to send is an integer (i.e., q.norm/min is an integer).

In the case of Deficit Round Robin (DRR), assume that the quantums for both queues are equal.

i. Ten packets from S1, followed by one packet from S2, followed by ten packets from S1, followed by one packet from S2. You can select more than one scheduling algorithm.
   (a) The router could be using WRR but not DRR
   (b) The router could be using DRR but not WRR
   (c) The router could be using either WRR or DRR
   (d) None of the above

ii. One packet from S1, followed by one packet from S2, followed by one packet from S1, followed by one packet from S2.
   (a) The router could be using WRR but not DRR
   (b) The router could be using DRR but not WRR
   (c) The router could be using either WRR or DRR
   (d) None of the above

iii. 1000 packets from S1, followed by 100 packets from S2.
   (a) The router could be using WRR but not DRR
   (b) The router could be using DRR but not WRR
   (c) The router could be using either WRR or DRR
   (d) None of the above

Initials: