University of California, Berkeley
College of Engineering
Computer Science Division – EECS

Spring 2018 Anthony D. Joseph and Jonathan Ragan-Kelley

Midterm Exam #2 *Solutions*

March 22, 2018
CS162 Operating Systems

Your Name:	
SID AND 162 Login:	
TA Name:	
Discussion Section Time:	

General Information:
This is a **closed book and one 2-sided handwritten note** examination. You have 110 minutes to answer as many questions as possible. The number in parentheses at the beginning of each question indicates the number of points for that question. You should read **all** of the questions before starting the exam, as some of the questions are substantially more time consuming. ***Make your answers as concise as possible.*** If there is something in a question that you believe is open to interpretation, then please ask us about it!

Write all your answers on the attached answer sheet on the last page of the exam (only the answer sheet will be graded), and put your **name & SID on both the answer sheet & the exam**. Before turning in the exam, **tear off the answer sheet**.

Good Luck!!

QUESTION	POINTS ASSIGNED	POINTS OBTAINED
1	20	
2	15	
3	26	
4	16	
5	23	
TOTAL	100	

DO NOT WRITE ANSWERS ON THIS PAGE

1. (20 points total) Short Answer.

a. (12 points) True/False and Why? **CIRCLE ANSWER ON ANSWER SHEET.**

i) A page fault can cause an unrecoverable error for a process.
TRUE*. If the page does not exist or the process does not have permission to access it. The correct answer was worth 1 point and the justification was worth an additional 1 point.*

ii) Consider a system where multiple threads are using locks to protect shared data. If deadlock occurs, we can resolve the deadlock by killing a thread without impacting the correctness of the program.
FALSE*. Abruptly killing a thread without any concept of rollback violates consistency of shared data. The correct answer was worth 1 point and the justification was worth an additional 1 point.*

iii) Clock algorithm, as an approximation of LRU, will always result in fewer page faults when more physical memory is added for the same access pattern.
FALSE*. False. Clock is a non-stack FIFO-based algorithm, so it is susceptible to Belady's anomaly. Consider the case where all use bits are set and it degenerates to FIFO.*

iv) Using lottery scheduling, each job receives lottery tickets proportional to runtime to ensure better response times.
FALSE*. Shorter jobs actually get more tickets to provide better response time. The correct answer was worth 1 point and the justification was worth an additional 1 point.*

v) A system that is running slowly on compute tasks, but has low CPU utilization can greatly improve its performance by adding more memory.
TRUE*. Slow performance with low CPU utilization indicates thrashing, which can be easily solved with more memory. The correct answer was worth 1 point and the justification was worth an additional 1 point.*

vi) Conflict misses cannot occur in a fully associative cache.
TRUE*. Blocks in a fully associative cache are not mapped to a specific location. Thus, it is impossible to have conflict misses. Only capacity misses are possible. The correct answer was worth 1 point and the justification was worth an additional 1 point.*

b. (4 points) Copy-on-Write.

i) Briefly, in one to two sentences, describe how to implement copy-on-write using virtual memory.
Mark pages as write-disabled/read-only. When a process tries to write, it will fault to the OS which can then copy a writable copy.

ii) In a multiprocessor system with per-core TLBs, you run a program that creates several forked processes, each with multiple threads, and notice

DO NOT WRITE ANSWERS ON THIS PAGE

disappointingly slow performance. Briefly, in one to two sentences, provide a possible explanation for why copy-on-write could be responsible for the poor performance.

Any time a thread in a forked process tries to write to a page, it will be copied to a new physical frame (page table update) and have its permissions updated. This results in a TLB shootdown, invalidating all (now stale) entries in all cores and thus causing slowdown.

c. (2 points) In the `start_process()` function for Pintos, why is the kernel is able to directly modify the `esp` variable returned by the `load()` function without performing an explicit translation? Briefly, in one sentence, provide an explanation.

User virtual memory is a subset of kernel virtual memory.

d. (2 points) Suppose we want to avoid invalidating TLB entries between context switches between different processes. Briefly, in one sentence, explain how we could avoid flushing the TLB.

Add the process ID to TLB entry.

DO NOT WRITE ANSWERS ON THIS PAGE

2. (15 points total) Distributed Computing with Banker's Algorithm. Suppose different TA teams write 3 distributed jobs (A162, B189, and C186) to run in a datacenter. Each job requires 3 resources (CPUs, network bandwidth, and disks) to run, initially available in the quantities indicated in the table below. Before submitting a job to the datacenter, the responsible TA team has to declare their job's maximum possible usage for each resource as specified in the table below. The datacenter always runs banker's algorithm before granting each request.

	CPUs	**Network Bandwidth**	**Disks**
Initial Quantity	10	15	17
A162	3	6	10
B189	5	13	3
C186	9	4	14

a. (5 points) Consider the following allocation:

	CPUs	**Network Bandwidth**	**Disks**
A162	1	3	8
B189	2	6	1
C186	5	2	5

Is this state safe? If so, give an example guaranteed safe execution. If not, give an example of requests that could deadlock the system. Show your work.

***SAFE**. A → B → C*

b. (5 points) Consider the following known safe allocation:

	CPUs	**Network Bandwidth**	**Disks**
A162	2	3	8
B189	4	5	2
C186	3	2	4

The CS186 TA's request an additional unit of bandwidth for their job, C186. Is it safe to grant this request? If so, give an example guaranteed safe execution. If not, give an example of additional requests that could deadlock the system. Show your work.

***UNSAFE**. Assume the loan is granted. A will complete, but B & C are unsafe. If B requests 8 bandwidths and C requests 4-6 CPUs. Note: if C requests a bandwidth, B only has to request 7-8, and if B requests a CPU, C only has to request 3-6.*

DO NOT WRITE ANSWERS ON THIS PAGE

DO NOT WRITE ANSWERS ON THIS PAGE

c. (5 points) The CS162 TA's want to guarantee their job, A162, is the first to be granted CPUs. Suppose they declare their CPU usage limit to be all 10 CPUs before submitting and then immediately request 10 CPUs. Briefly, in one sentence, explain whether they will succeed in being first and why or why not. *Assume the CS162 TA's cannot directly influence or coerce the other TA teams or the datacenter operators.*
No. Doesn't stop another TA team from requesting before the 162 TAs and getting the resources, since banker's algorithm only requires any (not all) job be able to finish.

DO NOT WRITE ANSWERS ON THIS PAGE

3. (26 points) Caching.

Suppose we have a four way set associative physically addressed cache of size 256KB and 16B blocks, on a machine that uses 32-bit physical addresses. We advance the clock hand before inspecting the use bit for clock algorithm.

Consider the following ordered memory accesses

```
1.     0xB3DA5C25
2.     0xE2DA5C21
3.     0xEA445C2B
4.     0xF4205C25
5.     0xD0035C71
6.     0xB3DA5C2B
7.     0x01115C25
8.     0xEA445C21
9.     0xE2DA5C2B
10.    0xF4205C25
11.    0xB3DA5C21
```

a. (6 points) What is the format of the physical address as used by the cache (i.e., how many bits are allocated to the tag, index, offset):

Tag = 16 bits	Index = 12 bits	Offset = 4 bits

b. (6 points) If the cache uses an LRU replacement policy:

i) How many compulsory misses will occur (*list the line numbers, if any, where misses happen*):

6; Lines: 1,2,3,4,5,7

ii) How many capacity misses will occur (*list the line numbers, if any, where misses happen*):

0

iii) How many conflict misses will occur (*list the line numbers, if any, where misses happen*):

3; Lines: 9,10,11

c. (6 points) If the cache uses a Clock algorithm replacement policy:

i) How many compulsory misses will occur (*list the line numbers, if any, where misses happen*):

6; Lines: 1,2,3,4,5,7

DO NOT WRITE ANSWERS ON THIS PAGE

ii) How many capacity misses will occur (*list the line numbers, if any, where misses happen*):
0

iii) How many conflict misses will occur (*list the line numbers, if any, where misses happen*):
1; Lines: 11

d. (6 points) If the cache uses a MIN algorithm replacement policy:

i) How many compulsory misses will occur (*list the line numbers, if any, where misses happen*):
6; Lines: 1,2,3,4,5,7

ii) How many capacity misses will occur (*list the line numbers, if any, where misses happen*):
0

iii) How many conflict misses will occur (*list the line numbers, if any, where misses happen*):
1; Lines: 11

e. (2 points) For this sequence of pages accesses, is using the Clock algorithm optimal? Why or why not?
MIN is the ideal page replacement policy and for this sequence of page accesses, clock produces the same number of page faults. Therefore, clock is optimal.

4. (16 points) Address Translation for RAM & Rem.

Consider a multi-level memory management scheme using the following format for virtual addresses (18 bits total):

Virtual Page # (4 bits)	Virtual Page # (5 bits)	Offset (9 bits)

a. (4 points) If the physical address space is 16 bits, what will X and Y be in the following format?

Physical Page # (____*7* bits)	Offset (____*9* bits)

b. (4 points) How many PTE's are in the first level page table? The second level?

First Level: *2^4 = 16 PTE's*
Second Level: *2^5 = 32 PTE's*

c. (8 points) Page table entries (PTE) are 16 bits in the following format, stored in big-endian form in memory (i.e. the MSB is first byte in memory):

Physical Page #	Unused (3)	Writable	Kernel	Dirty	Use	Directory	Valid

Using the scheme above, and the physical memory table on the next page, translate the following addresses. Assume that the Page Table Pointer points to **0x3000.** Intermediate page table entries should have the directory bit set. If you encounter an error, write **"Error"** in the Translated Physical address box instead of an address.

DO NOT WRITE ANSWERS ON THIS PAGE

Page Table Pointer: **0x3000**

Physical Memory

Address	**+0**	**+1**	**+2**	**+3**	**+4**	**+5**	**+6**	**+7**	**+8**	**+9**	**+A**	**+B**	**+C**	**+D**	**+E**	**+F**
0x0000	00	2A	A3	32	4A	BC	CD	DE	A1	A4	A3	AB	BC	A1	A3	3A
0x0010	AA	BB	CC	DD	EE	FF	11	22	33	44	55	66	77	88	99	00
…																
0x2000	A1	00	A3	00	A5	00	A7	00	A9	00	AB	00	AD	00	AF	00
0x2010	00	B1	00	B3	00	B5	00	B7	00	B9	00	BB	00	BD	00	BF
…																
0x3000	20	00	42	03	F0	03	60	00	20	03	F0	00	00	08	42	10
0x3010	00	12	00	14	00	16	00	18	42	12	42	16	42	18	42	04
…																
0x4200	12	32	00	54	56	01	78	02	9A	AB	03	CD	DE	04	32	00
0x4210	12	32	A3	A2	A1	DA	DD	1E	75	12	91	23	37	12	81	7C
…																
0x6000	DE	00	32	00	9A	AB	03	CD	56	01	78	02	12	32	00	54
0x6010	37	12	81	7C	75	12	91	23	A1	DA	DD	1E	12	32	A3	A2
…																
0xF000	A2	A1	FD	EF	98	01	CD	2A	56	14	32	12	65	54	42	32
0xF010	23	12	82	32	12	33	01	23	45	54	AB	CD	EA	12	32	12
…																

Virtual Address	**Translated Physical Address**
0x1024F (example)	*"Error"*
0x0442F	*0x562F*
0x0842D	*0x982D*
0x0CF1A	*"Error"*

DO NOT WRITE ANSWERS ON THIS PAGE

DO NOT WRITE ANSWERS ON THIS PAGE

0x0442F*: first 4 bits are "0001" so we look at index 1 in "0x3000" which is "0x4203" (in the new memory table). Last 2 bits are "11" so it is valid and a directory which is fine. Top 7 bits are taken and we 0 out the offset so we go to address "0x4200." The next 5 bits are "00010" so we look at the index 2, which is "0x5601." This has last bit as 1, which is valid, so we take the top 7 bits of "0x5601" and add that to the offset "0x02F" which gets us "0x562F."*

***0x0842D:** first 4 bits are "0010" so we look at index 2 in "0x3000" which is "0xF003". This has last 2 bits "11" which is fine. Top 7 bits are taken with 0 offset, so we got to "0xF000". We are looking for index 2 because next 5 bits are "00010", so we get "0x9801". This has valid bit, so it is good. we take top 7 bits here and add it to the offset, so we get "0x982D".*

***0x0CF1A:** Look at index 3 cause first 4 bits are "0011" and we find "0x6000". Looking at the last 2 bits, we realize this will cause some kind of error.*

***0x1024F:** Look at index 4 cause first 4 bits are "0100". We get "0x2003". Last 2 bits look good, so we got to address "0x2000". Look at index 1 because next 5 bits are "00001" and we find "0xA300". We see the last bit is not valid, so this will Error*

DO NOT WRITE ANSWERS ON THIS PAGE

5. (23 points) Advanced Bikeshare Scheduling.

In this problem, we want you to apply what you have learned about scheduling algorithms to scheduling resources in a new domain – bikeshare scheduling. Ford GoBikes are now in Berkeley and they need **your** help in figuring out how to schedule the resources in their system.

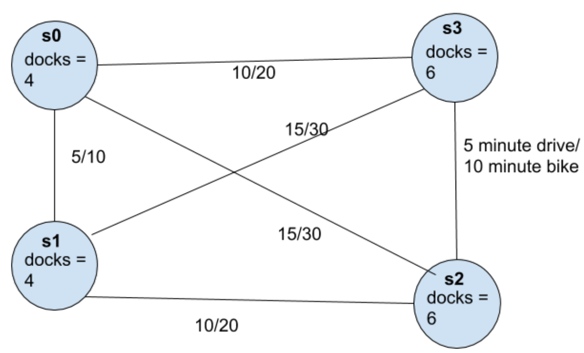
In a bikeshare system, the *bikes* are parked in *stations* each with some number of *docks* for bikes. A potential *rider* walks up to a station, checks out a bike from a dock, rides to a destination station, returns the bike to an empty dock, and walks off. In order to avoid blocking riders, the bikeshare operator periodically rebalances bikes by driving around in a *van* and adding or removing bikes from stations depending on demand.

Important Rebalancer Assumptions:

- Van has infinite capacity and infinite bikes and starts at s0 at 7:05
- Once the van arrives at a station, it stays there for 5 minutes to perform the rebalancing
- Multi-level priority uses the demand - (context switch time/10) to determine dynamic priority
- Pre-emptive algorithms pick up or drop off one bike on every visit; otherwise full rebalance to optimal state on every visit
- the optimal state is 50% full, every station starts in optimal state
- the demand can be represented by abs(n_bikes - n_empty_slots)/docks, and
- the context switch time is the drive time between stations + unloading/reloading time

Trips by riders

Start station	**Start time**	**End station**	**End time**
s1	7:00	s2	7:20
s3	7:05	s0	7:25
s3	7:10	s3	7:30
s3	7:20	s2	7:30



a. (18 points) Fill out the table ***in the answer sheet*** with the location of the van in each time slot, assuming each of the classic scheduling algorithms applied to this new context. Use **T** if the van is in transit. We have filled in Round-Robin for you.

DO NOT WRITE ANSWERS ON THIS PAGE

DO NOT WRITE ANSWERS ON THIS PAGE

<table>
<tr><th>Time</th><th>FCFS</th><th>Round-robin</th><th>Most imbalanced first (non-preempt)</th><th colspan="2">Most imbalanced first (preempt)</th><th>Multi-level priority</th></tr>
<tr><td>7:06</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>7:11</td><td>s1</td><td>s1</td><td>s1</td><td colspan="2">S1</td><td>s1</td></tr>
<tr><td>7:16</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>7:21</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>7:26</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>7:31</td><td>s3</td><td>s3</td><td>s3</td><td colspan="2">s3</td><td>s3</td></tr>
<tr><td>7:36</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>7:41</td><td>s2</td><td>s2</td><td>s2</td><td colspan="2">s2</td><td>s2</td></tr>
<tr><td>7:46</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>7:51</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>s3</td></tr>
<tr><td>7:56</td><td>T</td><td>T</td><td>T</td><td colspan="2">T</td><td>T</td></tr>
<tr><td>8:01</td><td>s0</td><td>s0</td><td>s0</td><td colspan="2">s0</td><td>s2</td></tr>
<tr><td>8:06</td><td></td><td>T</td><td></td><td>T</td><td>T</td><td>T</td></tr>
<tr><td>8:11</td><td></td><td>T</td><td></td><td>T</td><td>T</td><td>T</td></tr>
<tr><td>8:16</td><td></td><td>s3</td><td></td><td>T</td><td>s3</td><td>T</td></tr>
<tr><td>8:21</td><td></td><td>T</td><td></td><td>s2</td><td>T</td><td>s0</td></tr>
<tr><td>8:26</td><td></td><td>s2</td><td></td><td>T</td><td>s2</td><td></td></tr>
<tr><td>8:31</td><td></td><td></td><td></td><td>s3</td><td></td><td></td></tr>
</table>

b. (3 points) Briefly, in two to three sentences, explain what is common among the algorithms that work well in this scenario.
Non pre-emptive algorithms work much better than pre-emptive algorithms.

c. (2 points) Briefly, in two to three sentences, explain what are the characteristics of the problem that makes these algorithms particularly suitable.
Due to the large context switch time.

DO NOT WRITE ANSWERS ON THIS PAGE

6. (0 points) Just For Fun.
   a. What does the "d" in "adj" stand for?
   b. Who is your favorite anime character?

DO NOT WRITE ANSWERS ON THIS PAGE

DO NOT WRITE ANSWERS ON THIS PAGE

[No exam material on this page]
[Scratch paper]

DO NOT WRITE ANSWERS ON THIS PAGE

[No exam material on this page]
[Scratch paper]

DO NOT WRITE ANSWERS ON THIS PAGE

ANSWER FORM

Name		SID	

P1 (20 points): Short Answer

a.i) **TRUE** / **FALSE** ________________________________

a.ii) **TRUE** / **FALSE** ________________________________

a.iii) **TRUE** / **FALSE** ________________________________

a.iv) **TRUE** / **FALSE** ________________________________

a.v) **TRUE** / **FALSE** ________________________________

a.vi) **TRUE** / **FALSE** ________________________________

b.i) ________________________________

____

b.ii) ________________________________

____

c) ________________________________

d) ________________________________

P2 (15 points): Distributed Computing with Banker's Algorithm

a)	b)

c) **YES** / **NO** ________________________________

P3 (26 points): Caching

a) Tag: ____________ Index: ____________ Offset: ____________

b.i) Misses:______ Lines:____________ b.ii) Misses:______ Lines:____________

b.iii) Misses:______ Lines:____________

c.i) Misses:______ Lines:____________ c.ii) Misses:______ Lines:____________

c.iii) Misses:______ Lines:____________

d.i) Misses:______ Lines:____________ d.ii) Misses:______ Lines:____________

d.iii) Misses:______ Lines:____________

e) ________________________________

ANSWER FORM

P4 (16 points): Address Translation for RAM & Rem

a) Physical Page #: _______________ Offset: _______________

b) First Level: _______________ Second Level: _______________

c)

0x0442F		0x0CF1A	
0x0842D			

P5 (23 points): Advanced Bikeshare Scheduling

a) Time	FCFS	Most imbalanced 1st	Multi-level priority
7:06			
7:11			
7:16			
7:21			
7:26			
7:31			
7:36			
7:41			
7:46			
7:51			
7:56			
8:01			
8:06			
8:11			
8:16			
8:21			
8:26			
8:31			
8:36			

b) ____
____
____
____

c) ____
____
____
____

P6 (0 points): Just For Fun

a) ____

b) ____