ECE 254 Final Exam Solutions

J. Zarnett

January 2, 2017

(1A)

First-In First-Out

Step	cache[0]	cache[1]	cache[2]	Fault?
1	1	—	—	✓
2	1	2	—	✓
3	1	2	3	✓
4	4	2	3	✓
5	4	2	3	
6	4	2	3	
7	4	1	3	✓
8	4	1	3	
9	4	1	3	
10	4	1	3	
11	4	1	2	✓
12	3	1	2	✓
13	3	7	2	✓

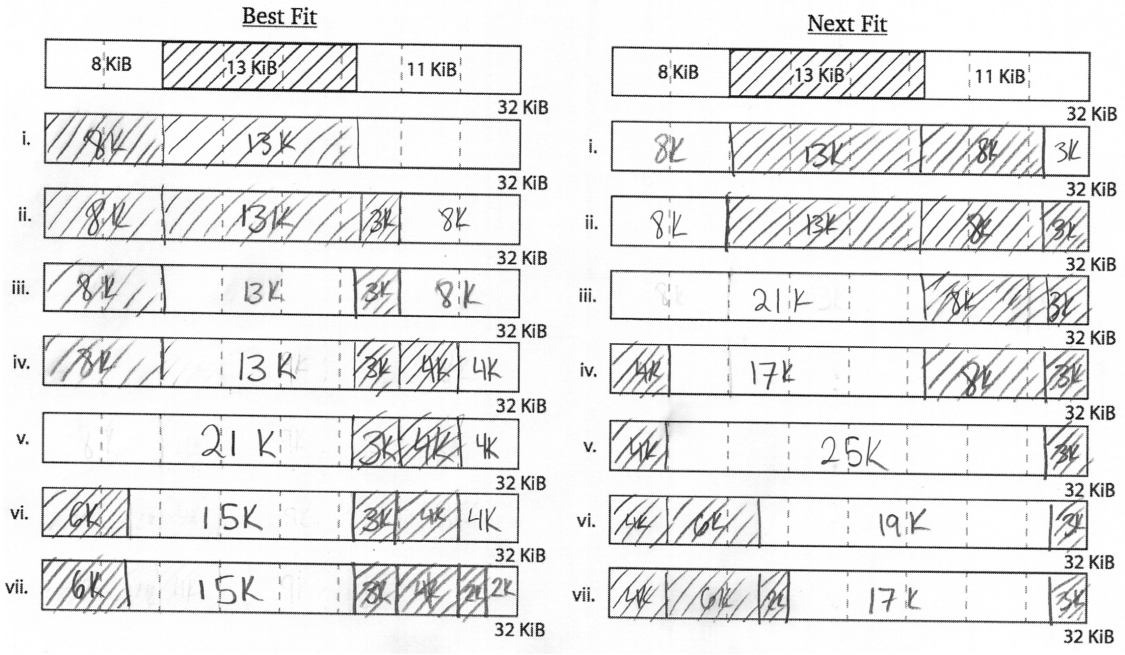
FIFO Page Faults: 8

Least Recently Used

Step	cache[0]	cache[1]	cache[2]	Fault?
1	1	—	—	✓
2	1	2	—	✓
3	1	2	3	✓
4	4	2	3	✓
5	4	2	3	
6	4	2	3	
7	4	2	1	✓
8	4	3	1	✓
9	4	3	1	
10	4	3	1	
11	4	2	1	✓
12	3	2	1	✓
13	3	2	7	✓

LRU Page Faults: 9

(1B)

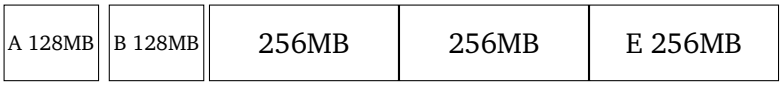


(1C)

In a 64 bit computer the address space is 2^{64} and with 4K pages then there would be $2^{64}/2^{12} = 2^{52}$ entries in the page table (at the most). If an entry is 8 bytes (2^3) then 2^{55} bytes of data are required to store this. Given that a computer with 16 GB of memory has some 2^{34} bytes of memory, there is no possible way we could store a page table with this many entries. (2^{55} is 32 Petabytes, which is an insane amount of memory.) Thus inverted page tables are necessary.

(1D)

Important to note that when D is deallocated it can be recombined with the 128 MB block to its left. But then the 256 MB block created cannot be combined with the other 256 block to its left because they are not from the same parent.



(2A)

A process in the queue twice will theoretically get to run about twice as often as a process in the queue only once.

If the current process gets blocked then its other entries need to be removed from the ready queue, otherwise it will be scheduled to run when it shouldn't!

(2B)

When the process is created it enters into the active queue for its priority level (124) and the bit 124 in the bitmap is set to 1 to indicate there is something in that queue. When the process is selected to run it runs on the CPU for up to a full timeslice at which point it is preempted ad moved to the expired queue. When all processes have moved to the expired queues, the expired queues become the active queues and the process will get another chance to run when its turn comes up. This repeats until the process finishes execution and terminates.

(2C)

$T_1 \rightarrow 0.5 + T_2 \rightarrow 0.2 + T_3 \rightarrow 0.143 + T_4 \rightarrow 0.1.$

Summing them up $U = 0.943$. The system is not overloaded because U is less than 1.

(2D)

Scheduling Algorithm	Execution Order	Turnaround Times	Waiting Times	Average Waiting Time
Highest Priority Period	$P_2 \rightarrow P_4 \rightarrow P_3 \rightarrow P_5 \rightarrow P_1$ (Swap of P_4 & P_3 is also OK)	P_1. 35 P_2. 2 P_3. 16 P_4. 9 P_5. 25	P_1. 25 P_2. 0 P_3. 9 P_4. 2 P_5. 16	10.4
First-In-First-Out	$P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5$	P_1. 10 P_2. 12 P_3. 19 P_4. 26 P_5. 35	P_1. 0 P_2. 10 P_3. 12 P_4. 19 P_5. 26	13.4
Round Robin	$P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5 \rightarrow P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5 \rightarrow P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5 \rightarrow P_1$	P_1. 35 P_2. 5 P_3. 30 P_4. 31 P_5. 34	P_1. 0+11+9+5=25 P_2. 3 P_3. 5+9+9=23 P_4. 8+9+7=24 P_5. 11+9+5=25	20
Shortest Process Next	$P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5 \rightarrow P_1$ (Swap of P_3 & P_4 is also OK)	P_1. 35 P_2. 2 P_3. 9 P_4. 16 P_5. 25	P_1. 25 P_2. 0 P_3. 2 P_4. 9 P_5. 16	10.4

Highest Priority Period and Shortest Process Next produce the same results. P_3 and P_4 are equal in both cases and they can go in either order. The solution above shows the two different orders, but either order is acceptable for either one.

(3A)

SSTF: 310, 283, 170, 420, 430

SCAN: 420, 430, 310, 283, 170

C-LOOK: 420, 430, 170, 283, 310

(3B)

- The file is spread out on disk into whatever block of space are free.
- The blocks are connected by pointers.
- The first block of the file contains pointers to the blocks on disk where the file is stored.
- We can have multiple levels of index block (linked scheme, multilevel index, or combined scheme).

(3C)

Device Driver: Software that connects to the kernel via a standard interface and allows standard kernel commands to be translated into device hardware commands.

Buffering: A section of memory used as the source or destination of data for performance reasons; often used to deal with the mismatch in speed between devices.

Spooling: A centralized buffer for devices like printers that can be used by only one process at a time until a job is complete.

Block Device: A device that operates on a chunk (block) of data at a time, such as a hard disk drive.

Character Device: A device that operates on one character at a time, or a linear stream of bytes, such as a keyboard.

(4A)

When we get to line 10 there will have been n calls to signal at line 8, n calls to wait at line 7, and one extra call to signal at line 4. Thus by the time we get to line 10, there is 1 additional call to signal and therefore the call to wait at line 13 will not block and there will be no deadlock. After that wait at line 13, the number of wait and signal calls is the same, making sure the system is back in its initial state; thus the barrier is reusable.

(4B)

- `count` not initialized
- Modify `index` outside critical section
- Assignment `start = &i` is bogus, start should be a new integer in each operation and not address of the thread counter because that value will keep changing.
- `pthread_exit` called with `*count` when it should be just `count`; wrong use of API
- `free(results)` is missing
- `pthread_detach` should be removed; can't detach a thread then join it