

CS 111 Midterm exam

CHEN; ALEXANDER QI-LI

TOTAL POINTS

89 / 100

QUESTION 1

1 Dirty bits 10 / 10

✓ - 0 pts Correct

- 10 pts No answer

- 5 pts Incorrect in explaining why dirty bit improves performance

- 5 pts Incorrect in explaining how dirty bit improves performance

- 2 pts Did not link performance increase to less disk I/O

QUESTION 2

2 ABI and system call interface 7 / 10

- 0 pts The subset relationship is clearly stated in the answer.

- 10 pts No answer

✓ - 3 pts Wrote down some sentences related to question, but didn't clearly mention system call interface is a subset of ABI.

- 1 pts More close to the answer but still missing clearly mentioning the subset relationship.

QUESTION 3

3 Shared libraries 10 / 10

✓ - 0 pts Correct

- 10 pts No answer

- 4 pts Missing details: multiple processes accessing the shared global data at the same time would be a problem.

QUESTION 4

4 System calls and trap instructions 8 / 10

- 0 pts Correct

- 10 pts No answer

✓ - 2 pts Did not mention transition of processor from

unprivileged mode to privileged mode

- 2 pts Did not mention OS runs appropriate code for the system call

- 2 pts Did not explain usage of trap handler

- 2 pts Did not mention OS will determine what trap was caused by trap instruction

- 2 pts Did not mention associated parameters preset by user application are saved

QUESTION 5

5 Working sets and page stealing 10 / 10

✓ - 0 pts Correct

- 10 pts No answer

- 5 pts Incorrect description of working set.

- 5 pts Incorrect description of page stealing algorithms.

- 3 pts insufficient description of page stealing algorithms.

- 3 pts Insufficient description of working sets.

- 2 pts Important element is that page stealing takes pages from processes whose working sets are too large and gives them to processes whose working sets are too small.

- 3 pts Goal of a working set is not to maximize the number of pages in memory, but to figure out the right number to have there.

- 2 pts Working set has nothing to do with TLB.

- 1 pts Just because a working set is large doesn't mean it isn't using its pages.

- 2 pts working set is not really about preventing thrashing, since that can occur even with properly implemented working sets.

- 3 pts Important to note that working sets are associated with processes and are controlled by their behavior.

- 3 pts Page stealing is used to build proper

working sets, not vice versa.

- **2 pts** Processes don't voluntarily release page frames. That's why it's called stealing.
- **2 pts** Page table usually bigger than the working set.
- **0 pts** Click here to replace this description.

QUESTION 6

6 Scheduling algorithm metrics 7 / 10

- **0 pts** Correct
- **10 pts** No answer
- **2 pts** Maximizing jobs completed does not necessarily translate to maximum throughput, by most definitions.
- **8 pts** Fairness not guaranteed by non-preemptive scheduling. Starvation not necessarily avoided, either.
- **1 pts** SJF has nothing to do with number of pages.
- **5 pts** Insufficient explanation of why metric is maximized.

✓ - **3 pts** Not for all non-preemptive algorithms.

Turnaround time won't be optimized for non-preemptive FIFO, for example. Question asked about non-preemptive algorithms in general, not just one example of such an algorithm.

- **5 pts** Turnaround time is not necessarily optimized. It's time of job arrival till time of job completion. With non-preemptive scheduling, one long-running job can kill the turnaround time of many other jobs.
- **0 pts** As stated in the test instructions, nothing on the back of the page is graded.
- **5 pts** Won't necessarily optimize time to completion. A long running job will not be interrupted, causing other short jobs to incur long times to completion. If you interrupted the long job for the short ones, average time to completion would improve.
- **10 pts** Did not specify a metric.
- **4 pts** "Minimizing context switches" isn't a performance metric, though doing so is likely to improve some metrics.

- **3 pts** Insufficient explanation of why metric is maximized.

- **10 pts** Response time may not be optimized with non-preemptive scheduling, since one long-running job can kill the response time of many other jobs.

- **2 pts** Won't also optimize average time to completion, since long-running job can kill time to completion of many other jobs.

- **4 pts** Throughput is typically defined as the amount of work produced by a system, not the number of jobs it completes. By the latter definition, non-preemptive scheduling doesn't optimize the metric, since you could finish many short jobs in the time it takes to finish one long job.

- **5 pts** Not clear exactly what you mean by "process speed".

- **4 pts** That's not the definition of mean response time. It's the average time to get some response from the system, not time to completion.

- **2 pts** Not very clear description of chosen metric.

- **4 pts** Non-preemptive scheduling is not likely to optimize number of deadlines met, since newly arrived jobs with short deadlines can't preempt a running job with a long deadline.

- **0 pts** Not the usual definition of time to completion, but correct as described.

- **10 pts** Round robin is not a non-preemptive algorithm

- **10 pts** "operations/second->output" makes no sense. Output is not a metric.

- **6 pts** Your assumptions are rarely true, and if not true, average time to completion is not optimized, by most definitions of that metric.

- **8 pts** Incorrect description of throughput.

Throughput is not the same as turnaround time, and turnaround time is not necessarily optimized by non-preemptive scheduling.

- **3 pts** Metric you're looking for is throughput, not "turnout" or turnaround time.

- **5 pts** Fairness not guaranteed (by any definition) for all types of non-preemptive scheduling, such as non-preemptive shortest job first.

- **2 pts** You're thinking of throughput, not total execution time.

- **10 pts** Round robin is not a metric, it's a scheduling algorithm, and not even a preemptive one.

- **0 pts** Click here to replace this description.

QUESTION 7

7 Worst fit and fragmentation 10 / 10

✓ - **0 pts** Correct

- **3 pts** Worst fit algorithms fit allocation requests into the largest free chunk of memory available, assuming no perfect fit is available. The remainder of that chunk will be as large as possible, meaning it will be well suited to match later requests.

- **3 pts** A best fit algorithm will choose the free chunk closest and larger in size to the requested allocation, which implies that the leftover free memory returned to the free list is likely to be a small chunk, poorly suited to matching future requests.

- **4 pts** The definition of external fragmentation is scattering small, useless chunks of free memory throughout the free list, so best fit is more likely to cause external fragmentation than worst fit.

- **10 pts** wrong answer

- **10 pts** No answer

QUESTION 8

8 Page tables for fork vs. shared memory

IPC 10 / 10

✓ - **0 pts** Correct

- **10 pts** No answer

- **5 pts** Major difference is fork results in copy-on-write, while shared memory IPC doesn't.

- **1 pts** new process has its own page table, but its contents are the same.

- **3 pts** No discussion of fork page table issues.

- **1 pts** Stack isn't shared in shared memory IPC.

- **2 pts** Fork need not be followed by exec, leading to COW issues.

- **1 pts** IPC shared memory almost always read/write, at least by one of the processes.

- **2 pts** Data segment also likely to change after fork.

- **0 pts** Not well worded, but I think you have the right idea.

- **10 pts** So what's the difference?

- **10 pts** What is the difference in their page table behavior?

- **3 pts** Difference won't be in the TLB.

- **2 pts** Copy-on-write issue.

- **5 pts** Not a thread issue. Copy on write is the main relevant mechanism.

- **4 pts** Shared memory doesn't share page tables. Just entries in different page tables point to the same page frame.

- **3 pts** IPC is not about libraries, it's about data.

QUESTION 9

9 Condition variables 7 / 10

- **0 pts** Correct

- **10 pts** No answer

- **4 pts** A condition variable is used to determine if some specific pre-defined condition has or has not occurred.

- **3 pts** If the condition does occur, one or more of the blocked processes will be unblocked and permitted to run.

✓ - **3 pts** The condition variable allows a process to wait for a specific condition without requiring the process to use a busy loop to check for the condition's occurrence.

- **10 pts** wrong answer

QUESTION 10

10 TLB misses 10 / 10

✓ - **0 pts** Correct

- **10 pts** No answer

- **3 pts** Missing case of invalid entry.

- **4 pts** Missing case of valid entry on disk.

- **3 pts** Missing case of valid entry in RAM.

- **1 pts** Page fault is on non-present, not invalid.

- **3 pts** Case with page on disk is present bit not set. Invalid bit is different.

- **4 pts** Different cases for valid page on disk and

valid page in RAM.

- **2 pts** What happens for an invalid entry?

- **1 pts** TLB is a cache of page table entries, not pages.

- **1 pts** Per test instructions, text on the back of the page is not graded.

- **2 pts** First step is to consult in-RAM page table.

- **1 pts** Disk isn't searched, since page table contains disk location of non-present pages.

- **2 pts** More details on not present case.

- **3 pts** Spatial locality does not play into TLB miss handling.

- **3 pts** Memory won't be searched. Either the page is present, not present, or not valid. Present pages have their PTE loaded, not present pages are fetched from disk, invalid pages cause an exception.

- **2 pts** Page table entry itself will indicate if page is on disk. No need to invoke clock algorithm.

- **1 pts** Dirty bit doesn't indicate whether a page is in memory or not. Present bit does. Dirty bit indicates if an in-memory page has been written.

- **1 pts** Invalid case is not that the page cannot be found, but that its PTE is marked invalid.

- **1 pts** How is it determined if a segmentation fault should occur?

- **2 pts** Not present pages are in the page table. They're just marked as "not present."

Midterm Exam
CS 111, Principles of Operating Systems
Winter 2018

Name: Alexander Chen

Student ID Number: 404 837 697

This is a closed book, closed note test. Answer all questions.

Each question should be answered in 2-5 sentences. DO NOT simply write everything you remember about the topic of the question. Answer the question that was asked. Extraneous information not related to the answer to the question will not improve your grade and may make it difficult to determine if the pertinent part of your answer is correct. Confine your answers to the space directly below each question. Only text in this space will be graded. No question requires a longer answer than the space provided.

1. In a virtual memory system, why is it beneficial to have a dirty bit associated with a page?

A dirty bit is used to show that a certain chunk of data in memory (often pages) has been written since it was loaded into memory. It shows that the page in memory is different than the corresponding page in the hard disk drive. It is beneficial to have a dirty bit so that we don't need to write to disk as often. Writing to disk only needs to occur when a page is ejected from memory. This has great performance benefits since disk is very slow.

2. What is the relationship between a system's Application Binary Interface and its system call interface?

A system's ABI is the binary executable that is run when a system call is performed. A system call is performed by the programmer using an API or more directly with a trap table and a system call number. The ABI, however, is only seen and executed by the machine. The programmer usually does not need to see the binary executable.

3. Why can't shared libraries include global data?

If shared libraries include global data, then they can no longer be shared correctly. If a shared library took data from multiple processes and stored it globally, then different processes can accidentally or maliciously access data they are not supposed to. Additionally shared libraries cannot be written. For example if the stdio library had global data storing its output buffer, multiple processes could fill this buffer and cause other processes to behave unexpectedly.

4. Describe how a trap instruction is used to implement a system call in a typical operating system.

On a trap instruction, the OS consults the trap table and the system call number which is stored in a register. The system call number dictates which system call to run from the trap table. The trap table links system call numbers to chunks of code that perform the system call. The trap table is setup by the OS and is loaded at boottime. When a program wishes to perform a system call, it loads the system call number into the register then uses the trap instruction. Once the OS takes over, it knows which system call to perform from the trap table and the system call number.

5. What is the relationship between the concept of working sets and page stealing algorithms?

Page stealing algorithms often try to optimize the size of working sets of processes. A working set is the set of pages a process has in memory. If the working set is too small, many page faults will occur. If the working set is too large the process could be taking up memory that would be better utilized by another process. Page stealing algorithms try to give page frames to processes whose working sets are too small by stealing page frames from processes whose working sets are too large. Thus these algorithms try to reduce page faults and thrashing.

6. Name a performance metric that is likely to be maximizable using non-preemptive scheduling. Why is this form of scheduling useful to maximize this metric?

Between the two performance metrics that were introduced (turnaround time and response time) turnaround time is much more likely to be maximizable through non preemptive scheduling.

Using the textbook's assumption that processes arrive at the same time, non-preemptive scheduling maximizes turnaround time. It does so because once a job is started, it is finished to completion, not interrupted like it would be with preemptive scheduling.

Non-preemptive SJF is the optimal scheduling algorithm given the arrival time assumption.

7. Why does a worst fit algorithm for managing a free list handle external fragmentation better than a best fit algorithm?

The worst fit algorithm leaves larger chunks of free memory. The best fit algorithm finds the smallest chunk of memory that fulfills a request, thus leaving very small chunks of free space once the chunk of memory has been allocated.

The worst fit algorithm finds the largest chunk of memory, thus leaving the largest chunk of memory after allocation occurs. Since the leftover memory is still large, it can still be used by other processes or another memory request.

8. Both shared memory IPC and the processes' data areas after a Linux fork() operation would require the page tables of two processes to point to the same physical page frames. What would be different about the two cases (other than being caused by IPC vs. forking)?

After a Linux fork(), two processes point to the same page frames. However, after one of the processes writes to a page, it is copied. This is called copy-on-write and results in the two processes no longer pointing to the same page frames. Thus, these processes cannot communicate simply through forking, because one process cannot write to where the other can read. Shared memory IPC is different because it allows 2 processes to share page frames until the IPC is closed. This allows one process to write to a location where the other process can read and allows communication.

9. What is the purpose of a condition variable?

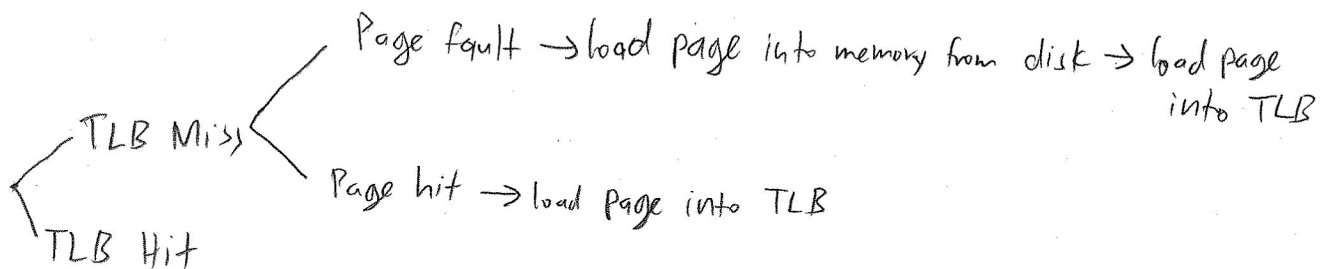
The purpose of a condition variable is to allow one thread to notify another when a certain condition occurs. For example, one thread may want to stop running until another thread carries out an I/O request. Using condition variables, the 1st thread is able to yield the CPU and the 2nd thread is able to notify the 1st when the I/O request is complete.

10. In a system using demand paging, what operations are required when a TLB miss occurs? What are the possible outcomes of those operations?

When a TLB miss occurs, the process must check its page table to see if the page is present in memory.

If there is also a page fault then the page must be retrieved from disk. If there is a page hit, then the page must be loaded into the TLB. Either way, the TLB will be checked again and the page should be there.

Possible outcomes:



It is also possible that the invalid bit is set at the PTE in which case the process would likely be killed.