

Final Exam Review ECE397A Operating Systems May 14, 2002

Dear Students,

This review has been put together to help you prepare for the final exam. You should use this AND the midterm review document to prepare for the exam. Don't hesitate to send me or the TAs email if you have additional questions.

The final exam will be held May 20, 2002, 8:00 (sorry guys), in GSMN 64 (Goessmann Lab. Addition 64). Try to be there 15-20 minutes earlier. Additional information may be announced on the web or sent to the class mailing list, please therefore check the class webpage regularly.

The Exam will be 1.5 hours and closed-book. All materials covered in the classroom, even if not in textbook, are required for the exam. Also make sure that you check the homework problems and the labs, you can get questions that relate to them.

All the material required for the midterm is also required for the final. 30% of the exam questions will be from the midterm material. Most of the questions related to that will be in form of short questions and/or appear combined with other topics such as for example deadlocks. Make sure that you revisit the exercises and code examples for synchronization and Java threading.

The remaining 70% of the questions will be from the new material we covered after the midterm. The following sections are required from the textbook for that part:

Chapter 8: Deadlocks – all sections

Recommended exercises: 8.3, 8.4, 8.5, 8.6, 8.8, 8.9

Chapter 9: Memory Management –all sections

Exercises: 9.1, 9.2, 9.3, 9.5, 9.7, 9.8, 9.9, 9.10, 9.11, 9.12, 9.13, 9.15, 9.16, 9.17, 9.18

Chapter 10: Virtual Memory –all sections with exception of 10.7.5-10.7.6

Exercises: 10.1, 10.2, 10.3, 10.4, 10.5, 10.6, 10.8, 10.9, 10.10, 10.11, 10.13, 10.14, 10.15, 10.16, 10.17, 10.19, 10.20

Chapter 11: File Systems –all sections with exception of 11.6, 11.7, 11.8, 11.9, 11.10

Exercises: 11.1, 11.3, 11.6, 11.7, 11.8, 11.9, 11.10, 11.12, 11.15, 11.16, 11.22

The article that I handed out related to memory management in current microprocessors is recommended but not required.

The exam questions will be of the following nature (note list is NOT exhaustive):

- (1) Short questions. Require a sentence or two as answers. For example, what is the role of a TLB? What is the difference between a regular cache and a TLB? Why do we have different page table formats? What are their advantages and disadvantages? What are the necessary conditions for deadlocks to occur? Is the OS aware of caching? What is the difference between deadlock avoidance and deadlock prevention? How does deadlock recovery works? Why do we construct a resource-allocation graph? What is the advantage and disadvantage of an inverted page table? What is the difference between a page, a segment, a cache block/line? What is Belady's anomaly? What is trashing? What is swapping? Can a segmented approach also be paged? What is demand based paging? How can protection be provided to your files? What is a symbolic (file/directory) link? How can we make sure a symbolic link is erased once a file that it points to is deleted? Compare various memory allocation algorithms. Is LRU always better than FIFO? Which one is easier to implement in hardware: LRU or FIFO? Why is a second-chance algorithm only replacing victims after a second access and not the first time? What is starvation? Give an example of a deadlock situation. Draw the resource-allocation graph and shortly explain why has deadlock occurred. What is a mount point in a file system? What is a partition? How does protection of files work in Unix? What about NT? Why do we have distributed file systems such as NFS? What is a possible problem with an acyclic-graph based directory structure? Give example of system calls used to access files. What happens during an file open() call?
- (2) Problems that require analytical analysis. For example, express the effective latency of a memory access analytically. Assume that there is a TLB, you know the page fault rate, the TLB hit rate, the percentage of page table accesses from cache vs main memory, the cache hit rate. Assume that the cost of accessing main memory and latency of the cache is known. Add other variables as you need to express what is happening during a page fault.
- (3) Problems that require programming or analysis of codes. Problems that require programming in pseudo-code, C, Java of a solution that has synchronization and may deadlock. I can provide a solution and ask you to determine if it can deadlock or if it can starve processes/threads. I can ask you to fix the problem by changing the synchronization part. I can ask you to represent and analyze the deadlock problem with resource-allocation graphs or similar approach.
- (4) Problems related to memory management techniques. I can ask you to compare/evaluate replacement techniques for various reference strings. I may ask you to design a new scheme based on existing ones. I can ask you to compare various memory allocation techniques and related implementations.
- (5) Design of a virtual memory system. For example, I can ask you to design a demand based paging system. I can ask you to draw (at block level) a design that

has certain characteristics, e.g., a TLB, two levels of caches, a page table type, paging/segmentation or other support, disk, etc. I will give you some parameters and ask you to design a system that meets certain design objectives, e.g., maximum page table size. I can ask you to explain what happens in your design under certain circumstances such as page faults, context-switch, thread-switching, cache misses, and TLB misses. I can ask you about how protection is provided and about the efficiency of your scheme. I can ask you about how memory is allocated during a replacement. I could ask you more subtle questions, e.g., can you have a cache miss and a TLB hit at the same time? What does that mean? What about if you have a cache hit and a page fault during a data access?

- (6) Simple design problems related to file systems. For example, I can ask you to compare the efficiency of various file access methods under a constructed scenario. I can ask you to show (or explain) an example of memory-mapped files under a certain memory management scenario. I can ask you to compare various directory structures and protection schemes.

Some of the exam questions may be similar to those from the list of exercises. Questions can be asked related to the examples given in class.

The best way to prepare is to study the notes, read the material from the textbook, and try to solve (at least some of) the exercises listed. Check all the programming assignments and homeworks you had. Make sure you understand how computer architecture and OS interacts. In general, I will be testing your ability to leverage this material and your understanding of key concepts.

This is what I won't ask: I will not require you to recall the syntax of system calls. I will not require you to recall the exact implementation of algorithms from the textbook. I will not require you to remember precise memory management details from various microprocessors/OS. I will be very flexible in reading your code, i.e., you don't need to put exact package names and/or remember special Java APIs beyond basic Java.

Good luck and have a good summer!

Professor Moritz