Retrieval Exercises
for
CS 3733 Operating Systems

Instructor: Dr. Turgay Korkmaz
Department Computer Science
The University of Texas at San Antonio

Topics: Early systems and OS overview

Skim Chapters 1-2 of SGG

Read Chapter 1 of USP

tk01-os-introduction.pptx

OS INTRODUCTION

Retrieval Exercises: OS Introduction

- What is OS?
- What are the goals of OS?
- What are the two key mechanisms to interact with the kernel, and how do they work?
- Differences between System calls and Library function calls? How they work, what information is stored in executable file etc…
- What are the basic OS Structures/components/tasks?
- What are the differences between batch processing, multiprogramming, time sharing systems? Adv/Disadv?
- What is a device driver (dd)?

Topics: Programs, command-line, storage class and Processes
(*SGG* 3.1-3.2; USP 2, 3)

tk02-program-process.pptx

PROGRAM AND PROCESS

Retrieval Exercises: program and process

- What is the difference between program and process?
- How does a program becomes a process?
- Draw a diagram to show how the state of a process changes?
- List at least four fields in PCB and explain their purposes?
- Using a diagram explain how does context-switching happens.
- Compare/contrast process vs. threads.
- What are the main parts/sections of a program image in the memory and what are the goals of these parts/sections?
- Be able to create/copy/free structures similar to command-line argument list
- What might be the adv/disadv of using static variables?
- Given example function that would not be thread-safe? Explain Why/how?
- What is the main purpose of using static before a variable and before a function in C?
- Explain the purpose of fork, exec, and wait in process creation and termination?
- Be able to draw the graph of process relations and show their output.
- What happens if the parent process quits without waiting for children?
- What does zombie mean and how are they cleared from the system ?

Topics: Process (CPU) Scheduling
(SGG[9ed] 6.1-6.3)
(SGG[older ed] 5.1-5.3)
Topics: CPU Scheduling Examples and Simulator
(SGG[9ed] 6.7 and web notes)

tk03-CPUScheduling.pptx
tk04-examples-simulator.pptx

CPU SCHEDULING

EXAMPLES AND SIMULATOR

Retrieval Exercises: CPU SCHEDULING

- What is the difference between long-term, short-term, and medium-term schedulers?
- When a running process moves out of CPU?
- Draw a state diagram and show how/where/when CPU scheduler is invoked to change the state of a process?
- Compare/contrast preemptive and non-preemptive scheduling and give examples.
- Using a diagram explain how does context-switching happens.
- What is CPU utilization? If CPU was busy for 100 ms during a second, what is CPU util?
- How can we increase CPU utilization?
- Compare/contrast CPU-bound and IO-bound processes? What are the challenges in scheduling such processes?
- What are the key performance criteria for scheduling algorithms? Explain turnaround time.
- Be able to explain the basic ideas/mechanisms behind FIFO, SFJ, PSFJ, RR, PR
- Be able to draw Gantt chart based on the given processes and scheduling algorithm and compute performance metrics (e.g., utilization, waiting time, response time, throughput, turnaround time etc)
- Why do we need multilevel queues and how to use them to provide different guarantees?
- What are the general approaches to evaluate the system performance: analytical (queuing theory) and Simulation…

Topics: Unix File and I/O

(USP Chapters 4 and 5)

(Optional SGG[9ed] Chapters 11, 12.1-5, 13)

tk05-IO-FS-USP-4-5.pptx

UNIX FILE AND I/O

Retrieval Exercises: Unix File and I/O

- What are the criteria for long-term information storage?
  - Large amount, survive the process, concurrent access
- What is a file? How does it make it easy to access information?
- Give/explain some of file attributes?
- How to give rwxr-xr– access to a file in Unix? Who can do what with that access permission?
- What does a partition means, how can we know where each partition is located on the disk?
- Disk Space allocation (contiguous, linked, indexed): how do they work? Adv/disadv?
- Free Disk space management: linked list, bitmap. Adv/disadv? Coding?
- Compare/contrast directory structures: single-, two-level, tree-structured.
- What does absolute and relative path means?
- A directory entry in Unix contains …. and ….
- Give the structure of inode, be able to compute file size, or write code to access data etc.
- What are the key IO system call on Unix?
- If a system call returns -1 and error is EINTER, how should we interpret this and what to do?
- Given a sequence of read - write, be able to generate possible outcomes.
- What is the difference between binary file and text file?

Retrieval Exercises: Unix File and I/O

- Describe File Descriptor Table (FDT) and System File Table (SFT), and explain how they are related? Draw a figure to show the relationship.
- How does **fork()** affect FDT or SFT? Be able to show the status of these tables after fork?
- What does redirection means. Open a file and make it point to standard input.
- What is the relationship between FILE pointers vs. File Descriptors?
- How are fopen, fclose, fread, fwrite, fprintf, fscanf different than open, close, read, write?
- What does buffering in stdio functions mean, how it may affect the program output?
- Directories are like regular files (Y/N)?
- How can we access directory entries?
- Why should we not directly access them using read/write then?
- What does hard link and soft/symbolic link means?
- How are they created/maintained in Unix?
- Show how the inode structure is created/removed when link/unlink is used.

Topics: IPC and Unix Special Files
(USP Chapters 6 and 7)

tk06-IPC-SpecialFiles-USP-6-7.pptx

IPC AND UNIX SPECIAL FILES
PIPE - FIFO

Retrieval Exercises: PIPE - FIFO

- What does IPC (Inter-Process Communication) mean?
- Why do processes need to communicate/cooperate?
- What is a pipe? How can we create one in Unix?
- If we want to write 100 bytes into a pipe, can we do it **atomically**? How about read?
- How about **blocking** behavior of a pipe in case of read or write?
- When does a SIGPIPE signal might be generated?
- Given a piece of code, draw the pipe structure and FDs, or vice versa.
- Pros/Cons of pipes.
- What is a Named Pipe (FIFO)? What advantages it has over regular pipe?
- How does a FIFO works?
- What does client-server communication means? Why we cannot use regular pipe but FIFO?
- How about client-server processes on different machines? Can they communicate via FIFO? Why/why not?
- Create some related processes and connect them via pipes (e.g., ring)

Topics: Memory Management
(SGG, Chapter 08)

tk07-memory-management-SGG-08.pptx

MEMORY MANAGEMENT

Retrieval Exercises: Memory Management

- What is the goal of using base and limit registers?
- Explain address binding at compile, load, and execution time.
- Explain the concept of logical/virtual address and physical address? Can they be the same?
- What is the goal of Memory Management Unit (MMU)? Give an example.
- Compare/contrast Dynamic loading and dynamic linking. Give Adv/disadv.
- What does swapping mean? Why would you disable or enable it?
- Why do we need to allocate contiguous space for a process?
- What are the adv/disadv of First, Best, Worst fit in allocating contig. space
- Explain why/how external and internal fragmentations happen. How can we solve them?
- At the high level describe paging and basic hardware support needed.
- Given a virtual and physical addresses, show how to design a single, two-level paging systems.

Retrieval Exercises: Memory Management

- Discuss adv/disadv of using small/big size pages/frames.
- Explain pros/cons of paging. E.g. How does it enable page sharing?
- What is the goal of translation look-aside buffers (TLBs)?
- How does it work? Draw a diagram to show the basic architecture.
- Be able to compute Effective Access Time (EAT) under a given scenario.
- Explain the key ideas and motivation behind Hierarchical Page tables, Hashed Page Tables, Inverted Page Tables
- Be able to design a paging system with multiple levels

Topics: Virtual Memory
(SGG, Chapter 09)
9.1-9.7

tk08-virtual-memory-management-SGG-09.ppt

CH 9: VIRTUAL MEMORY

Allow the OS to hand out more memory than existing physical memory

Retrieval Exercises for Virtual Memory

- What is virtual memory?
- What are the goals and benefits of virtual memory?
- How to map virtual memory addresses to physical ones?
- What is Demad Paging? How does it work?
- What is the role of “valid-bit” in page table?
- What does page fault means? How does it happens and how is it be handled?
- Be able to compute EAT under given demand paging scenario. What are the dominant delay components?
- What does Copy-on-Write means? Explain benefits.
- Explain how/why page replacement is needed and handled.
- Be able to trace/count number of page faults under FIFO, Opt, LRU, LFU, MFU etc.
- Adv/Disadvantages different approximate implementation of LRU?
- What does locality means, how it impacts program performance in demand paging?
- How do two programs can share memory (consider memory-mapped files)?
- What are the major concerns when allocating frames to different size programs?
- Explain the difference between global and local allocation/replacement.
- What does Thrashing means? What strategies to use for avoiding it?
- …. Optional (memory allocation user/kernel, other issues improvements)

Topics: Threads
(SGG, Chapter 04)
(USP, Chapter 12)

tk09-threads-sgg-ch04-usp-ch12.ppt

CH 4: THREADS

A fundamental unit of CPU utilization

Retrieval exercises from Threads:

- What is a thread?
- What are the differences between thread and process?
- What are the benefits of threads?
- How does a process create/manages several threads?
- What are the differences between user level and kernel level threads?
- What might be the adv/disadv of using user or kernel threads?
- Draw a diagram to show how user threads can be mapped to kernel ones.
- Explain how does Light-Weight Process (LWP) approach work.
- Thread Libraries
  - How to create a thread in C and Java? What does Join do?
  - How to pass parameters to or return values from threads in pthread and c?
  - Give the expected output of a given program using multiple threads.
  - Life-time (state transitions) of Java threads
- Other issues?
  - Should we replicate all active threads upon fork(), why, why not?
  - When do you think we may need thread cancelation and how to deal with it?
  - What are the adv/disadv of thread pools.

Topics: Process (Thread) Synchronization
(SGG, Ch 5 in 9th Ed Ch 6 in Old Ed)
(USP, Chapter 13-14)

tk10-synchronization-java-sgg9th-ch05.ppt

PROCESS SYNCHRONIZATION

Get processes (threads) to work together
in a coordinated manner.

Retrieval exercises from Process Sync.

- What are the problems in concurrently accessing the shared data? Give an example.
- What is a Race Condition, when does it appear?
- What is critical section (CS) and why does it need to be executed atomically? What does atomic means?
- Describe the following requirements for CS solutions: Mutual Exclusion, Progress, Bounded Waiting.
- Compare/contrast preemptive and non-preemptive kernel approaches in solving CS problem?
- Given a CS solution (e.g, Peterson's sol), explain if it satisfies the above requirements!
- What are the common hardware solutions for synchronization? Disable inter, non-inter atomic inst

  GetAndSet, Swap…

  Do they satisfy the above requirements, how, why, why not?

  Generalized solution for synch of n processes
- What is a semaphore? Acquire/**Wait/**P/Down, Release/**Signal**/V/Up… Give example usages.
- How to implement semaphore? Block waiting processes instead of spinlock, why?
- What does deadlock and starvation means, how do they happen?
- What does priority inversion mean and how can it be avoided?
- What is a condition variable? How to use?
- What is a Monitor? How to use?

Topics: Signals

(USP, Chapter 8.1-8.6)

tk11-signals-USP-ch8-ver2.ppt

SIGNALS

Retrieval exercises from Signals

- What is a Signal? Is it a synchronous or asynchronous event?
- Compare/contrast signals and interrupts?
- Use the following keywords in appropriate sentences about signals: generate, deliver, lifetime, pending, catch, handler, ignore, mask, block, unblock, default action.
- How can you generate signals from command line and/or from your program?
- **Show how to block SIGINT and SIGQUIT. - sigprocmask()**
- **Show how to set an action for a signal, check if a signal is blocked or ignored? --sigaction()**
- Wait for signals: what do pause(), sigsuspend(), and sigwait() do?
- Blocking system calls (e.g., read) may return -1 when a signal is caught. Is this a real error how can we deal with it?
- In a signal handler, why should we save errno and then restore it at the end?
- Regarding async-signal safety, we said only certain system calls and library functions may be used safely in a signal handler. What could be the reason for that?
- How to handle signals in a multithreaded environment?

Topics: Communications, Networking,
TCP/IP, Socket Programming
(USP, Chapter 18)

tk12-communications-part-1-networking.ppt
tk12-communications-part-2-socket-prog.ppt

COMMUNICATIONS - NETWORKING

Retrieval exercises from Networking

- What is a protocol (in computer networking)?
- What are the main things that need to be specified in a networking protocol?
- What are the main layers of Internet protocol stack?
- What does encapsulation means?
- What are the advantages/disadvantages of layering design of networking stack?
- Explain the main functions of each layer: application, transport, network, link , physical
- Why do all the nodes in a network have the same common prefix in their 32-bit IP address?
- What are the main difference between TCP and UDP protocols?
- TCP is a “reliable, in-order byte stream” protocol. What do we mean by that?
- What are the two key fields in TCP and UDP packets? And how are they used?
- What is a socket?
- What information is needed to identify a UDP socket and TCP socket?

Retrieval exercises from Networking

- Explain the main components and their interaction in the client-server architecture.
- Can any host serve be a server and/or a client?
- How does client know the server (how does addressing issue is resolved?)
- Be able to comment on the following issues in client-server communications:
  - Blocking versus non-blocking
  - Buffered versus unbuffered
  - Reliable versus unreliable
  - Server architecture: concurrent versus sequential
  - Scalability
- What is socket?
- What are the main system calls and in which order are they called in a client/server application using TCP?
- What are the main system calls and in which order are they called in a client/server application using UDP?
- Be able to write a simple client-server application using TCP (e.g., client sends two integer numbers to the server and server sends back the sum of these numbers). You need to use right system calls in the right order; but, you don't need to remember/use the exact structures used/set as parameters in real programs. So give correct function names with generic parameters (e.g., IP address, port etc.)