# Review and Retrieval Exercises for CS 5523 Operating Systems

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

Grand tour of the major operating systems components

Operating System Structures & Services

ch01-sgg-tk.ppt

ch02-sgg-tk.ppt

# CH1: INTRODUCTION

# CH2: OPERATING-SYSTEM STRUCTURES

# 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 differences between batch processing, multiprogramming, time sharing systems? Adv/Disadv?
- What is a device driver (dd)?
- What are the basic OS Structures/components/tasks/services?
- What are different systems and OS types? (distributed, embedded …)
- What are different OS design approaches? +/-
- Differences between policies and mechanisms?

Programs in execution

ch03-sgg-tk.ppt

# CH3: PROCESSES

# 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?
- 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 ?

A fundamental unit of CPU utilization

ch04-sgg-tk.ppt

# CH 4: THREADS

# 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?
  - 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.

Pick one 'lucky' process from ready queue

ch05-sgg-tk.ppt

# CH 5 IN OLD ED:  CH 6 IN 9TH ED:

# CPU SCHEDULING

# 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 II-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…

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

ch06-sgg-tk.ppt

# CH 6 IN OLD ED:  CH 5 IN 9TH ED:
# PROCESS SYNCHRONIZATION

# 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?

Wait for someone who waits for you!

ch07-sgg-tk.ppt

# CH 7: DEADLOCKS

# Retrieval exercises from Deadlocks

- When/how does a deadlock happen? Give an example.
- What are the four conditions that must hold simultaneously for a deadlock to arise?
- Show how to use a resource-allocation graph to detect a deadlock?
- What are the main approaches/methods to handle deadlocks? Explain and give examples?
  - Prevention, Avoidance, Detection, Recovery, Ignore… programmers responsibility
- How to realize deadlock prevention? What are the adv/disadv?
- How to realize deadlock avoidance? What are the adv/disadv?
  - What does “Safe State” means? Given a set of process, max resource need, and current allocation, can yous how if they satisfy safety condition?
  - How to use resource-allocation graph to check “safe state”?
  - Be able to show how Banker’s Algorithm verifies if a given state is safe or not.
- How to realize deadlock detection?
- How to recover from deadlock?
  - Process Termination
  - Resource Preemption

Share the main memory among many processes

ch08-sgg-tk.ppt

# 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 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

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

ch09-sgg-tk.ppt

# CH 9: VIRTUAL 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 FIFI, 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)