CSC 4320/6320: Operating Systems

Georgia State University

Chapter 01: Introduction (Part 3)

Spring 2025

Instructor: Dr. Md Mahfuzur Rahman
Department of Computer Science, GSU

Review: Multiprogramming

Georgia State University

- Also called batch system
- Single user cannot always keep CPU and I/O devices busy
- Multiprogramming organizes jobs (code and data) so CPU always has one to execute
- A subset of total jobs in system is kept in memory
- One job selected and run via **job scheduling**
- When job has to wait (for I/O for example), OS switches to another job

Multiprogramming is a technique that increases CPU utilization by organizing jobs (code and data) so that the CPU always has a job to execute.

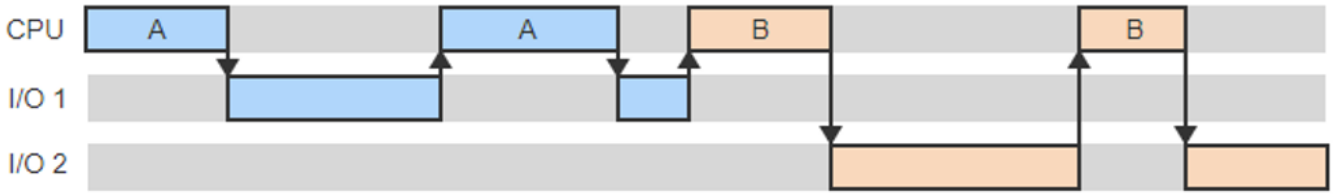
Review: Multitasking (Timesharing)

- The concurrent performance of multiple jobs.
- A logical extension of Batch systems– the CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨ **process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Review: Multiprogramming vs. Multitasking

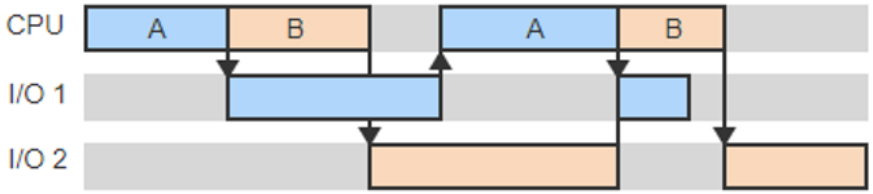
Georgia State University

Sequential execution



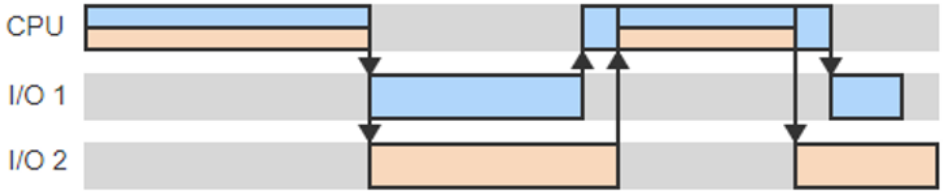
- Process A finishes, then process B starts

Multiprogramming



- Under multiprogramming, both A and B are active. When A needs I/O operation, B uses the CPU rather than making it idle.

Time-sharing



- When one processor is waiting for I/O, the other can use CPU at 100%. When both processes are running, the CPU is time-shared at 50% each

Let's think …

- What is the main benefit of multiprogramming over sequential processing?

A. Efficient utilization of CPU
B. Reduced computation time.
C. Both ✓

- Due to the overlapping of the CPU and I/O-bound computations, the total time to compute both A and B is significantly reduced.

Let's think more …

- What makes a time-sharing system able to run multiple processes concurrently (from the user perspective)?

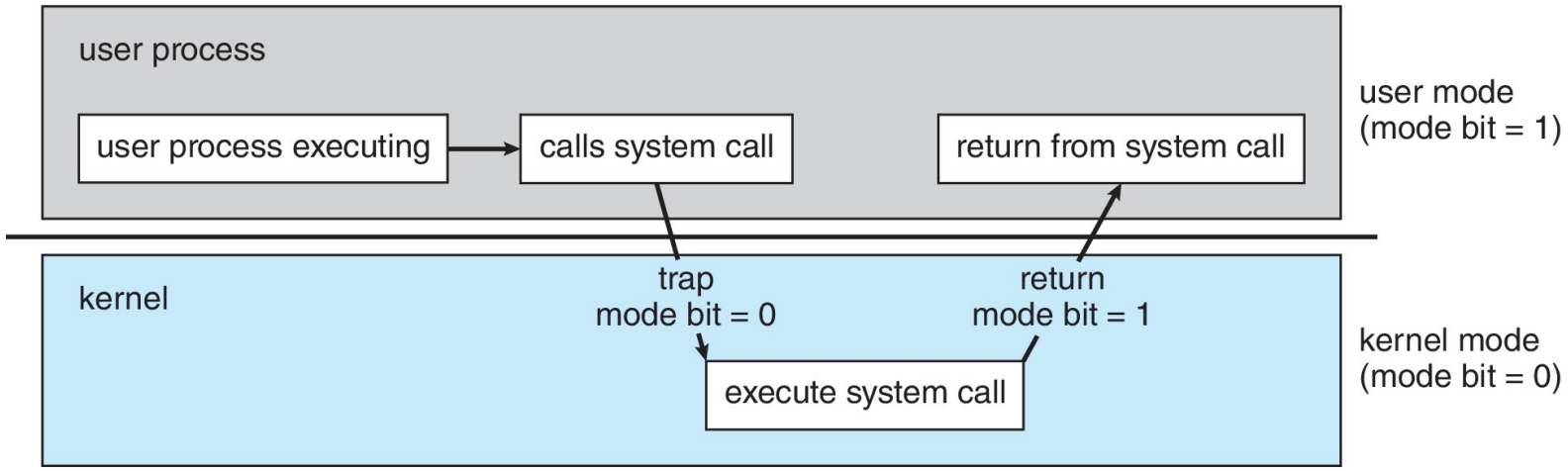
Ans: Time interval during which a process runs is much shorter than the required computing time. So, because of the frequent switching, the processes appear to be running smoothly (though at lesser speed)

Dual-mode Operation

Georgia State University

- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
- **Mode bit** provided by hardware
  - Provides ability to distinguish when system is running user code or kernel code.
  - When a user is running ⇨ mode bit is “user”
  - When kernel code is executing ⇨ mode bit is “kernel”
- How do we guarantee that user does not explicitly set the mode bit to “kernel”?
  - System call changes mode to kernel, return from call resets it to user
- Some instructions designated as **privileged**, only executable in kernel mode

Transition from User to Kernel Mode



User Mode/Kernel Mode – Consider it as a public area/private area of a facility

Let's recall

What is another term for kernel mode?

A) supervisor mode
B) system mode
C) privileged mode
D) All of the above ✓

Timer

- Imagine a user program to get stuck in an infinite loop or to fail to call system services and never return control to the OS

- Timer to prevent infinite loop (or process hogging resources)
  - Timer is set to interrupt the computer after some time period
  - Keep a counter that is decremented by the physical clock
  - Operating system set the counter (privileged instruction)
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time

Questions

1. A ____ can be used to prevent a user program from never returning control to the operating system.

A) portal
B) program counter
C) firewall
D) timer ✓

2. What statement concerning privileged instructions is considered false?

A) They may cause harm to the system.
B) They can only be executed in kernel mode.
C) They cannot be attempted from user mode. ✓
D) They are used to manage interrupts.

Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity;*** process is an ***active entity***.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically, system has many processes, (some user processes, some operating system processes) running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads

Process Management Activities

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

Memory Management

- To execute a program all (or part) of the instructions must be in memory
- All (or part) of the data that is needed by the program must be in memory
- Memory management determines what is in memory and when
  - Optimizing CPU utilization and computer response to users
- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

File-system Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each (storage) medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and directories
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Mass-Storage Management

- Usually, disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Mounting and unmounting (making it available/unavailable)
  - Free-space management
  - Storage allocation
  - Disk scheduling (Disks often receive multiple I/O requests simultaneously.)
  - Partitioning
  - Protection

Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, use the data and put a copy in the cache
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Characteristics of Various Types of Storage

Georgia State University

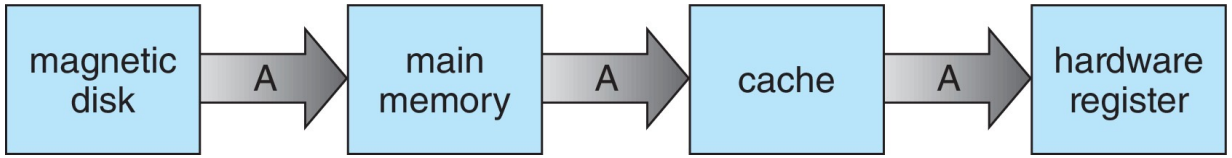
Level	1	2	3	4	5
Name	registers	cache	main memory	solid-state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25-0.5	0.5-25	80-250	25,000-50,000	5,000,000
Bandwidth (MB/sec)	20,000-100,000	5,000-10,000	1,000-5,000	500	20-150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Movement between levels of storage hierarchy can be explicit or implicit

Migration of data “A” from Disk to Register

Georgia State University

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy



- Multiprocessor environment must provide **cache coherency** in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist
  - Various solutions covered in Chapter 19

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices (only the device driver knows the peculiarities of the specific device)

Questions

1. Two important design issues for cache memory are ____.

A) speed and volatility
B) size and replacement policy ✓
C) power consumption and reusability
D) size and access privileges

2. A(n) _____ is the unit of work in a system.

A) process ✓
B) operating system
C) timer
D) mode bit

3. Program counter specifies the current instruction to execute.

True
False ✓

Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

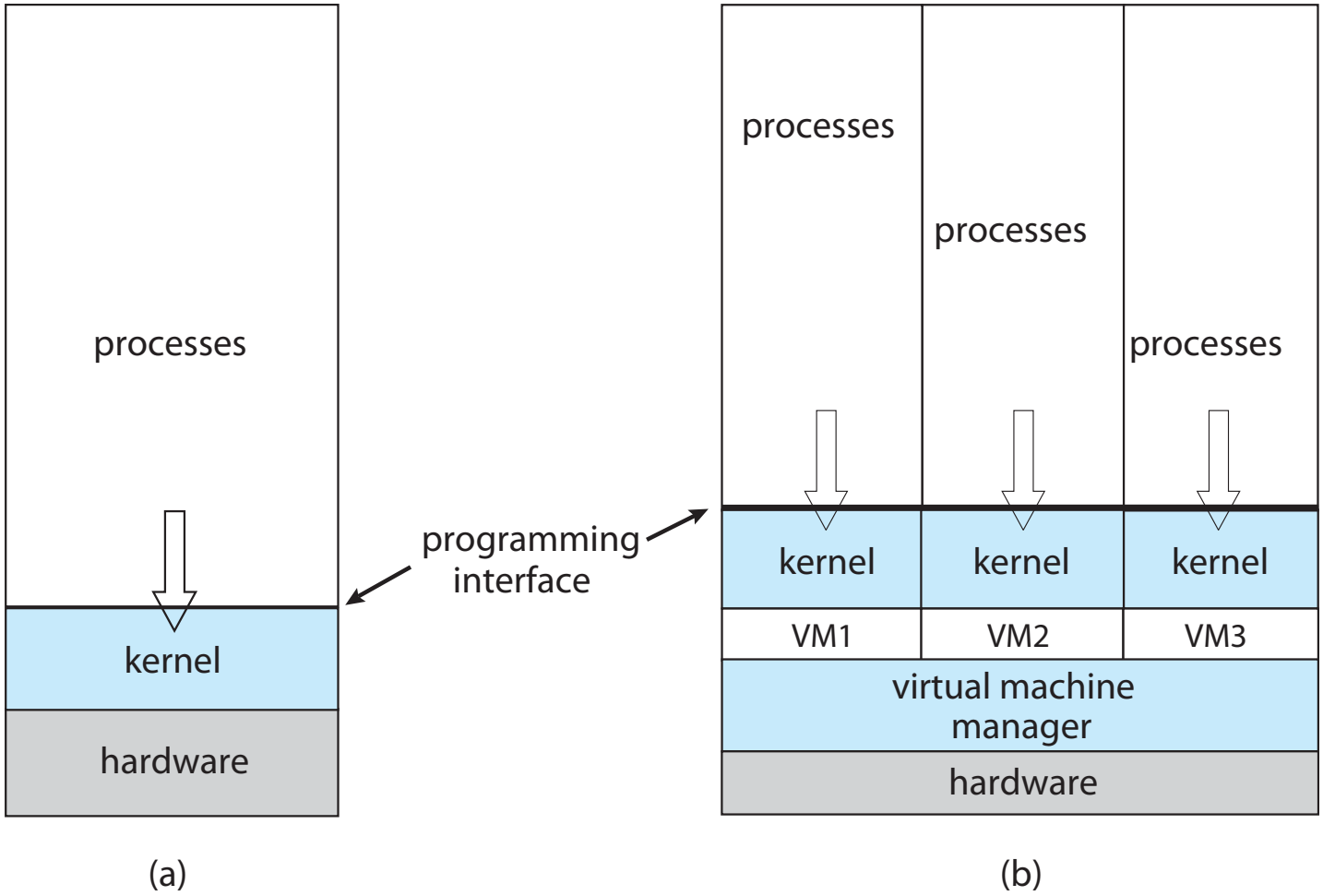
Virtualization

- **Virtualization** abstracts the hardware of a single computer into several different execution environments
- Allows operating systems to run as applications within other OSes
  - Vast and growing industry
- **Emulation** used when source CPU type different from target type (i.e. PowerPC to Intel x86)
  - Generally slowest method
  - When computer language not compiled to native CPU code – **Interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OSes also natively compiled
  - Consider VMware running WinXP guests, each running applications, all on native WinXP **host** OS
  - **VMM** (virtual machine Manager) provides virtualization services

Virtualization (cont.)

- Use cases involve laptops and desktops running multiple OSes for exploration or compatibility
  - Apple laptop running Mac OS X host, Windows as a guest
  - Developing apps for multiple OSes without having multiple systems
  - Quality assurance testing applications without having multiple systems
  - Executing and managing compute environments within data centers

Computing Environments - Virtualization



A computer running (a) a single operating system and (b) three virtual machines.

Questions

1. Virtualization software is one member of a class that differs from emulation.
   True ✓
   False

2. VMware application was a virtual machine manager (VMM).
   Yes ✓
   No

Some Questions

Security is equivalent to protection with no difference.

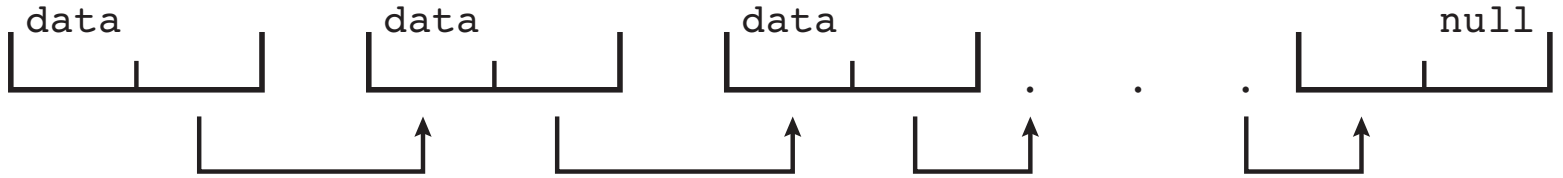
A. True
B. False ✓

A virtual machine can be switched like a process in the operating system.

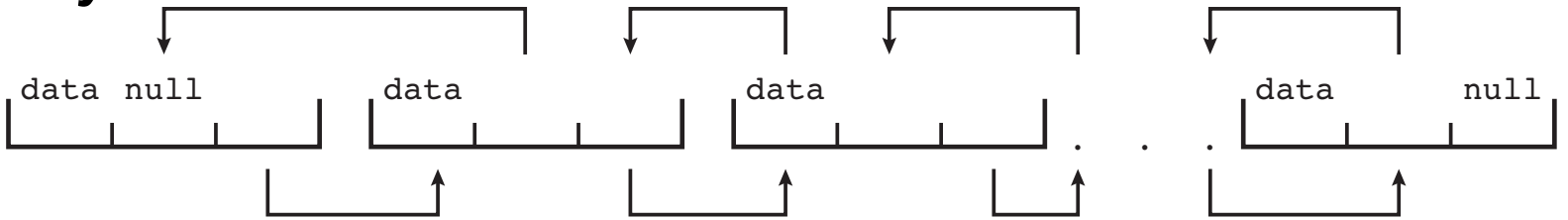
Yes ✓
No

Kernel Data Structures

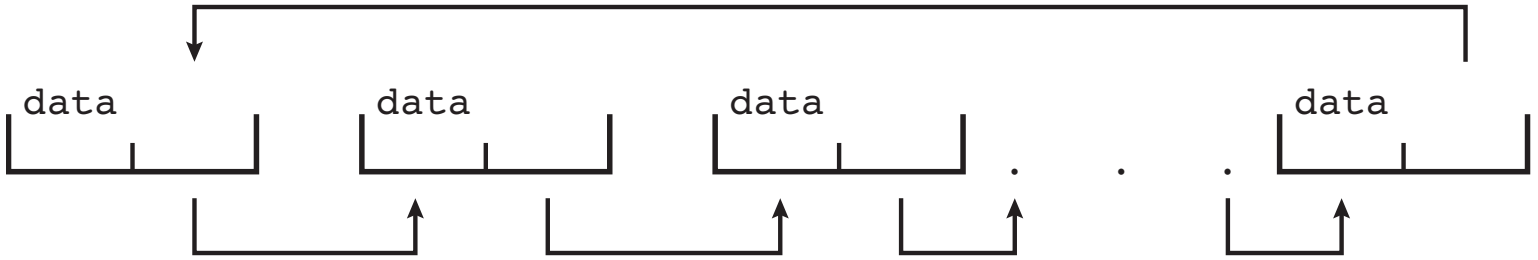
- Many similar to standard programming data structures, e.g. array, list, stack, queue, tree, etc.
- ***Singly linked list***



- ***Doubly linked list***

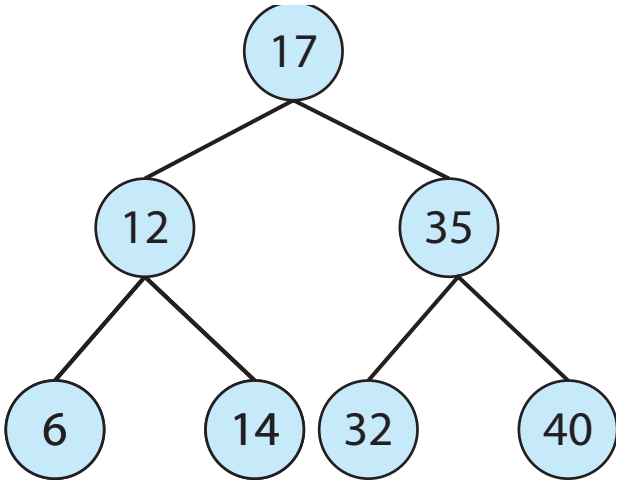


- ***Circular linked list***



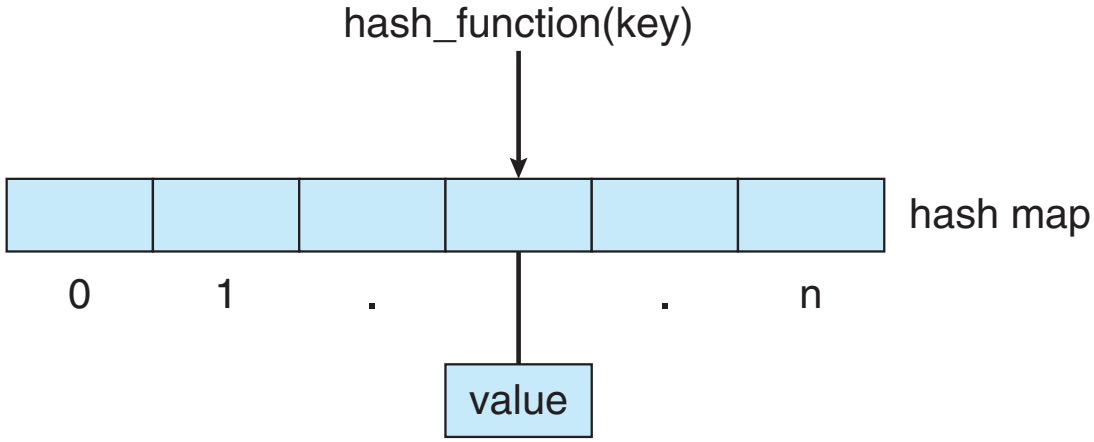
Kernel Data Structures

- **Binary search tree**
  left <= right
  - Search performance is *O(n) (worst-case, think how)*
  - **Balanced binary search tree** is *O(lg n)*



Kernel Data Structures

- **Hash function** can create a **hash map**



Example: username: password mapping (key: value)

e.g. hash_value = key % 10

Hash collision refers to the phenomenon of hash function in which two unique inputs result in the same output value.

- **Bitmap** – string of *n* binary digits representing the status of *n* items
- Linux data structures defined in ***include*** files `<linux/list.h>`, `<linux/kfifo.h>`, `<linux/rbtree.h>`

Georgia State University

Questions?