School of Computing and Information Technologies

Operating System Theory and Design

Assignment

To be attempted in Groups of 5-6 students

GROUP MEMBERS

NAME	ADM NO
GLORIA CHEROTICH	SCII/00828/2019
ISAAC WAMBIRI	SCII/00817/2019
MESHACK MAKIRA	SCII/00815/2019
MICHAEL ORINA	SCII/00825/2019
VINCENT KEMBOI	SCII/00832/2019

a) **Question 1**

a. Discuss process scheduling in the following operating systems

i. Unix operating system

A scheduler is the mode that divides the finite resource of processor time between the runnable processes on a system.

This is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

SCHEDULING WITH CRON

Cron, an automated scheduler in Linux executes jobs which are scheduled by the system , root or individual users. Information on different schedules in contained within a crontab file which is unique to each user.

Unix uses multilevel feedback queues. All runnable processes are assigned a scheduling priority that determines which queue they are placed in.

Each queue uses a round robin mechanism which is a variant of the FIFO algorithm . priorities are dynamically adjusted. When a nonrunning process with a higher priority becomes available it immediately preempts the running process if the current process is in user mode otherwise it will switch when current process exits.

ii. Windows XP operating systems

Windows XP uses a priority-based preemptive scheduling algorithm.

The dispatcher uses a 32-level priority scheme to determine the order of thread execution, divided into two classes - variable class from 1 to 15 and real-time class from 16 to 31, (plus a thread at priority 0 managing memory.)

There is also a special idle thread that is scheduled when no other threads are ready.

Win XP identifies 7 priority classes (rows on the table below), and 6 relative priorities within each class (columns.)

Processes are also each given a base priority within their priority class. When variable class processes consume their entire time quanta, then their priority gets lowered, but not below their base priority.

b) Question 2

a. Discuss the Bankers algorithms as used to handling deadlocks in operating system

The bankers algorithm is a resource allocation and deadlock avoidance algorithm that simulates the allocation for predetermined amounts of all resources then makes an “s-state” check to test for possible activities before making a decision whether allocation should be allowed to continue. When a new process is created in a computer system, the process must provide all types of information to the operating system like upcoming processes, requests for their resources, counting them, and delays. Based on these criteria, the operating system decides which process sequence should be executed or waited so that no deadlock occurs in a system. Therefore, it is also known as **deadlock avoidance algorithm** or **deadlock detection** in the operating system. **Available**: It is an array of length 'm' that defines each type of resource available in the system. When Available[j] = K, means that 'K' instances of Resources type R[j] are available in the system.

1. **Max:** It is a [n x m] matrix that indicates each process P[i] can store the maximum number of resources R[j] (each type) in a system.
2. **Allocation:** It is a matrix of m x n orders that indicates the type of resources currently allocated to each process in the system. When Allocation [i, j] = K, it means that process P[i] is currently allocated K instances of Resources type R[j] in the system.
3. **Need:** It is an M x N matrix sequence representing the number of remaining resources for each process. When the Need[i] [j] = k, then process P[i] may require K more

instances of resources type Rj to complete the assigned work. Nedd[i][j] = Max[i][j] - Allocation[i][j].

4. **Finish**: It is the vector of the order **m**. It includes a Boolean value (true/false) indicating whether the process has been allocated to the requested resources, and all resources have been released after finishing its task.

b. Discuss the strategies used to handle/manage deadlocks in the following operating systems

i. Windows operating system

Deadlock avoidance can be done with Banker's Algorithm.

Banker's Algorithm

Bankers's Algorithm is resource allocation and deadlock avoidance algorithm which test all the request made by processes for resources, it checks for the safe state, if after granting request system remains in the safe state it allows the request and if there is no safe state it doesn't allow the request made by the process.

Inputs to Banker's Algorithm:

1. Max need of resources by each process.
2. Currently, allocated resources by each process.
3. Max free available resources in the system.

The request will only be granted under the below condition:

1. If the request made by the process is less than equal to max need to that process.
2. If the request made by the process is less than equal to the freely available resource in the system.

ii. Linux Operating System

Mutual Exclusion

Shared resources such as read-only files do not lead to deadlocks.

Unfortunately some resources, such as printers and tape drives, require exclusive access by a single process.

Hold and Wait

To prevent this condition processes must be prevented from holding one or more resources while simultaneously waiting for one or more others. There are several possibilities for this:

Require that all processes request all resources at one time. This can be wasteful of system resources if a process needs one resource early in its execution and doesn't need some other resource until much later.

Require that processes holding resources must release them before requesting new resources, and then re-acquire the released resources along with the new ones in a single new request. This can be a problem if a process has partially completed an operation using a resource and then fails to get it re-allocated after releasing it.

Either of the methods described above can lead to starvation if a process requires one or more popular resources.

No Preemption

Preemption of process resource allocations can prevent this condition of deadlocks, when it is possible.

One approach is that if a process is forced to wait when requesting a new resource, then all other resources previously held by this process are implicitly released, (preempted), forcing this process to re-acquire the old resources along with the new resources in a single request, similar to the previous discussion.

Another approach is that when a resource is requested and not available, then the system looks to see what other processes currently have those resources and are themselves blocked waiting for some other resource. If such a process is found, then some of their resources may get preempted and added to the list of resources for which the process is waiting.

Either of these approaches may be applicable for resources whose states are easily saved and restored, such as registers and memory, but are generally not applicable to other devices such as printers and tape drives.

Circular Wait

One way to avoid circular wait is to number all resources, and to require that processes request resources only in strictly increasing (or decreasing) order.

In other words, in order to request resource Rj, a process must first release all Ri such that i >= j.

One big challenge in this scheme is determining the relative ordering of the different resources

c. How does deadlock avoidance work in Windows 10?

in deadlock avoidance, the request for any resource will be granted if the resulting state of the system doesn't cause deadlock in the system. The system continuously checks for safe and unsafe states.

Windows uses the bankers algorithm for deadlock avoidance.

c) Question 3

a. Describe disc (disk-arm) scheduling algorithms

First Come First Serve (FCFS)

In this algorithm, the requests are served in the order they come. Those who come first are served first. This is the simplest algorithm.

Eg. Suppose the order of requests are 70, 140, 50, 125, 30, 25, 160 and the initial position of the Read-Write head is 60.

Seek Time = Distance Moved by the disk arm = (140-70)+(140-50)+(125-50)+(125-30)+(30-25)+(160-25)=480

Shortest Seek Time First (SSTF)

In this algorithm, the shortest seek time is checked from the current position and those requests which have the shortest seek time is served first. In simple words, the closest request from the disk arm is served first.

Scan

In this algorithm, the disk arm moves in a particular direction till the end and serves all the requests in its path, then it returns to the opposite direction and moves till the last request is found in that direction and serves all of them.

Look

In this algorithm, the disk arm moves in a particular direction till the last request is found in that direction and serves all of them found in the path, and then reverses its direction and serves the requests found in the path again up to the last request found. The only difference between SCAN and LOOK is, it doesn't go to the end it only moves up to which the request is found.

C-SCAN

This algorithm is the same as the SCAN algorithm. The only difference between SCAN and C-SCAN is, it moves in a particular direction till the last and serves the requests in its path. Then, it returns in the opposite direction till the end and doesn't serve the request while returning. Then, again reverses the direction and serves the requests found in the path. It moves circularly.

C-LOOK

This algorithm is also the same as the LOOK algorithm. The only difference between LOOK and C-LOOK is, it moves in a particular direction till the last request is found and serves the requests in its path. Then, it returns in the opposite direction till the last request is found in that direction and doesn't serve the request while returning. Then, again reverses the direction and serves the requests found in the path. It also moves circularly.

d) ***A disk has the following cylinder requests: 2, 15, 30,9,16 and 10 in that order. Given that the disk arm is at position 14, describe the head movement using the following scheduling algorithms.***

i. **FCFS.**

=(14-2) + (15-2)+(30-15)+(30-9)+(16-9)+(16-10)

ii. **Shortest Seek Time First.(SSTF)**

=1+1+14+20+1+7=43

iii. **SCAN**

=4+1+7+2+13+1+14=42

iv. **LOOK**

=4+1+7+2+13+1+14=40

e) **Question 4**

The table below shows the arrival time and CPU burst of processes P1, P2, P3 and P4

Processes	*Arrival Time*	*CPU Burst*
P1	*0*	*10*
P2	*1*	*7*
P3	*4*	*5*
P4	*6*	*1*

With aid of a GANTT chart, determine the response time, waiting time and turn-around time of each process using shortest remaining time next and Round Robin algorithms

GANNT CHART

0	***2***	***4***	***6***	***7***	***9***	***10***	***12***	***14***	***15***	***16***	***18***	***20***	***22***
P1	P2	P3	P4	P3	P3	P2	P4	P4	P4	P1	P1	P1	P1

Question 5

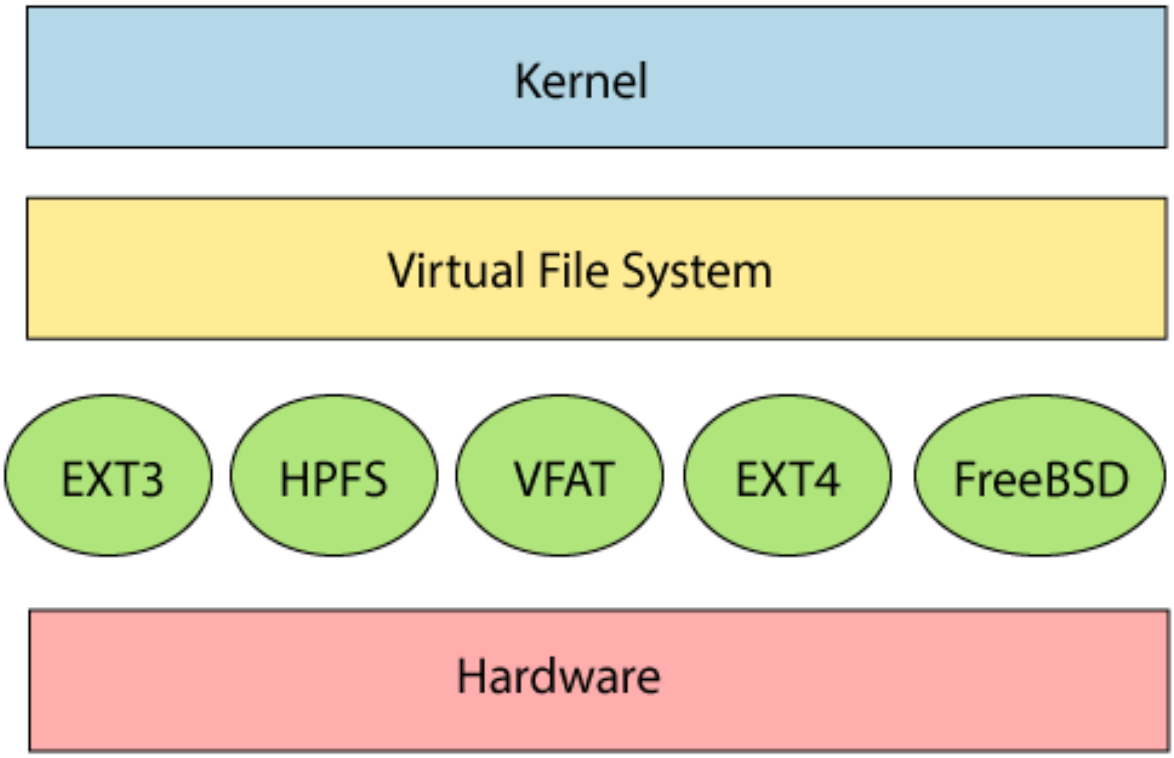
a) Describe the functions of a filesystem

- Control the data access mechanism (i.e how data stored and retrived)
- Manages the metadata about the Files / Folders (i.e. created date, size etc)
- Grants the access permission and manage the securities
- Efficiently manage the storage space
- File system manages user data that is how data is read, write and accessed.
- FS is the structure behind how our system stores and organizes data.
- FS stores all the information bout the file (metadata) like file name, the length of the contents of a file, and the location of the file in the folder hierarchy—separate from the contents of the file.
- The main role of FS is to make sure that what/whoever is accessing the data and whatever action is taken the structure remains consistent.

b) Discuss the structure of Linux filesystem

Linux file system has a hierarchal file structure as it contains a root directory and its subdirectories. All other directories can be accessed from the root directory. A partition usually has only one file system, but it may have more than one file system.

Linux file system contains two-part file system software implementation architecture.


Kernel
Virtual File System
EXT3
HPFS
VFAT
EXT4
FreeBSD
Hardware