FINAL TERM PREPARATION

PAST PAPERS 2022 SOLUTION

OPERATING SYSTEM CS604

SOLVED AND VERIFIED BY MASTERS

FOR ANY QUERIES FEEL FREE TO ASK

rimshazaibabbasi@gmail.com

Masters

CS604 Final Term Preparation:

Solved and verified by Masters

Subscribe our YouTube channel for more solved papers

Paper#1

22-9-2022

Q#1: How to handle be lady's anomaly in FIFO (3 marks)

Belady's Anomaly can be minimized **using a "stack-based algorithm"**. Stack-based algorithms make a priority list for the most used pages and fetch them easily. Least Recently Used (LRU) algorithm. Optimal Page Replacement Algorithm. 28-Aug-2022

https://www.scaler.com › topics › beladys-anomaly

What is Belady's Anomaly? - Scaler Topics

From handouts: page 205

Belady's anomaly

Stack Replacement Algorithms

These are a class of page replacement algorithms with the following property:

Set of pages in the main memory with n frames is a subset of the set of pages in memory with n+1 frames.

These algorithms do not suffer from Belady's Anomaly. An example is the LRU algorithm.

Consider the following example which shows that LRU does not suffer from Belady's anomaly for the given reference string.

- Number of frames allocated = 3
- Reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
- Number of page faults = 10

1	1	1	4	4	4	5	3	3	3

205 / 249 158.09%

Q#2: Types of mount (3 marks)

Mounting is a process by which a computer's operating system makes files and directories on a storage device (such as hard drive, CD-ROM, or network share) available for users to access via the computer's file system.

Mounting makes file systems, files, directories, devices, and special files available for use at a particular location. **Mount point** is the actual location from which the file system is mounted and accessed. You can mount a file or directory if you have access to the file or directory being mounted and write permission for the mount point
There are types of mounts:

- Remote mount
- Local mount

Remote mounts are done on a remote system on which data is transmitted over a telecommunication line. **Local mounts** are mounts done on your local system.

Q#3: What is P & V in semaphore (3 marks) page 111

Semaphores
Hardware solutions to synchronization problems are not easy to generalize to more complex problems. To overcome this difficulty we can use a synchronization tool called a semaphore. A **semaphore** S is an integer variable that, apart from initialization is accessible only through two standard atomic operations: wait and signal. These operations were originally termed P (for wait) and V (for signal). The classical definitions of wait and signal are:

Q#4: If safe sequence is not found, what does it mean? (3 marks)

If a safe sequence does not exist, then **the system is in an unsafe state, which MAY lead to deadlock**. (All safe states are deadlock free, but not all unsafe states lead to deadlocks.)

Q#5: Deadlock Prevention algorithm and safe state relation (3 marks)

Deadlock prevention: is a set of methods for ensuring that at least one of the necessary conditions cannot hold. These methods prevent deadlocks by constraining how processes can request for resources.

If a system is in a safe state, there can be no deadlocks. An unsafe state is not a deadlocked state; a deadlocked state is conversely an unsafe state. Not all unsafe states are deadlocks, however an unsafe state may lead to a deadlock state. Deadlock avoidance makes sure that a system never enters an unsafe state. The following diagram shows the relationship between safe, unsafe, and deadlock states.

Q#6: UNIX/LINUX Functional situations were given. We had to write system calls against them. (5 marks)

System Call	Description
access()	This checks if a calling process has access to the required file
chdir()	The chdir command changes the current directory of the system
chmod()	The mode of a file can be changed using this command
chown()	This changes the ownership of a particular file
kill()	This system call sends kill signal to one or more processes
link()	A new file name is linked to an existing file using link system call.
open()	This opens a file for the reading or writing process
pause()	The pause call suspends a file until a particular signal occurs.
stime()	This system call sets the correct time.
times()	Gets the parent and child process times
alarm()	The alarm system call sets the alarm clock of a process
fork()	A new process is created using this command
chroot()	This changes the root directory of a file.
exit()	The exit system call is used to exit a process.

Q#7:Write 2 conditions when to invoke deadlock detection (5 marks)

Detection Algorithm Usage

When should we invoke the deadlock detection algorithm? The answer depends n two factors:

1. How often is a deadlock likely to occur?
2. How many processes will be affected by deadlock when it happens?

Hence the options are:

- Every time a request for allocation cannot be granted immediately—expensive but process causing the deadlock is identified, along with processes involved in deadlock
- Periodically, or based on CPU utilization
- Arbitrarily—there may be many cycles in the resource graph and we would not be able to tell which of the many deadlocked processes "caused" the deadlock.

Q#8: If no free frames are there. A page is replaced to make space for desired page. Write steps of page replacement. (5 marks)

1. Find the location of the desired page on the disk
2. Find a free frame
    a) If there is a free frame use it.
    b) If there is no free frame, use a page replacement algorithm to select a victim frame.
3. Read the desired page into the newly freed frame; change the page and frame tables.
4. Restart the user process.

Q#9: A mailbox is used by sender S and receiver R for communication of message with their names. Write properties of communication channel/links. (5 marks) midterm page 47

Paper#2

19-9-2022

Mcqs Waqar ki file say

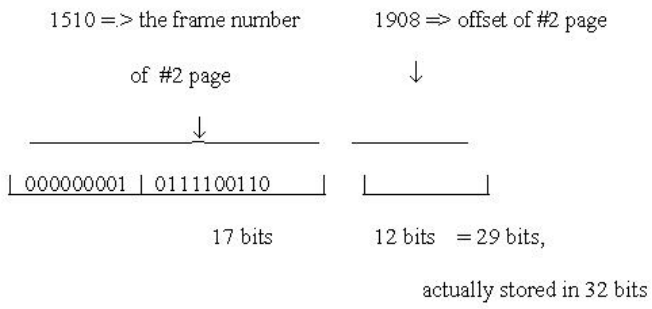
IPC kay tool five (5 marks) midterm page 48

Find the offset (5 marks)

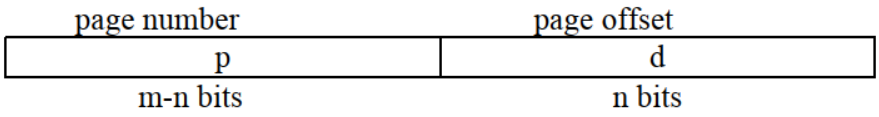
2. offset = A mod page_size

- this is the distance from the beginning of the page
- e.g. address in the process, A = 10,000
- page size = 4k
- page offset = 10000 mod 4k = 10,000 mod 4096 = 1908
- this calculation is done quickly on the computer since the page size is power of 2, e.g., 4k = 2^12
- to determine the page offset, mask out all but the rightmost 12 bits

first convert the number to base 2



From handouts



Example:

Assume a **logical address space** of 16 pages of 1024 words, each mapped into a physical memory of 32 frames. Here is how you calculate the various parameters related to paging.

No. of bits needed for **p** = ceiling [$\log_2$ 16] bits = 4 bits
No. of bits neede for **f** = ceiling [$\log_2$ 32] bits = 5 bits
No. of bits needed for **d** = ceiling [$\log_2$ 2048] bits = 11 bits
Logical address size = |p| + |d| = 4+11 bits = 15 bits
Physical address size = |f| + |d| = 5+11 bits = 16 bits

Find best fit and worst fit data given tha (5 marks)

Best-fit, first-fit, or worst-fit algorithms are the strategies used to select a hole from the set of available holes. Neither first fit, nor best fit is clearly best in terms of both time and storage utilization, but first fit is generally faster.

These algorithms suffer from the problem of external fragmentation. As files are allocated or deleted, the free disk is broken into little pieces. This situation results in **external fragmentation** of disk (similar to external fragmentation of main memory due to segmentation). Disk defragmenter utility needs to be used for removing external fragmentation.

Best-Fit Allocation is a memory allocation technique used in operating systems to allocate memory to a process. In Best-Fit, the operating system searches through the list of free blocks of memory to find the block that is closest in size to the memory request from the process. Once a suitable block is found, the operating system splits the block into two parts: the portion that will be allocated to the process, and the remaining free block.

Advantages of Best-Fit Allocation include improved memory utilization, as it allocates the smallest block of memory that is sufficient to accommodate the memory request from the process. Additionally, Best-Fit can also help to reduce memory fragmentation, as it tends to allocate smaller blocks of memory that are less likely to become fragmented.

Worst-Fit Memory Allocation :

In this allocation technique, the process traverses the whole memory and always search for the largest hole/partition, and then the process is placed in that hole/partition. It is a slow process because it has to traverse the entire memory to search the largest hole.

Semaphore ka tha spinlock ki definition or update karna tha given numeric ko (5 marks)

The main disadvantage of the semaphore discussed in the previous section is that it requires **busy waiting**. While a process is in its critical section, any other process that tries to enter its critical section must loop continuously in the entry code. This continual looping is clearly a problem in a real multiprogramming system, where a single CPU is shared among many processes. Busy waiting wastes CPU cycles that some other process may be able to use productively. This type of semaphore is also called a **spinlock** (because the process spins while waiting for the lock). Spinlocks are useful in multiprocessor systems. The advantage of a spinlock is that no context switch is required when a process must wait on a lock, and a context switch may take considerable time. This is, spinlocks are useful when they are expected to be held for short times. The

Find next fit (3 marks)

Algorithm for memory allocation using Next Fit

1. Enter the number of memory blocks.
2. Enter the size of each memory block.
3. Enter the number of processes with their sizes.
4. Start by selecting each process to check if it can be assigned to the current memory block.

Statement di thi us may tha y name kia hay es problem ka (3 marks)

Harddisk ko improve karney kay bare main tha (3 marks)

The following tips can help in boosting the speed of your hard drive.

1. Scan and clean your hard disk regularly.
2. Defragment your hard disk from time to time.
3. Reinstall your Windows Operating System after every few months.
4. Disable the hibernation feature.
5. Convert your hard drives to NTFS from FAT32.

Part 2. How to Increase Hard Drive Speed?

1. Solution 1: Delete Temporary Files.
2. Solution 2: Scan Hard Drive.
3. Solution 3: Defrag Your Hard Drive.
4. Solution 4: Enable Write Caching.
5. Solution 5: Partition Your Hard Drive.
6. Solution 6: Upgrade.

Paper#3

19-9-2022

Mcqs mostly from Junaid file

Types of storage un ko sort kr kay likhna tha kay jho storage most frequently used hoti hai at the top. (5 marks)

Primary storage devices

- Hard disk. ...
- Magnetic disk. ...
- Pen drive. ...
- SSD. ...
- Sd card. A Contactless Smart Card is what it's called. ...
- Memory card. It's commonly found in digital cameras, printers, gaming consoles, and other electronic devices. ...
- CD. Compact Disc is the name for it. ...
- DVD. Digital Versatile Disc is the name given to it.

The most common type of storage device is **optical**. It can be defined as any storage method that uses a laser to store and retrieve data from optical media.

Os multiprogramming single processor kay sath kesa karta hai (5 marks) midterm

Aik pre paging ka question tha

Pre-paging

An obvious property of a pure demand paging system is the large number of page faults that occur when a process is started. This situation is the result of trying to get the initial locality into memory. Pre-paging is an attempt to prevent this high level of initial paging. The strategy is to bring into memory at one time all the pages that will be needed.

Pre-paging may be an advantage in some cases. The question is simply whether the cost of using pre-paging is less than the cost of the servicing the corresponding page faults.

semaphores ka question tha aik uss ko do actions ka yani wait and signal ka syntax likhna tha.

operations were originally termed P (for wait) and V (for signal). The classical definition of wait and signal are:

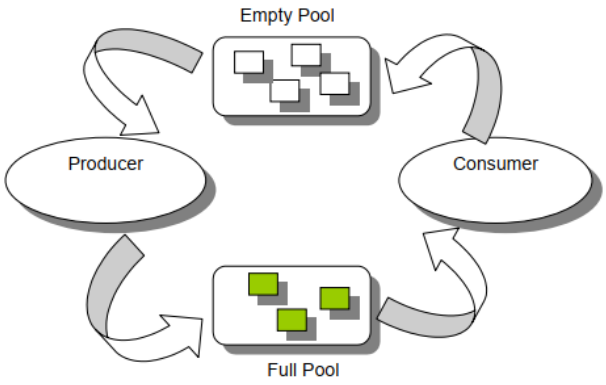
```
wait(S) {
  while(S<=0)
  ;// no op
  S--;
}
```

```
signal(S) {
  S++;
}
```

Bounded buffer may say tha

Bounded Buffer Problem

The bounded-buffer problem, which was introduced in a previous lecture, is commonly used to illustrate the power of synchronization primitives. The solution presented in this section assumes that the pool consists of n buffers, each capable of holding one item.



producer form and consumer form ka code given tha. Mistakes bataani thi.

The code for the producer is as follows:

```
do {
    ...
    produce an item in nextp
    ...
    wait(empty);
    wait(mutex);
    ...
    add nextp to buffer
    ...
    signal(mutex);
    signal(full);
} while(1);
```

Consumer code:

```
do {
    wait(full);
    wait(mutex);
    ...
    remove an item from
    buffer to nextc
    ...
    signal(mutex);
    signal(empty);
    ...
    consume the item in nextc
    ...
} while(1);
```

If the page-fault rate is very high. This leads to low CPU utilization. The operating system thinks that it needs to increase the degree of multiprogramming, because it monitors CPU utilization. Whether it would be good to add another process or not.. give reason in either case. (5 marks)

- Low CPU Utilization.
- The operating system will think that it needs to increase the degree of multiprogramming (bring back swapped processes into memory)
- This will make the problem even worse!!!

If a process does not have "enough" pages, the page-fault rate is very high. This leads to low CPU utilization. The operating system thinks that it needs to increase the degree of multiprogramming, because it monitors CPU utilization and find it to be decreasing due to page faults. Thus another process is added to the system and hence thrashing occurs and causes throughput to plunge.

A process is **thrashing** if it is spending more time paging (i.e., swapping pages in and out) than executing. Thrashing results in severe performance problems:

- Low CPU utilization
- High disk utilization
- Low utilization of other I/O devices

Paper#4

18-9-2022 CS 604

18.09.2022 (8:00 AM)

MCQ Waqar ke b or junaid ke b dunun ma say thay mostly, 3,4 in say alg say ay Subjective mix up.

Aik semaphore ka tha,

1. Read (pfdf [1], buf -5) is trha ke equation thee batana thaa kay properly execute ho ge ya ne.

```
#include <unistd.h>
ssize_t read(int fd, void *buf, size_t count);
```

```
#include <unistd.h>
ssize_t write(int fd, const void *buf, size_t count);
```

Sender receiver wala question tha aik midterm

1. Safety algorithm say type tha aik.

Safety Algorithm

The algorithm for finding out whether or not a system is in a safe state can be described as follows:

1. Let Work and Finish be vectors of length m and n, respectively. Initialize
   Work = Available and Finish[i] = false fori = 1, 2, …, n.
2. Find an i such that both
   a) Finish[i] = = false
   b) $Need_i$ <= Work
   If no such i exists go to step 4.
3. Work = Work + $Allocation_i$
   Finish[i] = true
   Go to step 2
4. If Finish[i] = = true for all i, then the system is in a safe mode.

This algorithm may require an order of m x n^2 operations to decide whether a state is safe.

2. Aik question thaa (15, 3922) on 20 byte is type ka tha k6 jis ma say logical address maloom krna tha.

Example:

Assume a **logical address space** of 16 pages of 1024 words, each mapped into a physical memory of 32 frames. Here is how you calculate the various parameters related to paging.

No. of bits needed for **p** = ceiling [$\log_2$ 16] bits = 4 bits
No. of bits neede for **f** = ceiling [$\log_2$ 32] bits = 5 bits
No. of bits needed for **d** = ceiling [$\log_2$ 2048] bits = 11 bits
Logical address size = |p| + |d| = 4+11 bits = 15 bits
Physical address size = |f| + |d| = 5+11 bits = 16 bits

Paging and thrashing related aik question

Thrashing

If a process does not have "enough" pages, the page-fault rate is very high. This leads to low CPU utilization. The operating system thinks that it needs to increase the degree of multiprogramming, because it monitors CPU utilization and find it to be decreasing due to page faults. Thus another process is added to the system and hence thrashing occurs and causes throughput to plunge.

A process is **thrashing** if it is spending more time paging (i.e., swapping pages in and out) than executing. Thrashing results in severe performance problems:

- Low CPU utilization
- High disk utilization
- Low utilization of other I/O devices

semaphore. A **semaphore** S is an integer variable that, apart from initialization is accessible only through two standard atomic operations: wait and signal. These operations were originally termed P (for wait) and V (for signal). The classical definitions of wait and signal are:

```
wait(S) {
  while(S<=0)
  ;// no op
  S--;
}
```

Bake 2 question yaad ne a rahay

Overall paper asan tha agr yaad keya ho to,,

Paper#5

sept-2022

Two atomic operations of Semaphore Is disk I/O to swap faster than the file system, justify ans

- A subtlety is that swap space is faster to access than the regular file system, because it does not have to go through the whole directory structure. For this reason some systems will transfer an entire process from the file system to swap space before starting up the process, so that future paging all occurs from the (relatively) faster swap space.
- Some systems use demand paging directly from the file system for binary code (which never changes and hence does not have to be stored on a page operation), and to reserve the swap space for data segments that must be stored. This approach is used by both Solaris and BSD Unix.

Bits required for 16 byte page size,

Example:

Assume a **logical address space** of 16 pages of 1024 words, each mapped into a physical memory of 32 frames. Here is how you calculate the various parameters related to paging.

No. of bits needed for **p** = ceiling [$\log_2$ 16] bits = 4 bits
No. of bits neede for **f** = ceiling [$\log_2$ 32] bits = 5 bits
No. of bits needed for **d** = ceiling [$\log_2$ 2048] bits = 11 bits
Logical address size = |p| + |d| = 4+11 bits = 15 bits
Physical address size = |f| + |d| = 5+11 bits = 16 bits

Frame allocation type and formulas

There are three major allocation schemes:

- **Fixed allocation**
  In this scheme free frames are equally divided among processes
- **Proportional Allocation**
  Number of frames allocated to a process is proportional to its size in this scheme.
- **Priority allocation**
  Priority-based proportional allocation

Here is an example of frame allocation:

Number of free frames = 64
Number of processes = 3
Process sizes: P1 = 10 pages; P2 = 40 pages; P3 = 127 pages

- **Fixed allocation**
  64/3 = 21 frames per process and one put in the free frames list
- **Proportional Allocation**
  - si = Size of process Pi
  - S = ∑ si
  - m = Number of free frames
  - ai = Allocation for Pi = (si / S) * m
    - a1 = (10 / 177) * 64 = 3 frames
    - a2 = (40 / 177) * 64 = 14 frames

Paper#6

23-9-2022

23-09-2022(11 Am)
Objective
McQ from junaid file (292-296)
McQ 81, 97, 114, 177, 186, 201, 228, 280, 289,

Subjective
Q#41
Vfork() different than fork()

vfork()

Several versions of UNIX provide a variation of the fork() system call—vfork() (for virtual memory fork). vfork() operates differently than fork() with copy on write. With vfork() the parent process is suspended and the child process uses the address space of the parent. Because vfork() does not use copy-on-write, if the child process changes any pages of the parent's address space, the altered pages will be visible to the parent once it resumes. Therefore, vfork() must be used with caution, ensuring that the child process does not modify the address space of the parent. vfork() is intended to be used when the

Q#42

Prepaging can be control by trashing?

Many other things can be done to help control thrashing. We discuss some of the important ones in this section.

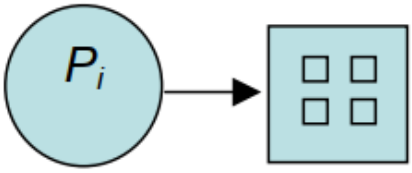
Pre-paging

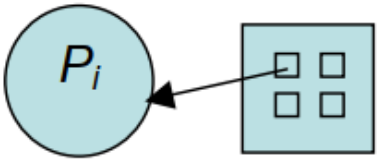
An obvious property of a pure demand paging system is the large number of page faults that occur when a process is started. This situation is the result of trying to get the initial locality into memory. Pre-paging is an attempt to prevent this high level of initial paging. The strategy is to bring into memory at one time all the pages that will be needed.

Pre-paging may be an advantage in some cases. The question is simply whether the cost of using pre-paging is less than the cost of the servicing the corresponding page faults.

Q#44

Part's and it's function? diagram given Pi(circle ○) and Ri (rectangle)And edges between Pi and Ri?

- Process 
- Resource Type with 4 instances 
- Pi requests instance of Rj 

- Pi is holding an instance of Rj 

Q#46

Paging and segmentation (Table given with some statement about paging and segmentation) Answer yes or no? clear concept of segmentation and paging very important topic

Q#47

Related question about semaphore? Repeated

Q#48

Related question about throughput? mid term

Q#49

Input, output and error redirection in Unix/Linux

Input, output and error redirection in UNIX/Linux

Linux redirection features can be used to detach the default files from stdin, stdout, and stderr and attach other files with them for a single execution of a command. The act of detaching defaults files from stdin, stdout, and stderr and attaching other files with them is known as input, output, and error redirection. In this section, we show the syntax, semantics, and examples of I/O and error redirection.

Q#50

Three Similarities between monitor and critical region? Handouts page 124,125

Critical regions

Although semaphores provide a convenient and effective mechanism for process synchronization, their incorrect usage can still result in timing errors that are difficult to detect, since these errors occur only if some particular execution takes place, and these sequences do not always happen.

To illustrate how, let us review the solution to the critical section problem using semaphores. All processes share a semaphore variable mutex, which is initialized to 1.

Monitors

Another high-level synchronization construct is the monitor type. A **monitor** is characterized by local data and a set of programmer-defined operators that can be used to access this data; local data be accessed only through these operators. The representation of a monitor type consists of declarations of variables whose values define the state of an instance of the type, as well as the bodies of procedures or functions that implement operations on the type. Normal scoping rules apply to parameters of a function and to its local variables. The syntax of the monitor is as follows:

Paper#7

21-9-2022 Cs604

Mcqs Junaid , Riz file Moazz file

1:Fork system call say related tha(193 page per answer ha is Ka) ?

vfork()

Several versions of UNIX provide a variation of the fork() system call—vfork() (for virtual memory fork). vfork() operates differently than fork() with copy on write. With vfork() the parent process is suspended and the child process uses the address space of the parent. Because vfork() does not use copy-on-write, if the child process changes any pages of the parent's address space, the altered pages will be visible to the parent once it resumes. Therefore, vfork() must be used with caution, ensuring that the child process does not modify the address space of the parent. vfork() is intended to be used when the

193

2:Safe state is necessary in deadlock prevention?

Deadlock can be prevented by eliminating any of the four necessary conditions, which are **mutual exclusion, hold and wait, no preemption, and circular wait**. Mutual exclusion, hold and wait and no preemption cannot be violated practically. 15-Feb-2022

https://www.scaler.com › topics › deadlock-prevention-in-...

Deadlock Prevention in Operating System (OS) - Scaler Topics

3: differentiate Response time / turnaround time

midterm

4: explain deadlock 4 condition

The following four conditions must hold simultaneously for a deadlock to occur:

1. **Mutual exclusion**: At least one resource must be held in a non-sharable mode; that is only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.
2. **Hold and wait**: A process must be holding at least one resource and waiting to acquire additional resources that are currently being held by other processes.
3. **No preemption**: Resources cannot be preempted. That is, after using it a process releases a resource only voluntarily.
4. **Circular wait**: A set $\{P_0, P_1 \ldots P_n\}$ of waiting processes must exist such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by P_2, and so on, P_{n-1} is waiting for a resource held by P_n, and P_n is waiting for a resource held by P_0.

5 : wait for graph banana tha resources allocation graph ko dakh ker or batana tha Ka deadlock ha ya Nahi ?

Page 132,133

6: ak frame say resulted tha calculation Ka ?

Here is an example of frame allocation:

Number of free frames = 64
Number of processes = 3
Process sizes: P1 = 10 pages; P2 = 40 pages; P3 = 127 pages

- **Fixed allocation**
  64/3 = 21 frames per process and one put in the free frames list

7:define 2 schemes of efficient implementation of page table? (170)

following schemes allow efficient implementations of page tables.

- Hierarchical / Multilevel Paging
- Hashed Page Table
- Inverted Page Table

8:thrashing Ka tha ak (page 207 per answer ha is Ka)

Thrashing

If a process does not have enough frames, it will quickly page fault. At this point, if a free frame is not available, one of its pages must be replaced so that the desired page can be loaded into the newly vacated frame. However since all its pages are in active use, the replaced page will be needed right away. Consequently it quickly faults again and again. The process continues to fault, replacing pages for which it then faults and brings back in right away. This high paging activity is called **thrashing**. In this case, *only one process is thrashing*. A process is thrashing if it is spending more time paging than executing.

Thrashing results on severe performance problems. The operating system monitors CPU utilization and, if CPU utilization is too low, the operating system increases the degree of multiprogramming by introducing one or more new processes to the system. This decreases the number of frames allocated to each process currently in the system, causing more page faults and further decreasing the CPU utilization. This causes the operating system to introduce more processes into the system. As a result CPU utilization

Paper#8

21-9-2022

Cs604 today paper MCQS from past papers

Subjective paper:

1)shortest seek time first ki diagram bnani thi or calculate krna tha waiting time and average time repeated

2)bounded buffer problem ko define krna tha or syntax dia hua tha us mai mistakes check krni thi repeated

3)wait for graph bnana tha repeated

Paper#9

20-9-2022 Cs604

20.9.2022 at 5pm

40 mcq

80 prcnt from wqar file

Directories Segmentation Fragmentation

Cpu scheduler Deadlock

Short or long question

Paper#10

19-9-2022

Cs604

Subjective

Deadlock techniques

Page table

1 numerical from lectures 32

Example:
Assume a **logical address space** of 16 pages of 1024 words, each mapped into a physical memory of 32 frames. Here is how you calculate the various parameters related to paging.

No. of bits needed for **p** = ceiling [$\log_2$ 16] bits = 4 bits
No. of bits neede for **f** = ceiling [$\log_2$ 32] bits = 5 bits
No. of bits needed for **d** = ceiling [$\log_2$ 2048] bits = 11 bits
Logical address size = |p| + |d| = 4+11 bits = 15 bits
Physical address size = |f| + |d| = 5+11 bits = 16 bits

1 question related to diagram

Which technique is used in page table in the main memory?

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. The process of retrieving processes in the form of pages from the secondary storage into the main memory is known as paging. 23-Jan-2023

https://www.geeksforgeeks.org › paging-in-operating-syst...

Paging in Operating System - GeeksforGeeks

What operation are used in a file

To define a file properly, we need to consider the operations that can be performed on files. Six basic file operations. The OS can provide system calls to **create, write, read, reposition, delete, and truncate** files.

What is mounting and their types Repeated

Paper#11

18-9-2022

Date: 18 September 22; Time: 11:00

I/O swap space

Swap space can be useful to computers in various ways:

- It can be used as a single contiguous memory which reduces I/O operations to read or write a file.
- Applications that are not used or are used less can be kept in a swap file.
- Having sufficient swap files helps the system keep some physical memory free all the time.
- The space in physical memory which has been freed due to swap space can be used by OS for some other important tasks.

Semaphore wait and signal

two logics Deadlock detect and recover with two types commands for "Read, rewind and close"

systems. Next to each operation are UNIX system calls or commands for the corresponding operation.

- Create — `mkdir`
- Open — `opendir`
- Read — `readdir`
- Rewind — `rewinddir`
- Close — `closedir`
- Delete — `rmdir`
- Change Directory — `cd`
- List — `ls`
- Search

Paper#12

25-03-2022

While managing computer system hardware efficiently, sometimes it seems reasonable to service all requests that are close to current head position identify the technique which will work on this policy. (3 marks)

Disk Scheduling

One of the responsibilities of the operating system is to use the computer system hardware efficiently. For the disk drives, meeting this responsibility entails having a fast access time and disk bandwidth. The access time has two major components. The seek time is the time for the disk arm to move the heads to the cylinder containing the desired sector. The rotational latency is the additional time waiting for the disk to rotate the desired sector to the disk head. The disk bandwidth is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer. We can improve both the access time and the bandwidth by scheduling the servicing of disk I/O requests in a good order. Some of the popular disk-scheduling algorithms are:

- First-come-first-serve (FCFS)
- Shortest seek time first (SSTF)
- Scan
- Look
- Circular scan (C-Scan)
- Circular look (C-Look)

If it is not possible to find a safe sequence after running safety algorithm for requesting purpose. What does it mean? (3 marks)

Safe State

A state is safe if the system can allocate resources to each process in some order and still avoid a deadlock. More formally a system is in a safe state only if there exists a safe sequence. A sequence of processes $<P_1, P_2 \ldots P_n>$ is a safe sequence for the current allocation state if, for each P_i, the resources that P_i can still request can be satisfied by the currently available resources plus all the resources held by all the P_j with $j < i$. In this situation, if the resources that P_i needs are not immediately available, then P_i can wait until all P_j have finished. When they have finished, P_i can obtain all of its needed resources, complete its designated task, return its allocated resources and terminate. When P_i terminates, P_{i+1} can obtain its needed resources and terminate. If no such sequence exists, then the system is said to be unsafe.

If a system is in a safe state, there can be no deadlocks. An unsafe state is not a deadlocked state; a deadlocked state is conversely an unsafe state. Not all unsafe states are deadlocks, however an unsafe state may lead to a deadlock state. Deadlock avoidance makes sure that a system never enters an unsafe state. The following diagram shows the relationship between safe, unsafe, and deadlock states.

Paper#13

10 sept 20202 Cs604..Current Final ppr

11 mcqs moaz r waqar file mn sy

thy..baki bhi easy thy..5 qs 3 mrks
k r 5 qs 5 mrks k thy

1. preepmtive kesy use hota hy deadlock situation men.

A deadlock in OS is a situation in which more than one process is blocked because it is holding a resource and also requires some resource that is acquired by some other process. The four necessary conditions for a deadlock situation are mutual exclusion, **no preemption**, hold and wait and circular set. 23-Feb-2022

3. **No preemption**: Resources cannot be preempted. That is, after using it a process releases a resource only voluntarily.

3. access and classes btani thi unix ki.

In Unix, file permissions, which establish who may have different types of access to a file, are specified by both access classes and access types. **Access classes are groups of users, and each may be assigned specific access types**. The access classes are "user", "group", "other", and "all". 15-Nov-2019

4 . ik r numerical tha us men logical address find krn tha.
5. ik directory di hoi thi r btana tha k is sy commnd dety hen Long
shortest Remaining Time First waly topic mn sy numerical type aya tha... midterm
ariving time r burst time dia hoa tha...wating time r us ki avrg find krni thi with gantt char.
monitor r critical section ki 3 smiralities btani then.. repeat
2 schemes btani thhen..
page table ko easy bnany k lea.
deadlock avoidace mn safe sequence pta krna zaruri

hota hy ya nai.

Yes its important to find safe sequence otherwise there will be a deadlock

ik r tha k pages full hen r page fault bhi zaida hen...to r pages add krny sy ki throutput increase hoga CPU ka?

No throughput will decrease.

Paper#14

3 sept 2022

cs604 9:30 3 sep

How process share memory in segments

> **A process creates a shared memory segment using shmget**(2). This call is also used to get the ID of an existing shared segment. The creating process sets the permissions and the size in bytes for the segment. The original owner of a shared memory segment can assign ownership to another user with shmctl(2).

Producer consumer ka code tha missing fill krna tha (imp) repeat

two method name in deadlock recovery

> **Recovery from Deadlock**
> When a deadlock detection algorithm determines that a deadlock exists, several alternatives exist. One possibility is to inform the operator that a deadlock has occurred, and to let the operator deal with the deadlock manually. The other possibility is to let the system recover from the deadlock automatically. There are two options for breaking a deadlock. One solution is simply to abort one or more processes to break the circular wait. The second option is to preempt some resources from one or more of the deadlocked processes.

types of access in Unix and classes in Unix promisson long

nanosec m value di hoi the t effective find krna tha

Example 2

T_{mem} = 100 nsec
T_{TLB} = 20 nsec
Hit ratio is 98%
$T_{effective}$ = 0.98 (20 + 100) + 0.02 (20 + 2*100) nanoseconds = 122 nanoseconds

how to we can improve the cpu utilization
parameters deay thy un me s batana tha k kis improve kea ja sakhta justify krna tha
deadlock four conditions Batani the
or mcqs kuch 4 5 past s thy bki handouts s

Paper#15

6 sept 2021
16 MCQS All From Past (Waqar,arsalan,junaid)
Q17:overlays problem
Q18:Deadlock recovery
Q19:Semaphore registers
Q20:modes of access and types of classes in unix

In Unix, there are two main modes of access control for files and directories:

User-based access control: This mode of access control allows the owner of a file or directory to specify which other users can access it and in what way. The owner of a file can specify access permissions for three types of users: the owner themselves, members of their group, and all other users.

Role-based access control: This mode of access control allows the system administrator to specify which users can access specific files or directories based on their role within the organization. For example, an administrator may define roles such as "managers", "developers", and

"administrators" and then specify which files and directories each role can access.

Q21:Find Logical address and Page offset (numerical)

To find the page number and page offset from a given logical address, the following steps can be taken:

Determine the page size: This is the size of the page in bytes. It is typically a power of 2 and is specified by the operating system.

Split the logical address into two parts: The most significant bits represent the page number, while the least significant bits represent the page offset.

Calculate the page number: To do this, take the most significant bits of the logical address and convert them to a decimal number. This decimal number represents the index of the page in the page table.

Calculate the page offset: To do this, take the least significant bits of the logical address and convert them to a decimal number. This decimal number represents the offset within the page.

For example, let's say we have a system with a page size of 4KB (4096 bytes) and a logical address of 0x12345678. To find the page number and page offset, we would follow these steps:

Page size = 4096 bytes
Logical address = 0x12345678
Page number = 0x12345000 (most significant 20 bits, or 5 hex digits)
Page offset = 0x678 (least significant 12 bits, or 3 hex digits)
Therefore, the page number is 0x12345000 and the page offset is 0x678.Q22:Detail note on semaphore

Paper#16

6 sept 2021
Share information with segment strategy
Deadlock cycle no permission say related tha aik short (3 marks)
Long 2 aye thay aik diagram thi jis ki coordination batani thi. Aur dushra Linux ka copy or write batana tha.

Paper#17

6 sept 2021

Source code to executable code steps
File protection
UNIX/LINUX may virtual memory ka concept, esa kuch question tha
Page replacement (algorithms)
Paged segmentation
Logical to physical address translations(steps)
Deadlock prevention me safe state essential hai ya nahi
Ek table me values thi uss may say average wait time or first come first serve algorithm say calculation karni thi.
Values di hoi thi uska gant chart banana tha.
Segmentation kay 2 registers kay name.

Paper#18

1 sept 2021
80% MCQS from past moaz file Name of three deadlock method Access class mode user
Long 3 Similarities Critical region and monitoring region?
Enlist Bits of protection?

Paper#19

31 august 2021

Virtual memory is extremely large than main memory.
It's yes it not justify your answer.

Table of Page and segment. Answer with yes or not.

If the five philosopher are hungry at the same time and they have chopstick in his or her left hand to write the techniques that the philosopher Ray the dinner without any deadlock condition occur. Write at least three quotations.

Deadlock avoidance.

Paper#20

Kinds of semaphores

Primitives are defined as:

Send(A, message) – send a message to mailbox A.
Receive(B, message) – receive a message from mailbox B.

Linux, Unix Commands for rewind, read and close.

If HR is hit ratio and MR is miss ratio, the effective access time is given by the following equation Teffective = HR (TTLB + Tmem) + MR (TTLB + 2Tmem)

Paging ka ek table tha wo fill karna tha.

Paper#21

Q1: Consider the following programming techniques and data structures. In demand paged environment, which of the following is/are good or bad?

1. **Stack**
2. **Hash table**
3. **Sequential search**

Suppose there's a deadlock occurs in the system and a deadlock detection algorithm detects a deadlock then how these deadlocks will be recovered? Mention any two method names.

Stack: In a demand paged environment, using a stack to allocate and deallocate memory is generally a good choice. This is because the stack is a Last-In-First-Out (LIFO) data structure, which allows for efficient memory allocation and deallocation. When a new page is needed, it can be easily added to the top of the stack, and when a page is no longer needed, it can be quickly removed from the top of the stack.

Hash table: In a demand paged environment, using a hash table can be both good and bad. On the one hand, hash tables provide fast lookup times, which can be beneficial in reducing the amount of page faults. However, hash tables can also be memory intensive, as they require a significant amount of memory to store the hash table itself and the linked lists that may be needed to handle collisions. Therefore, the effectiveness of using a hash table in a demand paged environment depends on the specific implementation and the size of the hash table.

Sequential search: In a demand paged environment, using sequential search is generally a bad choice. This is because sequential search involves searching through each element of a data structure until the desired element is found. This can be a slow and inefficient process, particularly if the data structure is large. In a demand paged environment, page faults can occur frequently, and a slow search algorithm can exacerbate the problem by increasing the time it takes to access the needed pages. Therefore, using a more efficient search algorithm, such as binary search, is typically a better choice.

Q2: Consider a scenario of demand paging system with the following utilizations:

A. **CPU = 10 %**
B. **Paging disk = 95%**
C. **Other I/O devices = 3%**

You are required to identify the situation of system by considering the above parameters and which of the following parameter will contribute to improve CPU utilization? Justify them. .

1. **Install a faster CPU**
2. **Introduce page replacement algorithms**
3. **Increase degree of multiprogramming**
4. **Increase hard disk storage**
5. **Decrease degree of multiprogramming**
6. **Install more main memory**

To improve CPU utilization in this scenario, one possible solution would be to increase the amount of main memory available to the system. By doing so, the frequency of page faults would decrease, and the system would spend less time swapping pages in and out of memory. This would lead to faster access times for the CPU, which would result in a higher CPU utilization. Therefore, option 6 (install more main memory) is the parameter that would contribute to improving CPU utilization in this scenario.

Option 1 (install a faster CPU) would not necessarily improve CPU utilization, as the system is already experiencing a low load on the CPU. Increasing the speed of the CPU would not necessarily speed up page operations.
Option 2 (introduce page replacement algorithms) could help to reduce the frequency of page faults, but it would not necessarily improve CPU utilization.
Option 3 (increase degree of multiprogramming) could potentially improve CPU utilization by allowing the system to run more processes simultaneously, but it would not necessarily address the underlying issue of frequent page faults.
Option 4 (increase hard disk storage) would not necessarily improve CPU utilization, as the system is already experiencing a high load on the paging disk. Adding more storage would not necessarily speed up page operations.
Option 5 (decrease degree of multiprogramming) could potentially reduce the load on the system, but it would also reduce the amount of work the system can do at any given time, and it would not necessarily address the underlying issue of

frequent page faults.

Q:3 Consider a swapping system in which memory consists of the following hole sizes in memory order: 10KB, 4 KB, 20 KB, 18 KB, 7 KB, 9 KB, 12 KB, and 15 KB. Which holes is taken for successive segment requests of: (a) 12 KB (b) 10 KB (c) 9 KB For best fit and worst fit?

We can solve this problem using the best fit and worst fit memory allocation algorithms.

Best fit: 9KB , 10KB, 12KB

Worst Fit: 12KB, 10KB, 9KB

Q:4 You have studied the concepts of semaphore in details. Answer the following questions. Classify the semaphores into at least two categories based on waiting behavior. And when a specific semaphore is more useful as compared to other.

Semaphores can be classified into two categories based on their waiting behavior:

Blocking semaphores: A blocking semaphore, also known as a binary semaphore, blocks the calling process until the semaphore is available. If the semaphore is not available, the process is put into a waiting queue until it becomes available. Once the semaphore is released, one waiting process is allowed to proceed.

Non-blocking semaphores: A non-blocking semaphore, also known as a counting semaphore, does not block the calling process if the semaphore is not available. Instead, the process simply returns an error code and continues executing. Non-blocking semaphores are typically used for synchronization between processes or threads.

When a specific semaphore is more useful than the other depends on the specific use case. Blocking semaphores are typically used in situations where mutual exclusion is required, such as protecting a critical section of code from concurrent

access by multiple processes or threads. Non-blocking semaphores, on the other hand, are typically used in situations where multiple processes or threads need to access a shared resource concurrently, and a blocking semaphore would cause unnecessary delays and reduce performance .In general, blocking semaphores are more useful when there is a need for mutual exclusion, and non-blocking semaphores are more useful when there is a need for coordination and synchronization between multiple processes or threads.

Q:5 Consider the following table for 2 resources and 4 processes. Identify whether system is in safe state or not? If yes then determine the safe sequence from given information.

	Current Allocation		Max		Available		Need	
	A	B	A	B	A	B	A	B
P0	2	0	2	4	2	7	0	4
P1	3	2	9	2			6	0
P2	1	1	4	1			3	0
P3	1	2	6	4			5	2

In Linux operating system, you are required to write down the working of copy on write technique with respect to virtual memory.

We can use the banker's algorithm to determine whether the system is in a safe state.
First, we calculate the total resources available in the system by adding up the current allocation and the available resources.
A = 2 + 3 + 1 + 1 = 7
B = 0 + 2 + 1 + 2 = 5
Using the banker's algorithm, we can see that the system is in a safe state and the safe sequence is P1, P3, P2, P0.
Copy on write (COW) is a technique used in virtual memory to conserve memory by sharing pages of memory between processes. In COW, when a process requests a new page of memory, the operating system creates a copy of the page, but the copy is not immediately created. Instead, the page is marked as read-only and shared between the requesting process and the original process that owns the page.

Q: 6In dining philosopher problem, assume that all five philosophers get hungry at the same then all will pick up their right chopstick and everyone will be waiting for his/her left chopstick. In that case, no

one will get the left chopstick and they will get starve. This problem will create deadlock situation. Many remedies have been suggested to resolve this deadlock problem. You are required to give at least three solutions for such type of problem.

Here are three possible solutions to avoid deadlock in the dining philosopher problem:

1. Resource hierarchy solution: Assign an order or priority to the chopsticks, such as a numbering system or a hierarchy based on the seating arrangement. Each philosopher must pick up the lower-numbered chopstick first before picking up the higher-numbered chopstick. This eliminates the possibility of a circular wait and guarantees that the resources will be allocated in an orderly fashion.
2. One chopstick as shared resource: Instead of each philosopher holding two chopsticks, allow the philosophers to share a common chopstick between two adjacent philosophers. This solution eliminates the possibility of circular wait as no two adjacent philosophers can pick the common chopstick at the same time.
3. Timeout solution: Set a timer for each philosopher while waiting for the chopstick. If a philosopher cannot acquire both chopsticks within a specified time, they release the chopsticks and return to thinking state. This ensures that no philosopher starves and all get an equal opportunity to eat.

Q:7 Consider a scenario when a process does not have adequate pages because the page-fault rate is very high. This situation leads to low CPU utilization. Will CPU utilization decreases due to high page fault rate? Justify your answer

CPU utilization can decrease due to a high page fault rate because when a process requires a page that is not present in physical memory, a page fault occurs. The operating system then has to load the required page from the disk into physical memory, and the process has to wait until the page is loaded. This waiting time reduces the CPU utilization because the

CPU is not executing any instructions during this time.
If the page-fault rate is very high, it means that the process is experiencing a large number of page faults, which translates to a longer waiting time for the required pages to be loaded. As a result, the CPU has to wait for a longer time for the process to resume execution, leading to a decrease in CPU utilization.

Q8: Name three schemes that allow efficient implementation of page table?

There are several schemes that can be used to efficiently implement page tables in operating systems. Three of the most commonly used schemes are:

1. Hierarchical Page Tables: In hierarchical page tables, the page table is divided into multiple levels, with each level having a smaller size than the previous level. Each level of the page table contains a page directory or page table, and the page directory or table entries point to the next level of the page table. This scheme is efficient because it allows the page table to be stored in a smaller space and requires fewer memory accesses to find a page table entry.

2. Inverted Page Tables: In inverted page tables, instead of having a page table for each process, a single page table is used for all processes in the system. The page table contains entries for each physical page frame, and each entry contains information about the process that owns the page frame and the virtual address range mapped to the page frame. This scheme is efficient because it reduces the amount of memory required to store page tables and reduces the time required to search for a page table entry.

3. Multi-level Page Tables: Multi-level page tables are similar to hierarchical page tables, but they use multiple levels of page tables of the same size. Each page table entry points to another page table, and the last level of the page table points to the physical page frame. This scheme is efficient because it allows the page table to be divided into smaller tables, which reduces the amount of memory required to store the page table and speeds up the search for a page table entry.

Paper#22

You are required to identify the problems in case of following three piece of code.

P0	P1
signal(S);	wait(S);
...	...
wait(S);	signal(S);
...	...

b)

P0	P1
wait(S);	wait(Q);
wait(Q);	wait(S);
... .	..
signal(S);	signal(Q);
signal(Q);	signal(S);

c)

P0	P1
wait(S);	wait(S);
...	...
wait(S);	signal(S);
...	...

If it is not possible to find a safe sequence after running the safety algorithm for a requestingprocess then what does it means?

a) There is a potential deadlock situation. If P0 executes the signal(S) operation before P1 executes the wait(S) operation, then P1 will be blocked indefinitely. Similarly, if P1 executes the wait(S) operation before P0 executes the signal(S) operation, then P0 will be blocked indefinitely.

b) There is a potential deadlock situation. If P0 executes the wait(S) operation before P1 executes the wait(Q) operation, and P1 executes the wait(S) operation before P0 executes the wait(Q) operation, then both processes will be blocked indefinitely. This is known as a circular wait condition.

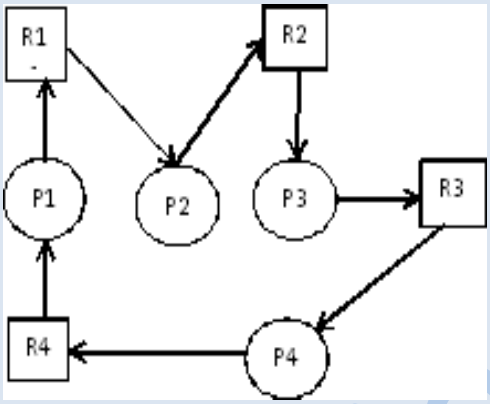
c) This code segment does not have any apparent problems as it involves mutual exclusion. The wait() and signal() operations are in the correct order and there is no potential for deadlock. However, the exact context of this code segment should be analyzed to ensure that it is being used correctly.

It means system is not in stable. This system will not end as a safe system and there will bedeadlock.

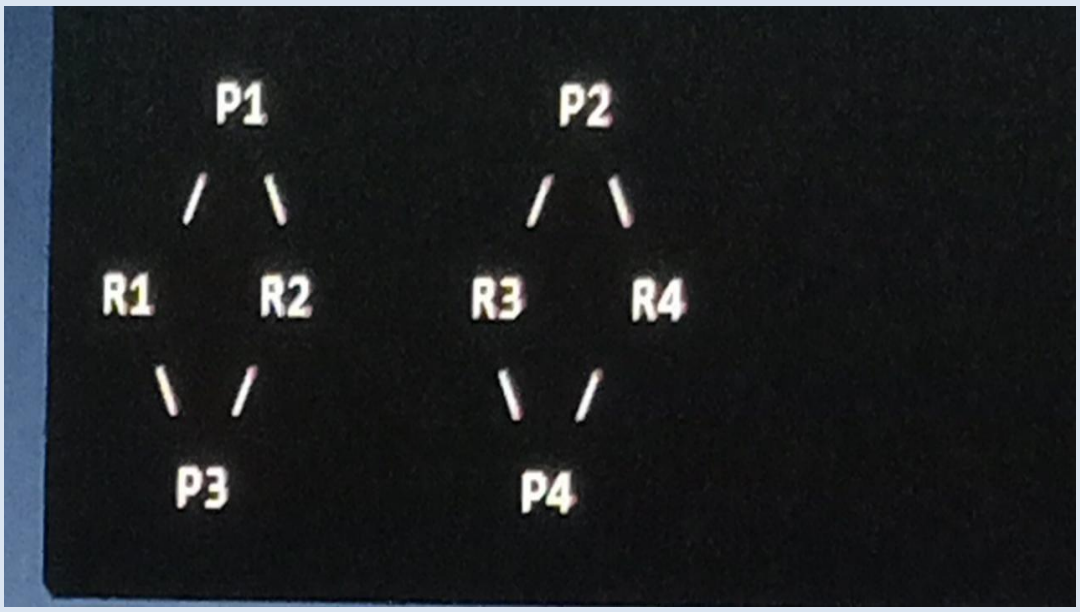
Let us consider a page size of 16 bytes and process address space of 32 pages and physicaladdress space of 64 frames. Calculate the following.

a) Number of bits needed for ‘p’.
b) Number of bits needed for ‘f’.
c) Number of bits needed for ‘d’.

Consider a system with 4 processes and 4 resources with single instance of eachresource type. Given below is the Resource Allocation graph. You are required to construct wait-for graphfrom Resource Allocation graph and find out whether the deadlock exists or not?



To construct the wait-for graph from the resource allocation graph, we start by drawing a node for each process and resource in the system. We then draw edges between nodes to represent requests and allocations of resources. The resulting wait-for graph is:



To determine whether a deadlock exists, we look for cycles in the wait-for graph. If there is a cycle, then there is a circular wait and a deadlock exists.
In this case, we can see that there is a cycle in the wait-for graph: P1 → R2 → P3 → R1 → P1. This cycle represents a circular wait, where each process is waiting for a resource that is held by another process in the cycle. Therefore, a deadlock exists in this system.

Q: You are required to calculate the following parameters assuming logical address space of 8 pages of 1MB and physical address space of 16 frames. (repeat)

No. of bits required for p
No. of bits required for f
No. of bits required for d
No. of bits required for logical address
No. of bits required for physical address

Q: You have studied the concepts of semaphore in details. Answer the following questions. Classify the semaphores

into at least two categories based on waiting behavior. And when a specific semaphore is more useful as compared to other. (repeat)

Q: Describe the operations that a file owner can use for 'file protection'. Also mention some operations that we can perform on a file.

Ans: File protection refers to the various methods used to limit access to a file and protect it from unauthorized modification or deletion. The specific operations available for file protection may depend on the operating system and the file system in use, but some common operations that a file owner can use for file protection are:

1. Changing file permissions: File owners can change the permissions of a file to control who can access it and what actions they can perform on it. File permissions typically include read, write, and execute permissions for the owner, group, and other users.
2. Setting ownership: File owners can set the ownership of a file to control who has administrative control over it. By default, the owner of a file is the user who created it.
3. Setting access control lists (ACLs): Access control lists are more fine-grained than traditional Unix-style permissions, allowing file owners to specify access permissions for individual users or groups.
4. Encrypting files: File owners can encrypt files to protect their contents from unauthorized access. Encryption can be used in conjunction with other protection measures like access control lists.

Q. Is disk I/O to swap space generally faster than file system? Justify the statement with valid reasons in

either case.

Disk I/O to swap space is generally faster than that to the file system. It is faster because swap space is allocated in much larger blocks, and file lookups and indirect allocation methods are not used.

Q. Is it necessary to have a reference count within a file descriptor in order to implement soft links? Justify the answer. 3 marks

Ans: No, it is not necessary to have a reference count within a file descriptor in order to implement soft links.

A soft link, also known as a symbolic link, is a type of file that contains a path to another file or directory. When a user tries to access a soft link, the operating system transparently redirects them to the target file or directory. Soft links are useful for creating aliases or shortcuts to files that are located in different directories. Therefore, implementing soft links does not require a reference count within a file descriptor. The file system itself is responsible for managing the reference count and ensuring that the soft link and target file remain valid and accessible