Assignment → II

Name :- Soumyadip Mondal
Roll :- 33200118011
Dept :- CSE (3rd year, 4th sem)
Sub :- Operating System
Sub. code :- PCC - CS - 502

Prime Merit
Page No.
Date

17. In multiprogramming environment: several processes may complete for a finite number of resources. A resources is requested by a process, but if the resources is not available at that time, the process enters the waiting state, because the resource it has requested are held by other waiting processes. This situation is called a deadlock.

Necessary Conditions :-

A deadlock situation can arise if the following 4 conditions hold simultaneously in a System.

(a) Mutual exclusion :- At least one resource must be held in a nonsharable mode: that is only one process at 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.

(b) Hold and wait :- A process must be holding at least one resource and waiting to require aditional resources that are currently being hold by other process.

(c) No preemption :- Resources can't be preempted; a resource can be released only voluntarily by the process holding it, after that process has complete it's task.

(d) Circular wait :- A set {$P_0, P_1, \ldots\ldots P_n$} of waiting processes must exist such that P_0 is waiting for a resource hold by P_1, P_1 is waiting for a resource held by P_2, ------, P_{n-1} is waiting for a resource held by P_n.. and P_n is waiting for the resources held by P_0.

Scanned with CamScanner

18.

Processor	Allocation			Max			Available		
	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	5	3	3	3	2
P_1	2	0	0	3	2	2			
P_2	3	0	2	9	0	2			
P_3	2	1	1	2	2	2			
P_4	0	0	2	4	3	3.			

(i) What is the content of need matrix
(ii) Is the system is safe state.
(iii) If a request from process P_1 arrives for (1,0,2) can this request be granted immediately.

(i) Need Matrix :-

Process	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

(ii) Available :-

	A	B	C
	3	3	2
P_1	5	3	2
P_3	7	4	3
P_4	7	4	5
P_0	7	5	5
P_2	10	5	7

The system is in safe state.

(iii) P_1 requests (1, 0, 2)

Allocation P_1 = 302
Need P_1 = 020
Available = 2 30.

A	B	C	
2	3	0	
5	3	2	(P_1)
7	4	3	(P_3)
7	4	5	(P_4)
7	5	5	(P_0)
10	5	7	(P_2)

Request of P1 can be granted immediately.

Prime Merit
Page No.
Date

19. Logical address — Physical address

Logical address	Physical address
(i) An address generated by the CPU is a logical address	(i) Address actually available on memory unit is a physical address
(ii) The set of all logical address generated by a program is reffered to as logical address space	(ii) The set of all physical address corresponding to logical address is called physical address space.
(iii) The user program deals with virtual address	(iii) The user program never sees the real physical address.

20. In dynamic loading, a routine of a program is not loaded until it is called by the program. All routines are kept on disk in a relocatable load format. The main program is loaded into memory and is executed. Other routines methods or modules are loaded on request. It makes better memory space utilization and unused routines are never loaded.

21. Linking is the process of collecting and combining various modules of code and data into a executable file that can be loaded into a memory and executed. OS can link system level libraries time, then it is called dynamic linking.

In static linking libraries linked at compile time. So program code size becomes bigger where as in dynamic linking libraries linked at execution time so program code size remains smaller.

22. First-fit:- Allocate the first hole that is big enough. Searching can start either at the beginning of the set of holes or at the location where the previous first fit search ended. We can stop searching as soon as we find a free hole that is large enough.

Best fit:- Allocate the smallest hole that is big enough. We must search the entire list, unless the list is ordered by size. This strategy produces the smallest leftover hole.

Worst fit:- Allocate the largest hole. Again, we must search the entire list, unless it is sorted by size. This strategy produces the largest leftover hole, which may be more useful than the smaller leftover hole from a best-fit approach.

23. As processes are loaded and removed from memory, the free memory space is broken into little pieces. It happens after sometimes that processes cannot be allocated to memory blocks remains unused. This process is known as fragmentation.

Types of fragmentation:-

(i) External fragmentation
(ii) Internal fragmentation.

→(i) External fragmentation:- Total memory space is enough to satisfy a request or to reside a process in it, but it is not contiguous so it cannot be used.

(ii) Internal fragmentation :- Memory block assigned to process is bigger some portion of memory is left unused as it cannot be used by another process.

24. Paging is a memory-management scheme that permits the physical address space of a process to be non-contiguous. It avoids external fragmentation, and the need for compaction. It also solves the considerable problem of fitting memory chunks of varying sizes onto the backing stores; most memory management used before the introduction of paging, suffered from this problem. The problem arises because when some code fragments or data residing in main memory need to be swapped out, space must be found on the backing store. The backing store has the same fragmentation problems discussed in connection with main memory, but the access is much slower, so compaction is impossible. Because of it's advantages over earlier methods, paging in it's various forms is used in most operating system.

25. Segmentation is a technique to break memory into logical pieces, where each piece represents a group of related information. For exam, data segments for operating system and so on. Segmentation can be implemented using or without using paging.

Unlike paging, segment is having varying sizes and thus eliminates internal fragmentation. External fragmentation still exists but to lesser extent.

Prime Merit
Page No.
Date

26. Virtual memory is a technique that allows the execution of processes which are not completely available in memory. Here the program can be longer than physical memory. Virtual memory is the separation of ~~var~~ user logical memory from physical memory. This separation allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available.

27. Considering an executable program to be loaded from disk into memory, one option is to load the entire program in physical memory at program execution time. But the problem with this approach is that we may not initially need the entire program in memory. Suppose a program starts with a list of available options from which the user is to select. Loading the entire program into memory results in loading the executable code for all options, regardless of whether an option is ultimately selected by the user or not. An alternative strategy is to load pages only as they are needed. This technique is known as demand-paging.

A demand-paging system is similar to a paging system with swapping where processes reside in secondary memory. When we want to execute a process, we swap it into memory. Rather than swapping the entire process into memory, however, we use lazy swapping. A lazy swapper never swaps a page into memory unless that page will be needed.

Scanned with CamScanner

28. 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

Newest → 7 0 1 2 2 3 0 4 2 3 0 0 0 1 2 2 2 7 0 1

7 0 1 1 2 3 0 4 2 3 3 3 0 1 1 1 2 7 0

7 0 0 1 2 3 0 4 2 2 2 3 0 0 0 1 2 7

h h h h h

FIFO → page hit = 5

page fault = 15

page fault rate = 15/20 = 3/4.

7 0 1 2 0 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

Newest 7 0 1 1 1 1 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1

7 0 0 0 0 0 0 4 4 4 0 0 0 0 0 0 0 0 0 0

7 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 7 7 7

h h h h h h h h h h h

Optimal → page hit = 11

page fault = 9

page fault = 9/20.

7 0 1 2 0 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

Newest → 7 0 1 1 1 2 3 3 3 2 2 2 2 2 2 2 2 2 7 7 7

7 0 0 0 0 0 0 0 0 3 3 3 3 3 3 0 0 0 0 0

7 2 2 2 2 2 4 4 4 0 0 0 1 1 1 1 1 1 1

h h h h h h h h

page hit = 8

page fault = 12

page fault rate = 12/20 = 3/5.

29. In computer storage, Belady's anomaly is the phenomenon in which increasing the number of page frames results in an increase in the number of page faults for certain memory access patterns. This phenomenon is commonly experimented when using the first-in-first-out page replacement algorithm.

Example:-

page req. →	3	2	1	0	3	2	4	3	2	1	0	4
Newest page →	3	2	1	0	3	2	4	4	4	1	0	0
		3	2	1	0	3	2	2	2	4	1	1
			3	2	1	0	3	3	3	2	4	4
	F	F	F	F	F	F	F	H	H	F	F	H

page fault 29

page req. →	3	2	1	0	3	2	4	3	2	1	0	4
Newest page →	3	2	1	0	0	0	4	3	2	1	0	4
		3	2	1	1	1	0	4	3	2	1	0
			3	2	2	2	1	0	4	3	2	1
				3	3	3	2	1	0	4	3	2
	F	F	F	F	H	H	F	F	F	F	F	F

30. If the number of frames allocated to a low-priority process falls below the minimum number required by the computer architecture, we must suspend that process's execution. We should then page out its ~~item~~ remaining pages, freeing all its allocated frames. This provision introduces a swap-in, swap-out level of intermediate CPU scheduling.

In fact, looking at any process that does not have enough frames. If the process does not have the number of frames it needs to support pages in active use, it must replace a page that will be needed again right away. Consequently, it quickly faults again, again, and again, replacing pages that it must bring back immediately. This high paging activity is called thrashing. It is spending more time in paging than executing.

Prime Merit
Page No.
Date

32. In unix based operating system each file is indexed by an ~~inode~~ inode. Inode are special disk blocks, they are created when the file system is created. The number of Inode limits the total number of files/directions that can be stored in the file system.

33. Direct Memory Access (DMA):-
Many computers avoid burdening the main CPU with programmed I/O by offloading some of this work to a special purpose processor. This type of processor is called, a direct memory access (DMA) controller. ~~A~~ ~~direct~~ special control unit is used to transfer block of data directly between an external device and main memory, without intervention by the processor. This approach is called direct memory access.

34. Kernel I/O Subsystems is responsible to provide many services related to I/O. Following are some of the services provided.

- Scheduling:- Kernel schedules a ~~set~~ set of I/O request to determine a good order in which to execute them. When an application issues a blocking of I/O system call, the request is placed on the queue for that device. The kernal I/O schedular rearranges the order of the queue to improve the overall system efficiency and the average response time experienced by the applications.

- Buffering:- Kernel I/O Subsystem maintains a memory area known as buffer that stores data while they are transferred between two devices or between a device with an application operation

- **Caching:-** Kernel maintains cache memory which is region of fast memory that holds copies of data. Access to the cached copy is more efficient than access to the original.

- **Spooling and device reservation:-** A spool is a buffer that holds output for a device, such as a printer, that cannot accept interleaved data streams. The spooling system copies the queued spool files to the printer one at a time.

- **Error handling:-** An operating system that uses protected memory can guard against many kinds of hardware and application errors.

35. **Authentication:-** Authentication refers to identifying each user of the system and associating the executing program with those user. It is the responsibility of operating system to create a protection system which ensures that a user who is running a particular program is authentic.

Program threat:- Operating system's processes and kernel do the designated task as instructed. If a user program mode these process do malicious tasks then it is known as program threat.

System threat:- They refer to misuse of system ~~system~~ services and network connections to put user in trouble. System threats can be used to launch program attack. System threats creats such an environment that operating system resources/files are misused.