Operating system

Part Review: Some questions

Review – Computations

- **CPU** Scheduling
  - **FCFS**, **SJF**, SRJF, RR, Priority, MLQ, MLFQ, Lottery, HRRN
- **Synchronization**
  - Semaphore based problems
  - Banker's algorithm
- **MM** replacement algorithms
  - **FCFS**, **Optimal** (MIN), LRU, Clock (2^{nd} chance), LFU, MFU
  - MM Placement algorithms: Best, Worst, 1^{st}, Next
- **Disk** Scheduling
  - **FCFS**, **SSTF** (Shortest seek time first), Scan, C-Scan, Look, C-Look
- **File** system
  - blocks needed, File size

Typical questions

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

- Consider the following set of processes, with the length of the CPU-burst time given in milliseconds:
- The processes are assumed to have arrived in the order P_1, P_2, P_3, P_4, P_5, all at time 0.
  1. Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, a non-preemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1) scheduling.
  2. What is the turnaround time of each process for each of the scheduling algorithms in 1?
  3. What is the waiting time of each process for each of the scheduling algorithms in 1?
  4. Which of the schedules in part a results in the minimal average waiting time (over all processes)?

- A multiprogramming system has one CPU and two IO devices – IO1 and IO2. The competition for CPU follows the **preemptive priority** scheme, which means the job with higher priority could interrupt the running job with lower priority and get the CPU. While, the **competition for IO devices is non-preemptive**.
- Now there are three jobs – J1, J2 and J3 – stored in memory, and they arrive at same time. J1 has the highest priority and J3 has the lowest priority.
- Their ordered requests for CPU and IO devices are listed as follows:
  - **J1: IO2(30ms), CPU(10ms), IO1(30ms), CPU(10ms).**
  - **J2: IO1(20ms), CPU(20ms), IO2(40ms).**
  - **J3: CPU(30ms), IO1(20ms).**
- Please draw the **Gantt diagram** and answer the following questions:
  1. Compute the turnaround time of those three jobs;
  2. Compute the CPU utilization rate after all three jobs are finished;
  3. Compute the utilization rate of IO1 after all three jobs are finished.

- There are 4 processes sharing 18 resources. For dynamic avoid deadlock, the number of resources which every process could request at most is ()

  A）4　　B）5　　C）6　　D）7

- When using a counting semaphore to control the usage of 5 printers shared among many processes, which value could not occur for the counting semaphore?

  A. -6 B.-5 C. 5 D. 6

General rules to cope with CS problem using semaphores

1. Find the types of actors
   - To determine the **processes**
2. Recognize the shared resources between actors
3. Infer the constraints based on the situations when actors use those shared resources
   - **ME or SCH**?
     - To determine semaphores and their initial values
     - To determine the code (nested for ME, and scattered for SCH)
4. Use semaphores to finish those processes

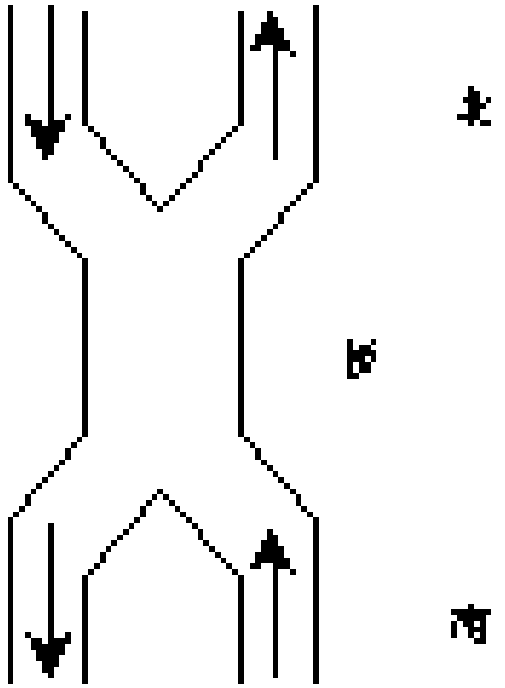
- Readers-Writers Problem is a quite popular synchronization problem in OS. One of its typical sub-problems is the **First Readers-Writers Problem**, in which readers have priority over writers. That is, unless a writer has permission to access the object, any reader requesting access to the object will get it. (Note this may result in a writer waiting indefinitely to access the object, AKA starvation.) Following is an unfinished pseudo code for this problem, please complete it.

```
Begin
 readcountmutex, wmutex: semaphore;  // 两个互斥信号量
 readcount: Integer;                 // 面向Reader的计数器
 readcountmutex = wmutex = 1;
 rcount = 0;

 Cobegin
  Process procedure Reader
   begin
    repeat
     // do something
     P(readcountmutex);
     _____________________;
     if (readcount ==1) then P(wmutex);
     V(readcountmutex);
     perform read operations;
     P(_____________________);
     readcount:= readcount - 1;
     if (_____________________) then V(_____________________);
     V(_____________________);
     // do something else
    until false;
   end
```

```
  Process procedure Writer
      begin
       repeat
        P(wmutex);
        perform write operations;
        V(wmutex);
       until false
      end
  CoEnd
End
```

- 10.Figure 1 illustrates a bridge, and the arrows show the directions of the corresponding cars. Only one car is allowed on the bridge at any time, but several cars are allowed to pass the bridge one by one if they are for the same direction. You are required to fill the blanks in following code which is used to cope with this synchronization problem using P and V operations



Var integer mutex， availn，avails;

availn = m;
avails =0;
mutex = 1;

Wrong

Var integer mutex， availn， avails;

// availn (for North) and avails (for South) are semaphores to synchronize the passing cars

availn = m; // **when it's >0, it means only the cars from this direction could pass the bridge**

avails =0; // **when it's 0, it means only the cars from this direction are forbidden to pass the bridge**

mutex = 1; // mutual exclusion for accessing the bridge

COBEGIN

Car for South:

BEGIN

P(_____);

P(_____);

Cross the bridge;

V(_____);

V(_____);

END;

Car for North:

BEGIN

P(_____);

P(_____);

Cross the bridge;

V(_____);

V(_____);

END

- Assume that the main memory is organized using pure paging. The page table (uncompleted) is shown as Figure 2.

Page table	
0	4
1	
2	
3	3
4	1

For each of the following decimal logical addresses, the **physical addresses** according to them are shown as follows.

logical addresses	physical addresses
0	2048
600	88
1024	0

(1) Compute the page size.

(2) Fulfill the page table.

(3) For each of the following decimal logical addresses, compute the **physical addresses** according to the page table.

(3-1) 1600

(3-2) 2500

An OS uses demand paging system in memory management. Assume the capacity of the main memory which a process could be allocated to is 300 byte, which is divided into 3 frames. The process will access the following Logical address byte series: 115, 228, 120, 88, 446, 102, 321, 432, 260, 167（Attention: 115B is only equivalent to 1 page）.

Please Answer：

(a) Writing down the page-reference string.

(b) Analyze the page replacement situation and calculate the page fault frequency when LRU and FIFO algorithm is used

Disk scheduling algorithms

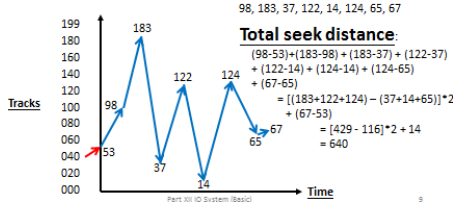
- We illustrate them with a request queue (0-199).
  - 98, 183, 37, 122, 14, 124, 65, 67
  - After visiting 40, current Head pointer is at 53
- FCFS, SSTF, SCAN, C-SCAN, LOOK, C-LOOK

FCFS [先来先服务算法]

- ***First come, first serve (FCFS):*** requests are served in the order of arrival

  \+ Fair among requesters

  \- Poor for accesses to random disk blocks

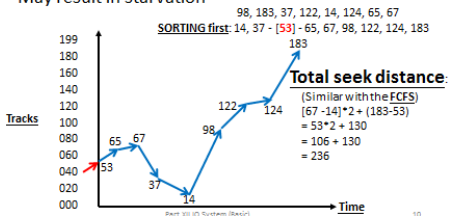


SSTF [最短寻道时间优先]

- ***Shortest seek time first (SSTF):*** picks the request that is closest to the current disk arm position

  \+ Good at reducing seeks

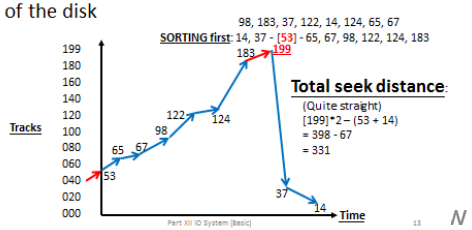
  \- May result in starvation



SCAN

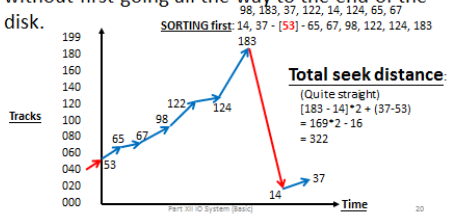
- ***SCAN:*** takes the closest request in the direction of travel (an example of elevator algorithm)

  \- a new request can wait for almost two full scans of the disk



C-LOOK

- Variation of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.



N for

Typical questions for file system + Device management

- A hard disk has 40G, Its each block size is 1K，and each table entry of FAT needs 20 bits，then Its FAT (File Allocation Table) need （ ） memory space
  - A）100M　　B）120M　　C）140M　　D）160M
  - How about using bit map?

- Here is a file system, which adopts multi-level index structure to support the search some records in a file. The block size is 512 bytes, and 3 bytes are used to represent the block number. If the cost to the logic block number in a physical block is not considered, please figure out the largest size of a file when using 2-level and 3-level index structures.

12. Given a hard disk whose size is 500GB (We use 1GB=2^{30}B here), and the block size is 8KB. 32 bytes are used to locate one block. Please answer following questions: (10 points)

a. If bit map (bit vector) is used to manage the hard disk which should be also stored in hard disk, what's the number of blocks used to store that bit map?

b. Now FAT (File Allocation Table) is used to the blocks of the hard disk, and the pointer size to locate a block is 32 bytes. If the blocks for the FAT itself are connected following linked list idea as Fig. 2 (each pointer uses 32 bytes), how many blocks should be used to store the FAT?

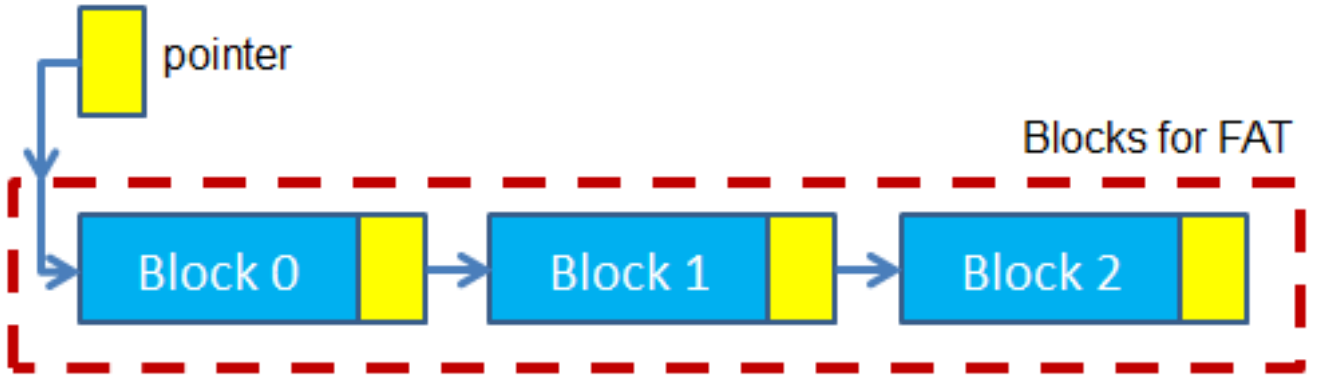


Fig. 1 Linked list idea to organize the data blocks for a file

c. Now we use 2-level index model to organize the data of a file as Fig. 3, and each index table (inner or outer) can only occupy one block at most. What's the largest size of a file following this kind of organization?

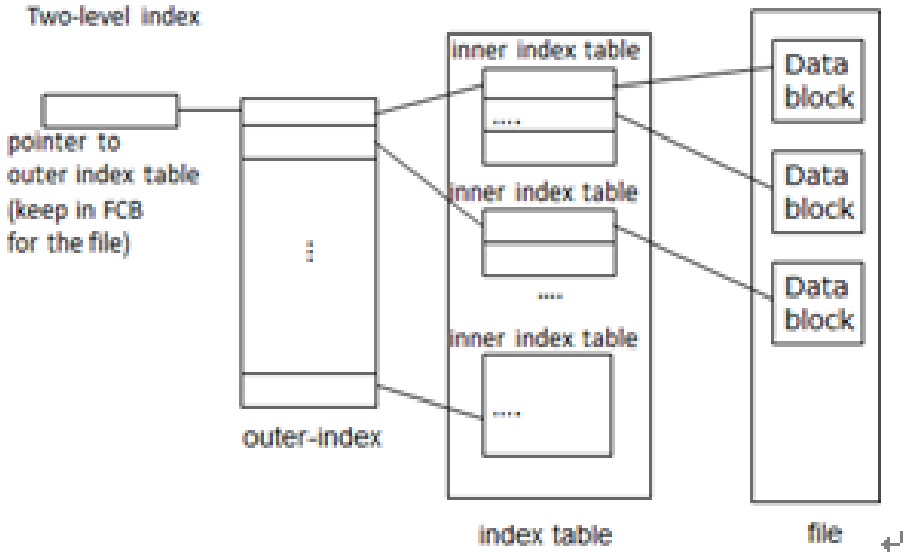


Fig. 3 2-level indexed structure to organized the data blocks for a file

d. When using i-node (shown in Fig. 4) to organize the data or a file, we use all the bytes of one block to store an i-node. In that i-node, 5KB is used to represent the attributes of the file, and the rest bytes are used for indirect pointers. If the rest bytes are evenly assigned to those 3 indirect pointers (namely, for instance, if we have 12*32 bytes, single indirect type uses 4*32 bytes, double indirect type uses 4*32 bytes, and triple indirect types uses 4*32 bytes), What's the largest size of a file following this kind of organization?