UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2004

BSc Honours Degree in Mathematics and Computer Science Part II
MSci Honours Degree in Mathematics and Computer Science Part II
for Internal Students of the Imperial College of Science, Technology and Medicine

*This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science*

# PAPER MC114

# OPERATING SYSTEMS

Tuesday 4 May 2004, 14:30
Duration: 120 minutes

*Answer THREE questions*

Paper contains 4 questions
Calculators required

***Section A*** *(Use a separate answer book for this Section)*

1a Briefly explain the concept of cylinder skew.

b How much cylinder skew is needed for a 6,000-rpm disk with a track-to-track seek time of 2 msec and 200 sectors per track?

c A request arrives at the I/O module to read a block on cylinder 15. While the seek to cylinder 15 is in progress, new read requests come in for cylinders 1, 36, 16, 34, 9 and 12, in that order. They are entered into a table of requests.

i) What data structure would you use to represent this list of requests and why?
ii) Using the FCFS algorithm, calculate the total number of cylinders traversed to read the data.
iii) Using the SSF algorithm, calculate the total number of the cylinders traversed to read the data.
iv) Using the Elevator algorithm with SSF, calculate the total number of cylinders traversed to read the data.

d Outline a good scheduling algorithm for a disk in which the seek time is much faster than the rotational delay.

*The four parts carry, respectively,* 10%, 20%, 55%, and 15% *of the marks.*

2 Consider the following pseudocode for a producer-consumer system:

```
const int N=100;
int count = 0;
```

```
void producer()
{
    int item;

    while (TRUE) {
        item = produce_item();
        if (count==N) sleep();
        insert_item(item);
        count = count + 1;
        if (count == 1)
            wakeup(consumer);
    }
}
```

```
void consumer()
{
    int item;

    while (TRUE) {
        if (count == 0) sleep();
        item = remove_item();
        count = count - 1;
        if (count == N-1)
            wakeup(producer);
        consume_item(item);
    }
}
```

Assume `sleep` and `wakeup` are system calls to stop the code processing and resume its processing again respectively. Further, `insert_item` and `remove_item` are routines that place and extract items to and from a buffer data structure held in memory. Bear in mind that both processes can be preempted.

a Briefly describe an execution scenario to illustrate a major problem with the code as implemented above.

b Show how would you rewrite this code using semaphores to protect the buffer.

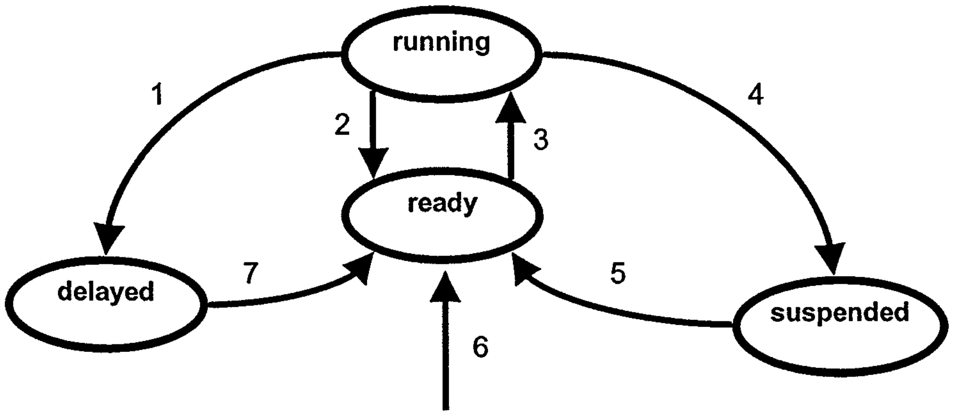
c How would the execution of the code in (a) differ if it were executed on a shared-memory multi-processor (that is, two CPU's sharing a common memory)?

d Suppose processes communicate with each other using a message-passing system that uses mailboxes. When sending to a full mailbox or trying to receive from an empty one, a process does not block. Instead it receives an error-code. The process responds to the error code by reattempting the operation until it succeeds. Briefly, comment on this scheme in terms of potential for data loss and starvation.

*The four parts carry, respectively 25 % of the marks each.*

*Section B (Use a separate answer book for this Section)*

3a In the Simple Kernel operating system, each process can be in one of four states. The diagram below shows these process states, together with the possible transitions between states, which have been numbered to help you refer to them.



i) Briefly describe three different scenarios that can lead to a running process being de-scheduled.

ii) Which process state transitions can be caused by a process calling, respectively, the P() and V() operations on a semaphore?

b A large university computer science department uses the same operating system across all its machines. This OS allows several key parameters of its scheduling algorithm to be tuned on a per-machine basis in order to adjust to varying requirements. You have been asked to help with tuning these parameters for two machines: `sharedserver`, which is a machine intended for many users to log in, read email and edit a few files and `numbercruncher`, which is a machine intended for one researcher to run some long-running CPU-intensive simulation jobs which occasionally perform a lot of disk I/O due to paging.

For each of the following parameters of the scheduling algorithm, state which you would choose, for both `sharedserver` and `numbercruncher`, in order to make the best possible use of the machines and keep their users happy.

i) You can choose the time slice assigned to a process when it is selected to run: short (10ms), medium (100ms) or long (500ms).

ii) You can choose the policy for assigning priorities to processes: either static or dynamic. The static option is similar to that used in Simple Kernel. The dynamic priority option reduces the priority of processes that have recently used a lot of CPU time, as in traditional UNIX scheduling.

iii) You can choose the priority assigned to system processes handling file I/O: high or low, assuming user processes have a default priority of medium.

*The parts carry, respectively, (30%, 20%) and (10%, 20%, 20%) of the marks.*

4 This question concerns memory management.

a Consider a dynamic partition storage management system which allocates each process a contiguous region of physical memory corresponding exactly to the amount of memory the process requires for its code and data.

i) State three problems of this dynamic partition memory management system that are overcome by using paging.

ii) State two ways in which paging incurs an overhead compared with the memory management system described above.

b Consider a paged memory management system with 32-bit virtual addresses, 32-bit physical addresses and a page size of 4MB. How much memory is required for holding the page table? State any assumptions you have to make.

c Assume that the following program fragment is running on a computer system using paging, with a word size of 4 bytes (i.e. integers are 4 bytes), a page size of 4KB and 8MB of available physical memory.

We represent a drawing in 2D space by a collection of points and other data, such as edges connecting the points. Points consist of two integer co-ordinates x and y and we assume that the other data can be held in an array of 14 integers. The program fragment you are asked to consider is a loop which repeatedly modifies the x and y co-ordinates but leaves the other data unchanged.

```
/*  The type point is a structure consisting of
    integers x, y and other data. */
typedef struct {
  int x;
  int y;
  int otherdata[14];
} point;

/* Declare an array of points. /
point array[(1024*1024)];

for( int count = 0; count < 100; count++ ) {
  for( int i = 0; i < (1024*1024); i++ ) {
    array[i].x = /* new value */;
    array[i].y = /* new value */;
  }
}
```

Explain briefly why this program performs very poorly and suggest a simple change to this program that would significantly improve its performance.

*The parts carry, respectively, (30%,20%), 20% and 30% of the marks.*