74

RAJARATA UNIVERSITY OF SRI LANKA

FACULTY OF APPLIED SCIENCES, MIHINTALE

B.Sc. in Information and Communication Technology

Fourth Year Semester I Examinations –Oct./Nov 2015

ICT 4405 - Parallel and Cluster Computing

Time: THREE(03) hours

Answer ALL questions.

Instructions

1. Time allowed 3 hours.
2. Answer **all** questions.
3. The maximum attainable mark for each question is given in brackets.
4. Start answering each of the main questions on a new page.
5. This examination accounts for 50% of the module assessment.
6. This is a closed book examination.
7. Assume reasonable values for any data not given in or with the examination paper. Clearly state such assumptions made on the script.
8. In case of any doubt as to the interpretation of the wording of a question, make suitable assumptions and clearly state them on the script.

Question 1 (25 marks)

(i) State whether the following statements are TRUE or FALSE. Give one sentence justification for your answer.

Write the corresponding sub question number and the answer in your answer book.

[2 × 5]

a) It is possible for one process/thread to lock the mutex and for another process/thread to unlock it.

b) When all the threads/processes are blocked it is called a deadlock.

c) When a problem is parallelized at data level, large data units could lead to thrashing.

d) During the process of Parallel Algorithm design, Partitioning phase decomposes computation/data into small tasks/chunks while considering available no of CPUs and memory.

e) "Fat Tree" is one of the techniques to interconnect a large cluster of computing nodes.

(ii) Recommend a suitable solution pattern to parallelize the following code snippets. Provide a suitable justification for each case. State any assumptions.

a)

```
for(k = 1, k < 500; k++){
     x[k] = y[k - 1] + 1;
     y[k] = z[k - 1] + 2;
}
```

[3]

b)

```
x = readData("xIn.txt");
y = readData("yIn.txt");
for(k = 1, k < x.getSize()(; k++){
     x[k] = x[k] + y[k - 1];
}
writeData("xOut.txt", x);
```

[4]

(iii) The following table shows the current allocation of resources for 3 processes (*P*, *Q*, and *R*) and their maximum resource requirements. Is the current state is safe or unsafe? Show the steps. [3]

	Has	Max
P	2	7
Q	1	4
R	2	5

Free: 3

(iv) Are the use of following 2 methods free from deadlocks (i.e., when you use them is there any risk of a deadlock)? Discuss. [5]

```
public void method1() {
    synchronized (String.class) {
        System.out.println("Aquired lock on String object");

        synchronized (Integer.class) {
            System.out.println("Aquired lock on Integer object");
        }
    }
}

public void method2() {
    synchronized (Integer.class) {
        System.out.println("Aquired lock on Integer object");

        synchronized (String.class) {
            System.out.println("Aquired lock on String object");
        }
    }
}
```

77

Question 2 (25 marks)

(i) Suppose a weather forecasting program typically takes **24 hours** to produce tomorrow's weather forecast. Therefore, the Meteorology department is forced to run it sufficient time ahead and only once a day. As more severe weather events are reported recently, meteorology department is interested in producing weather forecasts more frequently and with less turnaround time. They are thinking of achieving this by benefiting from more advanced hardware and software. Before deciding to upgrade their computing facility, the meteorology department wants your feedback on the following concerns they have.

a) "We want to provide a new weather forecast every **6 hours**. How many CPU cores are required, if we go for an *n*-core, shared memory design?"

It was noted that a fraction ***f*** of the weather forecasting program cannot be parallelized. ***f*** accounts for 30% of the program's execution time. The remaining code ***p*** is parallelized.

Hint: Amdahl's law in the context of concurrent programming is given as:

$$\frac{1}{1-p+\frac{p}{n}}$$

[2]

b) "Should we upgrade our server to *n*-cores or should we replace the server with a cluster of nodes having a total of *n*-cores?"

Suggest your recommendation while considering the advantages and disadvantages of each approach. State any assumptions. [3]

c) "As the modern GPUs provide much better performance, power, and cost advantage, should we buy a high-end GPU card than upgrading our server to *n*-cores?"

Suggest your recommendation while considering the advantages and disadvantages of each approach. State any assumptions. [3]

(ii) Consider the following program with 2 threads.

```
int i;
Thread 1                                  Thread 2
  i = 0;                                  i = 0;
  i++;                                    i--;
  printf("%d ", i);                       printf("%d ", i);
```

a) Provide 4 possible outcomes of the above program. [4]

b) Following solution based on a Condition Variable is suggested as a solution to make sure that the above program generates only "1, -1" as the output.

```
int i;
CV;
Lock l;
Boolean done = False;

Thread 1                                 Thread 2

    i = 0;                                   l.lock();
    i++;                                     if( !done )
    printf ("%d ", i);                           cv.wait()
    l.lock();                                l.unlock()
        done = True;                         i = 0;
    l.unlock();                              i--;
    cv.notify();                             printf ("%d ", i);
```

c) Briefly explain the functionality of this program and discuss whether the given code satisfy safety and liveness properties while attempting to print "1, -1". [8]

d) Give a semaphore-based solution that makes sure the above program generates only "1, -1" as the output. [5]

Question 3 (25 marks)

(i) Give an example for each of the following cases where the particular implementation of readers and writers solution becomes useful.

a) Readers-writers solution that gives priority to readers. [3]

b) Readers-writers solution that gives priority to writers. [3]

(ii) Static or dynamic load balancing is essential in most systems to increase the resource utilization. What type of load balancing would you recommend for the following problems?

a) A Producer-Consumer problem where the producer continues to generates a random integer i and multiple consumers are used to calculate the Fibonacci sequence of a given integer i (i.e., FIB(i)). [3]

b) Calculating the area under a given curve (i.e., integration) by breaking it into a large set of trapezoids. [3]

(iii) n-body interaction is one of the most common simulations run using GPUs. It can be used to simulate movement of objects such as planets during the Big Bang. For example, given the masses and locations of planets following equation can be used to calculate the forces that a planet i experience due to another planet j.

$$f_{i,j} = \frac{Gm_i m_j}{d_{i,j}^2}$$

Where G is the gravitational constant, m_i and m_j are masses of the 2 planets, and $d_{i,j}$ is the distance between 2 planets. These pairwise forces can be represented as a matrix F.

$$F = \begin{bmatrix} 0 & f_{0,1} & \dots & f_{0,n-1} & f_{0,n} \\ f_{1,0} & 0 & \dots & f_{1,n-1} & f_{1,n} \\ \dots & \dots & \dots & \dots & \dots \\ f_{n-1,0} & f_{n-1,1} & \dots & 0 & f_{n-1,n} \\ f_{n,0} & f_{n,1} & \dots & f_{n,n-1} & 0 \end{bmatrix}$$

Where n is the number of planets. Once the force matrix is calculated, it can be used to calculate the acceleration of each planet using Newton's second law (i.e., $F = ma$). Which intern can be used to calculate the velocities and new locations of planets after a given time t. Then this process can be repeated again and again to calculate the location of planets at time t, $2t$, $3t$, and so on.

a) Outline a CUDA kernel to calculate the force matrix F. Your solution should also include the code required to invocate the Kernel function. Assume n = 1,000,000. [10]

Hint: a typical CUDA supported GPU can only handle 1,024 threads per bloc

b) Is it worthwhile to calculate both $f_{i,j}$ and $f_{j,i}$ in a GPU? Briefly discuss. [3]

Question 4 (25 marks)

(i) The Root Mean Square (RMS) is one of the several kinds of averages. It is often used in many engineering and statistical applications, e.g., in electrical engineering. RMS of n real numbers $x_1, x_2, x_3, \ldots x_n$ can be calculated as follows:

$$RMS(X) = \sqrt{\frac{\sum_{i \in n} x_i^2}{n}} = \sqrt{\frac{x_1^2 + x_2^2 + x_3^2 + \ldots + x_n^2}{n}}$$

Outline an MPI program (using pseudo code) that can be used to calculate the RMS of one million real numbers. Once the calculation is complete, mean should be stored on a variable at process 0. [12]

Use relevant MPI functions that are given in the Appendix. Note that it is impractical to create one million concurrent processes/threads.

(ii) A cluster is parallel or distributed processing system with a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource". [7]

Using a suitable diagram and a real-world example discuss the architecture of a typical cluster. You are required to illustrate the key hardware, software, and networking components.

(iii) Briefly discuss how middleware is able to provide the following layers in cluster computing.

a) Single System Image (SSI) [3]

b) Availability [3]

81

Appendix – MPI Functions

```
int MPI_Init(int *argc, char **argv)

int MPI_Comm_size(MPI_Comm comm, int *size)

int MPI_Comm_rank(MPI_Comm comm, int *rank)

int MPI_Finalize()

int MPI_Send (void *buf,int count, MPI_Datatype datatype, int dest, int
        tag, MPI_Comm comm)

int MPI_Recv (void *buf,int count, MPI_Datatype datatype, int source, int
        tag, MPI_Comm comm, MPI_Status *status)

int MPI_Reduce(void *sendbuf, void *recvbuf, int count, MPI_Datatype
        datatype, MPI_Op op, int root, MPI_Comm comm)

int MPI_Allgather(void *sendbuf, int sendcount, MPI_Datatype sendtype, void
        *recvbuf, int recvcount, MPI_Datatype recvtype, MPI_Comm comm)

int MPI_Allreduce (void *sendbuf, void *recvbuf, int count, MPI_Datatype
        datatype, MPI_Op op, MPI_Comm comm)

int MPI_Bcast( void *buffer, int count, MPI_Datatype datatype, int root,
        MPI_Comm comm)

int MPI_Gather(void *sendbuf, int sendcnt, MPI_Datatype sendtype, void
        *recvbuf, int recvcnt, MPI_Datatype recvtype, int root,
        MPI_Comm comm)

int MPI_Scatter(void *sendbuf, int sendcnt, MPI_Datatype sendtype, void
        *recvbuf, int recvcnt, MPI_Datatype recvtype, int root,
        MPI_Comm comm)
```

Operation Value	Meaning
MPI_MAX	Maximum
MPI_MIN	Minimum
MPI_SUM	Sum
MPI_PROD	Product
MPI_LAND	Logical and
MPI_BAND	Bitwise and
MPI_LOR	Logical or
MPI_BOR	Bitwise or
MPI_LXOR	Logical exclusive or
MPI_BXOR	Bitwise exclusive or
MPI_MAXLOC	Maximum and location of maximum
MPI_MINLOC	Minimum and location of minimum

------------------------ END OF THE PAPER ------------------------