CARDIFF UNIVERSITY
EXAMINATION PAPER

Academic Year:	2018/2019
Examination Period:	Autumn
Examination Paper Number:	CMT106
Examination Paper Title:	High Performance Computing
Duration:	TWO hours

Do not turn this page over until instructed to do so by the Senior Invigilator.

Structure of Examination Paper:

There are 4 pages.
There are 4 questions in total.
There are no appendices.

The maximum mark for the examination paper is 60 and the mark obtainable for a question or part of a question is shown in brackets alongside the question.

Students to be provided with:

The following items of stationery are to be provided:
ONE answer book.

Instructions to Students:

Answer THREE questions.

Important note: if you answer more than the number of questions instructed, then answers will be marked in the order they appear only until the above instruction is met. Extra answers will be ignored. Clearly cancel any answers not intended for marking. Write clearly on the front of the answer book the numbers of the answers to be marked.

The use of translation dictionaries between English or Welsh and a foreign language bearing an appropriate school stamp is permitted in this examination.

Q1. (a) Describe the differences between static, dynamic, and guided scheduling of a parallel *for* loop in OpenMP. [6]

(b) Give one reason why guided scheduling might be preferable to dynamic scheduling.[2]

(c) Consider the following fragment of OpenMP code, in which B is a one-dimensional integer array of size 8000 that has already been initialised to values in the range 0 to 1000, inclusive:

```
int chunk = 500;
int sum=0;
#pragma omp parallel shared(B,sum,chunk) num_threads(6)
{
        #pragma omp for schedule(static,chunk)
        for (int i=0; i < 8000; i++) sum = sum + B[i];
}
```

(i) Explain why the above code will not reliably determine the sum of the integers in array B. [2]

(ii) Modify the above code to correctly determine *sum*. [2]

(d) Consider the following piece of OpenMP code:

```
int n=15, chunk=12, i, j;
float z[n][n], y[n][n], zji;
#pragma omp parallel shared(default) private(zji,i,j) num_threads(6)
{
    #pragma omp for collapse(2) schedule(static,chunk)
        for(j=0;j<n;j++)
            for(i=0;i<n;i++){
                zji = z[j][i];
                y[j][i] = zji*(2.0*zji+3.0)+5.0;
            }
}
```

(i) Which elements of the array y will be evaluated by thread 3? [4]

(ii) If the *collapse* clause is removed from the code, which thread will evaluate y[8][10]? [2]

(e) Explain the main difference between a *private* and a *threadprivate* variable in OpenMP. [2]

Q2. (a) Describe the differences in scope and lifetime of variables stored in global, shared, and register memory on a CUDA device. [6]

(b) Explain three ways in which a host computer for a CUDA device coordinates and controls the actions of the device. [6]

(c) In the CUDA programming model, explain why threads in the same block can be synchronised, but not threads in different blocks. [2]

(d) The array B is of size 1128×2462. In a CUDA program, each element of B is evaluated by a single thread. A two-dimensional grid of threads is set up using thread blocks of size 16×32.

(i) What is the minimum number of blocks required in each of the two dimensions? [4]

(ii) How many threads will be inactive? [2]

Q3. (a) Explain carefully the difference between a *non-blocking send* and an *asynchronous send* in MPI. [4]

(b) In point-to-point communications, MPI receive routines have the address of a return status object as one of the arguments. Describe two different uses of a return status object in an MPI program. [2]

(c) What is meant by an *application topology*, and how are Cartesian mesh application topologies supported in MPI? [4]

(d) Describe the main sources of overhead in a message-passing parallel program running on a distributed memory computer. [6]

(e) A researcher writes an MPI program and runs it for a fixed problem size with different numbers of processes. She finds that as the number of processes increases from 1 the execution time initially decreases, but then reaches a minimum before starting to rise. Give a plausible explanation for this variation in execution time as the number of processes increases. [4]

Q4. (a) Consider the following piece of Occam code:

```
CHAN OF INT chan1, chan2 :
  PAR
    INT x :
    SEQ
      chan1 ! 2
      chan2 ? x
    INT s :
    SEQ
      chan2 ! 4
      chan1 ? s
```

(i) Describe the structure of this piece of code in terms of the processes and communication involved. [4]

(ii) What would be the outcome of executing this piece of code? Justify your answer. [3]

(b) A regular 4×4 mesh is mapped onto a 4-dimensional hypercube so that neighbouring nodes in the mesh are also neighbours in the hypercube. Determine the node number of the hypercube node at location $(3, 1)$ in the mesh. [4]

(c) Define metrics for assessing the resiliency and congestion in an interconnection network, and evaluate them for a $k \times k$ mesh network. [5]

(d) A parallel program has an execution time of:

$$T_{par}(M, N) = \frac{2M^3}{N^3} t_{calc} + \frac{M^2 \log_2 N}{N^2} t_{comm}$$

where M is the problem size, N^2 is the number of processors, t_{calc} is the time to perform one floating-point calculation, and t_{comm} is the time to communicate one floating-point number. Determine the self-speedup of the program, and comment of its scalability.[4]