Name: _____ **Student ID**: _____

Indian Institute of Technology Jammu
Major Examination
B.Tech VII Semester
November 27, 2019

Course Title: Intro to Parallel & Distributed Programming **Maximum Time: 2 Hr**
Course Code: CSL380 **Maximum Marks: 100**

Instructions:

- Conditions of Examination: Closed book; No dictionary; Non-programmable calculator is allowed.
- Use of mobile phones is strictly prohibited.

---

1. (a) **[2 marks]** What is the maximum collapse level that can be specified with the #pragma omp for in the following code in order to give the same result as the sequential code ?

```
#pragma omp for ..
for(i=0;i<100;i++)
  for(j=0;j<100;j++)
    for(k=10;k<100;k++)
      for(l=10;l<100;l++)
        arr[i][j][k][l] = arr[i][j][k][l] + (k-2) ;
```

   (a) collapse(1)
   (b) collapse(2)
   (c) collapse(3)
   (d) collapse(4)

   (b) **[2 marks]** Consider the following code (2 threads).

Thread 1:	Thread 2:
a=1;	b=3;
b=2;	a=4;

   Which of the following cannot be the final values of shared variables a and b under the sequential consistency model ?

   (a) a=1, b=2
   (b) a=4, b=3
   (c) a=1, b=3
   (d) All the specified options can be the final values

   (c) **[2 marks]** What is wrong with the following usage of locks assuming that the lock has been defined and initialized appropriately ?

```
omp_set_lock(&lock);
count++ ;
omp_unset_lock(&lock);
```

   (a) omp_set_lock and omp_unset_lock should be put in critical sections as lock is shared
   (b) flush should be introduced before and after omp_set_lock and omp_unset_lock

(c) both (a) and (b)
(d) There is nothing wrong with the code

(d) **[2 marks]** Consider the following program. Assume that OMP_NUM_THREADS = 4

```
#include <omp.h>
#include <stdio.h>
void main( int argc, char **argv )
{
   int x[4] ;
   x[0]=x[1]=x[2]=x[3]=0 ;

   #pragma omp parallel
   {
      int i ;
      int t = omp_get_thread_num() ;

      #pragma omp for schedule(static, 20)
      for( i = 0 ; i < 50 ; i++ )
         x[t]++ ;

      #pragma omp single
      printf( "%d", x[t] ) ;
   }
}
```

Then the output cannot be:

(a) 0
(b) 10
(c) 20
(d) 50

2. **[14 marks]** Consider a binary tree interconnect wherein for any node with rank j, its children nodes are the nodes with ranks $2 * j + 1$ and $2 * j + 2$. Assume that the total number of nodes, n, is $2^q - 1$ for some integer q, so that the interconnect is a complete binary tree. Write an efficient MPI program to implement the allreduce collective using sends and receives in 2 phases: In the first phase, data is reduced to the root (node 0) and in the second phase, data is broadcasted back to all the nodes from the root. (Carefully consider where you should use non-blocking calls).

3. (a) **[2 marks]** What are the fundamental differences between CPU and GPU architectures ?
   (b) **[2 marks]** What are the implications on CUDA / GPU programming due to these differences ?
   (c) **[2 marks]** Why is the memory bandwidth important in the design of GPGPU programs ?
   (d) **[2 marks]** What is thread divergence in GPU / CUDA programming ? Give one example along with the explanation on GPU execution of such code block.
   (e) **[3 marks]** In CUDA, how do we declare / define a variable in private memory, shared memory and global memory in GPU ? Give examples only.

4. (a) **[2+2+2 marks]** Write either OpenCL **or** CUDA kernels for matrix addition ($C = A + B$) in *three* different ways: 1 thread per element, 1 thread per row, and 1 thread per column. Here A and B are two square matrices of size $n \times n$. The matrices are given as $1D$ arrays (i.e., $A[0..n^2 - 1]$ etc.).
   (b) **[3 marks]** Also write the invocation of the kernels for each case from CPU or main thread.

5. **[20 marks]** Write a CUDA program to reverse a string of size n in place (i.e., without using some other memory spaces for reversing). Clearly write all portions including the CUDA kernel(s) for reversing the string, the main() function for the CPU portion.

6. (a) **[(1+1+1)+1 marks]** Define *fault*, *error* and *failure* in computing systems ? Also state their inherent relationship in one or two sentences.

   (b) **[2 marks]** Write the differences between *permanent* and *transient* hardware faults ?

   (c) **[3+2 marks]** Describe a method (with diagram if require) to achieve hardware fault-tolerance in computing systems ? Also mention the assumptions and/or the deficiencies of the method.

7. (a) **[3 marks]** Explain how checkpointing mechanism can bring fault-tolerance in computing systems.

   (b) **[2+2 marks]** Explain with example the *orphan* and *missing* messages with respect to distributed checkpointing.

8. (a) **[2 marks]** What is the Parallel Random Access Machine (PRAM) model ? (Please mention all the underlying assumptions).

   (b) **[2 marks]** What is the self-simulation property of the PRAM model ?

   (c) **[4 marks]** Show that the COMMON CRCW PRAM is self-simulating. (CRCW: Concurrent Read and Concurrent Write. An example with appropriate steps should be sufficient)

9. (a) Consider the problem of finding the largest number in an array of size n.

      (i) **[3+1 marks]** Write a parallel algorithm for the above problem. State the total number of steps in the proposed parallel algorithm.

      (ii) **[2 marks]** Compute the cost of the sequential algorithm and the above parallel algorithm.

      (iii) **[2 marks]** Find the number of processors that can provide a cost optimal execution / scheduling of your proposed parallel algorithm. Also briefly explain this cost optimal scheduling (with an example / diagram for $n > 100$).

   (b) Consider the problem of searching a number x in a sorted array A of size n. There is no duplicate number in the array A. Let us assume the number of processors is p. A parallel algorithm is designed as follows:

   Step The array A is divided into $p + 1$ parts and each part (except the last one) is assigned to one processor. The processor verifies whether x is within the part of the array (by comparing the values at the end-points). Only one part wins (Note: it can be the last one if none of the processor finds x is within its portion of array A). The winning part is copied at the front/beginning of A - and the new size of A is noted for next pass.

   The above **Step** is continued until the searching is complete.

      (i) **[2 marks]** Compute the number of steps and the cost of the above parallel algorithm.

      (ii) **[2 marks]** Is there any p for which the above parallel algorithm can give cost optimal execution / scheduling ? Justify your answer.