6. (a) Show step by step implementation of Odd-Even Sorting algorithm in the following sequence of elements [8, 2, 7, 3] using 4 processors. (4)

(b) Write the calling sequence of MPI_Sendrecv function and describe its parameters. What are the limitations of using MPISendrecv function in MPI programming and how can they be resolved? (5)

(c) Describe predefined reduction operations `MPI_MAXLOC` and `MPI_MINLOC`. Give an example for each. (6)

7. (a) Differentiate between uniform memory access (UMA) and non uniform memory access (NUMA) multicomputer. (4)

(b) Write the pseudocode for block matrix multiplication algorithm for two n X n matrices with a block size of $\frac{n}{q} \times \frac{n}{q}$. (5)

(c) How can loops be scheduled in OpenMP using the schedule clause? Explain with the help of its syntax. Also, explain the following scheduling strategies with suitable examples:

(i) Static Scheduling

(ii) Dynamic Scheduling (6)

(1500)

[This question paper contains 8 printed pages.]

Your Roll No.....

Sr. No. of Question Paper : 5567 **J**

Unique Paper Code : 2342013603

Name of the Paper : Introduction to Parallel Programming

Name of the Course : **B.Sc. (Hons.) Computer Science**

Semester : VI

Duration : 3 Hours Maximum Marks : 90

Instructions for Candidates

1. Write your Roll No. on the top immediately on receipt of this question paper.
2. **SECTION A** is compulsory.
3. Attempt **any 4** questions from **SECTION B**.
4. Parts of a question must be answered together.

P.T.O.

SECTION A

1. (a) Explain the fork/join parallelism with the help of a diagram. (3)

(b) Explain the Single Program Multiple Data (SPMD) model depicting its capabilities and the typical platforms that support it. (3)

(c) Suppose that a quad-core computer capable of running four processes at once (one process per core). Now, consider a program where 40% of the execution time is inherently sequential, while the remaining 60% can be divided into six equal parts that may be executed concurrently. What is the maximum speedup achievable when:

- Program is run using all four cores
- Program is run on a single core (3)

(d) Describe any two diverse applications of parallel computing. (4)

(e) Define the critical section in OpenMP. Differentiate between named and unnamed `#pragma omp critical` directives. (4)

(4)

(b) Consider the following OpenMP code segment: (5)

```
double area = 0.0, pi, x;
int i, n;
#pragma omp parallel for private(x, i)
for (i = 0; i < n; i++) {
    x = (i + 0.5) / n;
    area += 4.0 / (1.0 + x * x);
}
pi = area / n;
```

(i) Identify the race condition in the above code and explain why it occurs.

(ii) Modify the code to remove the race condition using a critical section.

(c) Write a parallel program using OpenMP to calculate the square of each element in an array using all three OpenMP data-sharing clauses: private, firstprivate and shared. Clearly specify which variables are associated with each clause and provide a justification for your choices. (6)

(c) Explain the following OpenMP functions : (6)

- `omp_get_num_procs`
- `omp_get_num_threads`
- `omp_get_thread_num`
- `omp_set_num_threads`

4. (a) Explain completely-connected network topology for parallel computers using an example. Why are sparser network topologies like linear arrays and meshes often preferred over completely-connected networks? (5)

(b) Explain Send and Receive Operations in message passing programming paradigm with an example. Describe how the idling overheads in blocking non-buffered send/receive communication can lead to deadlocks. (5)

(c) Write a program that prints out a "Hello World" message from each processor using the MPI functions MPI_Init, `MPI_Finalize`, `MPI_Comm_size` and `MPI_Comm_rank`. (5)

5. (a) Explain recursive task spawning in OpenMP and identify the common pitfalls associated with it.

(f) Consider a 5-stage pipelined processor with Instruction Fetch (IF), Instruction Decode (ID), Operand Decode (OD), Instruction Execute (IE), and Write Back (WB) stages. The processor has the ability to simultaneously issue two instructions in the same cycle. Draw the execution schedule for the following sequence of instructions executed by the processor. (4)

```
Load  R1,  @1000
Load  R2,  @1008
Add   R1,  @1004
Add   R2,  @100C
Add   R1,  R2
Store R1,  @2000
```

(g) What are threads? Rewrite the following code segment to provide a mechanism where threads can be scheduled on multiple processors. (5)

```
for (row = 0; row < n; row++)
  for (column = 0; column < n;
column++)
    c[row][column] =
```

```
dot_product(get_row(a,
row),get_col(b, col));
```

(h) Explain the structure of message passing programming paradigm. What are the two key attributes that characterize it? (4)

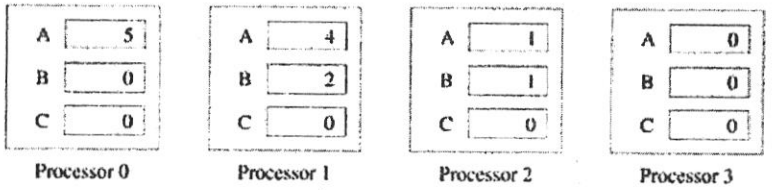
SECTION B

2. (a) Compare and contrast the multithreading and prefetching approaches used for hiding memory latency. (4)

(b) Explain the Single Instruction Stream Multiple Data Stream (SIMD) architecture using a diagram. Explain the following conditional statement executing in two steps on a SIMD computer with four processors. (5)

```
if (B==0)
   C=A;
else
   C=A/B;
```

Initial Values are as follows :



(c) Consider a system with a 1 GHz processor that uses DRAM with a 100 ns latency and a 32 KB cache with a latency of 1 ns (or one clock cycle). The task is to multiply two 32×32 matrices, A and B, where each element occupies 4 bytes (32 bits), and the CPU is capable of executing four instructions per cycle. Address the following :

(i) Compute the total number of floating-point operations needed to multiply matrices A and B.

(ii) Determine the peak computational throughput of the processor in GFLOPS. (6)

3. (a) Define the shared memory consistency model and the sequential consistency model in the context of compiler optimizations. (4)

(b) Write an OpenMP code segment for computing the cumulative sum of a list using the ordered directive. (5)