**CSCI-UA.0480-003**

# Parallel Computing

## Midterm Exam

## Spring 2015 (60 minutes)

**NAME:** **ID:**

- This exam contains **4 questions** with a total of 20 points in **4 pages.**
- The exam is open book/notes.
- If you have to make assumptions to continue solving a problem, state your assumptions clearly.

1. a) [2 points] State two advantages of multicore processors.

- Can reach higher performance without the high clock frequency (at its power consumption implication), assuming we have parallel programs.
- Can exploit different type of parallelism (given compiler and programmer support): task-level parallelism, data-level parallelism, … .

b) [2 points] State two advantages of single core processors.

- Easier to program
- Historically, with increasing clock frequency, can result in higher performance with no effort from the programmer.

c) [2 points] State two disadvantages of multicore processors.

- Hard to program.
- With increase in number of cores, there is contention on shared resources (e.g. shared cache, interconnect, memory controller, … ).

d) [2 points] State two disadvantages of single core processors.

- Does not scale anymore in terms of performance.
- Does no exploit other type of parallelism, such as task level parallelism.

2. [2 points] We have seen that a multicore processor is MIMD in Flynn's classification. Can a single core processor be anything else but SISD? If yes, give examples. If not, why not?

Yes, a superscalar processor and simultaneous multithreading architectures (i.e. processor with hyperthreading technology) can be considered MIMD.

3. [3 points] Describe 3 different scenarios where an MPI program can have a deadlock.

- A send call without the corresponding receive
- Collective calls not called by all processes of the communicator
- Deadlock due to out-of-order sends and receives

4. Suppose that MPI_COMM_WORLD consists of the three processes 0,1, and 2, and suppose the following code is executed (my_rank contains the rank of the executing process):

```
int x, y, z;

switch(my_rank) {

    case 0:
        x=0; y=1; z=2;
        MPI_Bcast(&x, 1, MPI_INT, 0, MPI_COMM_WORLD);
        MPI_Send(&y, 1, MPI_INT, 2, 43, MPI_COMM_WORLD);
        MPI_Bcast(&z, 1, MPI_INT, 1, MPI_COMM_WORLD);
        break;
    case 1:
        x=3; y=8; z=5;
        MPI_Bcast(&x, 1, MPI_INT, 0, MPI_COMM_WORLD);
        MPI_Bcast(&y, 1, MPI_INT, 1, MPI_COMM_WORLD);
        break;
    case 2:
        x=6; y=7; z=8;
        MPI_Bcast(&z, 1, MPI_INT, 0, MPI_COMM_WORLD);
        MPI_Recv(&x, 1, MPI_INT, 0, 43, MPI_COMM_WORLD, &status);
        MPI_Bcast(&y, 1, MPI_INT, 1, MPI_COMM_WORLD);
        break;
    }
```

a. [4 points] What will be the values of x, y, and z for each of the 3 processes after executing the above code?

| | **Po** | **P1** | **P2** |
|---|---|---|---|
| **X** | 0 | 0 | 1 |
| **Y** | 1 | 8 | 8 |
| **Z** | 8 | 5 | 0 |

b. [2 points] Is there a possibility that the communication among the 3 processes be executed out of order? If yes, explain the reason. If not, why not?

**No, because collective communication are blocking, and MPI_recv() is blocking.**

c. [1 point] What will happen if we execute the above code with: mpiexec –n 4 (and MPI_COMM_WORLD will then contain 4 processes)?

**The program will hang because the collective communication calls must be done by all the processes.**