

CSCI-UA.0480-003  
**Parallel Computing**  
**Midterm Exam**  
**Spring 2016 (60 minutes)**

NAME:

Net ID:

- This exam contains **7 questions** with a total of 20 points in **4 pages**.
- You are allowed only 1 paper (2 pages) of notes.
- **If you have to make assumptions to continue solving a problem, state your assumptions clearly.**

1.[1 pt] Describe a scenario, no need to write code, where having two MPI processes can still yield better performance than sequential program on a single-core processor.

**When one process is waiting for I/O, for example, the other one can make progress. In sequential code, the whole program will be waiting for I/O, making no progress.**

2.[1 pt] What is the main reason for moving from single core to multicore processors?

**It is the huge increase in power consumption/dissipation. We can no longer increase clock frequency of a core without huge increase in power.**

3. [2 pts] Suppose we have two MPI processes. Describe a scenario, no need to write code, where they would execute slower on a system with two cores than on a system with one core.

**The two processes need to have the following characteristics:**

- **There is a lot of communication between them.**
- **One of them is I/O bound (i.e. mostly doing I/O) and the other is compute bound (mostly doing computations).**

4.[2 pts] How could the following code sequence be changed to expose more parallelism but still achieve the same final result (i.e. at the end: x, a, b, and c have the same value as the sequential code)? In your solution, show the largest number of parallel tasks and what each task will do. Disregard any overheads, just show the parallelism. The optimal solution must not contain extra work than the original sequential code.

```
x++ ;  
a = x + 2;  
b = a + 3;  
c++;
```

**We can have four parallel tasks:**

**task 1: x++;**

**task 2: a = x+3;**

**task 3: b = x+6;**

**task 4: c++;**

5. [4 pts] In MPI, explain two ways where a process can send data to a *subset* of the processes in MPI\_COMM\_WORLD.

- Using a one or more MPI\_Send() and MPI\_Recv() pairs
- Forming a smaller communicator of the processes that need to receive the data, through MPI\_Comm\_split(), then use MPI\_Bcast() to that smaller communicator.

6. [6 pts] The following code snippet has one or more bug(s). Show and correct this (or those) bugs. No need to rewrite the code. Just state the bug and how you will fix it.

```
double a[100], b[100];
```

```
MPI_Init(&argc, &argv);
```

```
MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
```

```
if( myrank ==0){
```

```
    MPI_Recv( b, 100, MPI_DOUBLE, 1, 19, MPI_COMM_WORLD, &status );
```

```
    MPI_Send( a, 100, MPI_DOUBLE, 1, 17, MPI_COMM_WORLD );
```

```
}
```

```
else if( myrank ==1){
```

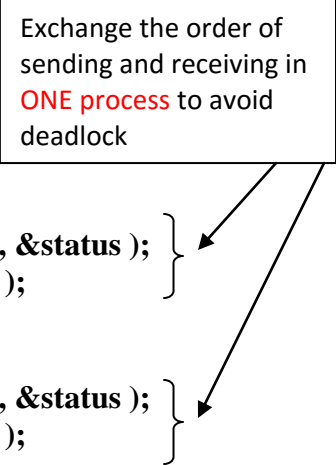
```
    MPI_Recv( b, 100, MPI_DOUBLE, 0, 17, MPI_COMM_WORLD, &status );
```

```
    MPI_Send( a, 100, MPI_DOUBLE, 0, 19, MPI_COMM_WORLD );
```

```
}
```

```
MPI_Finalize();
```

Exchange the order of sending and receiving in **ONE process** to avoid deadlock



**Another solution:**

You can change the type of a b to be: **int a[1000], b[1000];** and leave arguments of MPI\_Send() and MPI\_Recv() intact.

But you still need to change the order of communication in one of the processes.

7. In class we have seen that two overheads are load imbalance and synchronization. In the following two questions, disregard any overheads except synchronizations and load imbalance.

(a)[2 pts] If we, magically, eliminate load imbalance from our code, do synchronization points (i.e. barriers) still affect performance? Justify.

**No, performance will not be affected. The performance loss in barriers comes from the fact that some processes need to wait for others. If they all arrive at the same time, then no performance will be lost.**

**[Note: If you say that there will still be performance loss due to the overhead of call the barrier MPI, we will take it as correct.]**

(b)[2 pts] Since we cannot guarantee that we eliminate load imbalance, we decide to eliminate the barriers from our code. Does load imbalance still affect performance? Explain in 1-2 sentences only.

**Yes, load imbalance will still affect performance. For the task of the program to be accomplished, all processes must finish. So, even when there are no barriers, there is the *implicit* barrier of all processes must exit for the whole program to end.**