# ITCS 4145/5145 Parallel Programming
# Test 1
# 5:00 pm - 6:15 pm, Tuesday February 16th, 2012

Name:.........................................................................................

This test is closed book. Do not refer to any materials except those provided with the test. “Appendix A Basic MPI Routines” from the course text is provided, especially for Part II of the test.
Write your answers in the spaces provided.

Total /40
Part I /26
Part II /14

## Part I

Qu. 1 Answer each of the following briefly:

(a) Under what circumstances would Amdahl’s law indicate that no speed up is possible, i.e. a multiprocessor would not complete the computation faster than a single processor. 2

(b) What is meant by the phrase "embarrassingly parallel"? 2

(c) What does the -o option specify when used with the mpicc script? 2

(d) Name one MPI routine that does not have a communicator as a parameter (argument). 2

(e) Under what circumstance might an MPI_Send() operate as an MPI_Ssend()? 2

(f) Which MPI receive routine is used with MPI_SSend() to achieve synchronous message passing? 2

(g) What is a shared-memory multiprocessor system? 2

(h) In the command:

```
ssh coit-grid05.unc.edu -X
```

what does the -X option do? (Assignment 2) 2

(i) Why might statements not be executed in the order given in a program? 2

(j) Give one reason why one might create threads in Java by implementing the Runnable interface rather than extending the class Thread. 2

(k) In a synchronous iteration method when iterations are stopped when the solution values at consecutive iterations each differ by less that an value *e*, why is the solution obtained is not accurate to ± *e*? 2

(l) Explain the following `forall` statement: 2

```
forall (i = 0; i < 4; i++)
  a[i] = b[i];
```

(m) Assignment 2 asks you to install an X server on your client PC so that X11 graphics can be displayed. Which X server did you install (or what is one suggestion in the assignment instructions)? 2

# Part II

Qu. 2  Write an MPI program that implements a parallel version of bucketsort on *N* integers each having values between 0 and 99, using **two** MPI processes.

You may assume that there is a sequential sorting routine provided, sort(int *p, int n), where the parameter *p is the list to sort and n is the number of integers in the list. Sorting is done in place, i.e. the unsorted list becomes a sorted list.

Make whatever reasonable assumptions that are necessary but state them. You can assume the appropriate includes are declared.

*Provide comments in your code to help the grader! If I do not understand the code, I will assume it is incorrect.*

14