**CISC 372: INTRODUCTION TO PARALLEL PROGRAMMING**
**Fall 2007**
**Individual Lab Assignment 1**

**Due Date: start of class, Friday, September 7, 2007**

## 1 Objectives

The objective of this assignment is to become familiar with the use of MPI on the University of Delaware linux cluster, to learn how to time MPI programs, and to learn how to write experimental reports.

This lab is to be done individually, without group help. If you have problems, you should see the TA or the instructor, or post to the blog for general advice. Your writeups should be your own work as well as your efforts on the parallel programming and experimental studies.

## 2 Procedure

1. Read through Sections 1 and 2 of the *Beginner's Guide to MPI on the University of Delaware DEC Linux Cluster*. Try running the pings-all and spyall commands in pollock's public/bin directory. Set up a link to these commands for the future.

   Make sure that you can compile and run the example *hello world* program with several different numbers (2, 4, and 8) of processors, and understand how they work. The program can be found in `~pollock/372porsche/public` on porsche.

2. Playing Ping Pong: Copy the ping_pong.c program from `~pollock/372porsche06/public` on porsche, and compile and run the program. Run your program (with 2 processors) 3 times to collect timings. If you get spurious timings because other people are using the system, then do not include those spurious timings, but instead run your program again until you get 3 fairly close timings. Collect the data for all of your (nonspurious) raw timings for each message length.

3. Ping pong is a common way of timing the basic communication of a message passing system on a given system. Make copies and modify your copies of the program to time the basic communication of messages of type char, integer, and double. You will need to use the MPI types in the MPI calls for these types. For each data type, run the program (with 2 processors) 3 times (not including spurious timings) to collect timings. Try message sizes as large as possible to try to see the trends better. If the program will not run for larger message lengths for certain data types, change the program to stop before those lengths, and report up to the lengths that will run correctly. BE SURE TO RECORD ALL OF YOUR RECORDED TIMINGS (not just minimums) IN A TABLE TO SHOW YOUR RAW DATA. When you take your timings, run spyall first to determine whether others are running jobs. Try to run your timed runs when no one else is running.

4. Create a single graph that displays all of your timings. The horizontal axis should be the length of the messages sent, that is how many of the given data type were sent in a single message. The vertical axis should be the timings. The single graph should have a line for each data type with circles for the data points. The minimums should have circles on the line. The rest of the data should be points on the graph, but not necessarily on the lines. For this assignment, you can either create the graph by hand or on the computer using a tool that automatically creates the graph based on the data inputs. For later assignments, you need to use a computer to generate the graphs. Two different approaches to generating these graphs from output of your MPI programs are given on the CISC 372 web site.

5. The check_circuit program discussed in chapter 4 can be found in the same public directory as above. Copy this program, and make the following modifications: (1) Have process 0 print out the total number of solutions. (2) Insert timing statements into the program to time the portion of the program between MPI initialization and printing the total number of solutions, not including initialization and printing. (3) Comment out the printing inside the check_circuit function. Now run the program with

1,2,4, and 8 processors. Time for 3 data points for each number of processors, and create a graph similar to that created above for the ping pong program, including raw data.

### 2.1 Hints on Getting Good Timing Data

In order to get reliable timing data, you need to make sure that no one else is using the cluster. To do this, you should run the `spyall` command just prior to making a performance run. Also, from past experience, leaving the timings until last evenings before the assignment due date makes getting reliable timing runs very difficult to obtain due to the many other folks trying to get access to the cluster for timings.

## 3 Writing your Report

Your experimental report should consist of the following sections in this order. It is strongly recommended that you type your report using a word processor rather than handing in a hand-written report.

1. **Contact Information:** Title of lab, author, course number and semester, date.
2. **Project Summary:** In one paragraph, summarize the major steps of the lab for someone who did not have the lab spec and wants to know overall what you did.
3. **Analysis of Hello World:** Explain why the messages are printed in an arbitrary order. In English, describe an approach that might be used to force the messages to be printed in increasing order by rank.
4. **Ping Pong:**
   (a) **Collected Timing Data.** This section includes a script of a run of ping pong for two data sets, and the graph depicting the raw data pictorially indicating the minimums clearly.
   (b) **Analysis.** In English, explain your timing results for ping pong on different data types. You want to describe your observations about the way the timings vary as the length of messages increases, as well as how the timings vary between the different data types.
5. **Circuit Satisfiability:**
   (a) **Collected Timing Data.** This section includes the graph depicting the raw data pictorially indicating minimums clearly.
   (b) **Analysis.** In English, explain your analysis of what you observed about the timing results for circuit satisfiability.
6. **Conclusions:** This section consists of a discussion of what you learned overall about parallel computing from this lab. Discuss possible reasons for inconsistencies or discrepancies in your data versus what you would had expected to happen.
7. **Appendix:** Your code for the modified Ping Pong for one single data type, and the circuit satisfiability program.

Please staple all parts of your lab together, and label each piece. Be prepared to discuss your results in class.

## 4 Criteria for Evaluation

Your lab will be evaluated according to the following criteria:

1. 5 pts: lab report in order and easy to find parts
2. 10 pts: project summary

3. 10 pts: Modify code and run and collect stats on different data types (int, char, double).
4. 10 pts: compile, run, and collect stats on original ping pong program; (from script)
5. 5 pts: graph depicting results of different data types
6. 10 pts: Modify circuit satisfiability program and collect timing data
7. 5 pts: graph depicting results of circuit satisfiability experiment
8. 5 pts: analysis of hello world
9. 15 pts: analysis of ping pong data
10. 5 pts: analysis of circuit satisfiability data
11. 10 pts: conclusions
12. 10 pts: appendix with code

## 5 Extra Credit (8 pts)

Copy and modify the Hello world program so that the worker processes do not issue the print statements, but each sends a message back to process 0 containing their rank. After process 0 receives each message, it prints out a message containing the worker's rank. The program should produce output such as:

```
process 0 : Hello, world
process 1 : Hello back
process 2 : Hello back
process 3 : Hello back
```

where all print statements are issued by process 0. To earn the extra credit, turn in your program and a script of a run with 6 processes.