CHED National Center of Excellence in Information Technology Education
INSTITUTE OF COMPUTER SCIENCE
College of Arts and Sciences
University of the Philippines Los Baños
College 4031, Laguna, PHILIPPINES
☎ Phone (63-49) 536-2313 ✆ Fax (63-49) 536-2302 ✆ Campus VoIP 6123
✉ E-mail jppabico@uplb.edu.ph 💻 Web www.ics.uplb.edu.ph/jppabico

CMSC 180
Introduction to Parallel Computing
Second Semester AY 2023-2024

Laboratory Research Problem 04
Distributing Parts of a Matrix over Sockets

Introduction

Write a program that will create an open socket for sending and receiving data. Allow the program to read from the command line the port number to open and listen to. Also, let the program read from a configuration file a list of IP addresses and corresponding ports to communicate to. Run several instances of the program over several terminals or PCs. If the instances of your program are ran on the same PC, then that PC's IP address must be in the configuration file, and each instance must open and listen to different ports. If the instances of your program are ran on different PCs, then the IP addresses of these PCs must be in the configuration file together with a specific port. You define your own port. From among the running instances elect one instance as the master while the rest as the slave. The master will create a matrix **A** and distribute the corresponding submatrices to the slaves.

Research Activity 1: How to do it?

1. Write the main program `lab04` that includes the following:
   (1) Read n, p and s as user inputs (maybe from a command line or as a data stream), where n is the size of the square matrix, p is the port number, and s is the status of the instance (0 for master and 1 for slave);
   (2) If $s = 0$, then
      a. Create a non-zero $n \times n$ square matrix **M** whose elements are assigned with random non-zero positive integers;
      b. Read the configuration file to determine the IP addresses and ports of the slaves and the number of slaves t;
      c. Divide your **M** into t submatrices of size $n/t \times n$ each, $\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_t$;
      d. Take note of the system time `time_before`;
      e. Distribute the t submatrices to the corresponding t slaves by opening the port p and initiating communication with the IP and port of each slave;
      f. Receive the acknowledgment "ack" from each slave, for all slaves t;
      g. Wait when all t slaves have sent their respective acknowledgments;
      f. Take note of the system time `time_after`;
   (3) Else if $s = 1$,
      a. Read from the configuration file what is the IP address of the master;
      b. Wait for the master to initiate an open port communication with it by listening to the port assigned by the configuration file;
      c. When the master has initiated, take note of `time_before`;
      d. Receive from the master the submatrix $\boldsymbol{m}_i$ assigned to it;

e. Send an acknowledgment "ack" to the master once the submatrix have been received fully;
f. Take note of `time_after`;

(4) Obtain the elapsed time `time_elapsed:=time_after - time_before`;
(5) Output `time_elapsed` at each instance's terminal;
(6) Verify that each of the *t* slaves received the correct submatrix.
(7) Run all instances within one PC only but on different terminals.

2. Fill in the following table with your time readings (only from the master):

n	*t*	**Time Elapsed (seconds)**			**Average Runtime (seconds)**
		Run 1	**Run 2**	**Run 3**	
20,000	2				
20,000	4				
20,000	8				
20,000	16				
25,000	2				
25,000	4				
25,000	8				
25,000	16				
30,000	2				
30,000	4				
30,000	8				
30,000	16				

3. Repeat 1 and 2, but run all slave instances in a core-affine way. What happened?

4. Repeat 1 and 2, but run all slave instances on different PCs. What happened?

5. Is your implementation efficient? Did you use any of the communication techniques discussed in the lecture? If so, what is it (one-to-many broadcast, many-to-many broadcast, one-to-many personalized broadcast, many-to-many personalized broadcast)? If not, why not?

Note that because of 3 and 4, your report must include three tables.

Quiet reactions to possible complaints that may come up, and all other complaints will be ignored.

Complaint 1: *I do not know how to write socket programs*.
Reaction 1 (to self with sarcasm): *I am sure you learned to read the alphabet in the first grade by complaining.*

Complaint 2: *The programming language I used for LRP01 through LRP03 does not have a robust library for writing socket codes.*
Reaction 2 (annoyed talking to self like an schizophrenic): *Yeah, right. Why is your nose getting longer, Pinocchio?*