CS 495, Spring 2002
Assignment 3: Parallel Programming Using the Message Passing Model

Assigned: Tuesday, Feb. 19
Due: Thursday., Feb. 28, 9:00am

This assignment deals with parallel programming using the message passing model. The aim of this assignment is to understand the trade-offs involved in this model, and compare them with what we have learned about the shared memory model for writing parallel programs.

1 Policy and Logistics

Please work in groups of 2 people to solve the problems for this assignment. (Hand in one assignment per group.) There will be both electronic and hard copy hand-ins, as described below. Any clarifications and revisions to the assignment will be posted on Web page `assigns.html` in the class WWW directory. In the following, *HOMEDIR* refers to the directory:

`/afs/cs.cmu.edu/academic/class/15495-s02/public`

and *ASSTDIR* refers to the subdirectory *HOMEDIR*`/asst/asst3`.

2 Written Assignments

Please answer problem 2.7 (parts c and f only) in the textbook.

3 Programming Assignment: TSP using message passing

You are required to write a parallel Branch-and-Bound program for the Traveling Salesman Problem, using the Message Passing Interface (MPI) (recall that your previous assignment involved writing a shared memory solution for this problem). You should use the same input format as was used for the last assignment. You should have a command-line argument that says what number of processors to use. In addition, a command-line argument should also give the name of the input file.

Your report should include the following items:

1. A brief (approximately one page) description of how your program works. Describe the general program flow and all significant data structures. Compare and contrast with the design you used for the shared-memory version.
2. The solution to the problem given in `ASSTDIR/input/distances`. This file contains a 12 city problem.
3. Execution time and speedup (both total and computation) for 1, 2, 4, 8, 16, 24, and 32 processors on an NCSA Origin 2000 machine. (If you can get even more processors, that is great). Also include the speed-up graphs you had obtained for the shared-memory version (from assignment 2).
4. Discuss the results you expected and explain the reasons for any non-ideal behavior you observe. In particular, if you don't get perfect speedup, explain why. Also explain differences, if any, between your numbers for this assignment and for the previous assignment.

If the execution time for your program takes more than a few minutes on an Origin, double-check your program and algorithm. Once again, make sure your programs run on a uniprocessor before trying to run them on an Origin. Also, debug your programs using smaller numbers of cities (perhaps `ASSTDIR/input/dist4`) and small numbers of processors before trying larger runs.

4 Using MPI

A small introduction and tutorial is being handed out with this assignment; it can also be obtained from `ASSTDIR/tutorial.ps`. This tutorial gives only a very rudimentary overview to all the things that MPI allows you to do. To learn more about MPI, you should take the following online course:

Intoduction to MPI
http://webct.ncsa.uiuc.edu:8900/public/MPI/

The course is free; register online at the above URL, with any easy-to-remember login and password (these need not be the same as your login and password for the NCSA machines). Chapters 2-8 cover everything you will need for this assignment. You can log into your account any number of times, so the course material can also be used as an online reference.

5 Hand In

Electronic submission:

Your solution to the TSP. Do this by naming your file *last*`-tsp_mp.c`, where *last* is the last name of one of your group members, and copying this file to the directory

`/afs/cs.cmu.edu/academic/class/15495-s02/public/asst/asst3/handin`

Include as comments near the beginning of this file the identities of all members of your group. Also remember to put comments in your code.

Hard-copy submission:

1. Answers to the questions in Section 2.
2. Answers to the questions in Section 3.
3. A listing of your code.