CS/ECE 757 Parallel Computer Architecture (Spring 2016)

Homework 2

Due Monday 2/15/2016

Point-of-contact Chris Feilbach <crf@cs.wisc.edu> (for questions and handin)

You should do this assignment in groups of two. No late assignments.

In this assignment, you will gain experience in writing explicit parallel programs using two different programming styles - shared-memory and message passing. You will use Pthreads for shared-memory programming and MPI for message passing.

You will do this assignment on **ale-02.cs.wisc.edu**, **ale-03.cs.wisc.edu**, **ale-04.cs.wisc.edu**, **ale-05.cs.wisc.edu** or **ale-06.cs.wisc.edu** - each containing two Xeon x5550 processors which have 4 cores each for a total of 8 cores. We have given you individual accounts on this machine. Use them only for 757 homework assignments and, with instructor's permission, for your course project. The accounts and storage will be deleted at the end of the semester unless you obtain permission from the instructor for extending them.

Pthreads

POSIX threads or Pthreads is a library implementation of a standardized C threads API that enables shared memory programming. In particular, the API provides two major functionalities -

- Thread Management: Functions for creating, detaching, joining threads etc...
- Synchronization: Mutexes, Conditional variables etc...

For more information on the Pthreads, check this tutorial from Lawrence Livermore National Labs. A friendly primer on using pthreads by Tristan Ravitch is available here. The pthreads example program (maybe) demonstrated in class is available for review here.

MPI

MPI is a library specification for message-passing, proposed as a standard for use on parallel computers, clusters and heterogeneous networks. The MPI standard and related information is available here. An online reference to MPI can be found here. An MPI example program is available for review here.

It is strongly recommended that you download and run the examples **early** to get a hands-on experience of compiling, linking and running Pthread and MPI programs.

For Pthread programs

- Include pthread.h in all the source files that use Pthread library calls.
- Use the flags -lpthread for compilation and linking of your source files.

For MPI programs

- Include *mpi.h* in all the source files that use MPI library calls.
- Compile with the *mpiCC* compiler located at */usr/bin/*.
- Set your *LD_LIBRARY_PATH* to point to */usr/lib64/openmpi/*.
- To run your MPI program, use *mpiexec* located at */usr/bin/mpiexec*.

Programming Task: Ocean Simulation

We will re-use the two buffer version of Ocean Simulation program from the previous assignment.

Problem 1: Write Parallel Ocean using Pthreads(15 points)

Write a multi-threaded implementation of two-buffer Ocean using Pthreads. You could choose to parallelize only the simulation phase of your program and not the initialization phase. Make an argument for the correctness of your implementation.

Problem 2: Write Parallel Ocean using MPI (15 points)

For this problem, you will use MPI primitives to parallelize your sequential program. Note that the processes in message-passing programs do not share a single address space. So you will have to explicitly split your data structures between the various processes. Again, make an argument for the correctness of your implementation

For simplicity, you may assume that the dimensions of the grid are powers of two plus two, and that only N=[1,2,4,8,16] will be passed as the number of threads.

Problem 3: Analysis of Ocean (20 points)

Modify your programs to measure the execution time of the simulation phase of execution. Use of Unix's gethrtime() is recommended for Pthread programs. Use MPI_Wtime() for timing your MPI program.

Compare the performance of the simulation phase of your three Ocean implementations (Pthreads, MPI and OpenMP from Homework 1) for a fixed number of time steps (**100**). Plot the normalized (versus the sequential version of Ocean from Homework 1) speedups of your implementations on N=[1,2,4,8,16] threads for a **4098x4098** ocean. Note that the N=1 case should be the sequential version of Ocean, not the parallel version using only 1 thread. Repeat for an ocean sized to **8194x8194**.

What to Hand In

Please turn this homework in **on paper** at the beginning of lecture. Your answers to the discussion questions should be **typed up**, no handwritten notes will be accepted. A tarball of your entire source code including a Makefile and a README file, should be emailed to the **point-of-contact** (see top of page) before the beginning of lecture. Use subject line ***[CS/ECE 757] Homework 2***, so that email filters work properly.

- A printout of the source code of the simulation phase of Pthreads Ocean and a concise description of how you parallelized the program.
- A printout of the source code of the simulation phase of MPI Ocean and a concise description of the communication used in your program.
- Arguments for correctness of the two programs.
- The plots as described in Problem 3.
- A short paragraph on your experiences with parallelizing Ocean using the three different approaches (OpenMP, Pthreads and MPI). Which approach was easiest in terms of programming effort? Which approach would you choose considering the programming effort, performance and scalability of each implementation?

Important: Include your names on EVERY page.

Tips and Tricks

- Start early.

- Set up RSA authentication to save yourself some keystrokes. HowTo.
- Make use of the demo programs provided.
- You can use /proc/cpuinfo to learn many useful characteristics of your host machine.
- Run your programs multiple times to get accurate time measurements. This will help avoid incorrect results due to interference with other user's programs.
- While making your measurements do take care that no-one else is running his/her program on the machine. Otherwise, it will provide you as well as the other person wrong results as well as longer runtimes.
- Do not wait until the last day to run the experiments. You might not get time on the machine to get good results. Hint: The best time to run your expriments is 12-6am???
- Use piazza to ask general questions about the homework. Specific questions can be marked as private, and will be visible only to instructors.

Page last modified on January 27, 2016, at 04:27 PM, visited times