CSCI 3366 (Introduction to Parallel and Distributed Processing), Fall 2005

Homework 2

Assigned: October 1, 2005.

Due: October 10, 2005, at 5pm.

Credit: 30 points.

> *Note:* The HTML version of this document may contain hyperlinks. In this version, hyperlinks are represented by showing both the link text, formatted like this, and the full URL as a footnote.

1 Overview

First a reminder about something said in class: In doing this assignment, please *do not* discuss the problem or possible solutions with people who took this course last year. I want you to discover the potential pitfalls yourselves!

In class we briefly discussed approximating the value of π by "throwing darts" at a square of side 2 enclosing a circle of radius 1, counting how many darts fall within the circle, and then dividing that by the total number of darts to get the ratio between the area of the circle (π) and the area of the square (4). For this assignment, your mission is to write, for each of the programming environments we've talked about (MPI, OpenMP, and Java), a parallel program that performs this calculation.

2 Details

2.1 Sequential starter programs

To get you started, I have written sequential programs in C and Java that perform the desired calculation and print appropriate results:

- C program: monte-carlo-pi.c[1]. Also requires timer.h[2].
- Java program: MonteCarloPi.java[3].

Start by downloading these two programs, compiling them, and running them a few times to get a sense of what inputs you need to get a good approximation of π. Feel free to also tinker with the parts of the code that generate the sequence of random numbers; you may find some method you think will work better. (For the C program, review the man pages for `random` and `drand48`. For the Java program, review the API ("Java docs") for the `Random` class.)

[1] `http://www.cs.trinity.edu/~bmassing/Classes/CS3366_2005fall/Homeworks/HW02/Problems/monte-carlo-pi.c`

[2] `http://www.cs.trinity.edu/~bmassing/Classes/CS3366_2005fall/Homeworks/HW02/Problems/timer.h`

[3] `http://www.cs.trinity.edu/~bmassing/Classes/CS3366_2005fall/Homeworks/HW02/Problems/MonteCarloPi.java`

2.2 Parallel programs

The three programs you write (one each using MPI, OpenMP, and Java) should accept the same command-line input and produce the same output as my sequential programs, except that:

- The Java program should require an additional command-line argument, the number of threads to use.
- Rather than printing "sequential C/Java program results",
  - the MPI program should print "parallel MPI program results with N processes" (where N is replaced with the number of processes);
  - the OpenMP program should print "parallel OpenMP program results with N threads" (where N is replaced with the number of threads); and
  - the Java program should print "parallel Java program results with N threads" (where N is replaced with the number of threads).

  (You can make other changes to the output format if you like, but be sure your code prints the same information mine does.)

You can also make any changes you like to how the programs work internally.

It's up to you how you choose to parallelize the sequential code, but notice that in many respects the calculation here strongly resembles the one in the numerical integration example, so the approaches we used for that example might work well here too. The only thing that's tricky is deciding what to do about the random numbers (should all the processes/threads generate the same sequence of random numbers? should they generate different ones? if so, how?). For this assignment, I want you to make your best guess about what would be reasonable, implement that, and see how it works. After everyone has turned something in, we'll discuss in class your results and possible improvements.

You can make my grading job a bit easier by using the following names for your programs:

- `monte-carlo-pi-mpi.c` for the MPI program.
- `monte-carlo-pi-openmp.c` for the OpenMP program.
- `MonteCarloPiParallel.java` for the Java program.

2.3 Hints and tips

As with the previous homework, feel free to borrow code from any of the sample programs linked from the course sample programs page[4]. This page also contains links to my writeups about compiling and running programs on the lab machines. The course "useful links" page[5] has pointers to documentation on MPI, OpenMP, and Java.

[4] `http://www.cs.trinity.edu/~bmassing/Classes/CS3366_2005fall/SamplePrograms/index.html`
[5] `http://www.cs.trinity.edu/~bmassing/Classes/CS3366_2005fall/HTML/links.html`

3 What to turn in and how

Submit your program source code (three files, one per language) by sending mail to `bmassing@cs.trinity.edu`, with each file as an attachment. Please use a subject line that mentions the course number and the assignment (e.g., "csci 3366 homework 2"). You can develop your programs on any system that provides the needed functionality, but I will test them on the department's Fedora Core 4 Linux machines, so you should probably make sure they work in that environment before turning them in.