COMP 605: Introduction to Parallel Computing
Homework 6: Calculating $\pi$
Using Pthreads and OpenMP

Mary Thomas

Department of Computer Science
Computational Science Research Center (CSRC)
San Diego State University (SDSU)

Due: 04/23/15
Posted: 04/15/15
Updated: 04/09/15
Table of Contents

1. HW 6: Estimating π Using Numerical Integration
2. HW 6a: Estimating π Using Pthreads
3. HW 6b: Estimating π Using OpenMP
4. General Instructions
5. Example Outputs
6. Statistical Methods
Trapezoid Rule for Numerical Integration of a function

Solve the Integral: \( \int_a^b F(x)dx \)

Where \( F(x) \) can be any function of \( x \): \( f(x^2), f(x^3), \ldots \)

Ref: Pacheco (2011), Ch3.
Using Numerical Integration to Estimate $\pi$

- **Integral representation for $\pi$**

  \[
  \int_0^1 dx \frac{4}{1+x^2} = \pi
  \]

- **Discretize the problem:**

  \[
  \Delta = 1/N : step = 1/N_{\text{areas}}
  \]

  \[
  x_i = (i + 0.5)
  \]

  \[
  \Delta(i = 0, \ldots, N_{\text{areas}} - 1)
  \]

  \[
  \sum_{i=0}^{N-1} \frac{4}{1+x_i^2} \Delta \approx \pi
  \]

$\pi$ Formulae: http://en.wikipedia.org/wiki/Approximations_of_pi

Image: http://cacs.usc.edu/education/cs596/mpi-pi.pdf
Serial Code

#include <stdio.h>
#define NAREA 10000000

void main() {
    int i; double step,x,sum=0.0,pi;
    step = 1.0/NAREA;
    for (i=0; i<NAREA; i++) {
        x = (i+0.5)*step;
        sum += 4.0/(1.0+x*x);
    }
    pi = sum*step;
    printf(PI = %f\n,pi);
}
HW 6b: Estimating $\pi$ Using Numerical Integration: Pthreads Implementation (25 points)

- Write a Pthreads program that uses numerical integration to estimate $\pi$.
- The main thread should read in the total number of areas and print the estimate.
- You may write your own code, use Pacheco example (e.g. `mpi_trap4.c`), or a program found online.
- Use a shared variable for the sum of all the threads’ computations.
- To enforce mutual exclusion in the critical sections:
  - Choices: (i) busy-waiting; (ii) mutexes; (iii) semaphores.
  - If you are working alone, choose one method.
  - If you are working on a team of 2, choose two methods.
  - If you are working on a team of 3, you must test all three methods.
- See the General Instructions slides for details on parameter and scaling studies.
HW 6b: Estimating $\pi$ Using Numerical Integration: OpenMP Implementation (25 points)

- Write an OpenMP program that uses numerical integration to estimate $\pi$.
- Read in the total number of areas before forking any threads.
- To enforce mutual exclusion in the critical sections:
  - Choices: (i) critical; (ii) atomic; (iii) locks.
  - If you are working alone, choose one method.
  - If you are working on a team of 2, choose two methods.
  - If you are working on a team of 3, you must test all three methods.
- Use the reduction clause for calculating the total number of areas.
- Use default(none) directive and declare local/shared var scope.
- You may write your own code, use Pacheco example (e.g. mpi_trap4.c), or a program found online.
Use the method of *numerical integration* to estimate the value of \( \pi \).

See the *Trap* examples discussed in Pachecho 2011, Chs 3, 4, and 5.

Find a reference value for \( \pi \) to the limits of a *double precision* number.

Estimate \( \pi \) to the limits of a *double precision* number.

Calculate the value for \( \pi \) as a function of the number or areas used and number of threads.

Calculate the error of your estimate: \( Err = \pi_{ref} - \pi_{measured} \)

Use double precision for calculations and outputs.
Run the jobs on one node using the batch queue

Vary the number of threads \( #Thds \) used:

- Where \( #Thds = [1, 2, \ldots, Thd_{max}] \).
- What is the max number you can use? Why?

Vary the number of areas \( (N_{areas}) \) used:

- Where \( N_{areas} = 10^n \), and \( n = [1, 2, \ldots, N_{max}] \).
- Choose \( N_{areas} \), such that \( N_{areas} \) is evenly divisible by \( #Thds \).
- You can narrow down choices based on previous HW’s.

Time the job runs, calculate run time statistics.
General Instructions: Reporting

- You may work on teams of 1 to 3 people.
- Written report (this can be TEXT, Word, PDF Doc).
- Turn in your hard copy at start of class.
- Max length is 25 pages; can be two sided.
- Include an index, page numbering; number figures and tables (this can be manually done).
- Results: include plots, tables, discussions of the following
  - Timings as a function of \( \#PEs \) and \( N_{areas} \).
  - \( \pi \) calculations and error as a function of \( \#PEs \) and \( N_{areas} \).
  - Compare to MPI tests.
  - Include short description below each table or plot.
- Calculate the run time per point (word, or operation)
- Discussion about your results, including topics such as:
  - Compare to all results to your MPI version (include plots, tables).
  - How do you compare threads to Processors?
  - What advantages and disadvantages do you see with each approach?
- Reference key sources of information (Pacheco, lectures, Web, collaborators), both in the code and in your report.
Put homework into a directory: e.g.

\texttt{HOME/ < your\_username > /hw/hw6/hw6a}

Include:

- source code(s)
- compiled binaries
- batch scripts,
- include some output examples (small).

do not include libraries/demos/test code from development work.
Plot should contain labels and units for both horizontal and vertical axes, a title, and labels for family of curves.
Plot should contain labels and units for both horizontal and vertical axes, a title, and labels for family of curves.
Standard statistical variables used to describe the distribution of the data include:

- Max/Min (maximum/minimum values between all runs)
- Mean (average value)
- Median (central value)
- Variance (variance)
- StandardDeviation ($\sigma$) of the timings.

To test your codes:

- Run and time critical blocks
- Vary key parameters (packet/problem size, # of cores, etc.).
- Calculate the statistics at run-time.

Refs:

http://edl.nova.edu/secure/stats/