COMP 605: Introduction to Parallel Computing Homework 4: Shared Memory Programming: OpenMP

Mary Thomas

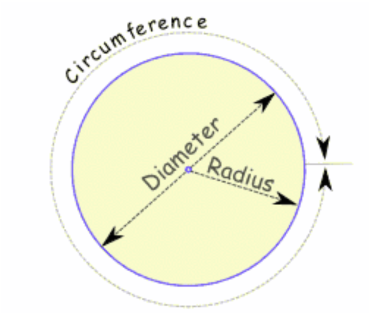
Department of Computer Science
Computational Science Research Center (CSRC)
San Diego State University (SDSU)

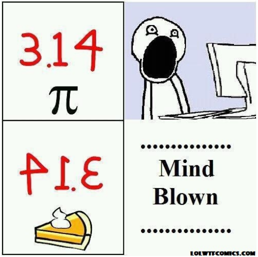
Due: 04/20/17
Posted: 04/10/17
Updated: 04/10/17

Table of Contents

1. HW #4, P1: Using Numerical Integration to Estimate π
2. HW #4, P2: Calculating Prime Numbers
3. What to Report/Turn in for both problems:

HW #4, P1: Using Numerical Integration to Estimate π



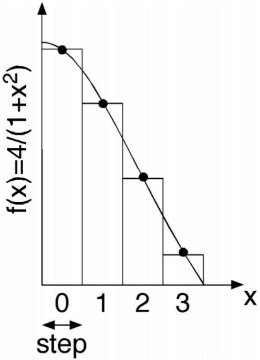


$$\pi = \frac{\textit{Circumference of a Circle}}{\textit{Diameter of a Circle}}$$

Image Source: `http://www.mathsisfun.com/numbers/pi.html`

HW #4, P1: Using Numerical Integration to Estimate π

- Integral representation for π
  $\int_0^1 dx \frac{4}{1+x^2} = pi$
- Discretize the problem:
  $\Delta = 1/N : step = 1/N_{areas}$
  $x_i = (i + 0.5)\Delta(i = 0, \ldots, N_{areas} - 1)$
  $\sum_{i=0}^{N-1} \frac{4}{1+x_i^2}\Delta \cong \pi$



π Formulae: `http://en.wikipedia.org/wiki/Approximations_of_pi`
Image: `http://cacs.usc.edu/education/cs596/mpi-pi.pdf`

HW #4, P1: Using Numerical Integration to Estimate π

```
#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 #4, P1: Instructions

- Write an OpenMP program that uses numerical integration to estimate π.
- Use OpenMP directives for the parallelism.
- You may write your own code, use Pacheco example (e.g. *mpi_trap4.c*), or a program found online.
- See the *Trap* examples discussed in Pachecho 2011, Chs 3, 4, and 5.
- Find a reference value for π to the limits of a double precision number.
- Estimate π to the limits of a double precision number.
- Calculate the value for π 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.

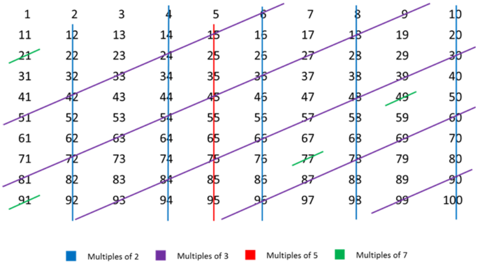
HW #4, P1: Instructions (cont.)

- Parse all key variables from the command line.
- Run the jobs using the batch queue
- Thread scaling: Vary the number of threads $\#Thds$ used:
  - Where $\#Thds = [1, 2, \ldots, Thd_{max}]$.
  - What is the max number you can use? Why?
  - Use *binding* to control the number of threads per core
- ProbSize Scaling:
  - Choose N_{areas}, such that N_{areas} is evenly divisible by #Thds.
  - Choose a few values for N_{areas} that allow scaling from 10^3 to $> 10^7$ or 10^8.
- Time the job runs, calculate run time statistics. Are the timings reproducible?

HW #4, P2: Calculating Prime Numbers

Develop an OpenMP version based on the Sieve of Eratosthenes approach to calculate all the prime numbers below some number N:

- Run jobs using the batch queue.
- Determine $N = [1, 2, 3, .., N_{max}]$ for tuckoo.
- Vary the number of threads
- Use thread binding for better performance
- Time the job runs



Img Src: http://mathworld.wolfram.com/SieveofEratosthenes.html

HW #4, P1: Instructions (cont.)

- Parse all key variables from the command line.
- Use OpenMP directives for the parallelism.
- Run the jobs using the batch queue
- Thread scaling: Vary the number of threads $\#Thds$ used:
  - Where $\#Thds = [1, 2, \ldots, Thd_{max}]$.
  - What is the max number you can use? Why?
  - Use *binding* to control the number of threads per core
- ProbSize Scaling:
  - Choose N, such that N is evenly divisible by #Thds.
  - Choose a few values for N that allows scaling from 10^3 to $>\ 10^7$ or 10^8.
- Time the job runs, calculate run time statistics. Are the timings reproducible?

What to Report/Turn in for both problems:

- Create the homework directory USER/hw/hw4 with correct access permissions.
- Short lab report with comments, figures and table labels.
- Explain your results for Thread and ProbSize scaling.
- Include relevant tables of your test data
- Evidence you ran your jobs using the batch queue (short/small job); examples of batch scripts
- Plot the runtime as a function of the number of threads or probsize.
- A copy of your code (single spaced, two sided, two column format is OK).
- Reference key sources of information in your report and code where applicable (Pacheco, lectures, Web,).