

Portland State University – ECE 588/688

Fall 2009

Homework 2

Due: Oct. 21, 2009

For this homework, you will write parallel programs using POSIX threads (Pthreads). For more information on Pthreads, you can check the tutorial:

<https://computing.llnl.gov/tutorials/pthreads>

There are a few steps that you need to do before you begin working on your homework:

1. Make sure you can login to nemo.ece.pdx.edu. If you don't have access to that machine, you should contact CAT (support@cat.pdx.edu).
2. Create a subdirectory under your home directory, and copy two files to it

```
mkdir hw2
cd hw2
cp /u/alaa/public_html/ece588/hw2/sum.c .
cp /u/alaa/public_html/ece588/hw2/procset.h .
```

The program sum.c represents a naïve implementation of for a parallel program sums up the integers from 1 to Max.

3. Compile the sample program

```
cc -lpthread sum.c -o sum
```

This compiles the program using the pthread library to the executable file “sum” using the default C compiler. You should make sure that you have /usr/local/bin in your path so that you can find the C compiler.

4. Run the sample program using different parameters to experiment with its performance. The program takes two arguments. The first is the maximum number to sum up to, and the second is the number of processors to run on. The number of processors should be from 1 to 8 since nemo has eight processors.

```
./sum 100000000 1
./sum 100000000 8
```

These commands sum up the numbers from 1 to 100 million on 1 and 8 processors, respectively, then prints the sum and the execution time.

Problem 1: (2 points)

Estimate the parallel speedup for the sample program when running on 2, 3, 4, 5, 6, 7, and 8 processors (compared to running on one processor only). You should use 100 million as the maximum number do that you run long enough to observe speedup. For accurate runtime estimates, try to use the machine when the user load is light, and try to run the program multiple times for each processor count and use the average.

Problem 2: (3 points)

Modify the sample program to compute the value of π with an algorithm similar to that provided in the parallel computing tutorial:

https://computing.llnl.gov/tutorials/parallel_comp/#ExamplesPI

You should turn in

- a) A text copy of your program
- b) A table containing the parallel speedup when using 100 million points to estimate the value of π on 2, 3, 4, 5, 6, 7, and 8 processors.

Problem 3: (5 points)

Modify the sample program to estimate the temperature of all points on a grid in a similar fashion to the algorithm provided in the parallel computing program:

https://computing.llnl.gov/tutorials/parallel_comp/#ExamplesHeat

You should use the following parameters:

- a) Grid size = 4000 x 4000
- b) Initial condition: All center points in the region (1500, 1500) to (2500, 2500) have a temperature of 200 degrees, and all other points have a temperature of zero.
- c) At each time step, the temperature of a point at coordinates (x,y) is computed from the temperatures of the neighboring points according to the following equation:

$$\begin{aligned} T(x,y) = & T(x,y) \\ & + C_x * (T(x+1,y) + T(x-1,y) - 2 T(x,y)) \\ & + C_y * (T(x,y+1) + T(x,y-1) - 2 T(x,y)) \end{aligned}$$

Where $C_x=0.2$ and $C_y=0.1$

- d) Run the program for 1500 steps. Note that depending on how you split your data, you may need to communicate information to neighboring processors after each time step.
- e) After the 1500 time steps are done, you should print the temperatures of the following points: (0,0), (1000,1000), (2000, 2000), (3000, 3000), and (4000,4000).

You should turn in a text copy of your program, as well as a table containing the parallel speedups similar to Problem 2. You will get a better score if you do a good job parallelizing your algorithm to get a high parallel speedup.

Important Note:

Please start early on this homework. Parallel programming can be time consuming. You should try to finish both Problem 1 and Problem 2 within the first week since Problem 3 will take some time to finish.