159.735 Assignment 3

Parallel Solution of the Heat Distribution Problem

Write a parallel program, that runs on a multi-core workstation, to solve for the distribution of temperatures across a printed circuit plate. The plate is initialized with a number of components and wires of fixed temperatures.

This can be done using the finite difference method whereby one iteratively finds a solution to Laplace's equation. The plate can be divided into a fine mesh of points. The updated temperature then depends upon the temperature of the points immediately above and below and to the left and right:

$$g_{i,j} = \frac{h_{i-1,j} + h_{i+1,j} + h_{i,j-1} + h_{i,j+1}}{4}$$

One then iterates over this equation until a suitable convergence condition is satisfied. A parallel implementation of this is to partition the plate according to the number of processors and work on each partition in parallel. This is an example of a problem requiring **local synchronization**. A processor working on a particular partition, must be synchronized with the processors working on the adjacent partitions.

The Assignment

1. A startup package, `heating.tar`, is available on the Stream site. Unpack the tar file and run the demo program (`heat_demo.cpp`). View the image produced by this program using the `ds9` FITS file viewer.

2. Complete the sequential version of the heat transfer solution program. It is recommended that you copy the `heat_demo.cpp` file to another file. You will also need to modify the `makefile` accordingly.

3. Develop an OpenMP parallel solution program that makes use of the cores available on your workstation. You will need to implement an appropriate partitioning strategy and you will need to ensure synchronization within the thread pool. Note that the number of iterations required to achieve convergence is not known beforehand, so you cannot fix this number. You will need to implement a strategy for detecting convergence and ensuring all threads terminate cleanly. Your parallel program should run in the same number of iterations as the sequential version, and produce the same output.

Submission

Please submit your C or C++ source code together with a report where you should address the following.

- State the architecture/platform/OS in which you chose to work on. Give the number of processor cores available on your machine.
- Provide a description of your program and how it works, emphasizing the parallelization strategy you employed and how you achieved synchronization between the threads, and your strategy for detecting convergence and ensuring clean termination of each thread.
- Run your program for different numbers of processors and appropriately chosen problem sizes. Does it obey Amdahl's Law and does it scale up for larger problem sizes in accordance with Gustafson's Law? Present your results in a way that best illustrates these points.
- Show a picture of the heat distribution of the metal plate after running the solution.

Due date: TBD, 2018

This assignment is worth 20% of your final grade