# Project #2 - OpenMP: 3D EW

SWE3021 Multicore Computing - Fall 2019

Due date: October 18 (Fri) 17:00pm

## 1 Introduction

In this project, you will improve the performance of 3D Elastic Wave Modeling (3D-EW) code using OpenMP.

3D-EW is a wave field extrapolation method to simulate the propagation of elastic wave in isotropic elastic medium. In this code, the P-wave and S-wave are simulated separately, so as to achieve a better understanding of the P-wave and S-wave propagation in the elastic medium. This method simulates the propagation of elastic waves through high-order finite difference methods.

## 2 Details

A sequential version of 3D-EW code is available at /home/swe3021/project2/ In this directory, you will find source codes and sample input and output files. `3dew.cpp` has about 400 lines of code with many nested for loops that you need to parallelize.

`3dew` program reads an input text file (para1.in, para2.in, etc) that specifies the configuration of simulation, i.e., the size of computational grid (nx, ny, nz), the number of time steps (lt), the number of edges (nedge), the number of shots (nxshot, nyshot), etc. Note that you do not have to fully understand how 3D-EW method works and how these parameters affect the simulation. If you do, you may come up with a more efficient implementation by making significant changes to the code, but it is not a requirement of this project. Even without fully understanding the logic, you should be able to parallelize and improve the given source code by analyzing the loop dependence.

While making changes to 3dew.cpp file, you may mess up its computation logic. To detect such errors, you need to compile and run "verify" program, which can be generated by running `make verify` command. This verify program reads two output files and verifies if they match. Suppose you run your parallel version of 3dew as follows.

```
$ ./3dew para1.in myout log
```

To verify this output, run the following command.

```
$ ./verify myout para1.out
```

If "verify" says diffmax is 0 or sufficiently close to 0, your output is correct. You may change the configuration file for various performance testing. Then you will need to run the sequential version as well as your parallel version so that you can compare both outputs using the verify program.

## 3 Grading

Similar to the first project, your 2nd project score will be given as follows.

1. If you do not submit, you will get 0 point.
2. If your code does not compile, you will get only 1 point.
3. If your code compiles and uses openMP directives but outputs are incorrect, you will get only 2 points.

4. The remaining 8 points are the performance score, which will be prorated by the following equation.

$$\frac{(the\ slowest\ code's\ execution\ time) - (your\ code's\ execution\ time)}{(the\ slowest\ code's\ execution\ time) - (the\ fastest\ code's\ execution\ time)}$$

## 4 How to Submit

To submit your project, you must run multicore_submit command in your project directory in swin.skku.edu server as follows.

```
$ cd your_working_directory
$ multicore_submit project2 ./
```

This command will submit all your files in your current directory to the instructor's account. For any questions, please post them in Piazza so that we can share your questions and answers with other students. Please feel free to raise any issues and post any questions. Also, if you can answer other students' questions, please don't hesitate to post your answer.