# CS 33 Spring 2013
# Lab 4: OpenMP

**Introduction**

In this lab assignment, you will improve the performance of existing code using optimization techniques and parallelization using OpenMP.

The lab handout contains the following files:

| | |
|---|---|
| `header.h` | Header with typedefs and function prototypes. |
| `edgedetect.c` | Primary source file (edit and submit this). |
| `main.c` | Source file containing main() and initialization code. |
| `Makefile` | Build script. |
| `img.bmp` | Sample input image. |
| `correct.txt` | Correct output data (for default inputs). |
| `submit` | Script to submit a source file. |
| `status` | Script to check submission status. |
| `results` | Script to check submission results. |
| `clear` | Script to clear submissions in progress. |

You will edit and submit only `edgedetect.c`.

**Grading**

Your grade for this assignment will be proportional to the amount of speedup you achieve. For full credit, you must achieve a 5X speedup. Extra credit will be awarded for speedup beyond that amount. To ensure fairness in measuring speedup, you will submit your code to be run on the Hoffman computing cluster.

**Compiling and Running**

| | |
|---|---|
| To compile normally: | `make seq` |
| To compile with OpenMP enabled: | `make omp` |
| To compile using a different source file: | `make omp SRC=try2.c` |
| To compile with gprof enabled: | `make seq GPROF=1` |
| To run with default options: | `make run` |
| To check that your output is correct: | `make check` |
| To remove the executable and output files: | `make clean` |

The generated executable is named `edgedetect`. By default, it will generate the image `out.bmp` and the list of points `pts.txt`. It also prints the time taken to run `detect_edges()`. If you like, you may experiment with different input images or different command-line parameters. `make check` assumes that you ran with default parameters using `make run`. If your output is not correct, a message will be printed saying your output differs from `correct.txt`.

**Submission**
To use the submission scripts, you must be logged in to `lnxsrv02`.

To submit a source file: `./submit edgedetect.c`
A unique cookie will be printed to allow you to identify your submissions.

To check the status of submissions in progress: `./status`
Your are limited to 1 submission in progress at one time.

To check the results of completed submissions: `./results`
You can also check the web scoreboard (see below).

To clear your submissions that have not yet completed: `./clear`

It may take several minutes before your submissions run. Before submitting a file, run on your local machine or lnxsrv to check if your changes have any effect on the execution time.

**Code Overview**
The code performs edge detection on a color input image and produces a black&white output image, where white pixels represent edges. An edge is a linear region of high contrast that corresponds to an object border or line in the input image. The main function is `detect_edges`, found at the bottom of `edgedetect.c`. The code contains a few comments which describe the main stages of the algorithm.

**Profiling**
When optimizing a large program, it is useful to profile it to determine which portions take up the largest portion of the execution time. There is no point in optimizing code that only takes up a small fraction of the overall computations.

`gprof` is a simple profiling tool which works with `gcc` to give approximate statistics on how often each function is called. `gprof` works by interrupting your program at fixed time intervals and noting what function is currently executing. As shown earlier, to compile with gprof support, simply add `GPROF=1` to the `make` command. Then when you run `edges`, it will produce the file `gmon.out` containing the raw statistics. To view the statistics in a readable format, run gprof with the name of the executable: `gprof edges`.

You can also measure execution time of portions of the program using the `timer_*` functions, defined in `header.h`. Check how they are used in `main()`.

**Scoreboard**
A full scoreboard is also available via the web, and is updated every 30 seconds:
`http://www.seas.ucla.edu/~vitanza/cs33s13/openmplab.html`