CS/SE 4F03 – ASSIGNMENT 1

NED NEDIALKOV

Due date: 28 January in class

1. Introduction

This assignment is about performing simple noise reduction on images using MPI and C. You can work in groups of at most two.

1.1. **PPM.** The Portable Pixmap Format (PPM) uses ASCII encoding of pixels for image files; for details, see https://en.wikipedia.org/wiki/Netpbm_format#PPM_example. Here we shall use the P3 encoding.

1.2. **Mean filter.** A simple filter for noise reduction in image processing is to replace a pixel by the average of the neighbouring pixels in a "sliding" window. For example, with a 3×3 window and the numbers on the left,

45	4	255
78	124	56
1	0	34

$\Rightarrow$

×	×	×
×	66	×
×	×	×

we obtain the center of the window on the right as (rounded to the nearest integer)

$$(45 + 4 + 255 + 78 + 124 + 56 + 1 + 0 + 34)/9 = 66.$$

Such a window goes through each entry and replaces it by the mean of the entries in the window, where the entry in the middle is included in computing the average.

1.3. **Median filter.** Another filter is to replace a pixel by the median of the pixels in the window. For details, see https://en.wikipedia.org/wiki/Median_filter.

1.4. **RGB.** In an RGB encoding, we take the average (or median) over each of the colors red, green, and blue independently. That is, the average of all red, all green, and all blue pixels in the window.

2. Programming

You need to read a PPM image file, apply filtering in parallel using MPI, and store the resulting image file. You are given C code for reading and writing PPM files; you can download it at http://www.cas.mcmaster.ca/~nedialk/COURSES/4f03/private/code.zip and use it in your program. This zipped file contains also a PPM file. You can generate such from JPG files using

```
convert  -compress none file.jpg  file.png
```

Date: 18 January 2015.

You have to write an MPI program that performs the following tasks:

- Process 0 reads a given input PPM file and distributes pixel data to p processes.
- Each process applies an $N \times N$ filter to the pixels distributed to this process.
- When finished, each process sends the result to process 0, which stores the filtered image in a file in PPM format.

You must have a `makefile` such that when `make` is typed, an executable with name `ppmf` is created in the current directory.

Your program must run as

```
mpirun -np p  ./ppmf input.ppm output.ppm N F
```

where

- `p` is the number of processes
- `input.ppm` is the name of the input file
- `output.ppm` is the name of the output file
- `N` specifies the size of the window, that is $N \times N$ window, where N is an odd integer ≥ 3.
- `F` is the type of filtering and can have a value `A` meaning mean filter, or a value `M` meaning median filter.

For example,

```
mpirun -np 4  ./ppmf input.ppm output.ppm 3 A
```

would run on 4 processes and apply a 3×3 mean filter.

3. Marking scheme

To obtain full marks, your program must

- work correctly with any number (the system allows) of processes, including $p = 1$, with $N = 3$ and mean filtering;
- each process must perform about the same amount of work as any other process; and
- you must follow the instructions exactly in this assignment, e.g. makefile, submission to SVN (see below) etc.

Bonus. You can obtain bonus marks as follows:

- up to 5% if you replace the read and write functions with much more efficient implementations;
- up to 15% if your program also does median filtering and N can take values 5, 7, etc.

Notes

- If your program does not compile, then it is worth at most 30%.
- If it compiles, but does not execute, your mark would be at most 40%.
- You have to follow the instructions precisely; otherwise, marks will be deducted. For example, if you name your executable differently from `ppmf` or/and a makefile is missing, then marks will be deducted. Similarly, if the files are not submitted to SVN as described below, marks will be deducted.

4. What to submit

- Your code using subversion to
  https://websvn.cas.mcmaster.ca/cs4f03-se4f03/username/A1
  The subdirectory `A1` must contain your C files and the makefile.
- Hard copies of your programs.
- A discussion on how your distribute the work.