CS 516, Spring 2020
Jetson TX-1 Project
Due date: Wednesday, March 25, 2020

The goal is to experiment with different forms of parallelism - multi-threaded parallelism using OpenMP and "vector" parallelism on GPUs using NVIDIA's CUDA language - and explore the **tradeoff space** of

Execution Time vs. Energy vs. Power

for a single image processing application, namely an image smoothing code (QuickShift). The project consists of an evaluation of QuickShift running on a single CPU, multiple cores, and the GPU of NVIDIA's single board, heterogeneous Jetson TX-1 system. The deliverable of the project is a written report that describes your experiments and presents the results. You may choose different aspects of the application to explore, for example different images and parameter settings. A deep understanding of the application's code is not required. Given the input image, the application reports the execution time in milliseconds. A working knowledge of Linux, C++ and OpenMP is desirable, latter if you want to change the number of cores used. Important metrics to report are:

Execution Time: reported by the application (milliseconds)

Power: voltage * current (reported by the powerstrip)

Energy: power * execution time; you may assume that the power is constant; in general, it is the integral under the power curve.

1. Jetson TX-1 board

A detailed description of the board that you will be using can be found online.
NVIDIA's TX-1 board: http://elinux.org/Jetson_TX1

Please feel free to just perform a google search to find more information about this system. All of you have been assigned a TX-1 board for the first homework, so please use this machine for the first project as well. You can remotely log into these systems from an iLab machine and run your application code. You need to be on a machine at Rutgers, so directly logging into the TX-1 boards from outside is not possible.

2. mFI power strips

The power supply of each board in our TX cluster is connected to a "smart" mPower Pro 8 power strip https://www.ubnt.com/mfi/mpower/ . This allows the measurement of power in real time. There is a web interface that allows instantaneous power readings. In addition, you can remotely log into the power strips and perform different logging commands, again **you should be on a ilab machine to do this**. This data logging script has been copied onto your TX-1 system. Please feel free to modify this script to match your needs.

3. QuickShift application

There are three versions of this code, one sequential (cpu), one using multi-threading with OpenMP (omp), and one using the 256 core GPU (gpu). The application has two parameters, tau and sigma, that can influence the quality of the program outcome. The application takes a single image as input, for example "flowers2.pnm":

The application computes pixels that have similar color values. Significantly different color values indicate object boundaries. The image within such boundaries is then smoothened, i.e., made identical. The notion of "identical" and the size of the regions that can be smoothened can be adjusted, resulting in different output image qualities. The code is written in C++ / CUDA. The outcome for tau=6 and sigma=10 (default values) is shown below:

4. How to get started?

Get cs516-proj1.tar from sakai Resources and copy it onto your TX-1 system. Create your project directory. Untar the project in this directory, say "make", and the executable "quickshift" is generated. To clean up, say "make clean". There is a single picture, flowers2.pnm, in the project distribution.

You can call quickshift on a file using different computing resources as follows:

```
./quickshift  --file  <input.pnm> --mode cpu
./quickshift  --file  <input.pnm> --mode omp
./quickshift  --file  <input.pnm> --mode gpu
```

If you just say ./quickshift, it gives you the possible command line options.

Values for sigma and tau can be provided using "—sigma" and "—tau" command line options. The default is 6 and 10, respectively. After you executed quickshift, the resulting image can be found in <input>-gpu.pnm, <input>-cpu.pnm, and <input>-omp.pnm

The basic image viewer on Ubuntu is called "eog". You can visually inspect the output to see whether they are all identical. You may also be able to use "display".

Information about power strips together with their passwords and the port assignments of each of our TX boards is provided in file PowerStripsInfo.txt in the csuser home directory on each TX-1 system. DO NOT SWITCH OFF YOUR POWER SUPPLY.

4. Grading

Grading is based on a written report (up to three pages). No code needs to be submitted.
You may "play" with different tau and sigma values for QuickShift, or use different sets of images that you may get from the internet. You should run quickshift on different numbers of OpenMP cores (2, 3, or 4, which is the default). Show what you did in a form that makes it accessible to a reader not familiar with the issues. Go for quality rather than quantity. Graphs are good tools to convey a trend and/or tradeoffs. Is there a lesson to be learned here?

5. Project Questions

All project related questions should be posted on piazza. Thanks!