Intel® Accelerate Your Code
Software Skills Contest

Problem to optimize and parallelize

Topic : Satellites, drones and cameras generate huge amounts of high resolution images every day. To get valuable information from these images, we need new codes and algorithms, highly optimized and parallel.
Power efficient servers and accelerators like the new Xeon PHI can do a fantastic job if the code is adapted to the processors. But, as for every real life project, coding with simple, portable and efficient frameworks is always a good long term strategy.

Problem : The goal of this contest is to find well defined color patterns in large quantities of high resolution images. Example : you want to find a tourist lost in the mountains, you know he's wearing a shirt with a specific logo, and you have all the high resolution images from a satellite/drone.
Patterns can be : a logo (usually : geometric shapes, specific colors), camouflage patterns (usually : specific shades of colors, specific shapes like leaves), a distinctive graphical feature, … etc.
Difficulty : But the pattern can be distorted (imagine a logo printed on clothes), the colors can be altered due to lighting conditions, the pattern can be seen from various angles and distances … it's a real life problem !
To simplify the problem :

- In our case, the patterns are well defined, we are not trying to find generic shapes like humans or faces. Only the pattern is interesting here, not the context.
- The patterns will usually show up in specific conditions for our specific use case. Example : if I am wearing a logo on my shirt, I am standing or sitting most of the time. You can suppose a drone camera would probably spot the logo more or less in the right orientation (top side up), with a minimal stretch but some distortion due to the viewing angle.
- Step by step : because the problem is really complex and require a lot of complex coding, the benchmarking procedure will proceed step by step. A first level of benchmark will propose exact scaling of the pattern, without rotations. The second will propose small rotations and no scaling, the third larger rotations. The last level of benchmark will propose real life photos with scaling and rotations. **You can still participate to the contest and focus on the first simple levels of benchmarks without solving the full problem. An incomplete result is still a result, and will be benchmarked.**

Sample code : To help you start, we are providing a very simple piece of code proposing a primitive answer to the problem. It is loading the pattern files, the image to analyze, finding the patterns and reporting the results in the right format. And the code is highly commented.
Bad news : This code is slow, not parallel, can only find non distorted or non rotated patterns, and the scaling algorithm is primitive.
Good news : The code works on our contest validation and benchmarking servers and comes with a Makefile, so we recommend you start from this skeleton code.
Use "`make`" to build for intel Core processors using intel compiler,
and "`make mic`" to build for Xeon PHI (if you have the Xeon PHI software stack installed)
Your goal : Your goal is to implement the necessary features, optimize and parallelize the code and algorithm. You are in charge of finding all the tricks relevant to the specific use case (example : if a pattern is using a very distinctive set of colors compared to the image, working on color detection before shapes may be a good idea).

Grading : We will evaluate (70% of the note) :

- If your software is able to return correctly (hint : check for parallel bugs !)
- The accuracy of your output. We accept non perfect results, as the problem is complex. But finding a maximum of patterns will get you bonus points. We also accept variability in the position of the pattern found.
- The speed and scalability of your software.

We will also evaluate :

- The documentation of the final submissions (15% of the note).
- The social interaction on Intel Developer Zone : forum/blog posts (15% of the note).
- Note : We consider posting your source code as open source on internet at the end of the contest is worth a perfect note for both documentation and social interaction. Send us the link after the end of the contest by email paul.guermonprez@intel.com

Input/Output :

Input parameters :

1. N = Number of threads your software should use. If we give 0 as value, it's up to you to find the optimal value for the machine.
2. MS = Maximum scale : to help you parallelize and save useless processing time, we will not ask you to find patterns scaled more than MS times.
   Example : if we give a pattern of 50x50 pixels and a MS parameter of 10, it means you don't have to find matches larger than 500x500 in the image.
3. IMG = Image file, BMP format. That's the image where you have to find patterns
4. PATTERNS = pattern files, BMP format. 1 or more patterns. The first 3 characters of the filename are the "pattern ID" (example "002logo.bmp" is pattern "2").

Example command line :

from the folder "test_case_1" in the problem zip file :

```
../run 4 4 img2.bmp 001template.bmp 002template2.bmp
```

Expected output :

tab delimited format, 3 columns, 1 line for each match in the image :

1. Column 1 : ID of the pattern found
2. Column 2 : x of the top left corner of the pattern found
3. Column 3 : y of the top left corner of the pattern found

Note : if a pattern is found several times in a small area, reporting it once is enough.

Example output :

```
1	9354	766
1	9775	521
2	3964	509
2	4052	245
```

Benchmark and validation :

Datasets : We are providing in the problem zip file a simple dataset to help you validate your code yourself.
We will publish larger datasets during the contest, available from the same site :
http://www.intel-software-academic-program.com/contests/ayc/2013-autumn/problem/
The dataset used for the final evaluation of codes will remain a secret.
Benchmark : You can upload your code to our benchmarking cluster :
http://www.intel-software-academic-program.com/contests/ayc/upload/
You need to rename your zip file with the ID you received by email after registration.
You can also upload from the command line, using our Makefile : insert your ID in the Makefile and type : `make submit`
Your last upload will be used for the final evaluation.
When you reach a certain level of performance with a dataset, you are upgraded to a new larger dataset on a larger machine.
Hardware : We use a variety of servers and Xeon PHI accelerators (60 cores, 240 threads). Your code has to be portable between the execution environments, but the large workloads will be run on the Xeon PHI only.
Xeon PHI has its own small operating system and libraries, all you can use is : intel compiler, C/C++, OpenMP, TBB, MPI, Cilk (also available on servers). We only accept native Xeon PHI code for this contest, no offloading. Note that “native code on Xeon PHI” is just regular C/C++ code able to compile with the Xeon PHI specific compiler. You don’t have to use assembly code or a complex framework. Also, we discourage you from using OpenCL, it’s not optimal for this problem and without offloading.
Procedure : To determine the ranks and the winners from the top, we will first start from a full complex benchmark on xeon-phi, then simpler benchmarks on xeon phi, then complex benchmarks on regular xeon servers, then simple benchmarks on regular xeon servers. That’s why you can still participate even if you don’t try to solve the complex problem, or do not code for xeon phi.

Links :

Contact :
paul.guermonprez@intel.com
twitter : https://twitter.com/intelswacademic

Register :
Professors and mentors can register their class from :
http://www.intel-software-academic-program.com/contests/ayc/mentors/
Participants can register from :
http://www.intel-software-academic-program.com/contests/ayc/register/

Courses and software tools :
http://software.intel.com/en-us/mic-developer
http://intel-software-academic-program.com/courses/#manycore
http://www.drdobbs.com/parallel/programming-intels-xeon-phi-a-jumpstart/240144160
http://software.intel.com/en-us/non-commercial-software-development

Forums :
English : http://software.intel.com/fr-fr/forums/accelerate-contest-english-forum
Problem description, code and datasets (new datasets uploaded during the contest) :
http://www.intel-software-academic-program.com/contests/ayc/2013-autumn/problem/
Terms and conditions of the contest :
http://www.intel-software-academic-program.com/contests/ayc/2013-autumn/tc
Privacy page :
http://www.intel.com/content/www/us/en/privacy/intel-online-privacy-notice-summary.html