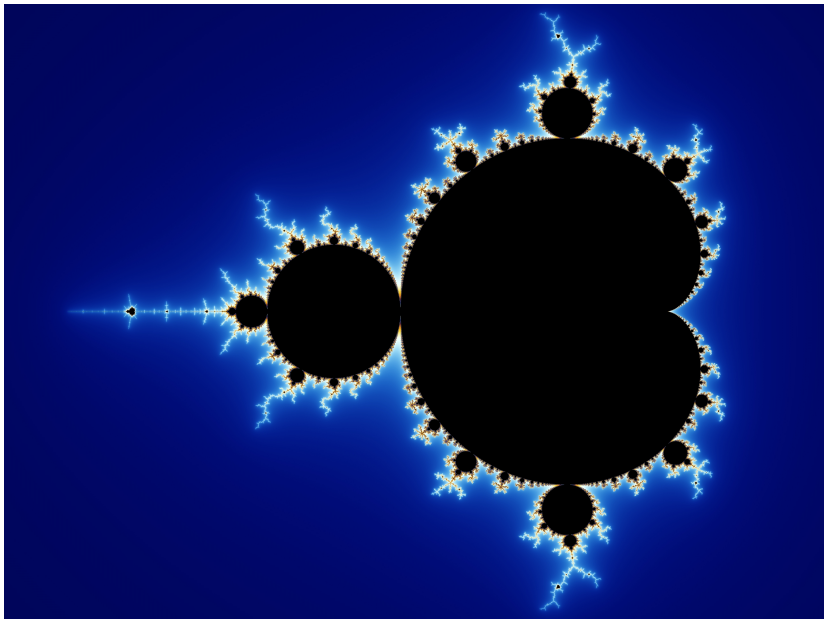
Lab #6: Mandelbrot set through OpenMP tasks

PCSE 2015

Justification

If you've never heard the name Mandelbrot set, you probably recognize the picture.



Its formal definition is as follows:

> A point c in the complex plane is part of the Mandelbrot set if the series x_n defined by
>
> $$\{x_0 = 0 x_{n+1} = x_n^2 + c$$
>
> satisfies
>
> $$\forall_n : |x_n| \leq 2.$$

It is easy to see that only points c in the bounding circle $|c| < 2$ qualify, but apart from that it's hard to say much without a lot more thinking. Or computing; and that's what we're going to do.

The typical way to generate a picture is to

1. take a coordinate
2. iterate up to a certain limit
3. if the sequence has not escaped the circle with radius 2 it may belong to the Mandelbrot set and we colour it black, otherwise we give it a colour depending on the iteration when it left the circle.

Lab exercise

In this exercise you are going to take an example program and extend it to use OpenMP tasks. The program has been set up as a loop around a routine that generates the coordinates, and then determines whether they are in the set or not. The output of the program is a graphics file that you can display on stampede with `display mandelpicture.ppm`.

- The makefile is set up so that the command
  ```
  make mandel MANDELVERSION=0
  ```
  make the executable, using a file `mandel_tools0.cxx`.
- The program takes commandline arguments. The following values are good for experimenting:
  ```
  mandel steps 300 iters 10000
  ```
  which gives a 300×300 pixel picture, and iterates $10,000$ steps maximum.
- You need to 'improve' the program by copying `mandel_tools0.cxx` to `mandel_tools1.cxx` and `mandel_tools2.cxx` and editing these files. Changing the `MANDELVERSION` parameter in the make command will use these new files.
- Step 1: use OpenMP tasks for the `for` loop. In the loop two things happen: coordinate generation, and iterating on the coordinates. Reason that only the second can be turned into tasks. This program should give you a decent speedup; approximately a factor of 4 on 8 cores.
- Step2: the reason for the imperfect speedup is false sharing. Can you image where that comes from? Fix this problem by making a small buffer of coordinates; for instance 128 coordinates. Then let each OpenMP task process that buffer, instead of a single coordinate.
  Do not make changes outside the `mandel_tools` *file.*