Parallel Computation of a Plot of the Mandelbrot Set

The goal of the project is to create parallel versions of a program to plot a Mandelbrot set and to investigate the performance of these parallel programs.

0. Generating the Postscript File.

The program "mandelbrot.c" (available for download from the course web site) generates a postscript file "mandelbrot.ps". We can send this postscript file to the printer or view them on screen by programs like "ghostview" (or "gv") or Acrobat Reader. Every pixel with coordinates (x, y) in the plot receives the grayscale value equal to the number of iterations of the map $z^2 + c$ to take a value larger than two, for $c = x + iy$, $i = \sqrt{-1}$, starting with $z = 0$. If the resolution is high enough, then the fractal nature of the plots is revealed when zooming in the viewer. Feel free to change the default noninteractive mode to experiment with different values of the resolution.

Notice that the program writes the grayscales directly to file (which is several Mb large) without first creating a large matrix of values for grayscales. To save printer toner, the grayscales are plotted "in inverse", i.e.: as $255 - n$ for n between 0 and 255.

1. Parallel Versions of the Program

We will create two versions of the program:

1) Every processor writes its own postscript file for its segment of the plot. When the parallel program returns from a run on p processors, there are p different postscript files. So on return, there are p postscript files.

2) Only the manager writes one large postscript file. The workers compute the grayscales and send this to the manager for plotting in the file. In this parallel version, compare the difference between blocking and nonblocking communication routines.

2. The Computation/Communication Ratio

We can compute the computation/communication ratio in two ways: (1) by adding timing statements to the code; or (2) by counting the flops and by counting the bytes sent. Use these two different ways to compute the computation/communication ratio of the two different parallel versions of the program.

3. Analysis of Speedup and Scaled Speedup

There are questions we would like to answer:

1) How much faster can we generate the same plot? To answer this question, we compute the speedup (using actual timings and flop counts) for p processors, for $p = 2, 3, \ldots$.

2) Given the same amount of time, how much can we increase the resolution of our plots, using p processors, for $p = 2, 3, \ldots$. This question concerns scaled speedup.

4. The deadline is Monday 20 February 2006 at 1PM.

There is work enough in this project for two, but it also fine if it is solved individually. A collaboration with three could be possible, but is often less productive.

Deliverables of this project are source of the programs, prints of the plots you created and a technical report with the answers to the questions.

The plot made by the default settings of the parameters in the program "mandelbrot.c" is below: