CS542200 Parallel Programming
Homework 2: Mandelbrot Set

Due: November 5, 2017 **23:59**

1 Goal

This assignment helps you to get familiar with OpenMP and know the differences between different memory architecture we have learned so far. Besides, this assignment also helps you knowing the importance of load balance. In this assignment, you need to parallelize the sequential program of Mandelbrot Set by implementing the following four versions of code:

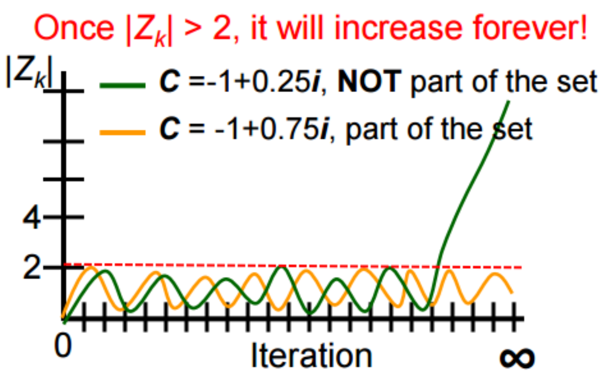
1. MPI with Static Scheduling
2. MPI with Dynamic Scheduling
3. OpenMP (Either Static or Dynamic Scheduling)
4. Hybrid parallelism – MPI + OpenMP (Either Static or Dynamic Scheduling)

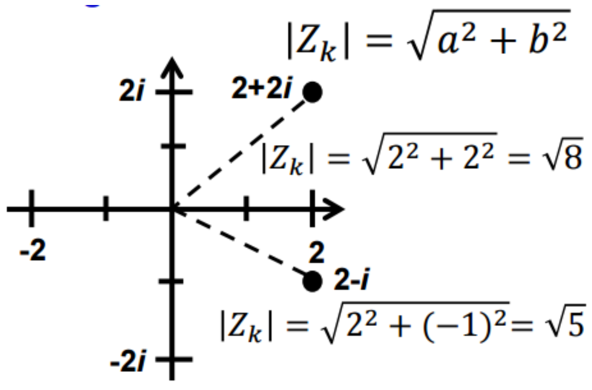
2 Problem Description

Mandelbrot Set is a set of complex numbers that are quasi-stable when computed by iterating the function:

$$Z_0 = C, Z_{k+1} = Z_k^2 + C$$

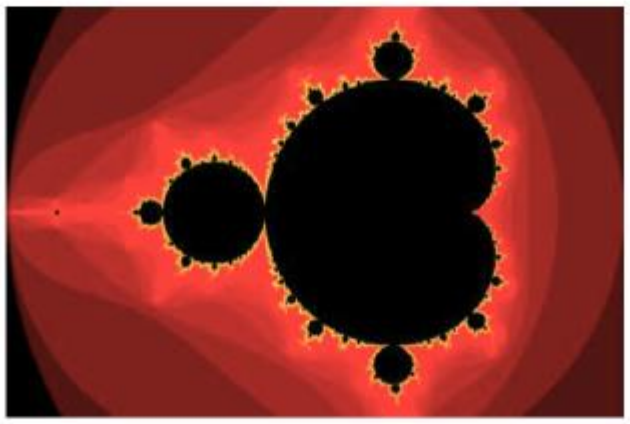
- C is some complex number: $\mathrm{C} = \mathrm{a} + \mathrm{b}i$
- Z_{k+1} is the $(k+1)_{th}$ iteration of the complex number
- if $|Z_k| \leq 2$ for any k, C belongs to Mandelbrot Set




$|Z_k| = \sqrt{a^2 + b^2}$
2i
2+2i
$|Z_k| = \sqrt{2^2 + 2^2} = \sqrt{8}$
-2
2
2-i
$|Z_k| = \sqrt{2^2 + (-1)^2} = \sqrt{5}$
-2i

What exact is Mandelbrot Set?

- It is fractal: An object that display self-similarity at various scale; Magnifying a fractal reveals small-scale details similar to the larger-scale characteristics
- After plotting the Mandelbrot Set determined by thousands of iterations:



For more information, please refer to lecture notes.

3 Input / Output Format

1. The value of the points is between 0-255
2. Your program accepts 8 input parameters:

```
./executable $num_threads $left $right $lower $upper $width
$height $filename
```

- $num_threads: number of threads per process [int, 1 ~ 12].
- $left: inclusive left bound of real-axis [double, -10 ~ 10]
- $right: non-inclusive right bound of real-axis [double, -10 ~ 10]
- $lower: inclusive lower bound of imag-axis [double, -10 ~ 10]
- $upper: non-inclusive upper bound of imag-axis [double, -10 ~ 10]
- $width: number of points in x-axis [positive integer]
- $height: number of points in y-axis [positive integer]
- $outfile: output filename

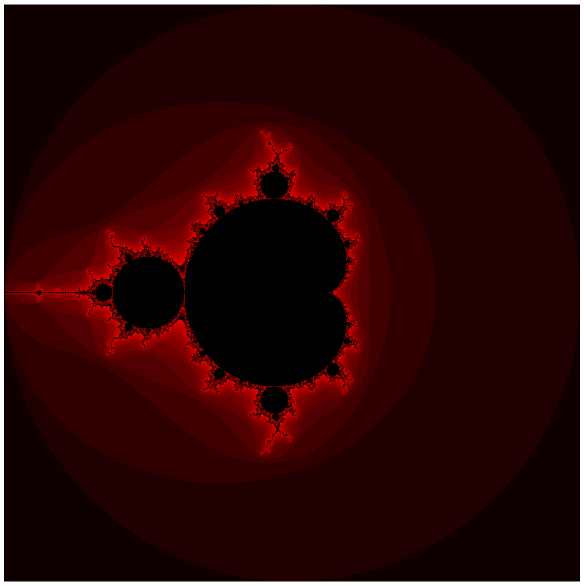
3. Output:

   The output should be a PNG file visualizing the Mandelbrot Set in the given range.

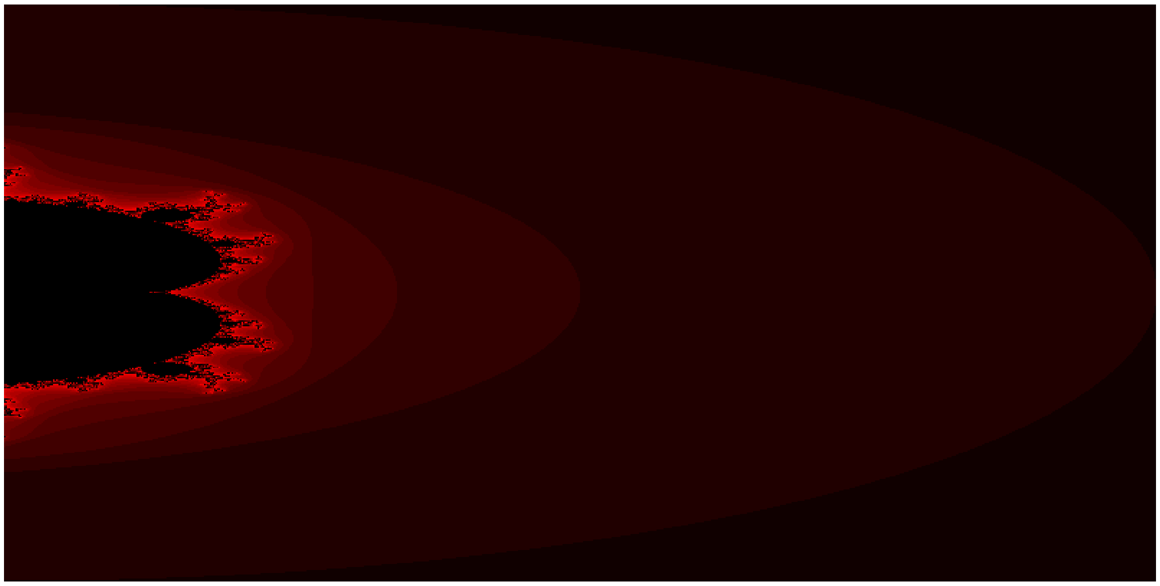
   We provide a sequential version to show how the pixels are rendered.

Example 1: **srun -n 2 ./ms_seq 4 -2 2 -2 2 400 400 out.png**

x axis: [-2, 2), 400 points = {-2, -1.99, -1.98, ..., 1.98, 1.99}



Example 2: **srun -n 2 ./ms_seq 4 0 2 -2 2 800 400 out.png**



4 REPORT

Report must contain the following contents, and you can add more as you like.

1. **Title, name, student ID**
2. **Implementation**

Explain your implementations, especially in the following aspects:

- ✓ How did you implement each of requested versions, especially for the hybrid parallelism and dynamic scheduling algorithm?
- ✓ How do you partition the task?
- ✓ What technique do you use to reduce execution time and increase scalability?
- ✓ Other efforts you've made in your program

3. **Experiment & Analysis**
Explain how and why you do these experiments? Explain how you collect those measurements? Show the result of your experiments in plots, and explain your observations.

i、 Methodology

(a). **System Spec (**If you run your experiments on your own machine**)** Please specify the system spec by providing the CPU, RAM, disk and network (Ethernet / InfiniBand) information of the system.

(b). **Performance Metrics**: How do you measure computing time of your program? How do you compute the values in the plots?

ii、 Plots: Scalability & Load Balancing

(a). Conduct strong scalability experiments, and plot the speedup. The plot must contain at least 4 settings (e.g., scales), and include both single node and multi-node environment.

(b). Design your own experiments to show how balance it is in each of your experiments by plots.

(c). Make sure your plots are properly labeled and formatted.

(d). You can configure proper problem size to ensure the experimental results are accurate and meaningful.

iii、 Discussion (Must base on the results in the plots)

(a). Compare and discuss the scalability of your implementations.

(b). Compare and discuss the load balance of your implementations.

iv、 Others

Conduct more interesting experiments to analyze performance. For example,

- What's the best distribution between MPI tasks & threads, why?
- What's the best distribution of cores between nodes, why?

4. Experience / Conclusion

It could include these following aspects:

- ✓ Your conclusion of this assignment.
- ✓ What have you learned from this assignment?
- ✓ What difficulty did you encounter in this assignment?
- ✓ If you have any feedback, please write it here. Such as comments for improving the spec of this assignment, etc.

5 GRADING

You are required to implement **four** different versions of Mandelbrot Set. Moreover, your grade will be judged by correctness, report, and demonstration as described below:

1. **Correctness** (50%)

During demo, TA will use a lot of different combinations of parameters to test your program, and check whether your output is correct. Also, please make sure you follow the different memory architectures to implement your programs as described before. The detail score distribution of each implementation is:

- MPI with Static Scheduling version (10%)
- MPI with Dynamic Scheduling version (15%)
- OpenMP version (10%)
- Hybrid version (15%)

Each implementation will be given several test cases. For each test case:

- Your implementation should produce the same output as the sequential version. We will accept your output if at least 99.6% of the pixels in the image are the same.
  (without rounding, please do not argue 99.55% ≈ 99.6%)
- Your implementation should be faster (or not significantly slower) than the sequential version. We'll make the judge timeout at 2x of the sequential version's execution time.

2. **Performance** (15%)

   Based on the fastest version.

3. **Report** (25%**)**

   Grading is based on your evaluation, discussion and writing. If you want to get more points, design or conduct more experiments to analyze your implementation.

4. **Demo** (10%)

   Demo will mainly focus on the following aspect:

   i、 Explain your implementation.

   ii、 Explain the key results and findings from your report.

   Your extra efforts. (why do you deserve more bonus points?)

6 SUBMISSION

Upload these files to apollo31:

- ~/homework/HW2/ms_mpi_static.c
- ~/homework/HW2/ms_mpi_dynamic.c
- ~/homework/HW2/ms_omp.c
- ~/homework/HW2/ms_hybrid.c
- ~/homework/HW2/Makefile

To see the time on the server, use the `date` command

Upload these files to iLMS:

- HW1_STUDENTID_Report.pdf

7 Reminder

1. **We provide a sequential version of Mandelbrot Set as well as Makefile under /home/pp17/ta/hw2 for your reference.**
2. Since we have limited resources, please **start your work ASAP**. Do not leave it until the last day!
3. **0 will be given to cheater** (even copying code from the Internet), but discussion on code is encouraged.
4. Asking questions through iLMS is welcomed!