|  |  |  |  |
| --- | --- | --- | --- |
| Hw2 report | | | |
| 學號: 109062233 | 姓名: 蘇裕恆 |  | 繳交時間: |

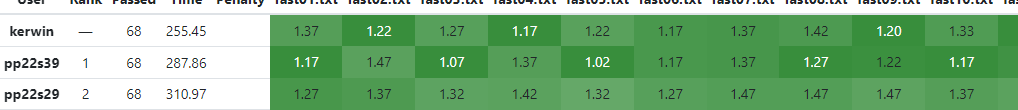
**Implementation :**

1. How you implement each of requested versions, especially for the

hybrid parallelism.

在一般的版本中，我使用了pthread 並結合了sse2的方法來達成SIMD的要求。

在hybrid 版本當中，我使用了omp與mpi的結合。基本上作法與在pthread 當中一樣，只不過每個的partition都是固定的。



沒有更進一步優化是因為在11/7保持rank 2 因此認為不需要太多的優化了。

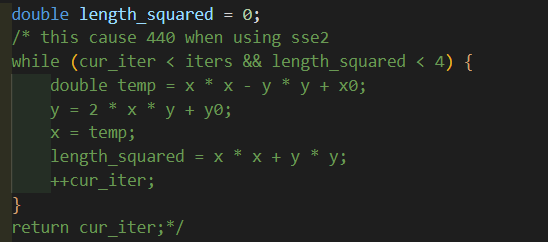
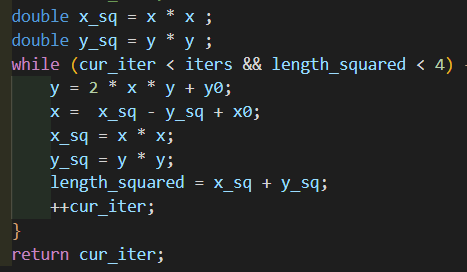
1. How do you partition the task?

兩種版本我都將width 與height 中比較長的那個做split之後再assign給不同的thread。

1. What technique do you use to reduce execution time and increase scalability?

我用的是很簡單的方法，就是將一個row 或是column當作一個Unit並使用類似omp dynamic的方式來做job assign，同時使用sse2來做double的execution time

1. Other efforts you made in your program

我將左邊的方式改為右邊 這樣可以少宣告一個temp 變數

<https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html>

<https://stackoverflow.com/questions/14492436/g-optimization-beyond-o3-ofast>

我同時參考了幾個CFLAGS來達成最優解。



我有試過Ofast，並會產生wa猜測是精準度問題使其無法正確被執行。

**Experiment & Analysis & Discussion:**

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.

Methodology

1. System Spec (If you run your experiments on your own machine)

Please specify the system spec by providing the CPU, RAM, storage

and network (Ethernet / InfiniBand) information of the system.

沒有，我使用的是apollo

1. Performance Metrics

How do you measure computing time of your programs? How do you

compute the values in the plots?

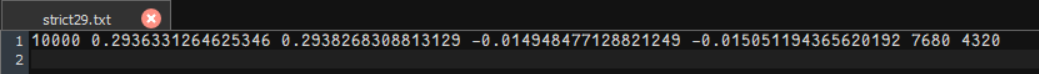
在單個node的pthread裡面 我使用的是 <sys/time.h> 裡面的gettimeofday()$

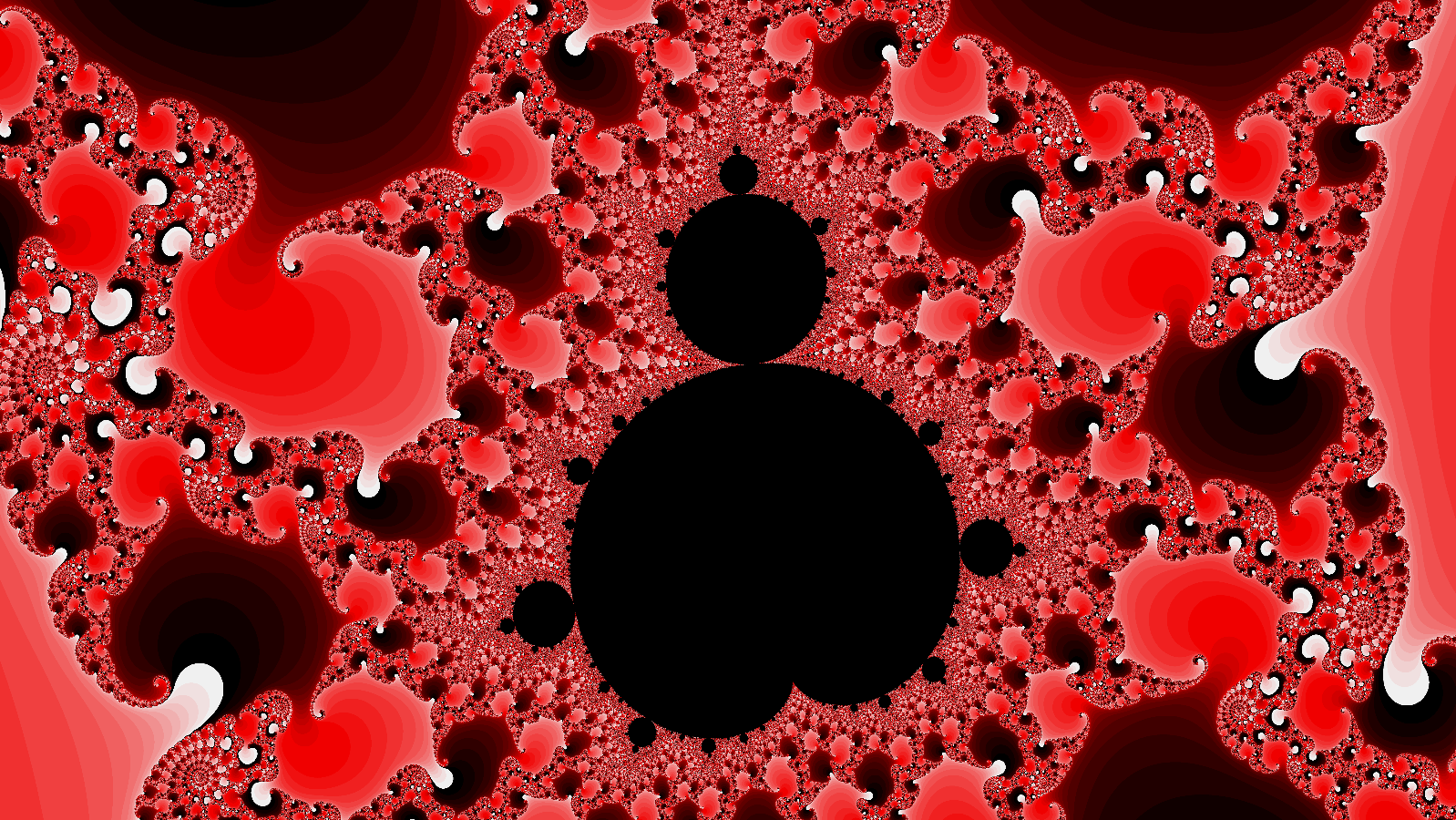
計算從抓完width , height … 開始，到write\_png 前( 我試過用<time.h>裡面的clock() 來做計算 結果發現並不會跑出正確的答案 ) 。

至於多個node作法跟上次一樣，都是使用<Wtime>來計算，而計算的地方與single node一致。

(於mpi init後至write\_png 前 )

ii、Plots: Scalability & Load Balancing





上述的圖為strict 29 。由圖可見其複雜性與精密性。因此我選擇這張圖做為baseline。

(a). Conduct strong scalability experiments, and plot the speedup. The

plot must contain at least 4 different scales (number of processes,

threads) for both single node and multi-node environments.

以下圖表 : 橘色皆為ideal狀態 藍色為實際值

Single node :

可以看到在single node裡面，我的scalability 非常的好，我想撇除在thread = 8 可能有一點差距以外基本上與ideal 都十分接近。推測是因為naïve方式很不錯。而且本身這份code很適合做平行計算。

MultiNode :

在multinode這邊我的scalability也不錯，因為在一開始就已經assign 好了每個node需要做哪些事情，因此我的communication time占比很少。猜測也是因此獲得不錯的scalability。

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

your experiments by plots.

我combine不一樣的Node數，發現基本上對於scalability 這次是非常成功的。在最差的數據中只與理論差距不到10%。

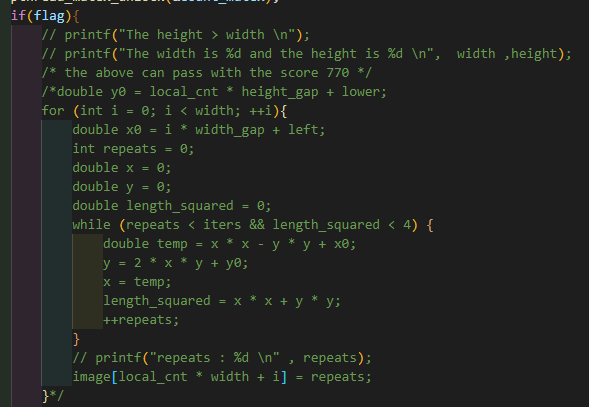
Others

You are strongly encouraged to conduct more experiments and

analysis of your implementations.

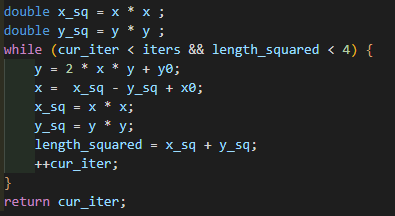
在這次作業裡面，總共做了幾個優化。

首先我用最naïve的方式來做for loop



並得到了770 的成績，接著我使用了sse2 將上述優化並變成了 450。

而底下是我的final output



我發現有一個temp會時時刻刻的在計算，以我們在計結的角度來看這樣會浪費掉指令。因此我將發放到外面。( sse2 的實作也有做更改 ) 並且調整Cflags 進而得到了成績的update

450 -> 416

至於CFlag，我試了幾種以後發現其對於整個影響並不算大 (猜測主要是因為最快速的優化同時會將精準度降地進而造成wa ( -ffast-math) 而其餘的在不影響整個正確性底下能做得十分稀少。

同時，沒有更進一步找尋cflag是因為在我前面的前提下，我無法判斷是因為server的奇蹟抑或是flag挑選的正確。

**Experience & Conclusion**

1. Your conclusion of this assignment、What have you learned from this assignment?

這次作業我對於openmp, pthread有這深層的了解，同時也對cflag 有個更多的想法，雖然在flag上無法對這次作業有多大的進步，但對於cflag的利弊我都有看到。

1. What difficulties did you encounter in this assignment?

Server過於老舊，讓每次跑出來的效果都有差距，整體會差到20~30%進而造成很多誤判與無法真正體現差距。