2022 NYCU OS HW2 report

運管 11 0713216 龔祐萱

Q1. (5pts)

Briefly describe your design for the add, multiple function of matrix, the thread management. Also, describe the number of threads in the Multi-thread program.

我先各自把加法和乘法分成 1, 2, 4, 5, 10, 20 個 thread，每個 thread 平分 500 筆 data 分別執行 5 次，取平均執行速度前三快的 thread 個數進行組合(結果如下表一、二)，因 thread 上限為 20 個，所以 thread 個數為 20 的選項不考慮。

表三為加法和乘法 thread 個數的組合，並分別執行五次取 real time 平均值的結果，可以知道當加法 thread 個數為 2(每個 thread 執行 250 筆資料)，乘法 thread 個數為 4(每個 thread 執行 125 筆資料)的時候，程式平均執行時間會最快，所以最後採用**加法 2 個 thread，乘法 4 個 thread 的組合**。

Real time和thread個數關係 (加法)			
Thread	user	system	real
1	0.079	0.003	0.1
2	0.077	0.002	0.082
4	0.074	0.003	0.08
5	0.074	0.005	0.08
10	0.076	0.004	0.088
20	0.083	0.005	0.084

表一

Real time和thread個數關係 (乘法)			
Thread	user	system	real
1	0.732	0.005	0.74
2	0.742	0.002	0.404
4	0.745	0.007	0.246
5	0.729	0	0.276
10	0.727	0.001	0.246
20	0.747	0.006	0.248

表二

Real time和thread個數關係(加法與乘法組合)			
Multiplitcation / Addition	4	5	10
2	0.252	0.28	0.26
4	0.266	0.304	0.274
5	0.304	0.312	0.272

表三

Q2. (15pts)

Try at least 3 kinds of number of threads, and compare the difference in time.(Take screenshots of the time of each case) Also, explain the results.

以下為不同 thread 個數組合的執行時間:

1. 加法 2 個 thread，乘法 4 個 thread

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.712u 0.002s 0:00.24 295.8%    0+0k 0+0io 0pf+0w
```

2. 加法 2 個 thread，乘法 5 個 thread

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.724u 0.005s 0:00.28 257.1%    0+0k 0+0io 0pf+0w
```

3. 加法 2 個 thread，乘法 10 個 thread

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.736u 0.005s 0:00.25 292.0%    0+0k 0+0io 0pf+0w
```

4. 加法 4 個 thread，乘法 4 個 thread

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.747u 0.006s 0:00.27 274.0%    0+0k 0+0io 0pf+0w
```

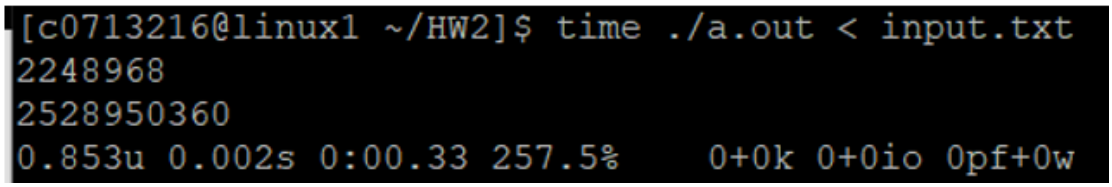
5. 加法 4 個 thread，乘法 5 個 thread

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.776u 0.009s 0:00.31 248.3%    0+0k 0+0io 0pf+0w
```

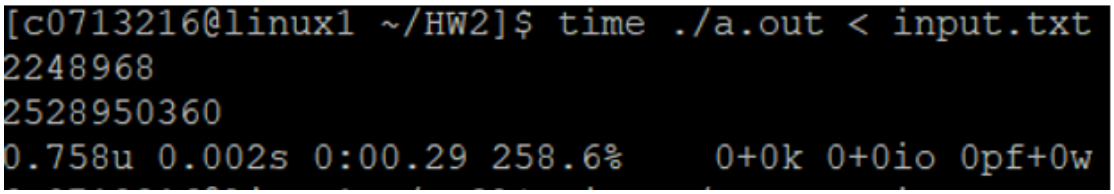
6. 加法 4 個 thread，乘法 10 個 thread

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.800u 0.010s 0:00.28 289.2%    0+0k 0+0io 0pf+0w
```

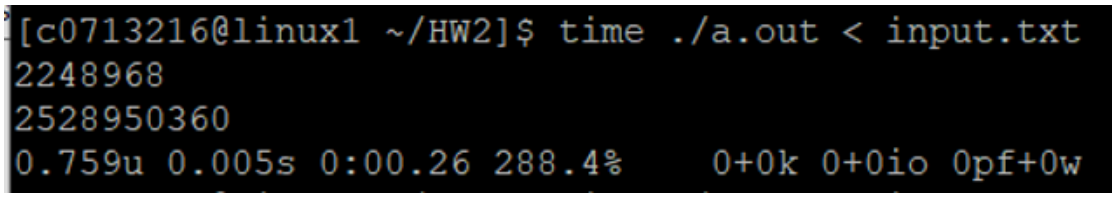
7. 加法 5 個 thread, 乘法 4 個 thread



8. 加法 5 個 thread, 乘法 5 個 thread



9. 加法 5 個 thread, 乘法 10 個 thread

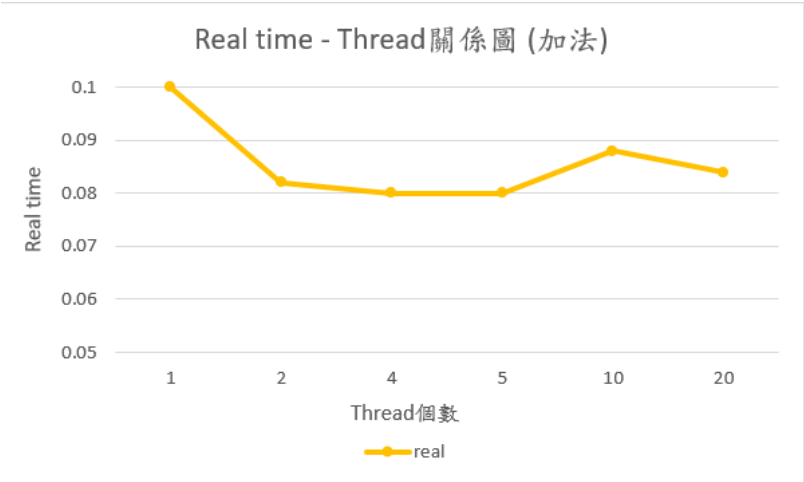


以上螢幕截圖為執行 5 次中的其中一次，為了使結果更直觀以表四(平均執行時間)來進行解釋。從表四可知加法為 2 個 thread 以及乘法為 4 個 thread 的組合的平均執行時間最快。

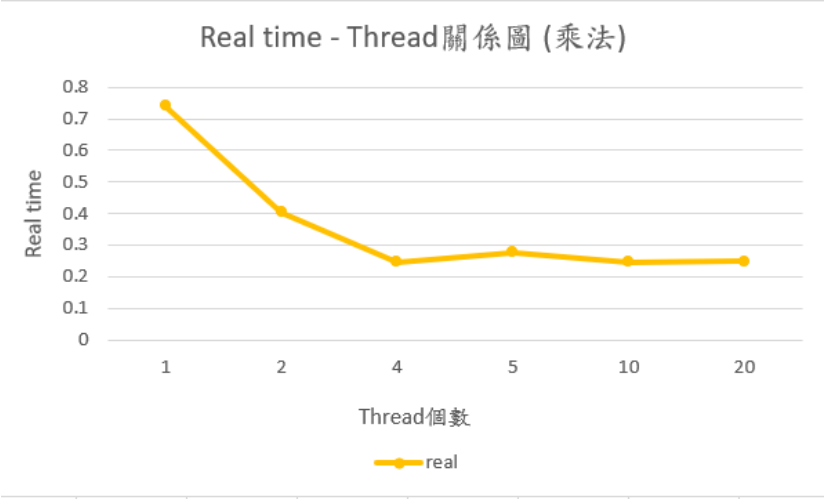
如果加法的 thread 固定為 2 個 thread，當乘法的 thread 個數增加時，平均執行時間也會增加，其中當乘法的 thread 為 5 時的平均值執行時間最長，從表六也可以知道，當乘法的 thread 為 5 時的平均執行時間也最長，可能是如果 thread 增加時， CPU 切換 thread 所需要的時間也會增加，因此即使把 500 筆資料分成多個 thread 來執行，整體的執行時間仍會增加。當乘法的 thread 個數固定，加法的 thread 增加時的情況也同理。

Real time和thread個數關係(加法與乘法組合)			
Multiplitcation / Addition	4	5	10
2	0.252	0.28	0.26
4	0.266	0.304	0.274
5	0.304	0.312	0.272

表四


Real time - Thread關係圖 (加法)
0.1
0.09
0.08
0.07
0.06
0.05
Real time
1
2
4
5
10
20
Thread個數
real

表五


Real time - Thread關係圖 (乘法)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Real time
1
2
4
5
10
20
Thread個數
real

表六

Q3. (10pts)

Show the best speedup between multi-thread and single-thread. (Take screenshots of the time of single-thread and multi-thread) Also, explain why multi-thread is faster.

1. Single-thread:

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.774u 0.003s 0:00.79 97.4%     0+0k 0+0io 0pf+0w
```

2. Multi-thread:

```
[c0713216@linux1 ~/HW2]$ time ./a.out < input.txt
2248968
2528950360
0.701u 0.010s 0:00.24 295.8%    0+0k 0+0io 0pf+0w
```

3. $\text{Speedup} = \frac{\text{real run time of single thread}}{real\ run\ time\ of\ multiple\ thread} = \frac{0.79}{0.24} = 3.2917$

4. Multi-thread 會比 single thread 快是因為如果把程式的 data 分成多個 thread 去執行的話，可以把 data 分成多個部分同時執行運算，不需要等前面的 data 執行完加法和乘法後才能繼續往下一個 data 進行運算，所以會比 single thread 快。