HW5報告

資工三 408410098 蔡x祥

演算法：

本次作業計算pi 使用上下逼近法來確定算出來答案的精準度。

原理:

類似積分的概念，將1/4圓從x軸上分成n等分,將n個長方形的面積加總。 每個長方形的長是固定的（r/n），寬則是每一長方形的兩個落於x軸的頂點中，若是取較小的x對應的高y，加總出來的面積就是upperbound (>1/4圓面積);如果是取較大的x對應的高y,加總的面積就是lowerbound(<1/4圓面積)。 當n->越大，則upperbound越趨近 lowerbound，，兩者亦越區近於1/4圓面積。 當n = 無窮大時,upperbound = lowerbound = 1/4圓面積。

實做:

這隻程式我計算半徑為2的圓的1/4面積，即是 2*2pi/4 = pi,故答案即是pi。 我將x軸從0-2分為1000000000份並開啟多個thread, 每個thread處理一樣多的upperbound中的小長方形們。 最後將其加總起來，就是upperbound的面積。 得到upperbound的面積後，lowerbound的面積就是把所有的upperbound中的小長方形左移一個unit後的面積，故只要將upperbound減去平移後多出來的左邊那一塊即可（就是x=0 y=2在upperbound中的那一塊）。 最後去比較upperbound跟lowerbound，相同到小數點第幾位數就是精準的範圍。 (1000000000 可以精準到小數點後第8位)

使用time 函數觀察結果：

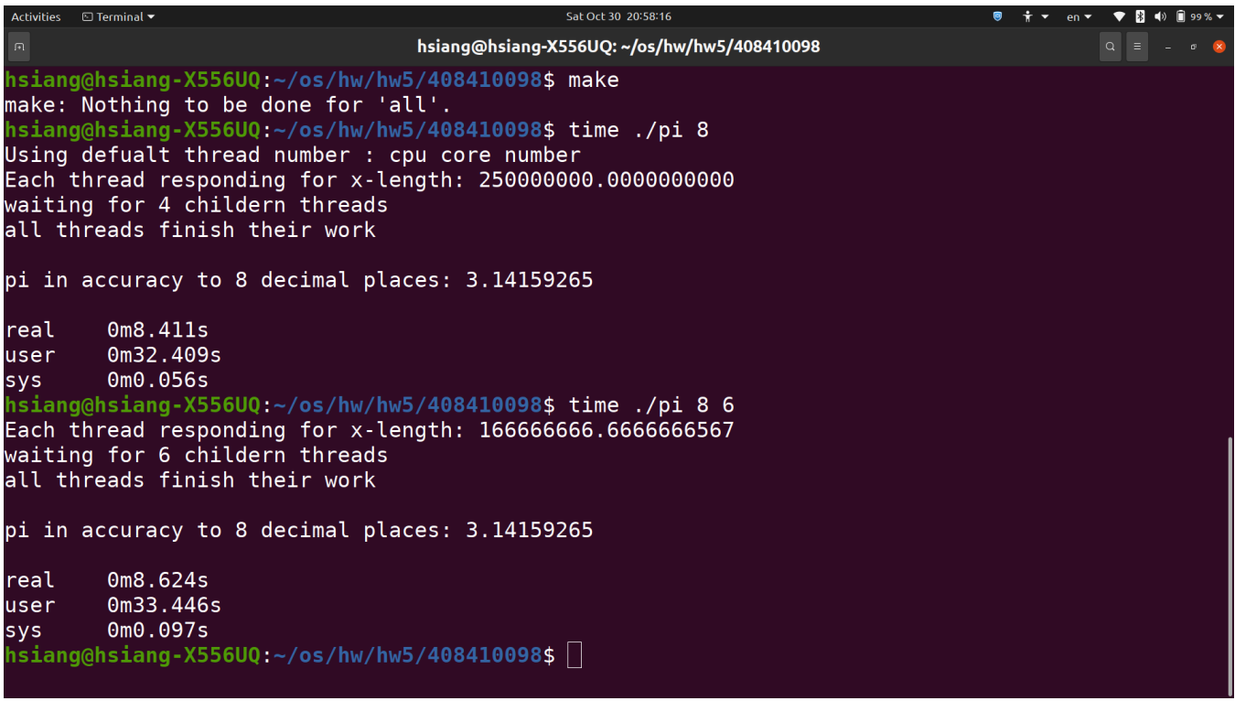
real time：於cpu上的計時之時間，是8.736s。

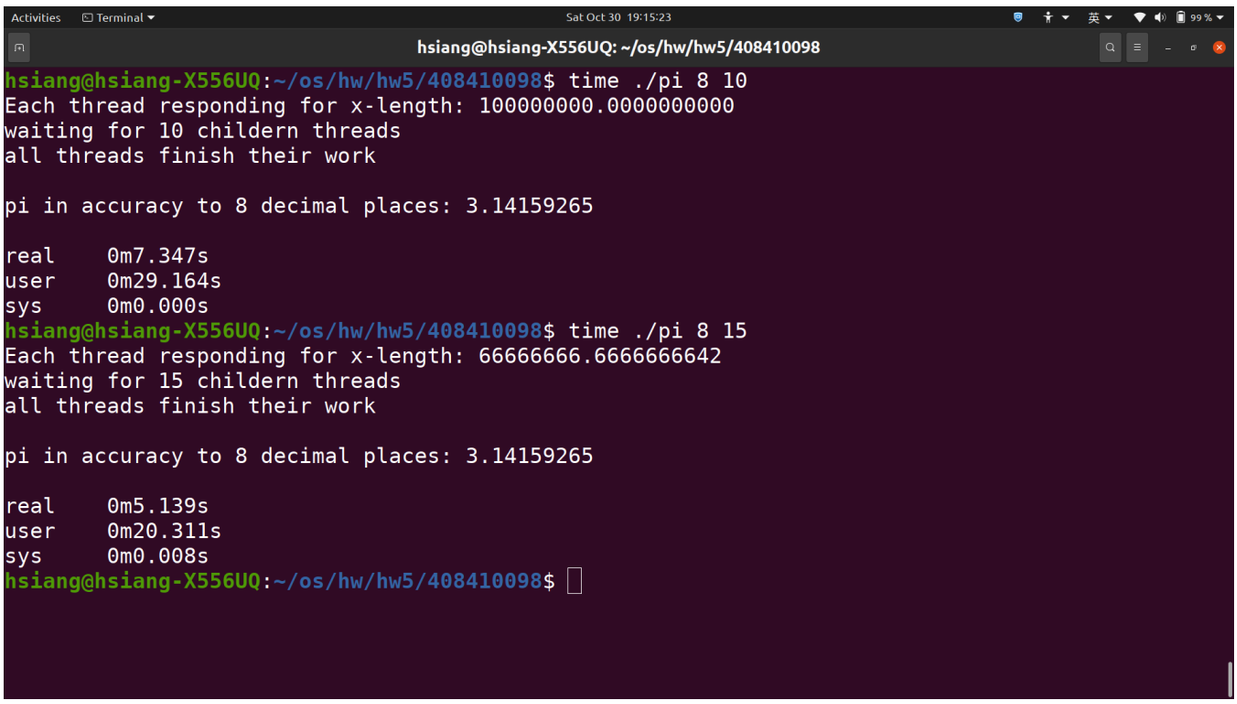
user time：運行於user mode的時間,是34.421s。

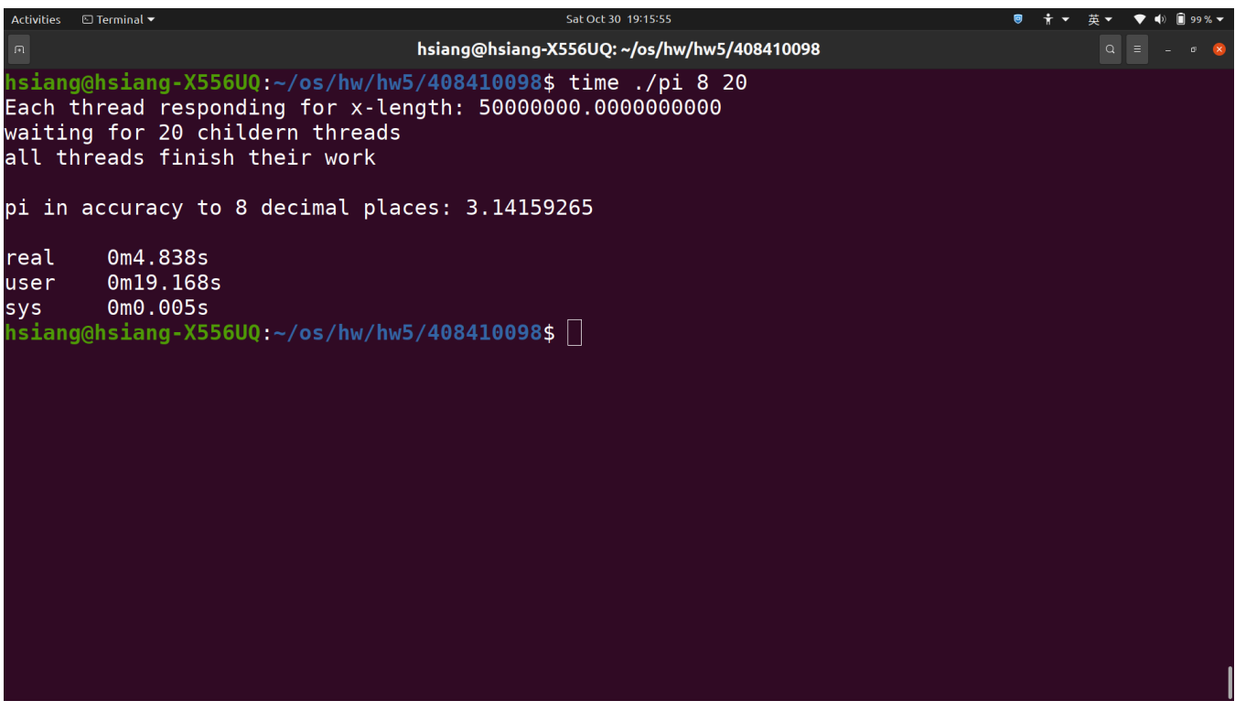
system time： 為核心運行的時間，是0.013s。

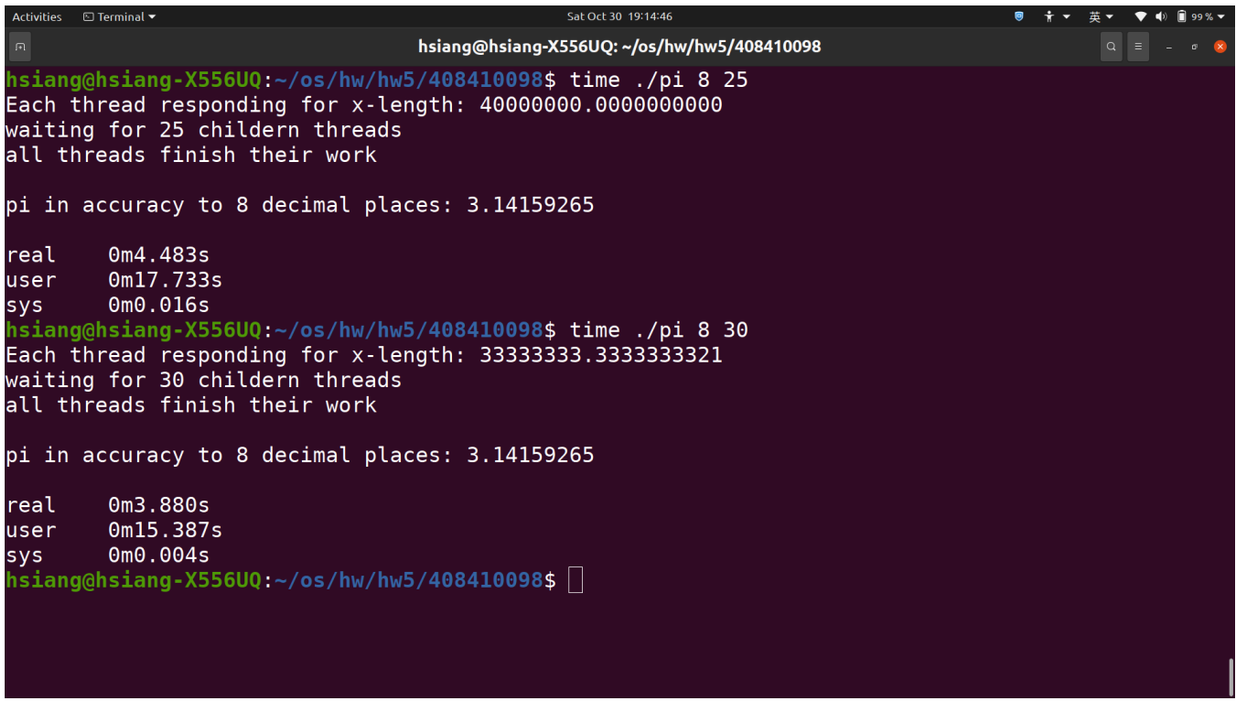
我自己的電腦是4核心，可以看到real time*4 約等於 user time。之所以沒有完全等於，可能是因為還有一些io等與硬體層之運算。

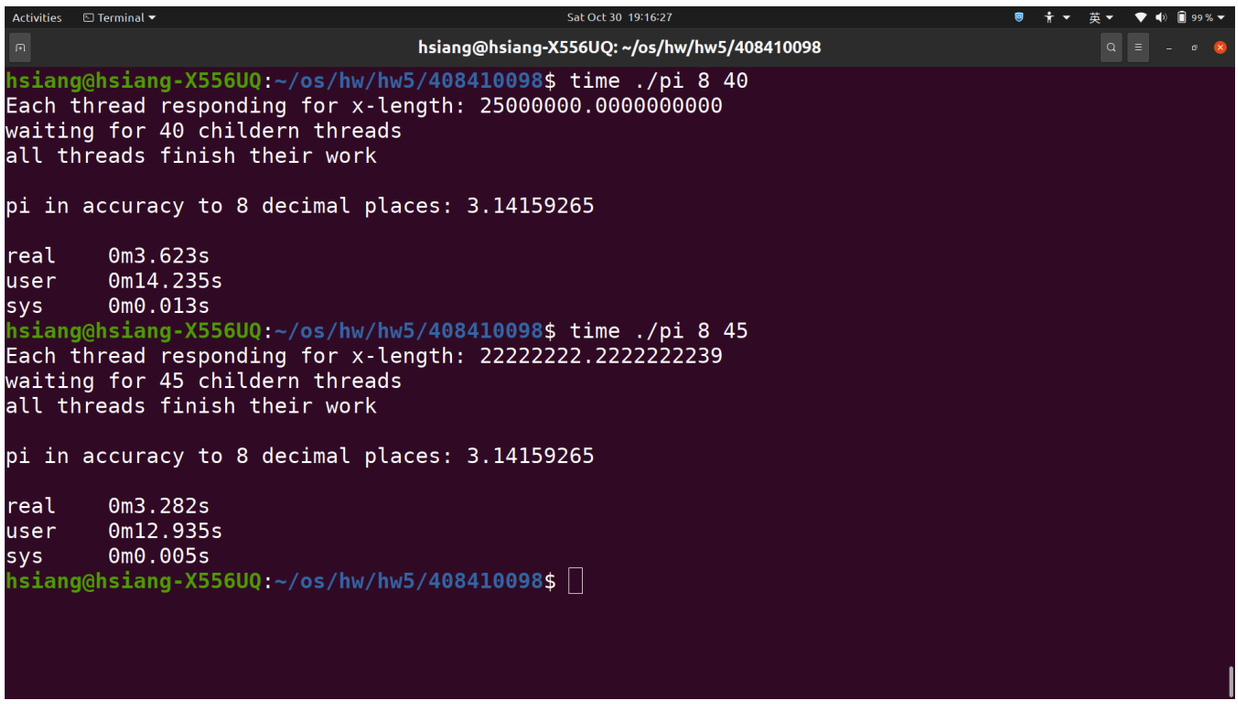
不同的thread數量


Activities
Terminal
Sat Oct 30 20:58:16
en
99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ make
make: Nothing to be done for 'all'.
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8
Using defualt thread number : cpu core number
Each thread responding for x-length: 250000000.0000000000
waiting for 4 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m8.411s
user 0m32.409s
sys 0m0.056s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 6
Each thread responding for x-length: 166666666.6666666567
waiting for 6 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m8.624s
user 0m33.446s
sys 0m0.097s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$


Activities
Terminal
Sat Oct 30 19:15:23
英
99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 10
Each thread responding for x-length: 100000000.0000000000
waiting for 10 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m7.347s
user 0m29.164s
sys 0m0.000s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 15
Each thread responding for x-length: 66666666.6666666642
waiting for 15 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m5.139s
user 0m20.311s
sys 0m0.008s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$

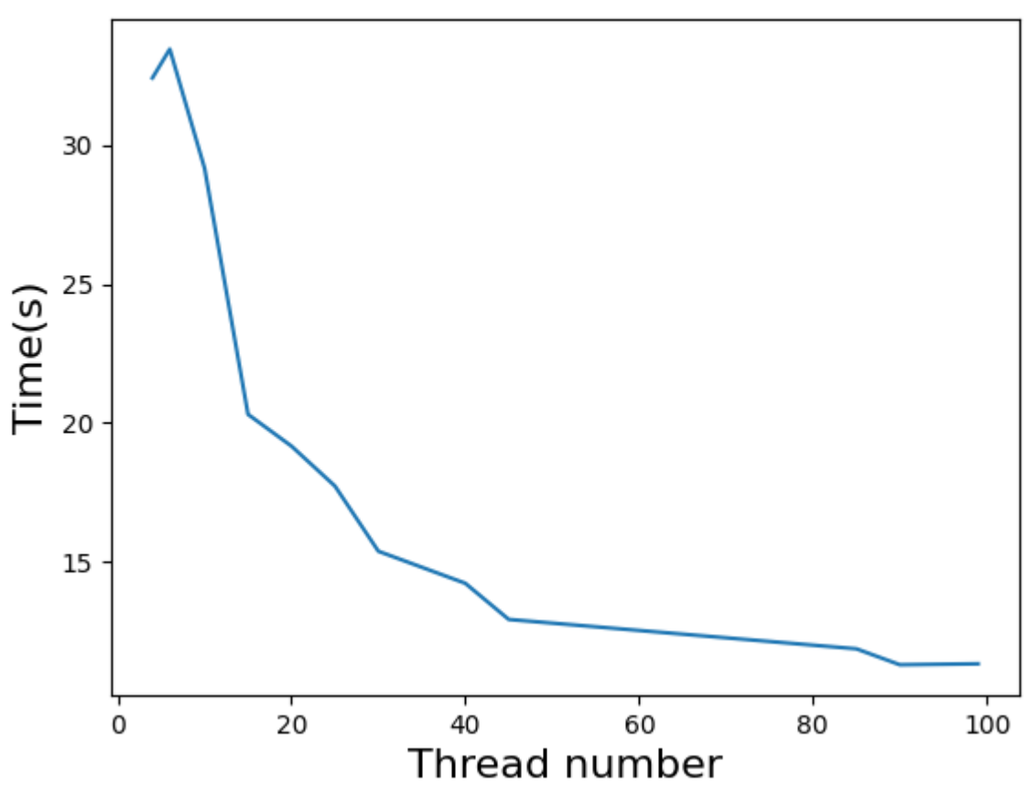

Activities
Terminal
Sat Oct 30 19:15:55
英
99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 20
Each thread responding for x-length: 50000000.0000000000
waiting for 20 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m4.838s
user 0m19.168s
sys 0m0.005s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$


Activities
Terminal
Sat Oct 30 19:14:46
英
99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 25
Each thread responding for x-length: 40000000.0000000000
waiting for 25 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m4.483s
user 0m17.733s
sys 0m0.016s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 30
Each thread responding for x-length: 33333333.3333333321
waiting for 30 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m3.880s
user 0m15.387s
sys 0m0.004s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$


Activities
Terminal
Sat Oct 30 19:16:27
英
99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 40
Each thread responding for x-length: 25000000.0000000000
waiting for 40 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m3.623s
user 0m14.235s
sys 0m0.013s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 45
Each thread responding for x-length: 22222222.2222222239
waiting for 45 childern threads
all threads finish their work
pi in accuracy to 8 decimal places: 3.14159265
real 0m3.282s
user 0m12.935s
sys 0m0.005s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$

```
Activities  Terminal  Sat Oct 30 19:14:15  英  99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 85
Each thread responding for x-length: 11764705.8823529407
waiting for 85 childern threads
all threads finish their work

pi in accuracy to 8 decimal places: 3.14159265

real    0m3.031s
user    0m11.878s
sys     0m0.021s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 90
Each thread responding for x-length: 11111111.1111111119
waiting for 90 childern threads
all threads finish their work

pi in accuracy to 8 decimal places: 3.14159265

real    0m2.862s
user    0m11.305s
sys     0m0.004s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$
```

```
Activities  Terminal  Sat Oct 30 19:13:45  英  99 %
hsiang@hsiang-X556UQ: ~/os/hw/hw5/408410098
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$ time ./pi 8 99
Each thread responding for x-length: 10101010.1010101009
waiting for 99 childern threads
all threads finish their work

pi in accuracy to 8 decimal places: 3.14159265

real    0m2.890s
user    0m11.339s
sys     0m0.037s
hsiang@hsiang-X556UQ:~/os/hw/hw5/408410098$
```



可以看到，thread數量到一定的大小之後，加速能力就減緩下來了。 原因為當thread 一多，cpu要花越多時間在context switch 多個thread上面，而造成thread的數量並非與效能線性成長。

演算法優化

在每一個thread中，他要算出的面積是它負責區段的所有小長方形面積總和。而我們的長是一樣的（等分成1000000000分），所以我將長提出來，先將所有的高加玩完後再一次性去乘長。算是有一點點小小的效能改進吧。