**Parallel Programming Exercise 6 – 13**

|  |  |
| --- | --- |
| **Author:** | 李明軒 ([b08505012@ntu.edu.tw](mailto:b08505012@ntu.edu.tw)) |
| **Student ID** | B08505012 |
| **Department** | Engineering Science and Ocean Engineering |

# Problem and Proposed Approach

(Brief your problem, and give your idea or concept of how you design your program.)

Problem:

Write a parallel program that reads from a file an where there is a matrix and play the game of life for j iterations, printing the state of the game once every k iterations.

Concept:

Each processors be responsible for certain numbers of rows.

Let m=number of rows, p=number of processors.

Distribute the row to each processor depends on n/p.

Each processor will play the game of life in their responsible part of rows and send it to processor 0. After receiving the change, processor 0 will broadcast the board to each processor.

# Theoretical Analysis Model

(Try to give the time complexity of the algorithm, and analyze your program with iso-efficiency metrics)

time complexity: Θ(m\*n\*j/p)

Iso-efficiency: mnj≧C[(p-1)(mn)+(p)(nj)]

T(n,1)= Θ(mnj), To(n,p)=(p-1)σ+pk=Θ((p-1)\*(mn)+p\*(nj))

σ=Θ(mn) , read input matrix

k=Θ(nj), processors send messages to processor 0 in every iterations.

M(n)=n^2, assume square matrix. M(C[(p-1)n/j+p])/p=C[p+ +2(p-1)n/j]

# Performance Benchmark (j=3,m=n=5,k=1)

(Give your idea or concept of how you design your program.)

Estimated execution time=χ\*j+t+λ

t= time to read input matrix, χ= time to compute one iteration , λ=time to broadcast

t≒0.0005,λ≒0.00001

χ≒ 0.000032(p=1)

χ≒ 0.000035(p=2)

χ≒ 0.000041(p=3)

χ≒ 0.000160(p=4)

χ≒ 0.000180(p=5)

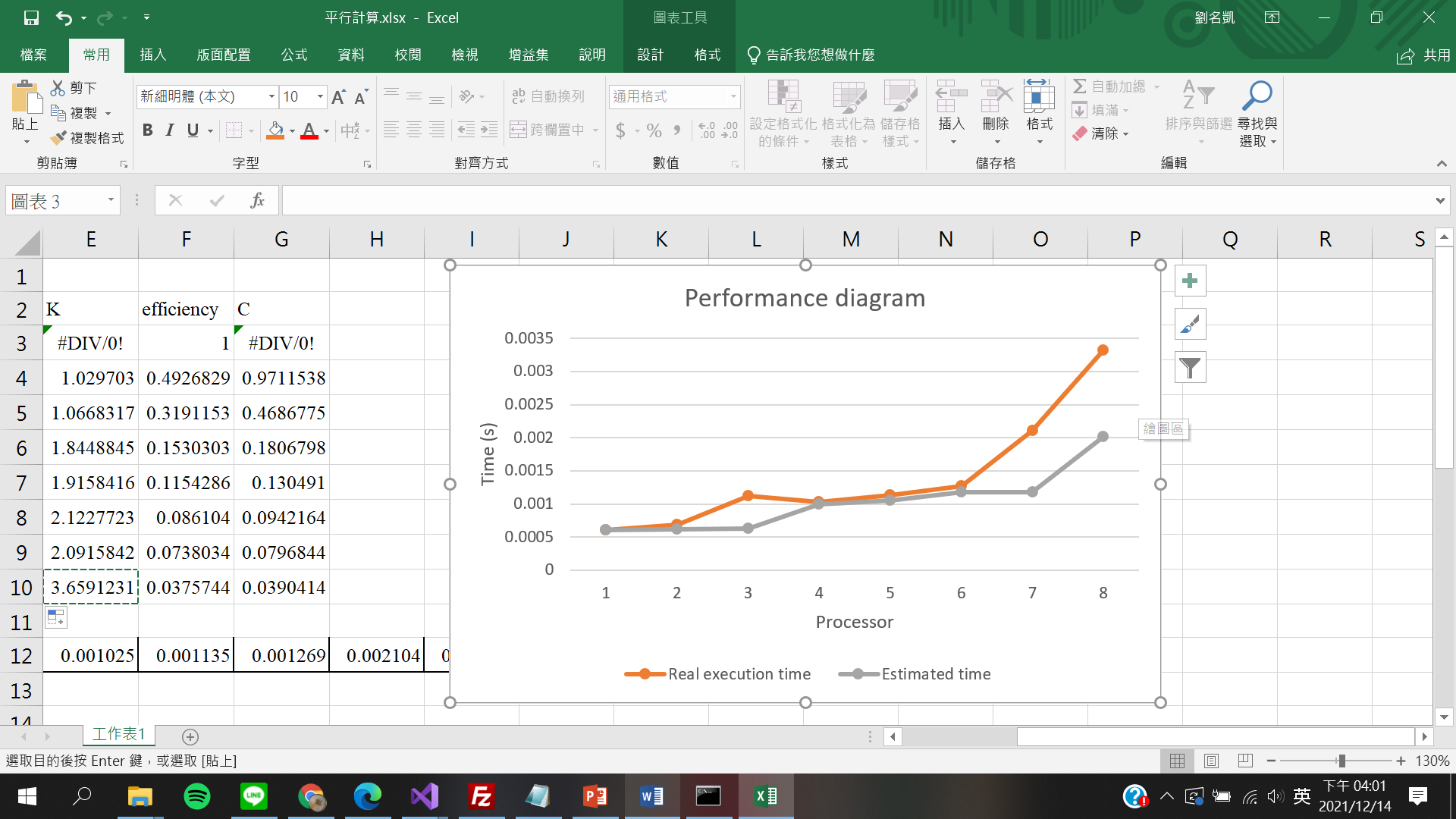
χ≒ 0.000221(p=6)

χ≒ 0.000221(p=7)

χ≒ 0.000502(p=8) (χinclude send and receive, so more processor doesn’t necessarily mean lesser computation time)

Table 1. The execution time

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Processors | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Real execution time | 0.000608 | 0.000685 | 0.001124 | 0.001025 | 0.001135 | 0.001269 | 0.002104 | 0.003315 |
| Estimate execution time | 0.000606 | 0.000615 | 0.000633 | 0.00099 | 0.00105 | 0.001173 | 0.001173 | 0.002016 |
| Speedup | 1 | 0.9853659 | 0.957346 | 0.6121212 | 0.5771429 | 0.516624 | 0.516624 | 0.3005952 |
| Karp-flatt metrics | x | 1.029703 | 1.0668317 | 1.8448845 | 1.9158416 | 2.1227723 | 2.0915842 | 3.6591231 |



# Conclusion and Discussion

1. What is the speedup respect to the number of processors used?

: As number of processor increase, speedup decrease. Because in my simulation, 5x5 matrix is too small. Execution time spend mostly on communication. Therefore, when number of processor increase, total computation time becomes longer.

1. How can you improve your program further more

: Currently, every processor has to send there computation result to processor 0 in every iteration. This may be a part to improve my program.

1. How does the communication and cache affect the performance of your program?

: In this program, the majority of total computation time is spent on communication, because 5x5 matrix is too small. Therefore, the more processor, the longer the total computation time.

1. How does the Karp-Flatt metrics and Iso-efficiency metrics reveal?

: e is increasing when number of processor increase. Parallel overhead(Send, Recv) limits the speedup. Because most of the total computation time is spent on communication.

And from the scalability function we can see that this program has very poor scalability.

**Appendix(optional):**

