**Parallel Programming Exercise 6 – 13**

|  |  |
| --- | --- |
| **Author:** | 許秉倫 ([b05901011@ntu.edu.tw](mailto:b05901011@ntu.edu.tw)) (感謝張凱捷同學協助) |
| **Student ID** | B05901011 |
| **Department** | 電機系 |

(If you and your team member contribute equally, you can use (co-first author), after each name.)

# Problem and Proposed Approach

Problem: In 1970, Princeton mathematician John Conway invented the game of Life. Life is an example of a cellular automaton. It consists of a rectangular grid of cells. Each cell is in one of two states: alive or dead The game consists of a number of iterations. During each iteration a dead cell with exactly three neighbors becomes a live cell. A live cell with two or three neighbors stays alive. A live cell with less than two neighbors or more than three neighbors becomes a dead cell. All cells are updated simultaneously. Figure 6.12 illustrates three iterations of Life for a small grid of cells. Write a parallel program that reads from a file an m x n matrix containing the initial state of the game. It should play the game of Life for *j* iterations, printing the state of the game once every *k* iterations, where *j* and *k* are command-line arguments.

Proposed Approach: 我參照張凱捷同學的做法，按row去切block，由processor 0負責讀檔，接著把n, m broadcast給每個processor，再把board scatter到每個processor。

每個iteration中，block會將邊界的row傳給上下兩個block，如此一來就可以獲得足夠的資訊來更新cell。

當要印出board時，我們將資料都gather到processor 0再一併印出。

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

# Theoretical Analysis Model

X: 更新一個cell的時間

n: # of row

m: # of col

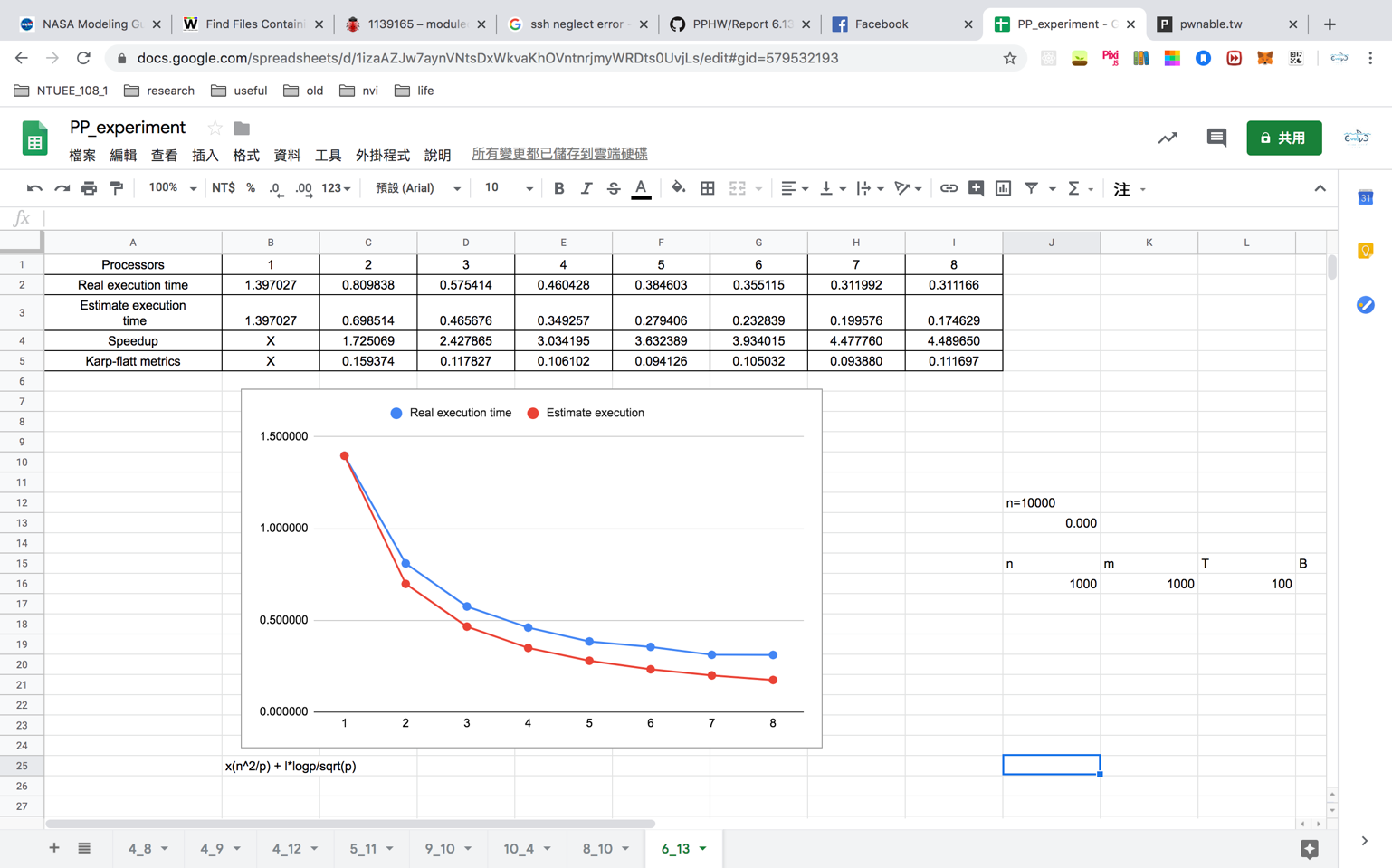
p: # of processors

Iso-efficiency:

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

# Performance Benchmark

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



# Conclusion and Discussion

1. Speedup隨processor數而提高
2. 改成block decomposition
3. Communication佔的比例不少，因為每個iteration都要交換資料
4. 由K可看出serial佔了不少比例。Iso可看出scalability還不錯。

(Discuss the following issues of your program

1. What is the speedup respect to the number of processors used?
2. How can you improve your program further more
3. How does the communication and cache affect the performance of your program?
4. How does the Karp-Flatt metrics and Iso-efficiency metrics reveal?

)

**Appendix(optional):**

(If something else you want to append in this file, like picture of life game)