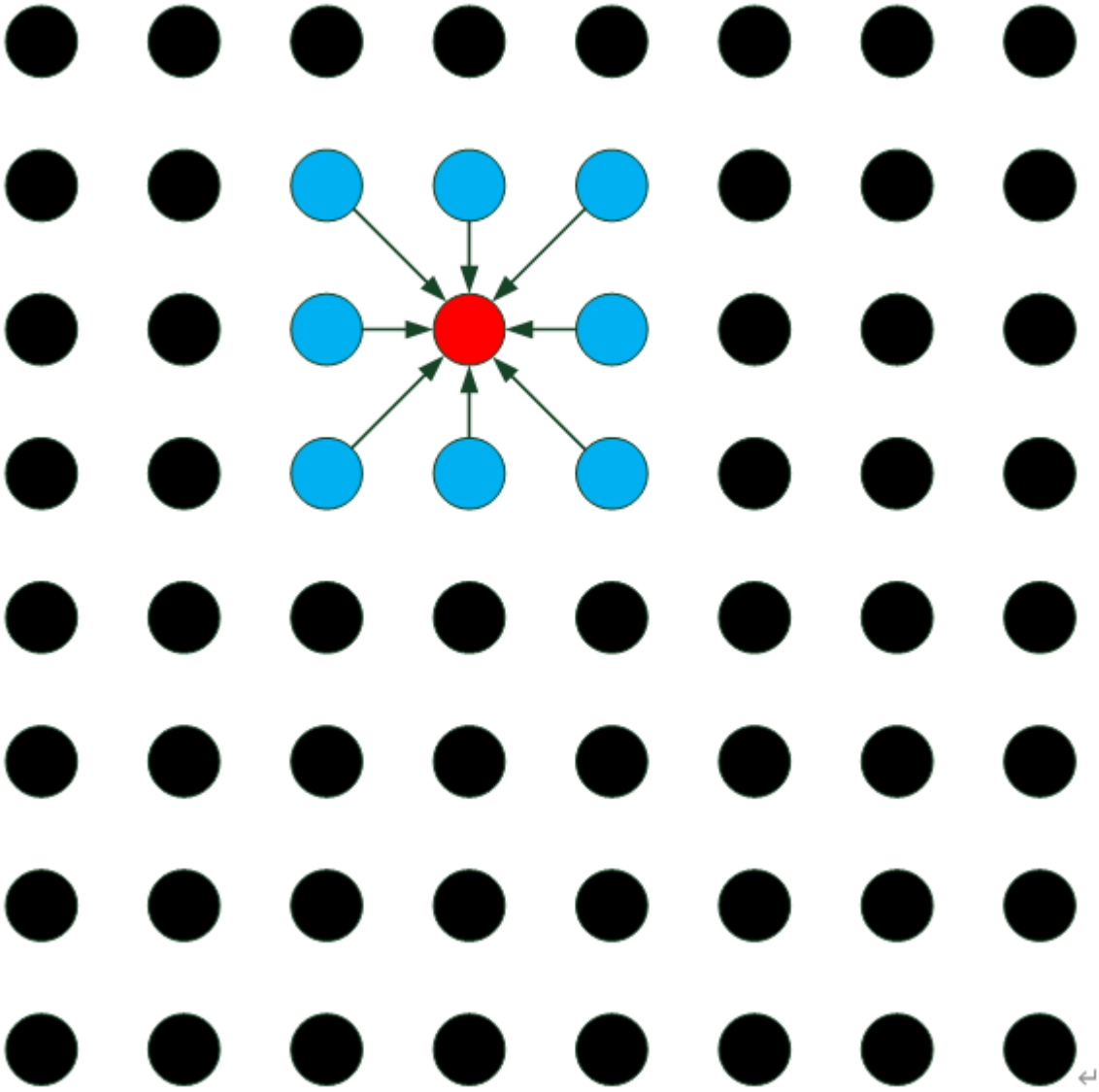
多處理機平行程式設計 2021 fall 作業二說明

- 多處理機平行程式設計 2021 fall作業二說明
  - 題目
    - 1. 影像平滑
    - 2. 平行化 odd-even sort
  - 如何使用叢集電腦
  - 如何交作業
  - 計分方式

題目

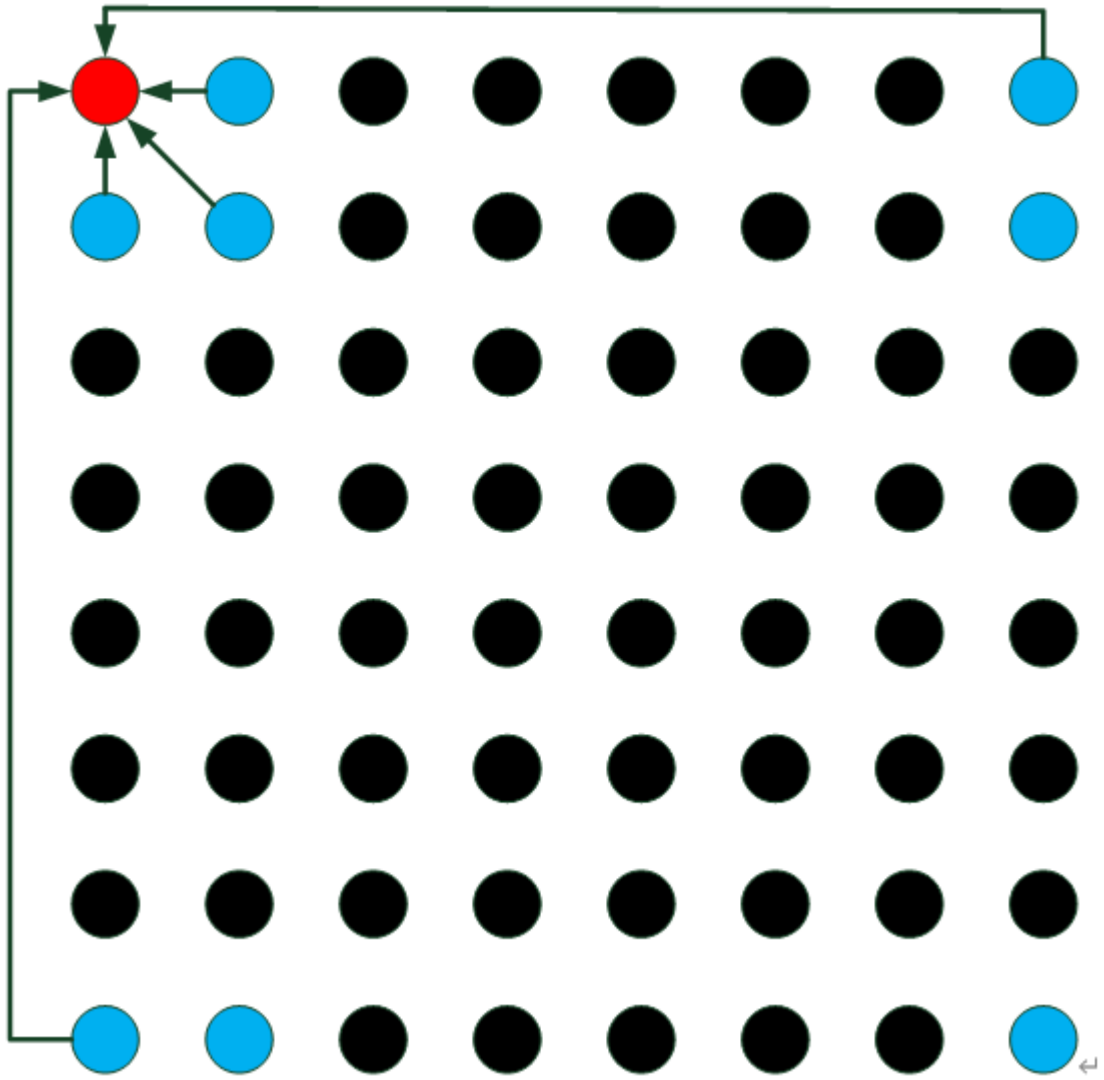
1. 影像平滑

(請使用 `MPI_Scatterv` 和 `MPI_Gatherv` 來實作) 數位化影像處理 (Digital Image Processing) 中的影像平滑運算 (Image smoothing) 可以消除或衰減影像中的雜訊或輪廓。其中一種方式為空間領域 (局部) 平均法，其方法如圖一所示：



▲圖一：像素平均示意圖

每一個點裡的像素都會與其周圍的像素進行平均。如果該像素點在邊界中，如圖二所示，則會與對面的像素進行平均運算。



▲圖二：邊界像素平均示意圖

現在我們有一個單機的平滑化運算程式，此程式讀取一個 BMP 圖檔 (位元深度 24 bits)，其像素資料儲存在一個結構矩陣裡：

$$A[Height][Width] = \begin{bmatrix} BGR & \dots & BGR \\ \vdots & \ddots & \vdots \\ BGR & \dots & BGR \end{bmatrix}$$

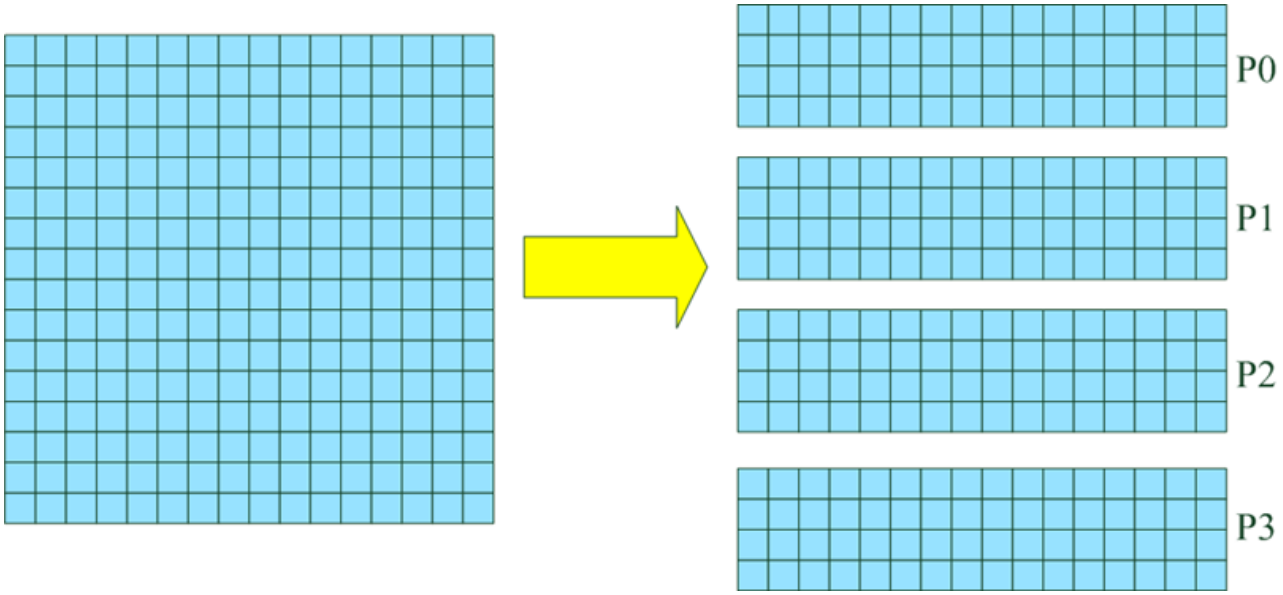
其中 B 儲存像素 Blue 的資料， G 儲存像素 Green 的資料，R 儲存像素 Red 的資料，且 BGR 的資料型態均為位元 (Char)。每次平滑運算會分別把 BGR 與周圍的 BGR 進行平均運算。舉例來說，當我們把圖三進行 1000 次平滑化運算後，會得到圖四的結果。請試著將此程式進行平行化，並且印出執行時間。程式碼與圖片請至 moodle 下載。原程式碼使用 C++ 撰寫，可以繼續使用 C++ 改寫，並改用 `mpiicpc` 編譯，或是自行改寫成 C。

▲圖三：原始圖片

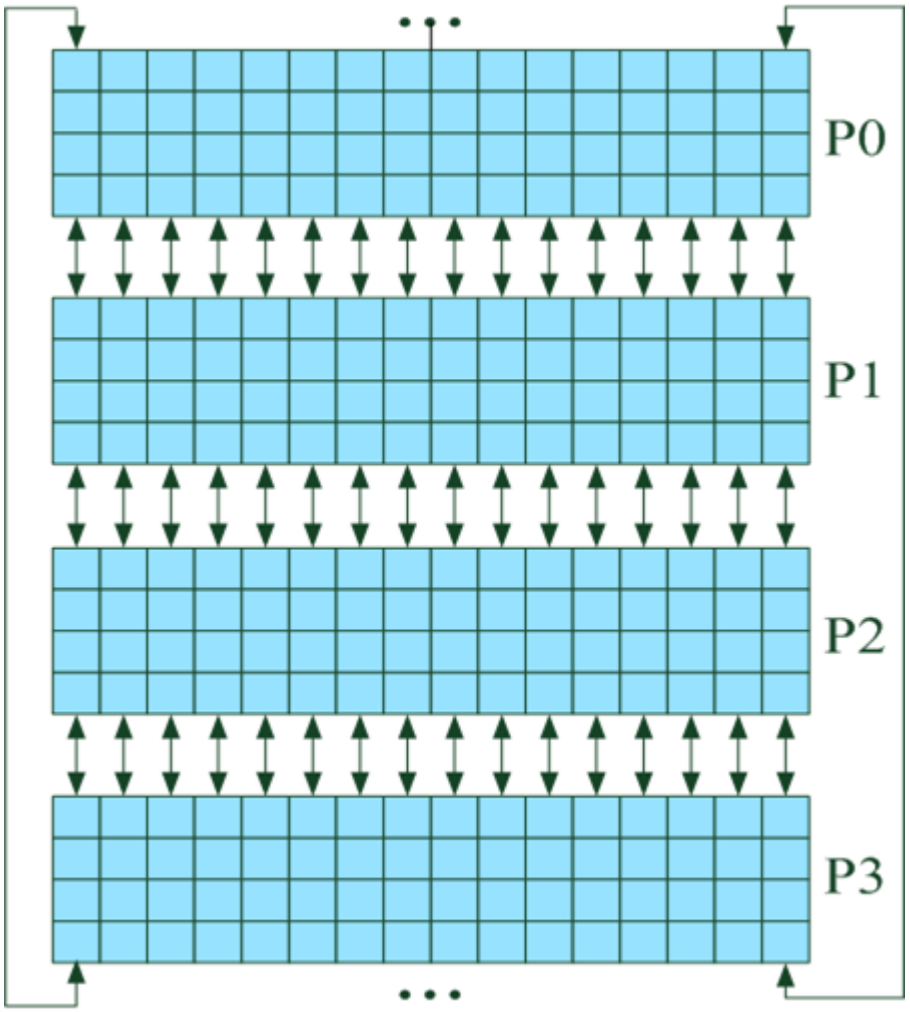
▲圖四：1000次平滑化後的結果

實作方法參考

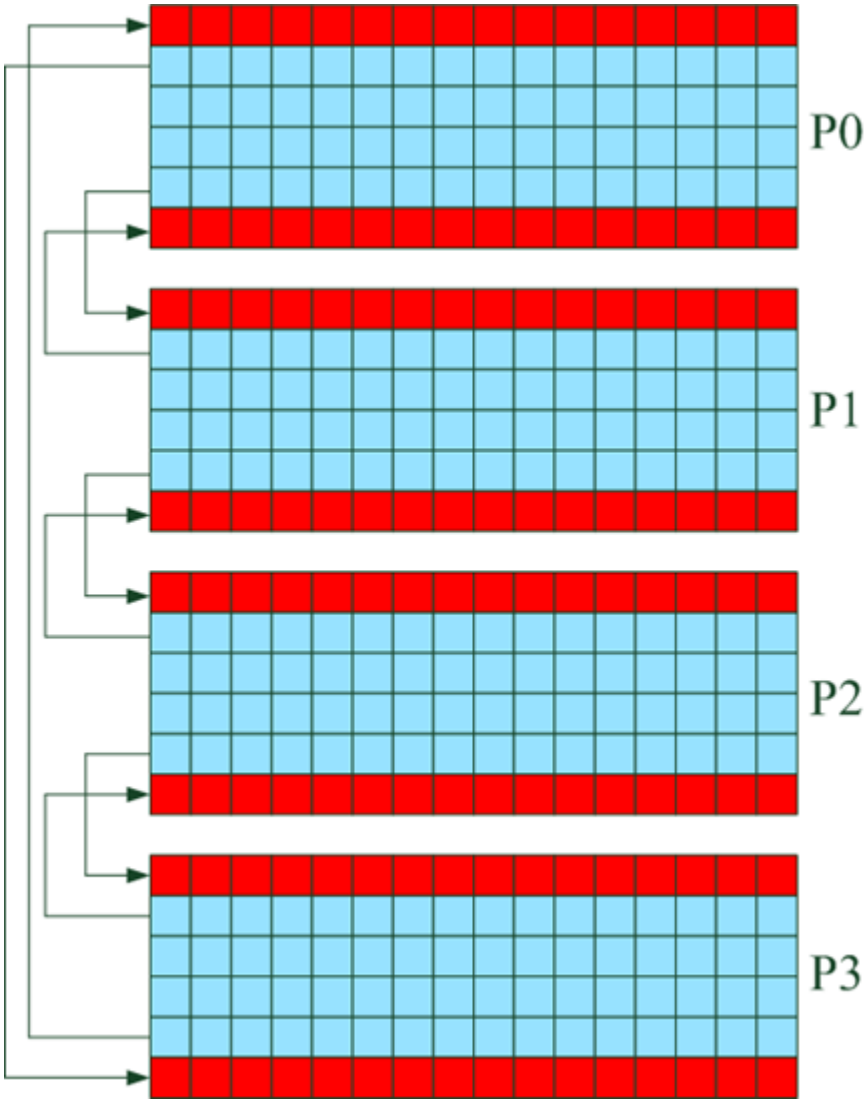
- 影像分割



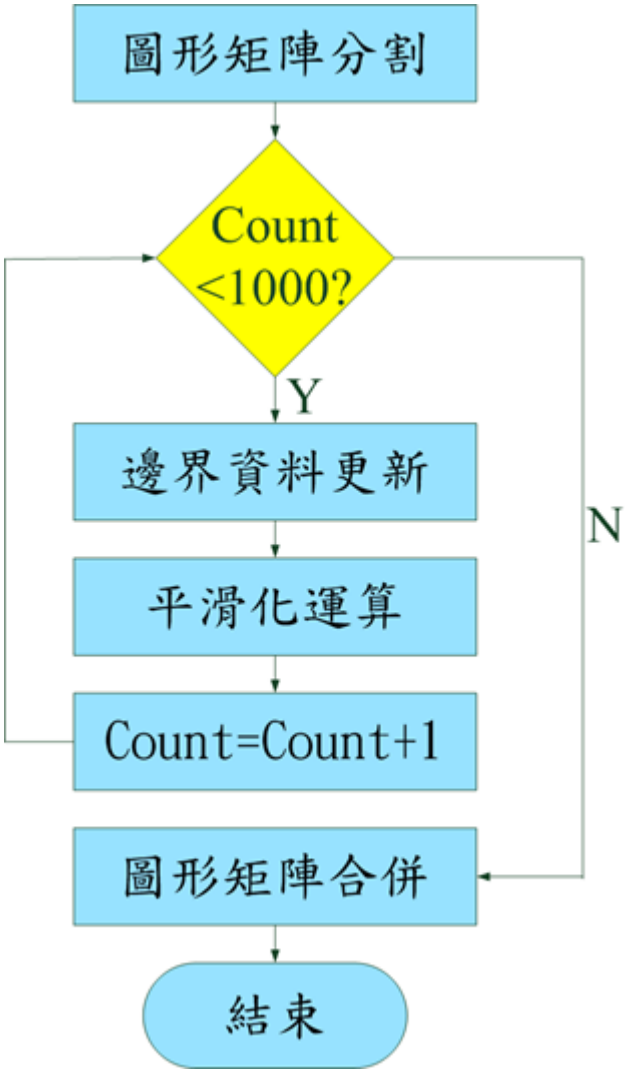
- 資料的傳送



- 設定邊界暫存空間



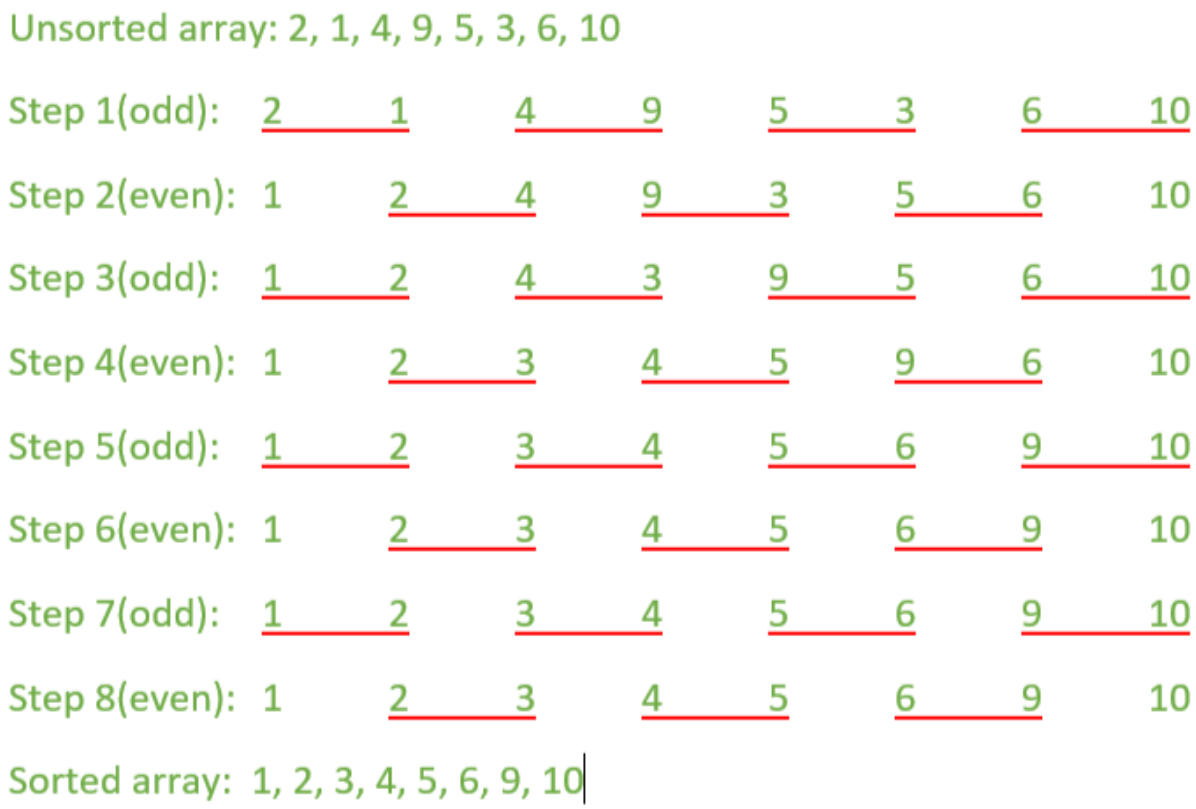
- 平行化流程



⚠ 注意重點

- 如果程式執行過久請自行取消 (`ctrl + c`)
- 此程式在 16 個 process 之下，大約執行 4.3 秒。

2. 平行化 odd-even sort (https://en.wikipedia.org/wiki/Odd%E2%80%93even_sort)



▲Image Source : Odd Even Transposition Sort / Brick Sort using pthreads (https://www.geeksforgeeks.org/odd-even-transposition-sort-brick-sort-using-pthreads/)

Parallel odd-even sort starts with `n / comm_sz` keys assigned to each process. It ends with all the keys stored on process 0 in sorted order. Write a program that implements **parallel odd-even sort**. Process 0 should read in `n` and broadcast it to the other processes. Each process should use a random number generator to create a local list of `n / comm_sz` ints. Each process should then sort its local list, and process 0 should gather and print the local lists. Then the processes should merge the global list onto process 0, which prints the result.

🎶 小提示：ch3 講義 p.140, 141

如何使用叢集電腦

請參照 作業一 (https://hackmd.io/@idoleat/ry9N6eMVF#How-to-use-real-cluster-to-test-your-MPI-program)

另外，伺服器目前有多裝了 `tmux` 和 `git` ，有需要的同學請自行服用。

如何交作業

1. Leave your code under your home directory (`mv yourfile.c ~`) with correct file name: `h2_problem1.c` (or `.cpp`) for problem 1 and `h2_problem2.c` for problem 2. Please don't copy others' code.

2. Upload a report (in `.md` or `.pdf`)(in Chinese or English) to moodle about
    - What have you done
    - Analysis on your result
    - Any difficulties?
    - (optional) Feedback to TAs

Deadline: 2021/10/29 23:59:59

Please report any server mis-configuration you found. TAs are new to System/Network administration. We will appreciate your report.

計分方式

- 程式 style 25%
    1. 巢狀結構需用階層式編排。
    2. 適當地使用空白列來區隔功能上無關的程式碼，使你的程式段落分明。
    3. 清楚且詳細的註解。
- 結果正確無誤 20%
  平行程式執行的結果需與循序程式執行的結果一致。可用 `diff` 指令驗證輸出結果是否正確， `diff` 會印出 2 個檔案的不同處。假設平行程式的輸出檔為 `output1.bmp` ，循序程式的輸出檔為 `output2.bmp` ，執行 `diff output1.bmp output2.bmp` 的結果應該沒有任何輸出
- 效能 30%
  時間計算方式如下（不計算圖檔輸出入時間）

    ```
    讀取圖檔

    MPI_Barrier(comm);
    Start_time = MPI_Wtime();
    …
    平滑化處理
    …
    MPI_Barrier(comm);
    Finish_time = MPI_Wtime();

    寫入圖檔
    ```

  要用 `MPI_Scatterv` 和 `MPI_Gatherv` 來實作
  此部份分數需在結果正確無誤的情況下才計分。
- 平行程式的觀察報告 25%
  請觀察作業中，改變不同的平滑化的次數與不同數量的處理程序對執行時間有何影響。
  請寫出觀察結果，並回答為何會有這種結果，寫在結果分析中