# Problem descriptions

在這個Assignment裡，我們要用三種MPI提供的API去實作平行程式的運行，分別是MPI\_Send()+MPI\_Recv()、MPI\_Isend()+MPI\_Irecv()、MPI\_Scatter()+MPI\_Gather()。而主要要求是從Rank 0(Master)傳送資料到Rank1~4(Slave)，他們進行接收資料後印出再傳出資料讓Master可以收到Slave傳輸的資料。所以一共需要5個processer 去執行對應的program，而Master傳出的資料為“Hi rank [slave\_rank], I'm 廖洺玄from Parallel

Programming Design Course in 2024“；Slave傳出去的資料為“Rank [slave\_rank] received. Thank you.“。在印出資料的同時也需要印出MPI\_Wtime()，來讓我們知道這個program在master或是slave的每一次收到資料花費多少時間。

# Code and explanations

## PPD\_PA1\_B0829002\_1.c

A computer screen shot of a program code

Description automatically generated

### Import library 和變數宣告

這部分用於印入C的Standard library和本次作業用到最重要的MPI library，然後Main function會把使用者執行程式後面連帶的arguments引入讓MPI在initialize parallel program的時候可以使用到。proId、numPro用來儲存下面再執行parallel program的時候是哪個node在執行、以及共有多少node。double的變數用於記錄parallel program開始時的時間。

A screen shot of a computer program

Description automatically generated

### 初始化Parallel Program

這部分告訴MPI從哪裡開始執行平行的運算，然後給他proId、numPro用來記錄開出來的node的process id和共有多少個node (process)需要初始化，並在一切初始化完成後開始執行parallel program並記錄下開始時間。

A computer screen shot of a program code

Description automatically generated

### Master Parallel Program

首先先給一個共用numPro個的char array讓他們可以儲存等一下要送出去的data，sprintf會把要傳給slave的資料印到buffer的memory address上，接下來就將資料傳給第i個slave，因為是synchronous的傳資料所以會等到slave接收到才往下等待接收那個slave傳過來的資料。收到後把slave傳過來的資料印出。

A screenshot of a computer program

Description automatically generated

### Slave Parallel Program

先initialize一個空的data讓slave可以接收從master傳過來的資料。MPI\_Recv收到資料後會讓Master的parallel program可以繼續，接下來slave會把花費時間和收到的資料印出。最後把要傳給master的資料印到data’s memory address上並傳給master後等待，master 接收。

A screenshot of a computer program

Description automatically generated

### 定義Parallel Program在哪部分結束

A screen shot of a computer

Description automatically generated

## PPD\_PA1\_B0829002\_2.c

A screen shot of a computer program

Description automatically generated

### Import library 、變數宣告和初始化Parallel Program

同PPD\_PA1\_B0829002\_1.c [[按我到之前的解釋]](#_Import_library_和變數宣告)

A screenshot of a computer program

Description automatically generated

### Master Parallel Program

這邊和前面的方法不太一樣，因為要用到Asynchronous的通訊，所以我們要設一個MPI\_Request type的變數去確認他已經收到data可以進行下一步了，而從master送資料到slave的流程大致上和先前的program差不多，唯一有差的地方是要帶&request[i-1]讓之後收到的slave可以確認偵測/等待函數用的。之後便會開始接收從slave傳回來的資料，MPI\_Wait確認收到後request後便會把收到的data印出。

A screenshot of a computer program

Description automatically generated

### Slave Parallel Program

這裡也跟先前的部分很像但不太一樣的地方是我們需要收data時確認request是否完全收完後把要傳送給master的資料印在data memory address上，並傳送回master。

A screenshot of a computer program

Description automatically generated

### 定義Parallel Program在哪部分結束

A screen shot of a computer

Description automatically generated

## PPD\_PA1\_B0829002\_3.c

### Import library 、變數宣告和初始化Parallel Program

同PPD\_PA1\_B0829002\_1.c [[按我到之前的解釋]](#_Import_library_和變數宣告)，多了一個define Macro的去定義每個要傳送的字串最長長度。

A screen shot of a computer program

Description automatically generated

### Master Parallel Program

在Master裡面我們先宣告好要傳送資料的string和之後slave要接收master傳送資料的變數以及最後master要接收從slave傳回來的資料。宣告好後，用sprintf把要傳送的string印到sendBuffer的memory address上，之後用MPI\_Scatter傳送到每個node。（Scatter的詳細運作請看Discussion）

A screen shot of a computer program

Description automatically generated

在從Slave Gather後我們會把傳回來的資料要出來。

A screen shot of a computer

Description automatically generated

### Slave Parallel Program

Slave把收到的資料印出來後，用Gather發送資料回Master，讓Master有可以印出的接收資料。（Gather的詳細運作請看Discussion）

A screen shot of a computer screen

Description automatically generated

### 定義Parallel Program在哪部分結束

A screen shot of a computer

Description automatically generated

# Sampled outputs

## PPD\_PA1\_B0829002\_1.c

A screenshot of a computer screen

Description automatically generated

A screen shot of a computer

Description automatically generated

## PPD\_PA1\_B0829002\_2.c

A screenshot of a computer screen

Description automatically generated

## PPD\_PA1\_B0829002\_3.c

A screen shot of a computer program

Description automatically generated

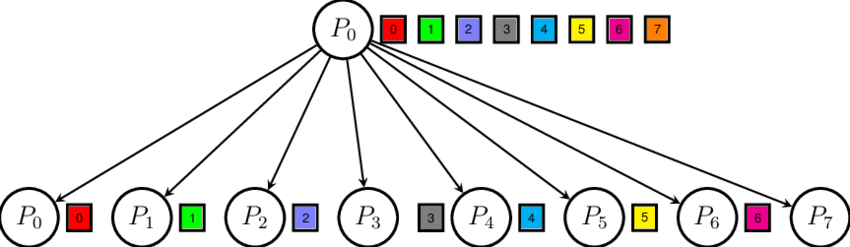
# Discussions or what I’ve learned

在這個Assignment裡面我學習到了如何利用MPI提供的API開muti-thread，讓我的Program可以平行運算。除此之外，之前在Unix修過的thread和inter-process communication的Blocking和Non-blocking也在第一個和第二個program用到了，雖然概念很像但是還是有一些值得注意的地方就是thread.h和mpi.h兩個提供的，例如thread.h要用到OS mutual exclusion的概念實作，但MPI有提供一個request可以去檢查是否已經傳送完成，相對thread.h上方便很多。

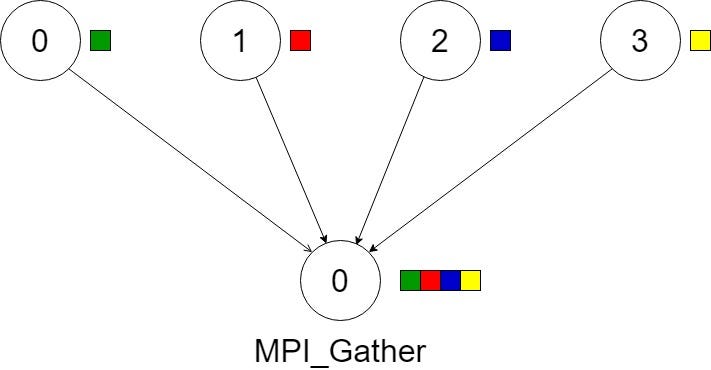
而在第一個和第二個program的Blocking和Non-blocking，其中分別代表的通訊方式為：

1. Blocking（阻塞通訊）意味著調用的函數會一直等待，直到操作完成後才返回控制權給program。這種方式在通訊過程中，發送方和接收方都需要同步操作。例如：MPI\_Send、MPI\_Recv，阻塞通訊的主要缺點是效率問題，特別是在延遲較高的情況下，它可能導致計算資源的浪費，因為處理器在等待完成通訊時可能什麼也做不了。
2. Non-blocking（非阻塞通訊）非阻塞通訊允許program在通訊操作尚未完成的情況下繼續執行，這有助於重疊計算與通訊，從而提升program的整體性能。例如：MPI\_Isend、MPI\_Irecv，非阻塞通訊需要額外的管理，因為你必須使用 MPI\_Wait 或 MPI\_Test 等函數來確認操作是否完成。這種方式適合於那些異步操作中的應用，特別是那些計算和通訊可以有效重疊的情況。[1]

在MPI\_Scatter和MPI\_Gather的功能應改是我覺得最實用的兩個平行API了，除了大幅提高我程式的精簡度以外，他用在資料傳送和收集的概念也非常有趣。MPI\_Scatter將資料從Master（or also called root）分發到參與的所有processes中，他的用法就是Master擁有一個包含資料的Array或vector（如果用C++寫得話），並將這個陣列均勻地切分成多個部分，每個部分分發給一個process（包含自己）。[2]



MPI\_Gather則會所有process中的資料收集到一個process中，用法大概就是每個process（包括Master）將其資料發送到master，master將接收到的資料組合成一個大陣列。[3]



# Reference

[1] M. Kirby, “IntroMPI.ppt,” CS 6230: High-Performance Computing and Parallelization – Introduction to MPI, https://users.cs.utah.edu/~kirby/classes/cs6230/IntroMPI.pdf (accessed Apr. 23, 2024).

[2] K. Hasanov and A. Lastovetsky, “Hierarchical optimization of MPI reduce algorithms,” *Lecture Notes in Computer Science*, pp. 21–34, 2015. doi:10.1007/978-3-319-21909-7\_3

[3] N. Joram, “Scatter and gather in MPI,” Medium, https://medium.com/nerd-for-tech/scatter-and-gather-in-mpi-e66b69366ee3 (accessed Apr. 23, 2024).