**Assignment 1**

**Group member:**

**Dian Li, Rui Wang, Ruonan Wu**

The idea of this program is to use MPI\_Irecv() to overlap the communication with the computation to calculate sum of each rows receive. We implemented our idea using the below graph:

|  |  |
| --- | --- |
| Rank 0 | Rank 0 generate and send rank 0 data (row1~2500);  Rank 0 receive and calculate sum; |
| Rank 1 | Rank 0 generate and send rank 1 data (row2501~5000);  Rank 1 receive and calculate sum; |
| Rank 2 | Rank 0 generate and send rank 2 data (row 5001~7500);  Rank 2 receive and calculate sum; |
| Rank 3 | Rank 0 generate and send rank 3 data (row 7501~10000);  Rank 3 receive and calculate sum; |

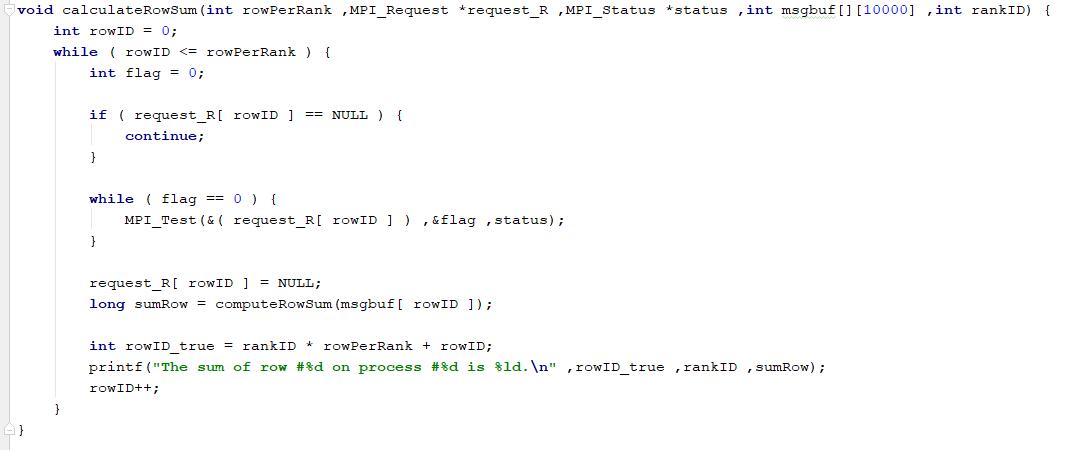
Use Rank 1 as an example, it can start to calculate the sum of the rows it already complete receiving. For example, if Rank 1 already received the data buffer of row 2600, it can start calculating the sum of row 2600 and at the same time, receive data for row 2601. This is where we can use the idea of “overlapping”. This overlap is within the same processor. Use MPI\_Test() to check if it has completed receiving data for the next row or not. Assume row 2599 data has already been received by Rank 1 from Rank 0, then:

|  |  |
| --- | --- |
| Rank 1 receive row 2600, meanwhile: | Rank 1 calculate sum for row 2599 |
| Rank 1 receive row 2601, meanwhile: | Rank 1 calculate sum for row 2600 |
| Rank 1 receive row 2602, meanwhile: | Rank 1 calculate sum for row 2601 |
| Rank 1 receive row 2603, meanwhile: | Rank 1 calculate sum for row 2602 |

Also, Rank 0, 1,2 and 3 can receive data from Rank 0 at the same time. This is “overlapping” between different processors. For example:

|  |  |  |
| --- | --- | --- |
| Rank 1 receive row 2600 | Rank 2 receive row 6000 | Rank 3 receive row 8000 |
| … | … | … |

This overlapping method is implemented using the following function *calculateRowSum()*:



In this function, if the request rowID is null. Then it means we are not receiving for that row, so breaks out of loop and continue receiving. If we have already received the data of that row, then we can use MPI\_test to see if it is a complete buffer transmission, and if it is, then next we can calculate its sum.

We have tested our code in intel MPI compilers and below is a screenshot proven that we have gotten the results from expectation:

