**中山大学计算机院本科生实验报告**

**（2024学年春季学期）**

**课程名称：并行程序设计**   **批改人：**

|  |  |  |  |
| --- | --- | --- | --- |
| 实验 | **2-基于MPI的并行矩阵乘法（进阶）** | 专业（方向） | **计算机科学与技术** |
| 学号 | **21307174** | 姓名 | **刘俊杰** |
| Email | **liujj255@mail2.sysu.edu.cn** | 完成日期 | **2024/4/2** |

## 实验目的

改进上次实验中的MPI并行矩阵乘法(MPI-v1)，并讨论不同通信方式对性能的影响。

**输入：**三个整数，每个整数的取值范围均为[128, 2048]

**问题描述：**随机生成的矩阵及的矩阵，并对这两个矩阵进行矩阵乘法运算，得到矩阵.

**输出**：三个矩阵，及矩阵计算所消耗的时间。

**要求：**

**1.**采用MPI集合通信实现并行矩阵乘法中的进程间通信；

1. 使用mpi\_type\_create\_struct聚合MPI进程内变量后通信；
2. 尝试不同数据/任务划分方式（选做）。

## 2.实验过程和核心代码

### 2.1实验过程

①首先0号进程接收矩阵A和矩阵B的维度，并初始化矩阵A、B、C。

②将矩阵A按行分给不同的进程计算，计算出平均每个进程负责的矩阵A的行数rows\_per\_proc。

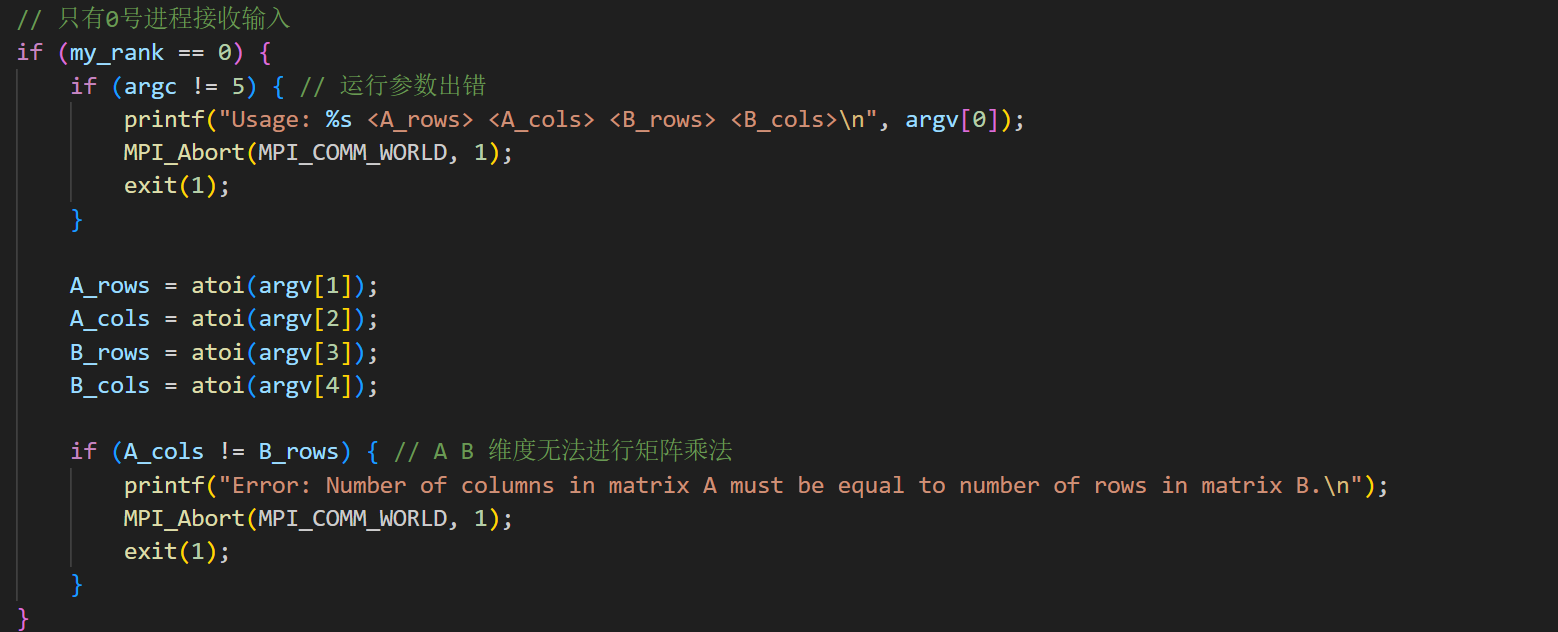
③利用MPI\_Scatter将每个进程负责的矩阵A的行分给每个进程,将矩阵A、B的维度和矩阵B用MPI\_Bcast广播给每个进程。

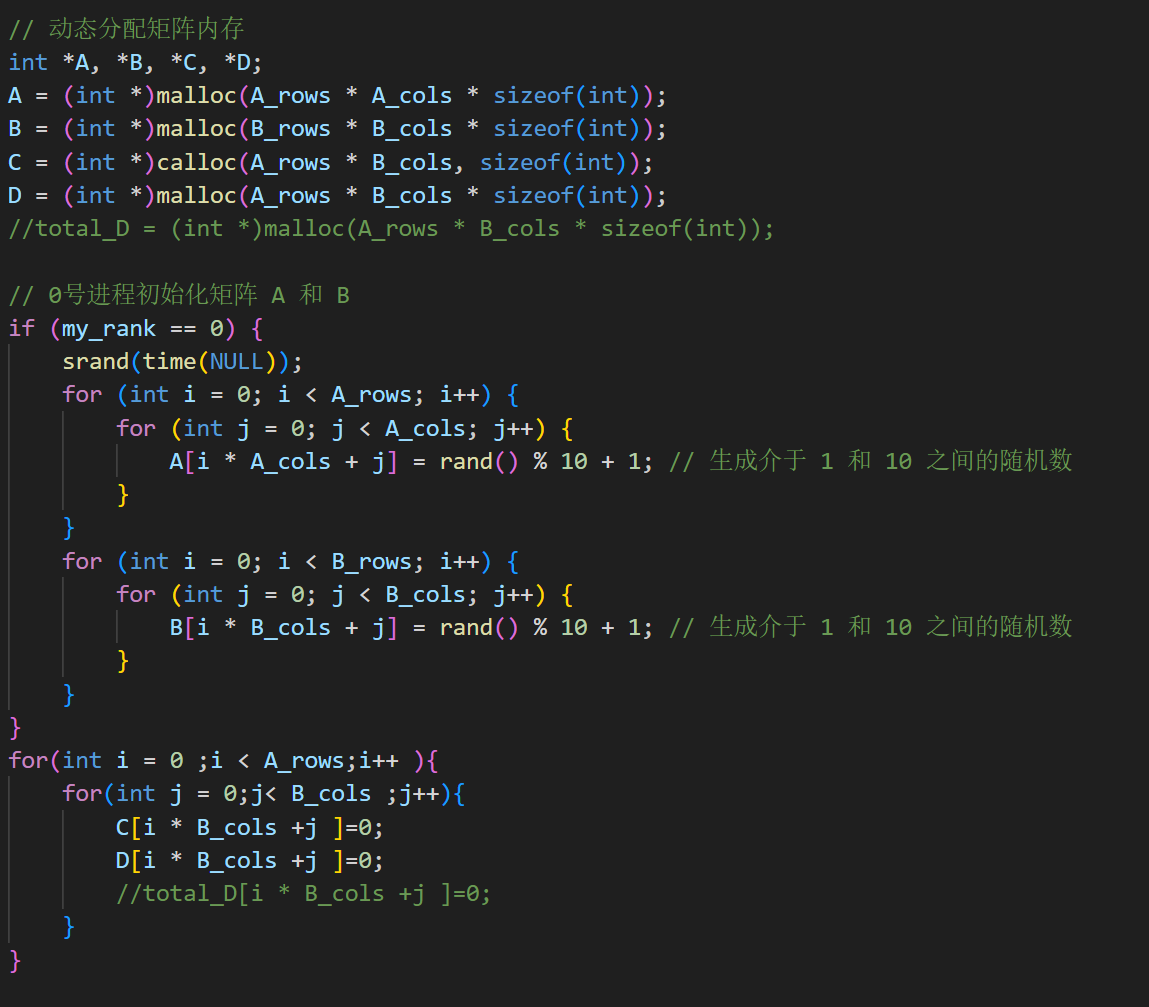
④每个进程计算出结果后，利用MPI\_Gather将结果归到0号进程中。

⑤0号进程输出结果和计算时间。

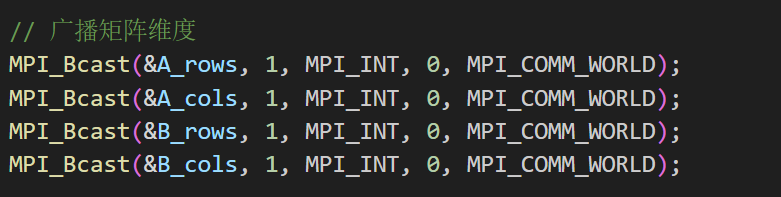
### 2.2 核心代码

①首先0号进程接收矩阵A和矩阵B的维度，并初始化矩阵A、B、C。

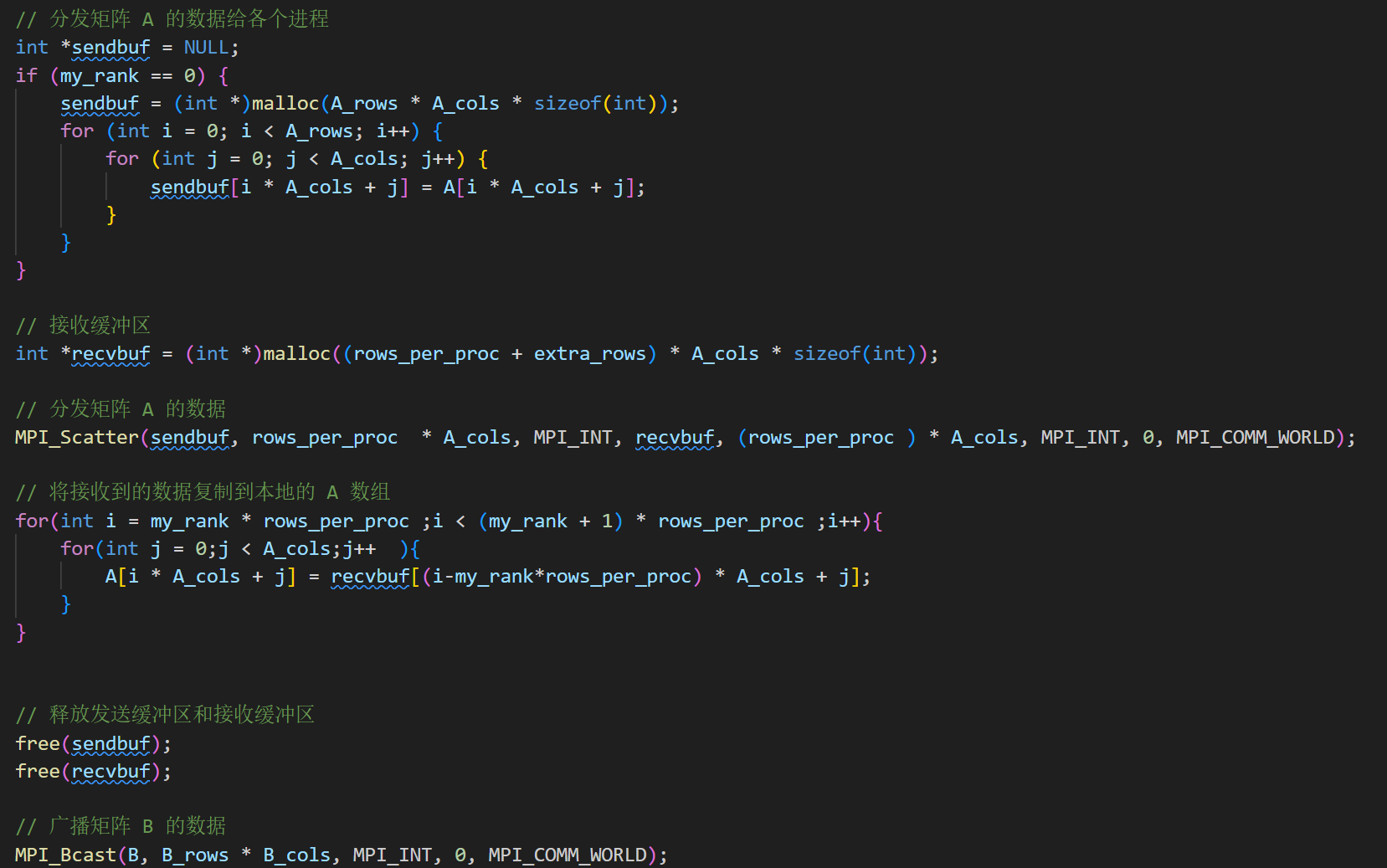




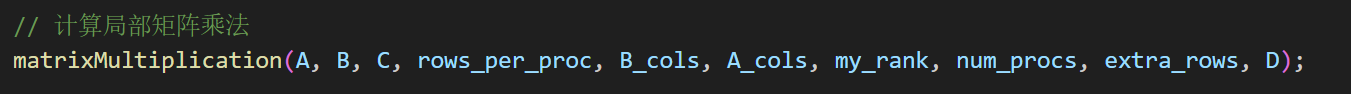
②将矩阵A按行分给不同的进程计算，计算出平均每个进程负责的矩阵A的行数rows\_per\_proc。

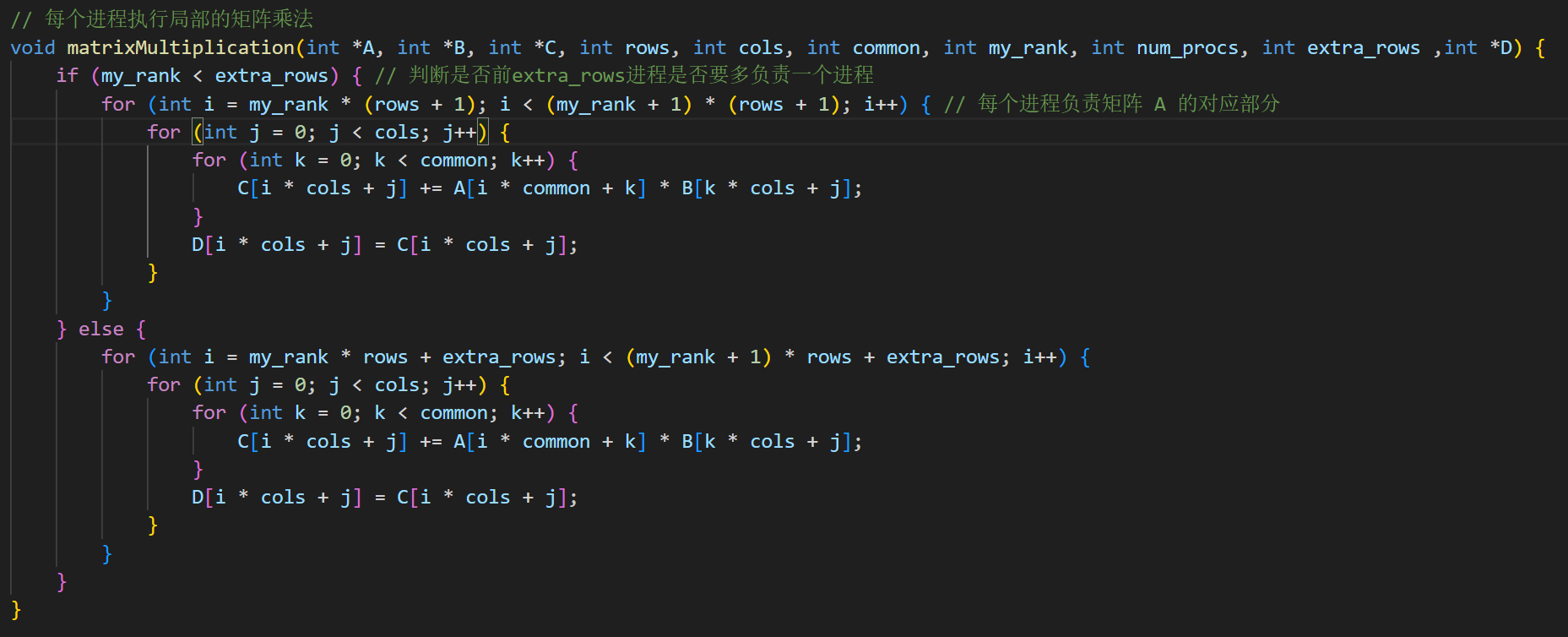


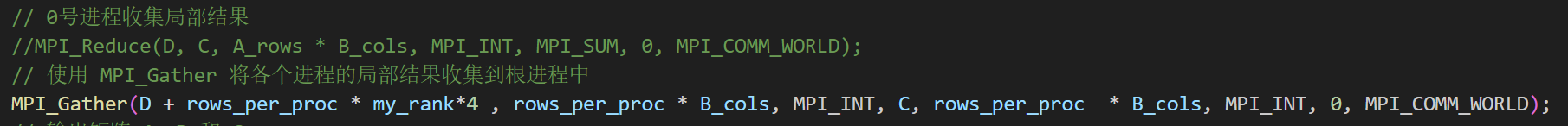
③利用MPI\_Scatter将每个进程负责的矩阵A的行分给每个进程,将矩阵A、B的维度和矩阵B用MPI\_Bcast广播给每个进程。



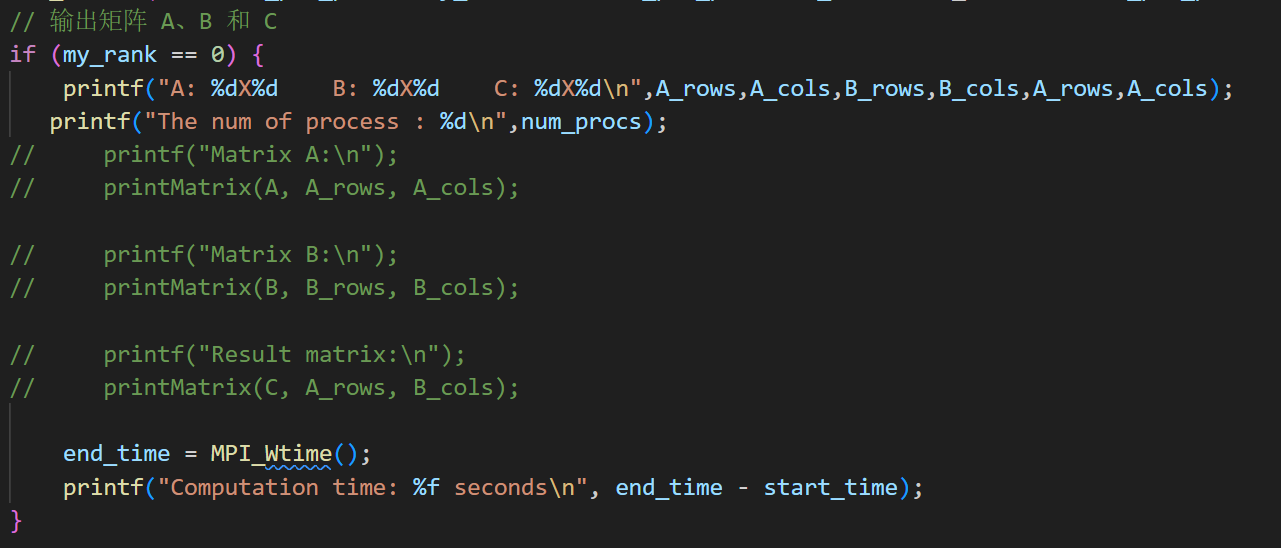
④每个进程计算出结果后，利用MPI\_Gather将结果归到0号进程中。





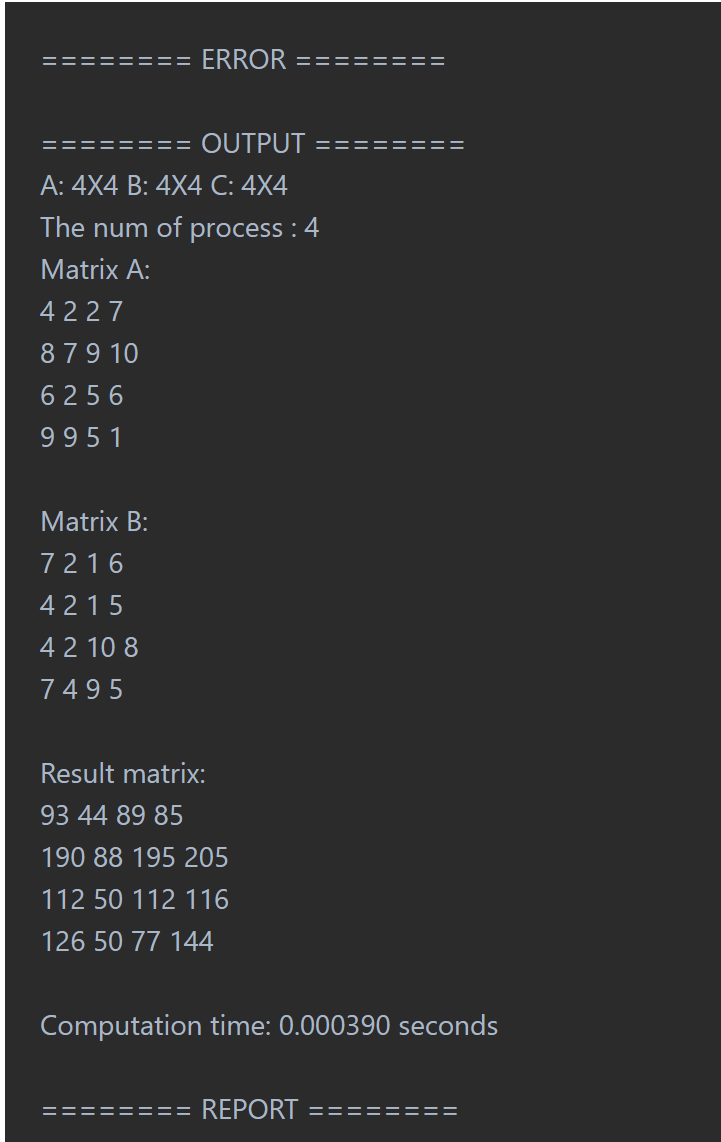


⑤0号进程输出结果和计算时间。



## 3实验结果

3.0 验证程序的计算正确性:



可以看到计算的结果是正确的

### 3.1 在虚拟机上进行实验

由于虚拟机资源有限，只有4个核，故只能跑4个进程

### 3.1.1 矩阵规模为128

### 屏幕截图 2024-04-02 175521

### 3.1.2 矩阵规模为256

### 屏幕截图 2024-04-02 175550

### 3.1.3 矩阵规模为512

### 屏幕截图 2024-04-02 175707

### 3.1.4 矩阵规模为1024

### 屏幕截图 2024-04-02 175729

### 3.1.5 矩阵规模为2048

### 屏幕截图 2024-04-02 180911

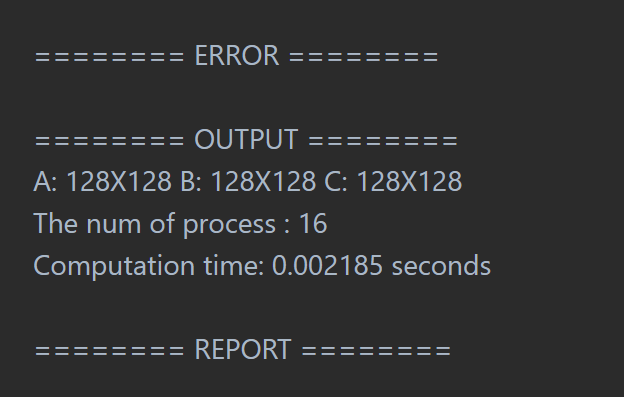
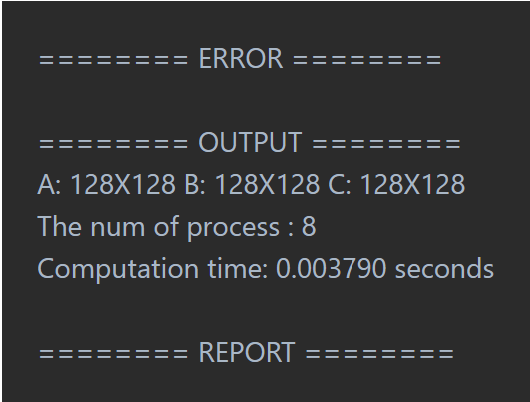
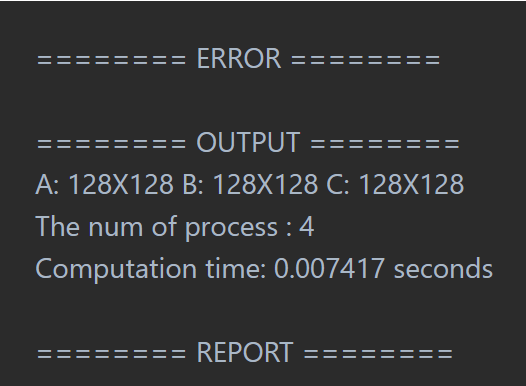
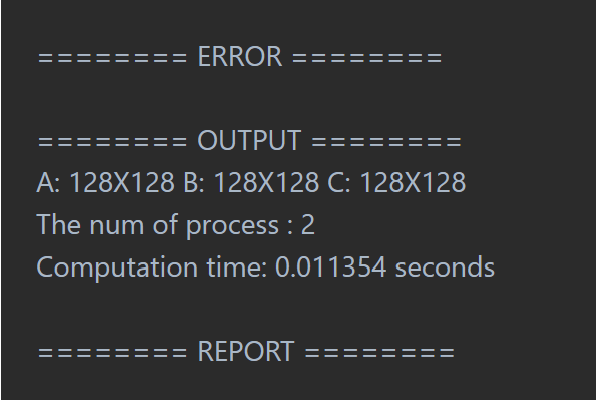
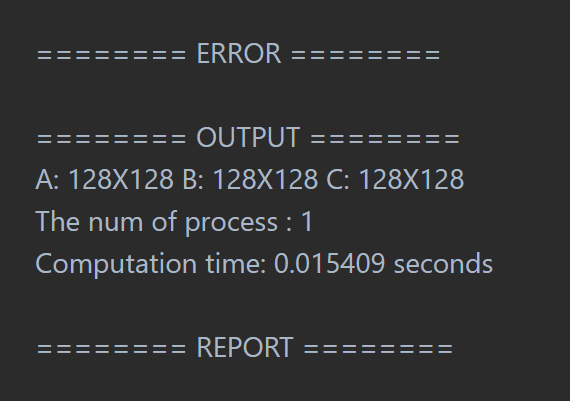
### 3.1.6 实验结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 进程数 | 矩阵规模 | | | | |
| 128 | 256 | 512 | 1024 | 2048 |
| 1 | 0.011057s | 0.067198s | 0.706773s | 19.0130648s | 252.25035s |
| 2 | 0.005244s | 0.036430s | 0.388127s | 6.86366s | 138.678854s |
| 4 | 0.007688s | 0.053572s | 0.412725s | 7.582389s | 105.216116s |

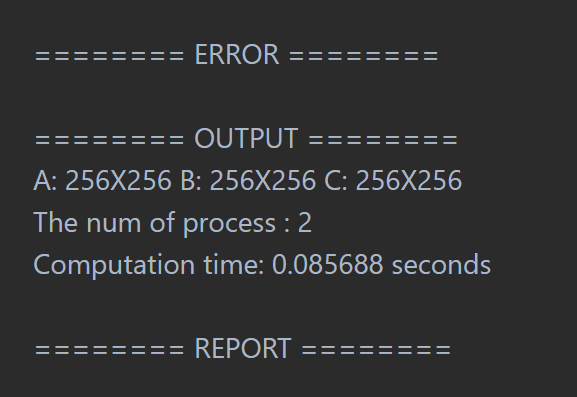
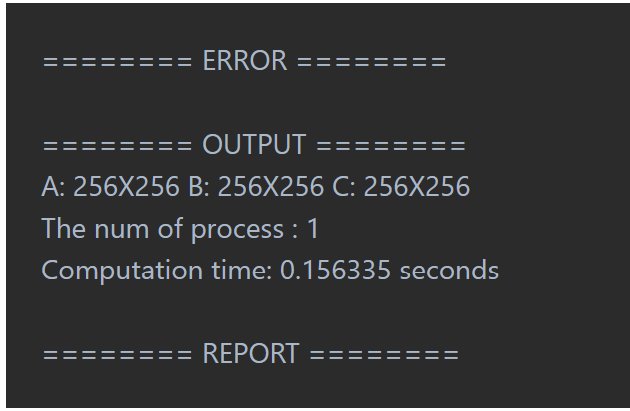
### 3.2 在超算习堂上进行实验

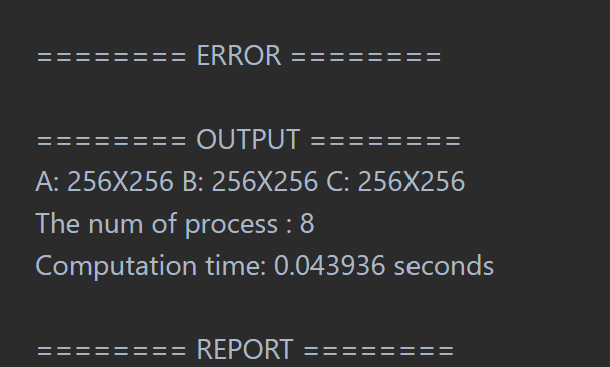
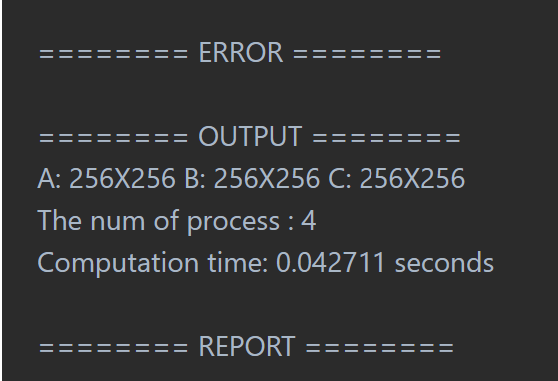
超算习堂自由编程可使用多个进程资源进行实验,但似乎运行时间过久就不行

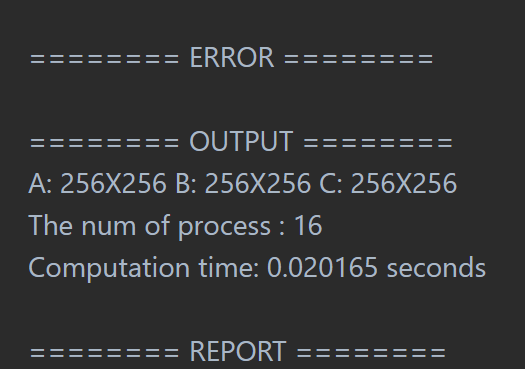
### 3.2.1 矩阵规模为128



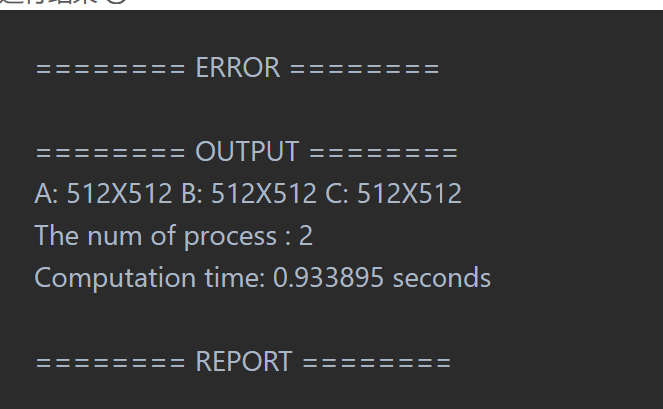
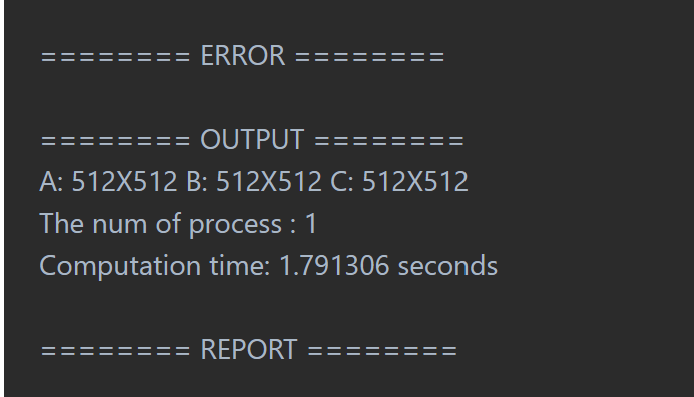
### 3.2.2 矩阵规模为256

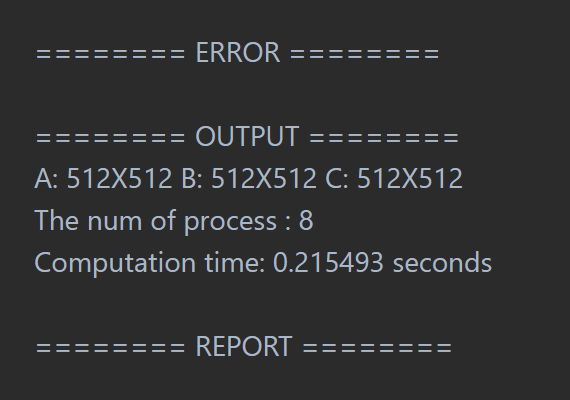
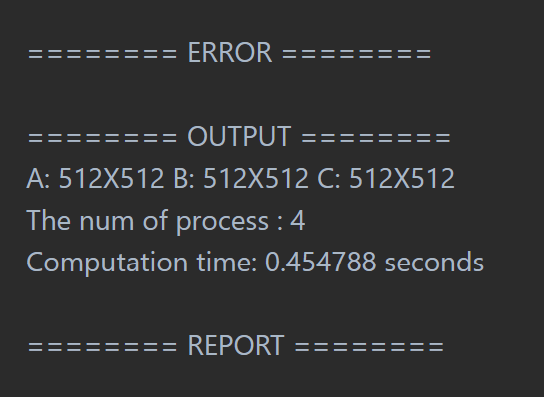


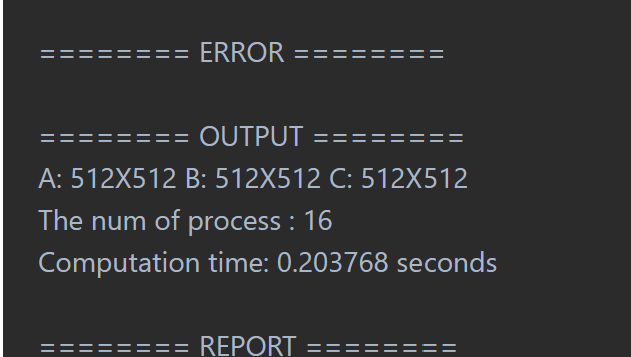




### 3.2.3 矩阵规模为512







### 3.2.4 结果对比

|  |  |  |  |
| --- | --- | --- | --- |
| 进程数 | 矩阵规模 | | |
| 128 | 256 | 512 |
| 1 | 0.015409s | 0.156335s | 1.791306s |
| 2 | 0.011354s | 0.085688s | 0.933895s |
| 4 | 0.007417s | 0.042711s | 0.454788s |
| 8 | 0.003790s | 0.043936s | 0.215493s |
| 16 | 0.002185s | 0.020165s | 0.203768s |

### 3.3 结论

## 4.实验感想

### 4.1 问题即解决方法

①Problem: MPI集合通讯不能用于二维数组

Solution: 将二维数组扩展为一维数组进行传播通讯

②Problem: 对于矩阵A的行数除进程数除不尽即有的进程接受需要负责的到A的矩阵行数会比平均每个负责的要多一行,而MPI\_Scatter只能给每个进程分发相同数目的数量，所以MPI\_Scatter不能处理这种情况。

Solution: 可以对矩阵A进行填充，MPI\_Scatter 将行分给对应进程后，进程从中提取对应的部分。(实验中未实现)

通信开销的重要性：在并行计算中，通信开销往往是性能的一个瓶颈。因此，选择合适的通信方式和优化通信模式对于提高性能至关重要。在这个实验中，我们使用了集合通信函数 MPI\_Gather 来收集结果，而没有直接采用点对点通信方式。这样做可以减少通信的次数，提高效率。

数据划分的影响：在并行计算中，如何划分数据对于负载均衡和性能的影响很大。合理的数据划分可以使得各个进程的计算量尽量均衡，避免出现单个进程负载过重或者大量空闲的情况。在矩阵乘法中，可以尝试不同的数据划分方式，比如按行划分或者按列划分，以找到最优的划分方式。

MPI 的灵活性和复杂性：MPI 提供了丰富的函数库和通信机制，可以实现各种各样的并行计算模式。然而，MPI 的灵活性也带来了一定的复杂性，需要开发人员有一定的经验和技能才能充分利用 MPI 实现高效的并行程序。