中山大学计算机学院本科生实验报告

课程名称：并行程序设计与算法

实验	MPI矩阵乘法进阶	专业（方向）	计算机科学与技术
学号	21307035	姓名	邓栩瀛
Email	dengxy66@mail2.sysu.edu.cn	完成日期	2024.4.1

1、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：m,n,k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m×n的矩阵A及n×k的矩阵B，并对这两个矩阵进行矩阵乘法运算，得到矩阵C

输出：A、B、C三个矩阵，及矩阵计算所消耗的时间t

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；使用`mpi_type_create_struct`聚合MPI进程内变量后通信；尝试不同数据/任务划分方式（选做）。

2. 对于不同实现方式，调整并记录不同进程数量（1-16）及矩阵规模（128-2048）下的时间开销，填写下页表格，并分析其性能及扩展性。

2、实验过程和核心代码

执行命令

```
mpic++ main.cpp -o main.out
# example
mpirun -np 1 ./main.out 128 128 128 1
```

核心代码

定义Matrix结构体

```
struct Matrix {
    int rows;
    int cols;
    double *data;
};
```

为 Matrix 结构定义 MPI 数据类型

```
    MPI_Datatype matrix_type;
    int block_lengths[2] = {1, 1};
    MPI_Aint offsets[2] = {offsetof(Matrix, rows), offsetof(Matrix, data)};
    MPI_Datatype types[2] = {MPI_INT, MPI_DOUBLE};
    MPI_Type_create_struct(2, block_lengths, offsets, types, &matrix_type);
    MPI_Type_commit(&matrix_type);
```

向所有进程广播矩阵A和矩阵B

```
    MPI_Bcast(A, m*n, MPI_DOUBLE, 0, MPI_COMM_WORLD);
    MPI_Bcast(B, n*k, MPI_DOUBLE, 0, MPI_COMM_WORLD);
```

计算矩阵乘法

```
    int local_m = m / size;
    double *local_C = new double[local_m*k];
    matrix_multiply(local_m, n, k, A + rank * local_m * n, B, local_C);
```

收集结果

```
    Matrix global_C;
    if (rank == 0) {
        global_C.rows = m;
        global_C.cols = k;
        global_C.data = new double[m * k];
    }
    MPI_Gather(local_C, 1, matrix_type, global_C.data, 1, matrix_type, 0,
MPI_COMM_WORLD);
```

3、实验结果

进程数	矩阵规模				
	128	256	512	1024	2048
1	0.008296	0.061404	0.481842	5.35154	87.0441
2	0.004713	0.030898	0.247896	3.08308	47.6167
4	0.003158	0.016422	0.128463	1.64606	25.681
8	0.003367	0.014603	0.092781	1.25253	19.8817
16	0.072847	0.182074	0.608492	1.87918	20.5694

实验结果分析：

1. 随着矩阵规模增加，计算量也随之增加，运行时间增加。
2. 并行计算能够降低运行时间。
3. MPI 中的数据传输涉及将数据从一个进程传输到另一个进程，通过使用 `mpi_type_create_struct` 聚合数据，可以将多个相关数据作为一个单元进行传输，而不是分开传输，可以减少通信次数和数据拷贝，从而提高通信效率。
4. 聚合数据结构需要额外的内存来存储额外的信息，会增加内存占用。
5. 由于CPU核心数目的限制（该电脑的核心数为8），因此当进程数为16时，性能不升反降。

4、实验感想

1. 通过完成这个实验，更加深入地理解并行计算的概念和原理，学会了如何利用多个处理单元同时处理数据，以提高计算效率。
2. 通过编写MPI程序，对MPI编程有了更深入的了解，学会了如何初始化MPI环境、创建通信域、进行进程间通信等基本概念和方法。
3. 在实验中，我尝试不同的并行计算策略，以优化程序的性能，通过选择合适的进程数量、通信模式和数据分布方式，以最大程度地提高程序的运行效率。