# 矩阵乘法cannon算法

11091222 余锋伟

## 1设计方法和模式

Cannon的算法基本思路是把矩阵分块，把A、B、C矩阵分成P\*P份，每次Aik\*Bkj加到Cij上。不断轮转，使得最终所有的Cij能得到应有的AB分块相乘的值。

算法的精髓在于，第一步把AB矩阵分完块之后，每次运算完只需要把A矩阵循环左移，B矩阵循环上移，所有的AB矩阵块又能再次进行乘法运算。因此只需要经过P-1次交换数据+P次分块乘法，就能得到最终的C矩阵。

我把cannon算法分为以下几个步骤：

1. 计算当然把矩阵分块的P是多少，把节点个数标号为Nodeij，其中0=<i,j<P。
2. 其中节点Nodeij得到分块后的Ai,j+i，Bi+j,j。
3. 节点Nodeij中的AB分块矩阵相乘，加到Cij中
4. 节点Nodeij中的A矩阵往Nodeij-1发送，B矩阵往Nodei-1,j发送。重复3步骤直到矩阵乘完P次。
5. 把结果输出到结果文件

### 1.1矩阵分发

矩阵分发我认为有两种方式，一种是主从模式，也就是说节点0读取矩阵的所有数据，然后分发分块矩阵到各自的节点。

另一种方式（也是我采用的）是让每个节点自行从文件系统里读取自己需要的矩阵块AijBij。这种方法的好处之一是由于集群使用的是并行文件系统，读取数据不会存在竞争关系。二是上一中方式需要 0开辟空间存储矩阵的所有数据，这使得内存有可能成为瓶颈。当每个节点只各自读自己的数据，那么即使矩阵再大，只要节点数足够多，能装得下N/(P\*P)的数据，就能进行计算。使得数据也并行存储。

### 1.2分块矩阵相乘

单节点矩阵乘法很容易想到使用多线程加速，在程序时间中，先获取当然计算机所能支持的并发数量，以此构造线程。

假设计算机有nump个可并行线程，对于第i个线程，它将计算第I,i+nump,i+2nump…行A与B的乘法，并加到C的i,i+nump,i+2nump…上。由于每个线程之间不存在相互影响，因此多线程矩阵相乘能达到最的并行效率。

另外，矩阵乘法中ijk的枚举顺序能影响到效率，这里采用的是ikj循环。原因在《深入理解计算机系统》中有分析，主要是ikj顺序能有效降低CPU的cache缺失率。

### 1.3 AB块的分**发**

由于每次矩阵做完乘法之后，都需要把A矩阵循环左移，B矩阵循环右移。因此如何分发数据对于程序的整体性能有较大影响。

我一开始想采用的是所有的节点都先使用MPI\_Send发送数据，再使用MPI\_Recv接收数据。这个时候发现程序死锁了，原因在于MPI\_Send是阻塞发送，所有的节点都卡在Send阶段了。因此这个时候需要错开每个节点的Send和Recv。我终于想到了一个好方法。

以A块的循环左传为例子，P=5时，使用

If (j % 2 == 0)

{

MPI\_Send

MPI\_Recv

}

Else

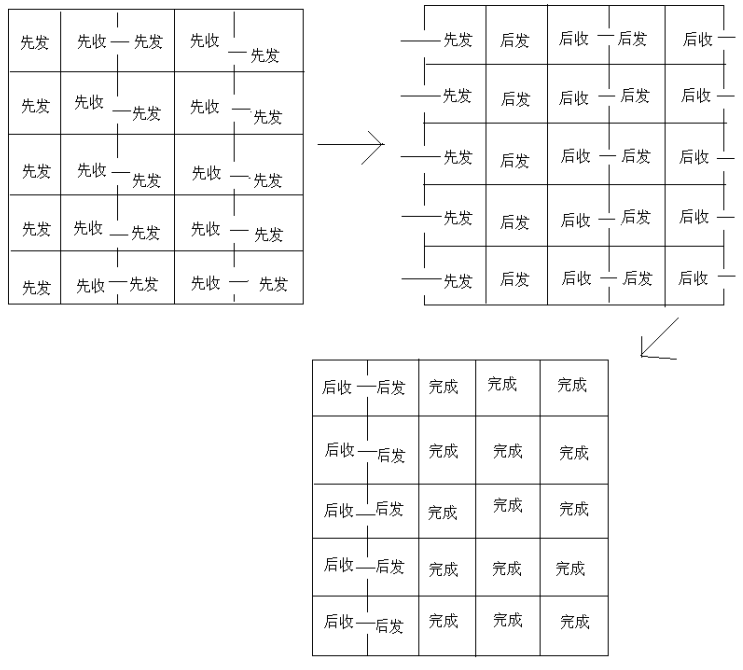
{

MPI\_Recv

MPI\_Send

}

表示奇数列先发再收，偶数行先收再发。于是通讯情况如下。

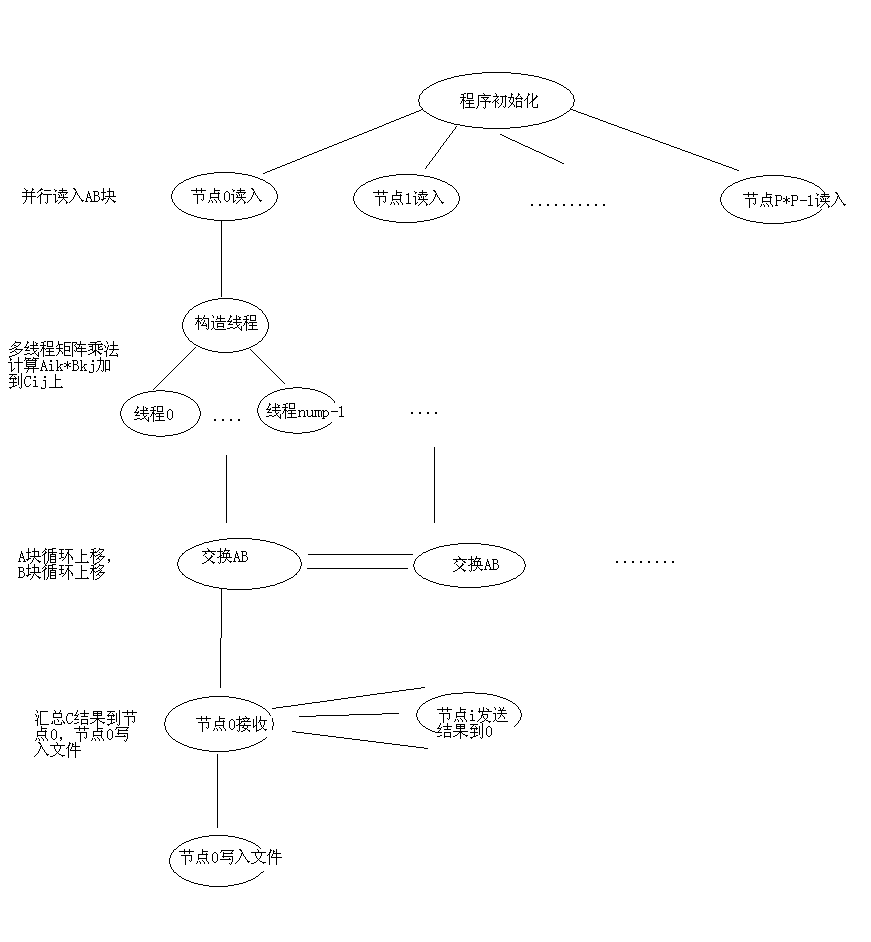


一共需要3次并行通信，原因是P=5为奇数。而当P为偶数时，只需要2次并行通信。并且该通信次数并不随着P的增大而增大。B块同理。

### 1.4输出结果

由于并行文件系统对于并行写支持的很不好，因此对于文件结果写入，采用主从式。即所有节点把它对于C的运算结果发送到0号节点，由0号节点写入文件。

## 2流程图



## 3复杂度

从流程图可以很容易分析出算法的复杂度。

### 3.1空间复杂度

假设矩阵一共有A=N1\*N2,B=N2\*N3,C=N1\*N3个数据，有个节点，那么每个节点需要开辟空间去存储A，B，C的分块数据。有个节点，所以，总的空间复杂度为，达到了理论上的最低空间复杂度。

### 3.2时间复杂度

* 每个矩阵块读入的时间是，并行效率较高，
* 多线程矩阵乘法时间是，其中是线程个数，说明多线程矩阵乘法能够达到理论最高的加速比。
* 假设一次数据交换(一块A或者一块B)的时间是，那么最坏情况下，需要并行3次交换，因此一次数据交换时间的上界是。
* 0号节点接受所有个节点发回的数据需要耗时，需要串行写入个数据。

总时间复杂度：

其中

## 4结果分析和总结

本次作业所有代码都有自己一个人设计编写而成，一共300行左右，感觉很有成就感。 在本次实验中，尽量设计出让每一步都高度并行的程序，经过测试。在1000×1000的数据中，普通乘法需要耗费12s时间，而使用我的方法进行计算，在开启4×4=16个节点计算时，耗费时间大约为0.9s，达到了超过12的加速比，十分接近理论上16的加速比。

这次是最后一次作业，感觉学到了很多东西，尤其是如何优化OI操作，如何设计更好的并行流程，以及如何优化通信复杂度。

感谢这一学期来老师和助教的辛苦教学！