并行与分布式作业

Foster 方法论

第四次作业

姓名： 何泽

班级： 18 级计科（超算方向）

学号： 18340052

> 利用 Foster并行程序设计方法计算1000x1000的矩阵与1000x1的向量之间的乘积, 要求清晰地呈现Foster并行程序设计的四个步骤。

- 串行算法

  对一个 `m x n` 的矩阵 `a` ，与一个 `n x 1` 的向量 `b` 相乘，结果为一个 `m x 1` 的向量 `c` 。

```
for (i = 0; i < m; i++) {
    for (j = 0; j < n; j++)
        c[i] += a[i][j] * b[j];
}
```

- `Foster` 并行设计
  - `Partitioning`

    按照数据划分，矩阵 `a` 第i行的第 `j` 个元素和矩阵 `b` 的第 `j` 个元素相乘，得到乘积后，同一行的相加作为矩阵 `c` 的一个元素。
  - `Communication`

    不同行的计算结果不需要通信，同一行的结果需要计算之后相加。
  - `Agglomeration`

    由于按照单个元素划分效率较低，将同一行的元素整合到一起，矩阵a的每一行和矩阵b计算内积，得到矩阵c的一个元素，矩阵a的每一行为一个计算任务。由于每一行计算的结果直接存入目标矩阵中，所以各任务间不需要通信，只是需要把结果矩阵定义为全局变量，每个线程计算完将线程内的计算结果存入即可。
  - `Mapping`

    由于各任务间不需要通信，只需要将任务数相对于处理器的个数平均分即可。
- 实现

  我是用的是 `C++` 的 `thread` 库函数
  - 每个线程调用的函数：

```
void mul(int p)
{
    int sum=0;
    for(int i=0;i<1000;i++)
        sum+=a[p][i]*b[i];
    c[p]=sum;
}
```

  传入一个参数p代表计算第p行
  - 主函数

```
 int main() {
     //随机生成大于100的数
     for(int i=0;i<1000;i++){
         for(int j=0;j<1000;j++){
             a[i][j]=rand()+100;
         }
     }
     for(int j=0;j<1000;j++){
         b[j]=rand()+100;
```

```
    }
    //开始多线程计算
    thread threads[1000];
    for (int i = 0; i < 1000; i++) {
        threads[i] = thread(mul, i );
    }
    for (int i = 0; i < 1000; i++) {
        threads[i] .join();
    }
    return 0;
}
```

可以运行，设计完成。