15.3

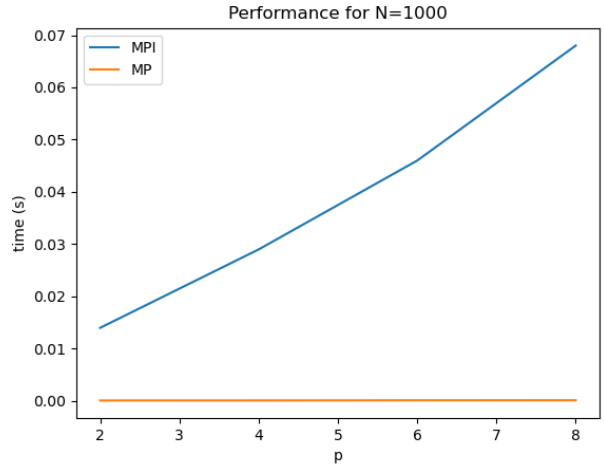
第二段代码

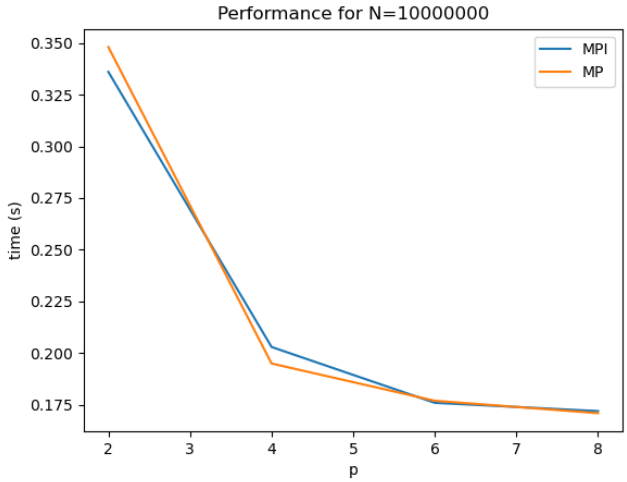
```
for(int=0;i<10;i++) buff[i] = data[32*2]
MPI_Send(buff,10,MPI_FLOAT,dest,tag,MPI_COMM_WORLD);
```

15.13

(1)当N=100万时，Ex_time为0.061s，精确到小数点后两位，即3.14
(2)
代码规模：
串行代码为30行，MP代码为32行，MPI代码为42行
性能表现
在控制进程数P不变的情况下，随着N的增加，两者有不同表现：

- MP在N较小时相对MPI有很大的优势，这很可能是由于OpenMP是共享内存的并行方式，而MPI是分布式内存的并行方式，MPI的通信开销过大
- 而当N变大时，MP的优势逐渐缩小，这是由于程序中的并行占比越来越大，通信开销占比越来越小
  Time-P关系图





- 可以看到在N=1000时，随着P的增加，运行时间不降反升，并且都大于串行算法的运行时间。这可能是由于在问题规模较小时，并行化带来的通信和同步开销大于并行计算的效率提升。
- 而在N=10000000时，二者表现相近，P=2时的执行时间是串行的$\frac{1}{2}$；P=8的执行时间约是串行算法的$\frac{1}{4}$。此时并行化确实提高了算法的效率

可扩展性

可扩展性是指并行程序的效率值**E**随着输入规模和进程数比例增长而保持不变的能力

- 二者都没有**强可扩展性**，即不增加问题的规模，将P翻倍，运行时间不能缩短为一半。
- 至于弱可扩展性，二者皆有不错的可扩展性，其中MPI的可扩展性强于MP，这是由于MPI随着N的增长，t增长得较慢
  根本原因可能是：MPI在规模小的情况下效率比较低，而规模大的情况效率比较高。所以在保持效率不变的前提下，P增长，规模只需要稍微增长即可。
  而造成MPI在规模小效率更低的原因是：分布式内存并行化带来的通信和同步开销大于共享内存型的MP。

实验数据：

```
paiCounter
N=1000000    |    pi=3.14     |    time=0.061s
N=10000000  |    pi=3.140   |    time=0.6s

MP
N=10000      |    pi=3.1~3.3   |    time=0.0003s
N=100000     |    pi=3.1       |    time=0.002s
N=1000000    |    pi=3.12~3.13 |    time=0.02s
N=10000000   |    pi=3.140     |    time=0.171s

N=1000       |    p=2          |    time=0.00010s
N=1000       |    p=4          |    time=0.00011s
N=1000       |    p=6          |    time=0.00013s
N=1000       |    p=8          |    time=0.00014s

N=10000000   |    p=2          |    time=0.348s
N=10000000   |    p=4          |    time=0.195s
N=10000000   |    p=6          |    time=0.177s
N=10000000   |    p=8          |    time=0.171s



MPI
N=10000      |    pi=3.1~3.3   |    time=0.068s
N=100000     |    pi=3.1       |    time=0.067s
N=1000000    |    pi=3.12~3.13 |    time=0.077s
N=10000000   |    pi=3.140     |    time=0.172s

N=1000       |    p=2          |    time=0.014s
N=1000       |    p=4          |    time=0.029s
N=1000       |    p=6          |    time=0.046s
N=1000       |    p=8          |    time=0.068s

N=10000000   |    p=2          |    time=0.336s
N=10000000   |    p=4          |    time=0.203s
N=10000000   |    p=6          |    time=0.176s
N=10000000   |    p=8          |    time=0.172s
```