**实验报告**

1. 实验名称：矩阵向量乘法并行实现的加速比及并行效率分析
2. 实验目的：

实现矩阵向量乘法的MPI程序，并测试矩阵向量规模在m=256, n=128时的一组加速比(7个规模)，进程规模分别设为p=2, 4, 8, 16, 32, 64, 128，同时计算出每个规模下的并行效率。矩阵向量规模m, n的值也可以大于该值。程序执行时间的统计采用timer.h里定义的GET\_TIME(start)和GET\_TIME(finish)获取一段代码的墙钟时间。

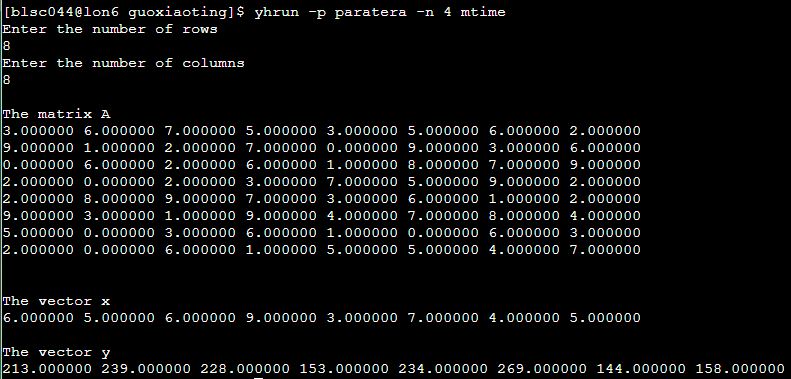
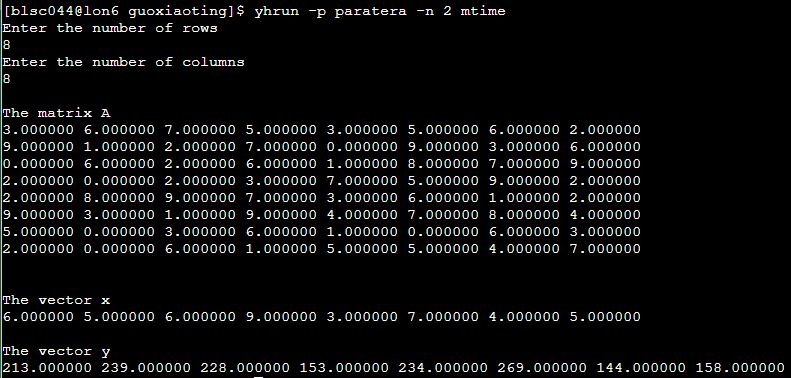
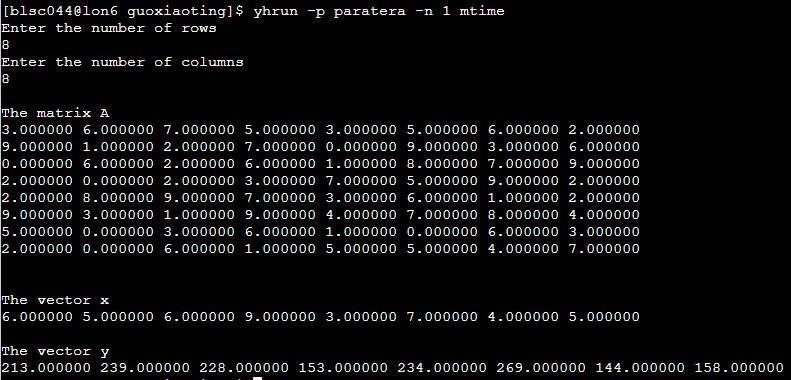
在命令行下用time命令统计以上程序的各项执行时间，并与之前的时间进行对比分析。

1. 实验平台：

广州超算安装的“天河二号”：CPU 型号 IntelXeonE5-269212C2.200GHz，采用 THExpress-2 高速互连；Paracloud云服务平台。

1. 实验步骤：
2. 自动生成指定规模的矩阵、向量

考虑到本实验涉及到的矩阵向量规模较大，所以改变手动键入的策略，通过与用户交互矩阵向量规模和运行进程数目采用随机数生成的方法来自动生成实验矩阵。而在随机数种子不变的情况下执行随机函数，生成的随机数是一样的（实验中以8\*8规模的矩阵和8\*1的向量并将其打印出来进行了验证），因此为保证改变进程规模而依然对同一矩阵向量进行运算，本实验采用默认的随机数种子，即srand(1)；



1. 时间获取函数

double global\_elapsed;

double my\_start, my\_finish, my\_elapsed;

MPI\_Barrier(comm);//执行路障函数，同步所有的进程

GET\_TIME(my\_start);

Mat\_vect\_mult(local\_A, local\_x, local\_y, local\_m, n, local\_n, comm);

GET\_TIME(my\_finish);

my\_elapsed = my\_finish-my\_start;//每个进程记下它所耗费的时间

MPI\_Reduce(&my\_elapsed, &global\_elapsed, 1, MPI\_DOUBLE, MPI\_MAX, 0, MPI\_COMM\_WORLD);

//利用MPI\_Reduce返回0号进程各个进程所耗费的最大时间值

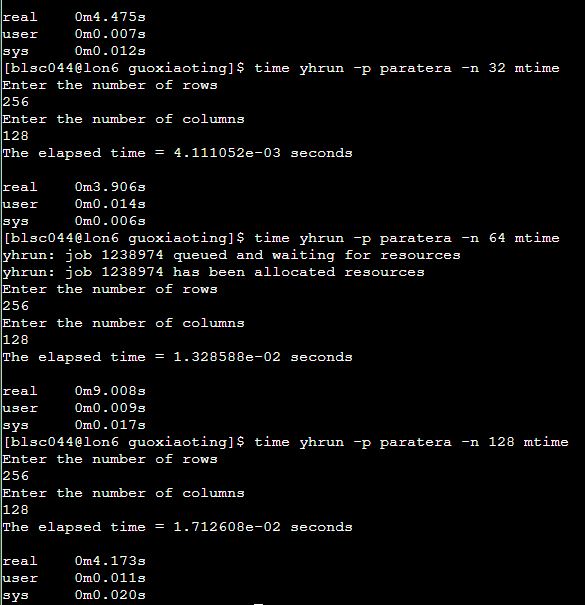
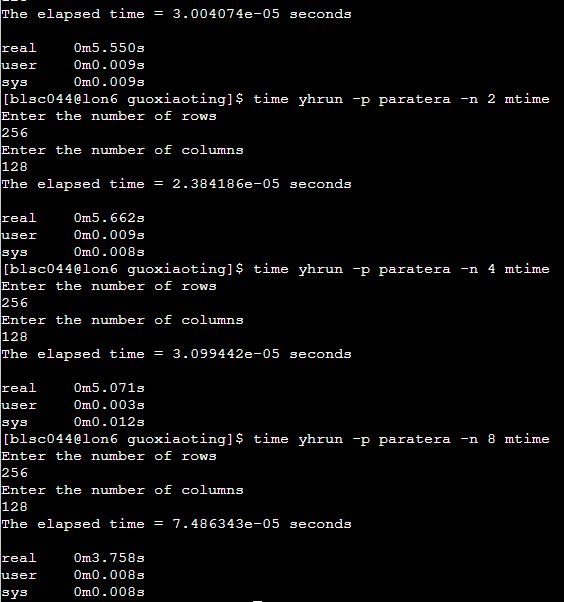
if(my\_rank==0)

printf("The elapsed time = %e seconds\n",global\_elapsed);

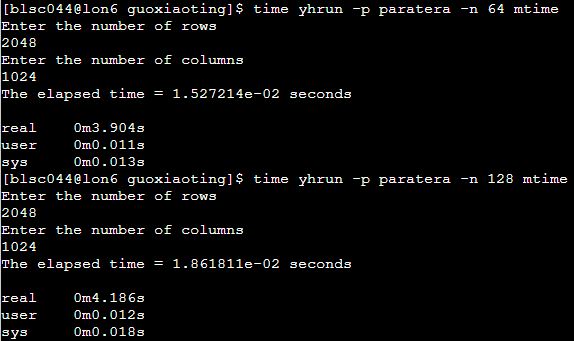
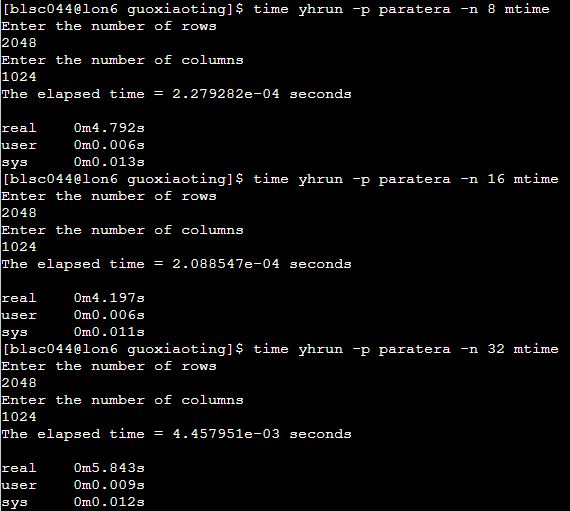
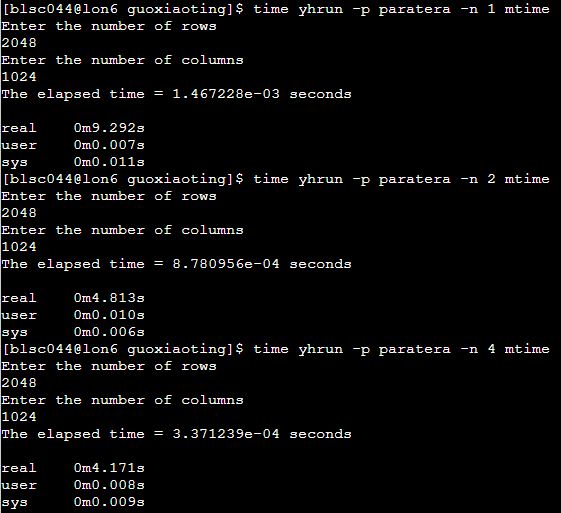
1. 登录paracloud平台上传本地代码，在天河上编译运行
2. 学习time命令的使用
3. 两种计时方式的运行结果如下：

在给定的256\*128矩阵规模实验下，发现实验结果与实验预期相差较大，没有预期的高效改善，故采用2048\*1024规模和4096\*4096规模的矩阵再次实验，具体结果如下：

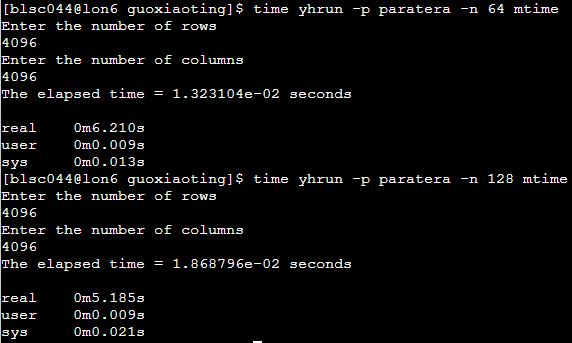
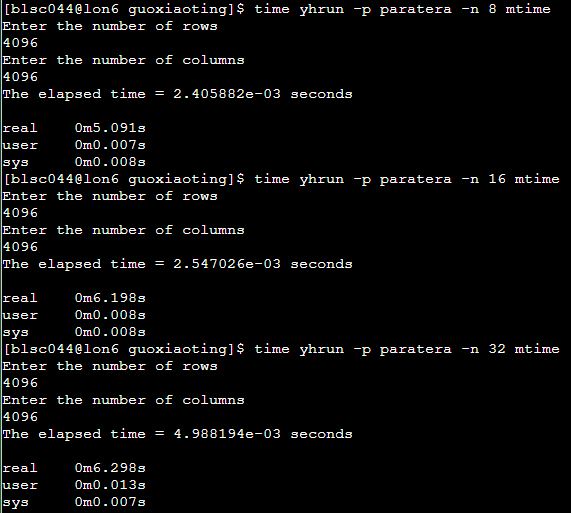
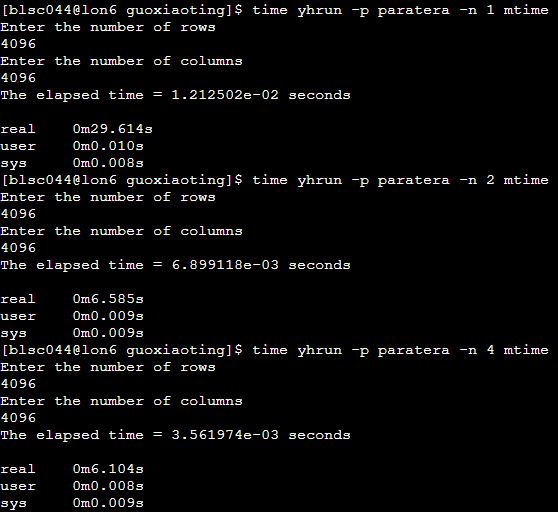
256\*128：



2048\*1024：



4096\*4096:



1. 加速比：

并行效率：

1. 实验结果：

m=256 n=128

|  |  |  |  |
| --- | --- | --- | --- |
| 进程数 | 执行时间/e^-5s | 加速比 | 并行效率 |
| 1 | 3.004 | 1 | 1 |
| 2 | 2.384 | 1.26 | 0.63 |
| 4 | 3.099 | 0.969 | 0.242 |
| 8 | 7.486 | 0.401 | 0.050 |
| 16 | 1.101e^1 | 1.004 | 0.063 |
| 32 | 4.111e^2 | 0.099 | 0.003 |
| 64 | 1.329e^3 | 0.113 | 0.002 |
| 128 | 1.713e^3 | 0.087 | 0.0007 |

m=2048 n=1024

|  |  |  |  |
| --- | --- | --- | --- |
| 进程数 | 执行时间/e^-5s | 加速比 | 并行效率 |
| 1 | 1.467e^2 | 1 | 1 |
| 2 | 8.781e^1 | 0.454 | 0.227 |
| 4 | 3.371e^1 | 1.183 | 0.296 |
| 8 | 2.279e^1 | 1.736 | 0.217 |
| 16 | 2.089e^1 | 1.909 | 0.119 |
| 32 | 4.458e^2 | 0.329 | 0.010 |
| 64 | 1.527e^3 | 0.353 | 0.006 |
| 128 | 1.862e^3 | 0.290 | 0.002 |

m=4096 n=4096

|  |  |  |  |
| --- | --- | --- | --- |
| 进程数 | 执行时间 | 加速比 | 并行效率 |
| 1 | 1.213e^3 | 1 | 1 |
| 2 | 6.899e^2 | 0.478 | 0.239 |
| 4 | 3.562e^2 | 0.926 | 0.231 |
| 8 | 2.406e^2 | 1.370 | 0.171 |
| 16 | 2.547e^2 | 1.295 | 0.081 |
| 32 | 4.988e^2 | 0.661 | 0.021 |
| 64 | 1.323e^3 | 0.917 | 0.014 |
| 128 | 1.869e^3 | 0.649 | 0.005 |

m=256 n=128的time计时

|  |  |  |  |
| --- | --- | --- | --- |
| 进程数 | Real/s | User/s | Sys/s |
| 1 | 5.550 | 0.009 | 0.009 |
| 2 | 5.662 | 0.009 | 0.008 |
| 4 | 5.071 | 0.003 | 0.012 |
| 8 | 3.758 | 0.008 | 0.008 |
| 16 | 4.475 | 0.007 | 0.012 |
| 32 | 3.906 | 0.014 | 0.006 |
| 64 | 9.008 | 0.009 | 0.017 |
| 128 | 4.173 | 0.011 | 0.020 |

1. 实验分析：

根据实验结果中三种不同规模大小的矩阵进行实验发现，对矩阵向量相乘算法的并行改进相对串行实现而言改善效果并不明显。可以发现，随着运算规模的增大，加速比大于1所需的进程数也在增加，加速比指标总是会经历增大、减小的波动趋势，其中可以明显地观测到，当进程数大于16个即分布在不同节点上时，加速比和并行效率大打折扣，说明在本实验的计算复杂性下，节点间通信开销的增加要大于并行算法的改进，并行效率几乎呈现递减的趋势。

Time命令执行下，real时间明显大于user时间和sys时间的总和，因为在并行程序中有很多进程之间的通信时间耗费。

(user)用户时间：主要耗费在不需要操作系统执行的用户代码和库函数上；在用户态执行的时间。

(sys)系统时间：主要耗费在需要操作系统执行的函数上；在内核态执行的时间。

CPU时间：程序执行代码的总时间。不包含进程等待过程的空闲时间。是用户时间和系统时间的总和。

(real)墙上时钟时间：进程运行的时间总量，其值与系统中同时运行的进程数有关，进程从开始到结束，始终走过的时间，包含进程在阻塞和等待状态的时间。