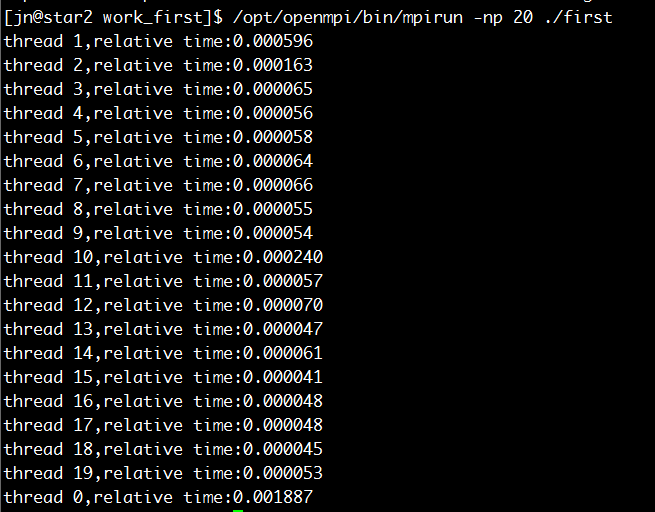
**并行计算第一次作业**

浙工大交换生

姜楠

实验结果：



|  |
| --- |
| #include<stdio.h>  #include<mpi.h>  int main(int argc,char \*argv[]){  int myid,numprocs;/\*进程编号，全部进程数量\*/  int tag=333; /\*传输所需的标记\*/  double start\_time=0,end\_time=0,source\_time;/\*开始时间，结束时间，上一个进程传输过来的时间\*/  MPI\_Status status;  MPI\_Init(&argc,&argv);/\*初始化\*/  MPI\_Comm\_rank(MPI\_COMM\_WORLD,&myid);/\*获取当前进程ID\*/  MPI\_Comm\_size(MPI\_COMM\_WORLD,&numprocs);/\*获取全部进程数量\*/  if(myid==0){ /\*进程0\*/  start\_time=MPI\_Wtime();/\*由进程0负责创建开始时间\*/  /\*将开始时间作为参数传送给进程1\*/  MPI\_Send(&start\_time,1,MPI\_DOUBLE,(myid+1)%numprocs,tag,MPI\_COMM\_WORLD);  /\*接受上一个进程传给自己的时间\*/  MPI\_Recv(&source\_time,1,MPI\_DOUBLE,(myid-1+numprocs)%numprocs,tag,MPI\_COMM\_WORLD,&status);  /\*计算传输一个回路所需的时间\*/  printf("thread %d,relative time:%f\n",myid,source\_time-start\_time);  }else{  /\*接受上一个进程传给自己的时间\*/  MPI\_Recv(&source\_time,1,MPI\_DOUBLE,(myid-1+numprocs)%numprocs,tag,MPI\_COMM\_WORLD,&status);  /\*获取当前时间\*/  end\_time=MPI\_Wtime();  /\*计算上一个节点传输给当前节点花费的时间\*/  printf("thread %d,relative time:%f\n",myid,end\_time-source\_time);  /\*将当前的时间作为参数传送给下一个节点\*/  MPI\_Send(&end\_time,1,MPI\_DOUBLE,(myid+1)%numprocs,tag,MPI\_COMM\_WORLD);  }  MPI\_Finalize();  } |