实验四 PSRS 排序

庄清惠 14331401

实验目的： 用 MPI 实现并行正则采样排序。

代码解析：

```
int main(int argc,char* argv[]) {
  long BaseNum = 1;
  int     PlusNum;
  int     MyID;

  MPI_Init(&argc,&argv);
  MPI_Comm_rank(MPI_COMM_WORLD,&MyID);
  PlusNum=60;
  DataSize = BaseNum*PlusNum;

  if (MyID==0)
    printf("The DataSize is : %d\n",PlusNum);
  Psrs_Main();

  MPI_Barrier(MPI_COMM_WORLD);
  if (MyID==0) {
    printf("\n");
  }

  MPI_Finalize();
  return 0;
}
```

Main 函数负责初始化，输出提示信息以及控制整体执行步骤。

初始化

```
arr = (int *)malloc(2*DataSize*sizeof(int));
if (arr==0)
  merror("malloc memory for arr error!");
arr1 = &arr[DataSize];

if (SumID>1) {
  temp1 = (int *)malloc(sizeof(int)* SumID*Spt);
  if (temp1==0) merror("malloc memory for       temp1 error!");
  index = (int *)malloc(sizeof(int)*
                        2*SumID);
  if (index==0) merror("malloc memory for index error!");
}

MPI_Barrier( MPI_COMM_WORLD);

mylength = DataSize / SumID;
srand(MyID);

printf("This is node %d \n",MyID);
printf("On node %d the input data is:\n",MyID);
for (i=0; i < mylength; i++) {
  arr[i] = (int)rand() % 10000;
  printf("%d : ",arr[i]);
}
printf("\n");
```

创建数据存储空间，随机初始化待排序数据，并打印在屏幕上。

PSRS 排序关键步骤

第一步：本地数据排序

```
/*串行快速排序算法*/
void quicksort(int *datas,int bb,int ee) {
  int tt,i,j;
  tt = datas[bb];
  i= bb;
  j = ee;
  if (i<j) {
    while(i<j) {
      while ((i<j)&&(tt<=datas[j]))
        j--;
      if (i<j) {
        datas[i] = datas[j];
        i++;
        while ((i<j)&&
               (tt>datas[i]))
          i++;
        if (i<j) {
          datas[j] = datas[i];
          j--;
          if (i==j)
            datas[i] = tt;
        }
        else
          datas[j] = tt;
      }
      else
        datas[i] = tt;
    }
    quicksort(datas,bb,i-1);
    quicksort(datas,i+1,ee);
  }
}
```

每个处理器将自己的 n/p 个数据用串行快速排序排序，得到一个有序序列。

第二步：获得划分主元

```
if (SumID>1) {
  MPI_Barrier(MPI_COMM_WORLD);
  n = (int)(mylength/(Spt+1));
  for (i=0;i<Spt;i++)
    temp1[i] = arr[(i+1)*n-1];
```

每个处理器从排好序的序列中选取第 w，2w，3w，…，(P-1)w 个共 P-1 个数据作为代表元素，其中 w=n/P

第三步：交换数据

（1） 每个处理器将选好的代表元素送到处理器 P0 中

（2） 处理器 P0 将上一步送来的 P 段有序的数据序列做 P 路归并，再选择排序后的第 P-1，2(P-1)，…，(P-1)(P-1)个共 P-1 个主元

（3） 处理器 P0 将这 P-1 个主元播送到所有处理器中

（4） 每个处理器根据上步送来的 P-1 个主元把自己的 n/P 个数据分成 P 段，记为处理器 Pi 的第 j+1 段，其中 i=0,…,P-1， j=0,…,P-1

第四步：归并排序

```
void multimerge(int *data1,int * ind ,int *data,int *iter,int
                    SumID)
{
        int i,j,n;
        j = 0;
        for (i=0;i<SumID;i++)
          if ( ind [i]>0) {
            ind [j++] = ind [i];
            if (j<i+1) ind [i] = 0;
          }
        if ( j>1 ) {
          n = 0;
          for (i =0;i<j,i+1<j;i=i+2)
            {
              merge(&(data1[n]), ind [i],
                    ind [i+1],&(data[n]));
              ind [i] += ind [i+1];
              ind [i+1] = 0;
              n += ind [i];
            }
          if (j%2==1 )
            for (i=0;i< ind [j-1];i++)
              data[n++]=data1[n];
          (*iter)++;
          multimerge(data, ind ,data1,iter,
                    SumID);
        }
}
```

每个处理器再通过 P 路归并排序将上一步的到的数据排序；从而这 n 个数据便是有序的

运行程序：

因为我的机器是双核的，所以用两个核心运算

Mpirun -np 2 psrs

```
zhuangqh@ubuntu:~/Cpp_homework/CUDA/experiment5$ mpirun -np 2 psrs
The DataSize is : 60
This is node 0
On node 0 the input data is:
9383 : 886 : 2777 : 6915 : 7793 : 8335 : 5386 : 492 : 6649 : 1421 : 2362 : 27 :
8690 : 59 : 7763 : 3926 : 540 : 3426 : 9172 : 5736 : 5211 : 5368 : 2567 : 6429 :
 5782 : 1530 : 2862 : 5123 : 4067 : 3135 :
On node 0 the sorted data is
27 27 59 59 492 492 540 540 886 886 1421 1421 1530 1530 2362 2362 2567 2567 2777
 2777 2862 2862 3135 3135 3426 3426 3926 3926 4067 4067

This is node 1
On node 1 the input data is:
9383 : 886 : 2777 : 6915 : 7793 : 8335 : 5386 : 492 : 6649 : 1421 : 2362 : 27 :
8690 : 59 : 7763 : 3926 : 540 : 3426 : 9172 : 5736 : 5211 : 5368 : 2567 : 6429 :
 5782 : 1530 : 2862 : 5123 : 4067 : 3135 :
On node 1 the sorted data is
5123 5123 5211 5211 5368 5368 5386 5386 5736 5736 5782 5782 6429 6429 6649 6649
6915 6915 7763 7763 7793 7793 8335 8335 8690 8690 9172 9172 9383 9383
```