# 实验五

学号：SA17011125

姓名：吴燕晶

1. 实验题目
2. 实验题目选择：枚举排序
3. 实验描述：枚举排序通常也叫做秩排序，算法的基本思想就是：对每一个要排序的元素，统计小于它的所有元素的个数，从而得到该元素在整个序列中的位置。
4. 实验环境
5. 虚拟机：VMware
6. 操作系统：Ubuntu16.04
7. 内存：4G
8. 处理器：4
9. 算法设计与分析
10. 串行算法

对于每一个要排序的元素统计小于它的所有元素的个数，从而得到该元素在整个序列中的位置。

以下是简要的代码描述：  
输入为：a[1], a[2],......, a[n]

输出为：b[1], b[2],......, b[n]

For i=1 to n do

K=1

For j=1 to n do

If a[i]>a[j] or (a[i]=a[j] and i>j)then

K+=1

Endif

Endfor

B[k] = a[i]

Endfor

1. 并行算法

假设对于一个长为n的输入序列使用n个处理器进行排序，只需要使每个处理器完成对其中一个元素的定位，然后将所有的定位信息集中到主进程中，由主进程负责完成所有元素的最终排位。

以下是代码的简要描述：

0号进程广播a[0]到a[n]给其他进程

For all pi where 1<=i<=n parallel do:

K =1

For j = 1 to n do

If (a[i] > a[j]) or (a[i] = a[j] and i > j) then

K = k+1

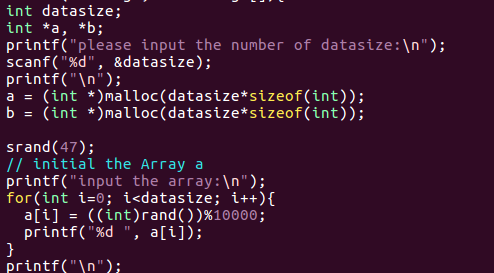
End if

end for

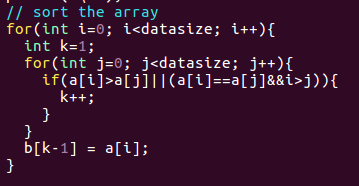
0号进程收集k并按序定位

end

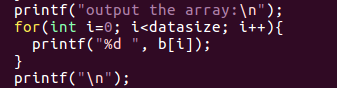
1. 核心代码
2. 串行代码
3. 初始化数组长度，输入数组和输出数组



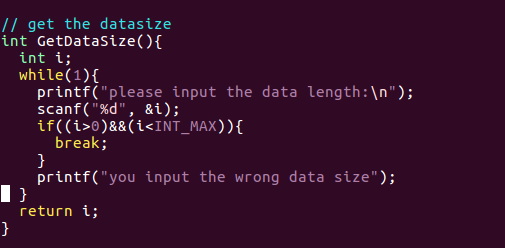
1. 排序算法



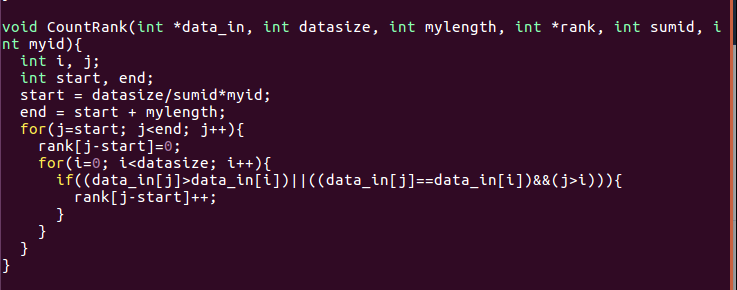
1. 输出排序后的结果



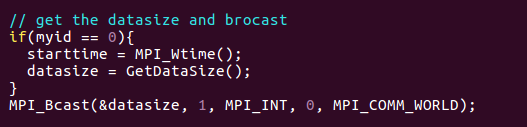
1. 并行代码
2. 获取datasize的函数



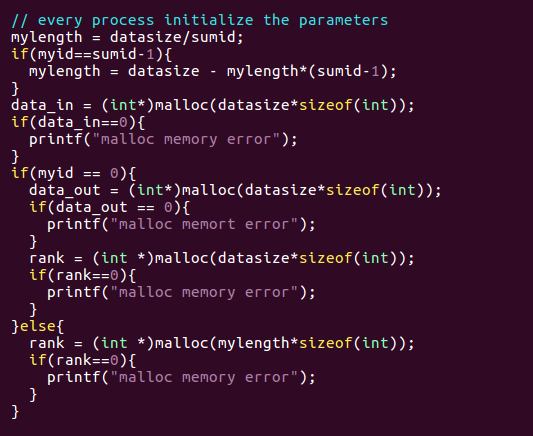
1. 每个进程计算它所分配的元素的rank



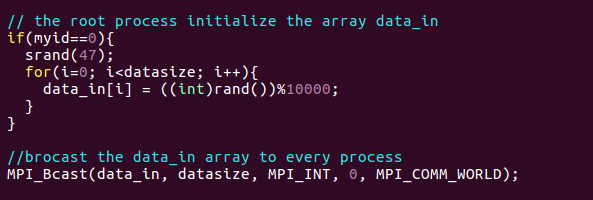
1. 主函数
   1. root进程获取初始时间和datasize,并将datasize广播给其他进程



* 1. 每个进程初始化自己的参数



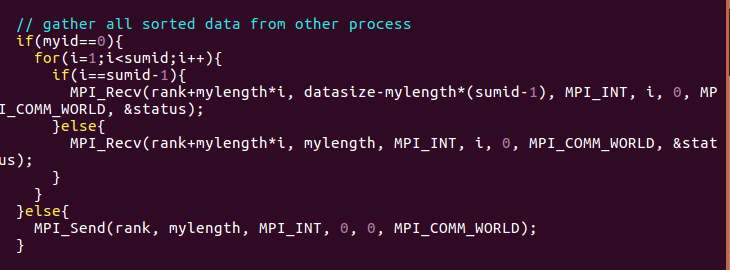
* 1. root进程初始化数组并将数组广播给其他进程



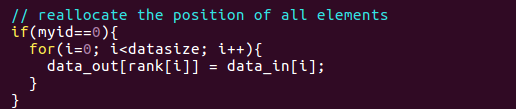
* 1. 每个进程计算自己分配到的元素的rank



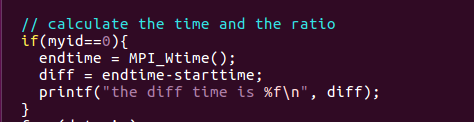
* 1. root进程收集其他进程计算的元素的rank



* 1. root进程对数组进行排序

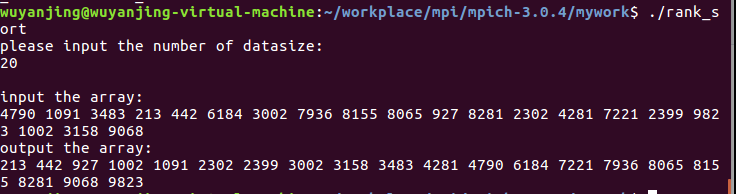


* 1. 计算程序所花费的时间



1. 实验结果
2. 串行实验结果

以下是数组长度为20时的排序结果，数组中的值是随机的。



1. 并行实现结果

用MPI实现

**运行时间**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **规模\进程数** | **1** | **2** | **3** | **4** |
| **N=100** | **1.704818** | **1.548937** | **1.194371** | **1.681934** |
| **N=1000** | **2.878434** | **1.862828** | **1.384115** | **1.929578** |
| **N=10000** | **3.952846** | **3.116346** | **2.065669** | **2.259054** |
| **N=50000** | **19.743864** | **11.001170** | **8.407078** | **7.422481** |

**加速比**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **规模\进程数** | **1** | **2** | **3** | **4** |
| **N=100** | **1** | **1.100637** | **1.427377** | **1.013606** |
| **N=1000** | **1** | **1.545195** | **2.079620** | **1.491742** |
| **N=10000** | **1** | **1.268423** | **1.913591** | **1.749779** |
| **N=50000** | **1** | **1.794705** | **2.348481** | **2.660008** |

1. 总结与分析
2. 从运行时间从可以看出，当进程数为1到3之间时，随着进程数的增加，不论什么规模，它的运行时间都在减少。而当进程数加到4的时候，虽然运行时间相对于进程数为1的时候会有减少，但是它相对于进程数为3，运行时间却在增加。特别的，当规模是100时，进程数为1和进程数为4的运行时间是大致相等的。这是因为随着进程数增加，进程之间的通信开销也增加了。
3. 从加速比上可以看出，当进程数为1到3之间时，随着进程数的增加，不论什么规模它的运行时间都在减少。同时可以看到随着规模的增加，加速比成增加的趋势。当进程数达到4的时候相比于之前，除了规模为50000的任务，其他任务的加速比都下降了。这是因为当进程数增加的时候，进程之间的通信开销增加。当规模为50000时，该任务的加速比在进程数在1到4之间时，随着进程数的增加，加速比也在增加。
4. 从以上的实验我们可以知道并行程序并不是进程数越多越好。在一定范围内增加进程数会使运行时间增加，但如果进程数过多的话，通信过于频繁，此时的通信开销过大，并行化计算所减少的时间不足以抵消通信开销，甚至会使计算时间比串行的还慢。总而言之，我们在并行化程序的时候要选择合适的进程数，这样才能够让并行化的效果达到最佳。