MPI并行优化的快速排序程序

一、算法的文字描述：

快速排序算法的功能是对若干数据以某一规定的比较大小的规则生成有序序列，因为是基于比较的平均时间复杂度最低的算法，故广泛应用于排序领域。快速排序算法以若干数据为输入，生成的有序序列为输出。

二、实验方法描述：

实验在Windows操作系统下，在VS中使用Microsoft MPI接口开发。

项目的包含目录设置为MPI安装目录下的Include文件夹，库目录设置为MPI安装目录下的Lib\x64文件夹，预处理器定义添加“\_CRT\_SECURE\_NO\_WARNINGS”和“MPICH\_SKIP\_MPICXX”，代码生成中的运行库设置为多线程调试（/MTd），链接器-输入中的附加依赖项中加入“mpi.lib”，并在配置管理器中新建x64平台。

三、优化方法详细描述：

开始时，进程0读取数据，所有非零进程等待接受任务指派。当某进程发现自己没有任务时，若有后继进程则通知所有后继进程无任务，然后结束自己的任务；当某进程发现自己的后继进程超过最大进程号将调用库函数std::sort并结束自己的任务；当某进程发现自己的任务数据量小于某一常整数N，则调用库函数std::sort并通知所有后继进程没有任务，然后结束自己的任务；否则，该进程将任务以快速排序算法Partition的规则二分，前一半留给自己，后一半交给后继进程。

即进程0二分后，将任务交给进程0、1并行执行；进程0、1二分后，将任务分别交给0、2和1、3并行执行；进程0、1、2、3二分后，将任务分别交给0、4，1、5，2、6，3、7并行执行；……

四、优化结果展示：

排序时间/ms：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数据量/10^x | 串行库函数sort | 并行进程数量 | | | | | | | | |
| 1 | 2 | 5 | 10 | 20 | 50 | 100 | 200 | 500 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 47 | 187 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 62 | 141 |
| 5 | 110 | 110 | 63 | 62 | 47 | 47 | 47 | 62 | 94 | 266 |
| 6 | 1015 | 1141 | 640 | 625 | 515 | 485 | 391 | 375 | 360 | 547 |
| 7 | 9984 | 10016 | 8500 | 8031 | 7750 | 6032 | 4469 | 4032 | 3390 | 3109 |

五、效果分析：

在串行快速排序算法中，T(n) = 2T(n/2) + n => T(n) = O(n logn)

理想的并行快速排序算法，T’(n) = T’(n/2) + n => T’(n) ≈ 2n - 1 = O(n)

但由于实际上，并行的进程数量有限，且不同进程任务不均匀、调度不均匀会导致部分进程任务结束后闲置而其他进程还在工作，而且MPI使用消息传递机制，需要对数据在进程之间进行多次拷贝。如此一来，本并行快速排序算法的时间复杂度应该介于二者之间。而根据数值结果，本并行快速排序算法在进程数较多的情况下确实优于串行快速排序算法；但在进程数过多的情况下，大量进程闲置，而进程数远多于机器允许的并行进程数量，导致多进程实则并发执行，浪费大量资源，运行速度反而降低。

六、结论：

在机器能接受的范围内，在没有闲置进程的情况下，工作的进程数量越多，本并行算法的效率越高。