实验报告

在Open MP平台上实现并比较多线程的归并排序和快速排序算法。

实验环境

CPU：Intel Core i5-7200U

内存：8GB

操作系统：Windows 10 64位教育版

编程语言：C++

编译环境：MSVC 2017 64-bit

算法分析

串行算法

包括归并排序和快速排序算法，其时间复杂度均为$O(n \lg n)$.

并行算法

并行快速排序

在串行算法的基础上将每次递归调用的两个quick_sort_parallel函数并行执行，$T_\infty(n) = \Theta(\lg n)$，并行度为$\Theta(n)$.

并行归并排序

针对Open MP的性能特点进行了优化，将原算法重写为循环形式，将每个循环并行执行，$T_\infty(n) = \Theta(n)$，并行度为$\Theta(\lg n)$.

结果分析

根据第五次作业的结果，取$n = 10^7$，在Debug模式下分别使用串行算法和并行算法（均使用4个线程）进行实验，每组实验均进行5次取运行时间的平均值，并记录其运行时间和计算结果，运行时间如下表（单位为ms）：

Merge sort	Parallel merge sort	Quick sort	Parallel quick sort
3735	1550	16538	15474

分析数据可得，当$n = 10^7$时，相比对应的串行算法，并行快速排序的速度提升较少，但并行归并排序的时间却减少了一半多，猜想这可能与Open MP对循环和递归的不同优化和递归过程中不断打开和关闭新线程所消耗的常数时间有关.

为了验证这个猜想，设计了quick_sort_new_parallel算法，该算法只在第一层的快速排序中进行多线程优化，理论上只是对递归进行了展开，并行度为$\Theta(1)$，但是最终运行时间却为11363ms，相比原并行算法还要快一些. 这说明递归并行在Open MP的多线程编程中会导致比较大的性能损耗，应该尽量使用循环并行.