多线程排序实验报告

李胜锐 2017012066

1 实验环境

Operating System: Windows10

Processor: 2.6 GHz 4-Core Intel Core i7

Memory: 4GB

Language: C++

IDE: Microsoft Visual Studio 2019

2 算法分析

2.1 多线程归并排序算法

递归调用排序阶段和递归调用归并时，都采用了多线程算法。因此：

$$PMS_1(n) = 2PMS_1(n/2) + \Theta(n) = \Theta(n \lg n)$$

$$\begin{aligned} PMS_\infty(n) &= PMS_\infty(n/2) + \Theta\left(\lg^2 n\right) \\ &= \Theta\left(\lg^3 n\right) \end{aligned}$$

2.1 多线程快速排序算法

在递归调用 n/2 规模的快速排序算法自身时，采用了多线程模式。因此：

$$MS'_1(n) = 2MS'_1(n/2) + \Theta(n) = \Theta(n \lg n)$$

$$MS'_\infty(n) = MS'_\infty(n/2) + \Theta(n) = \Theta(n)$$

3 结果分析

MULTI_THREAD / NUM_SIZE	YES	NO	
1000	0.000331	0.017748	merge sort
10000	0.003998	0.007181	
100000	0.044753	0.067171	
1000000	0.482935	0.73066	

MULTI_THREAD / NUM_SIZE	YES	NO	
1000	5.60E-05	8.80E-05	quick sort
10000	0.0007	0.000802	
100000	0.007728	0.00799	
1000000	0.083304	0.108492	

实验发现，采用多线程算法反而使得运算速度变慢。经过查阅资料得知，由于本地电脑 CPU 核数量有限，多线程算法的实际并行度会很低，远远达不到理论并行度的要求。并且，由于采用递归，多线程程序分配线程的次数非常高，造成了额外的时间浪费。

此外，快速排序算法运行速度比归并算法快 10 倍左右，这应该是由于快速排序无需额外分配内存，并且内存读写次数更少导致的。另外，归并算法相比快速排序多线程的并行度优势并没有体现出来。

4 实验总结

通过这次试验，发现在计算机实际运行的过程中，多线程算法由远远达不到理论上的并行度，反而容易因为分配分配线程的额外开销导致运行速度反而更慢。