

测试结果

单线程

```
Thread 0:Sort begin!  
Thread 0:Sort End!  
0.568
```

双线程

```
Thread 0:Sort begin!  
Thread 1:Sort begin!  
Thread 1:Sort End!  
Thread 0:Sort End!  
0.27
```

四线程

```
Thread 0:Sort begin!  
Thread 3:Sort begin!  
Thread 1:Sort begin!  
Thread 2:Sort begin!  
Thread 0:Sort End!  
Thread 1:Sort End!  
Thread 3:Sort End!  
Thread 2:Sort End!  
0.156
```

六进程

```
Thread 0:Sort begin!  
Thread 3:Sort begin!  
Thread 1:Sort begin!  
Thread 4:Sort begin!  
Thread 5:Sort begin!  
Thread 2:Sort begin!  
Thread 4:Sort End!  
Thread 2:Sort End!  
Thread 0:Sort End!  
Thread 5:Sort End!  
Thread 3:Sort End!  
Thread 1:Sort End!  
0.126
```

八线程

```
Thread 0:Sort begin!  
Thread 4:Sort begin!  
Thread 3:Sort begin!  
Thread 2:Sort begin!  
Thread 1:Sort begin!  
Thread 5:Sort begin!  
Thread 7:Sort begin!  
Thread 6:Sort begin!  
Thread 0:Sort End!  
Thread Thread 3:Sort End!  
1:Sort End!  
Thread Thread 4:Sort End!  
2:Sort End!  
Thread 5:Sort End!  
Thread 6:Sort End!  
Thread 7:Sort End!  
0.155
```

十六线程

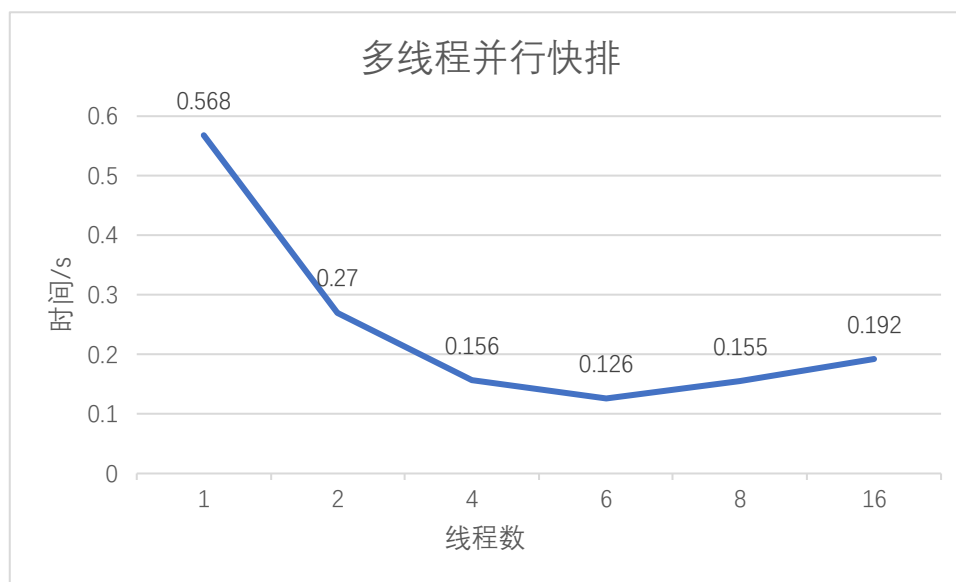
```

Thread 0:Sort begin!
Thread 8:Sort begin!
Thread 4:Sort begin!
Thread 9:Sort begin!
Thread 2:Sort begin!
Thread 14:Sort begin!
Thread 1:Sort begin!
Thread 12:Sort begin!
Thread 7:Sort begin!
Thread 10:Sort begin!
Thread 6:Sort begin!
Thread 15:Sort begin!
Thread 5Thread 13:Sort begin!
Thread 9Thread 4:Sort End!
Thread 14:Sort End!
Thread 12Thread 3:Sort begin!
Thread 0:Sort End!
Thread 15:Sort End!
:Sort End!
Thread 1:Sort End!
Thread 11:Sort begin!
:Sort End!
:Sort begin!
Thread 2:Sort End!
Thread 6:Sort End!
Thread 10:Sort End!
Thread 8:Sort End!
Thread 7:Sort End!
Thread 13:Sort End!
Thread 3:Sort End!
Thread 11:Sort End!
Thread 5:Sort End!
0.192

```

对应的打印信息和时间如上图所示，同时把对应的时间做成和线程数做成可视化折线图，结果如下图所示。

	A	B	C	D	E	F	G	
1	线程数	1	2	4	6	8	16	
2	时间/s	0.568	0.27	0.156	0.126	0.155	0.192	



可以看到，线程数为6的时候花费的时间是最少的，当开辟的线程数小于6或者大于6的时候，时间都会变长，效率都会变低。由进程的相关知识进行分析，当线程数小的时候，排序的并行度较低，不能很好的发挥多线程的优点，所以效率会不够好。当线程数太大的时候，线程切换以及线程同步的开销远大于并行度提升带来的好处，此时排序的效率又会大大减小。故当线程数达到一个适中的值的时候，并行度和线程切换和同步带来的开销达到平衡和最优的时候，此时的效率最高。

对于本次实验而言，对于600w数据的排序，其中线程数为6的时候效率是最高的。