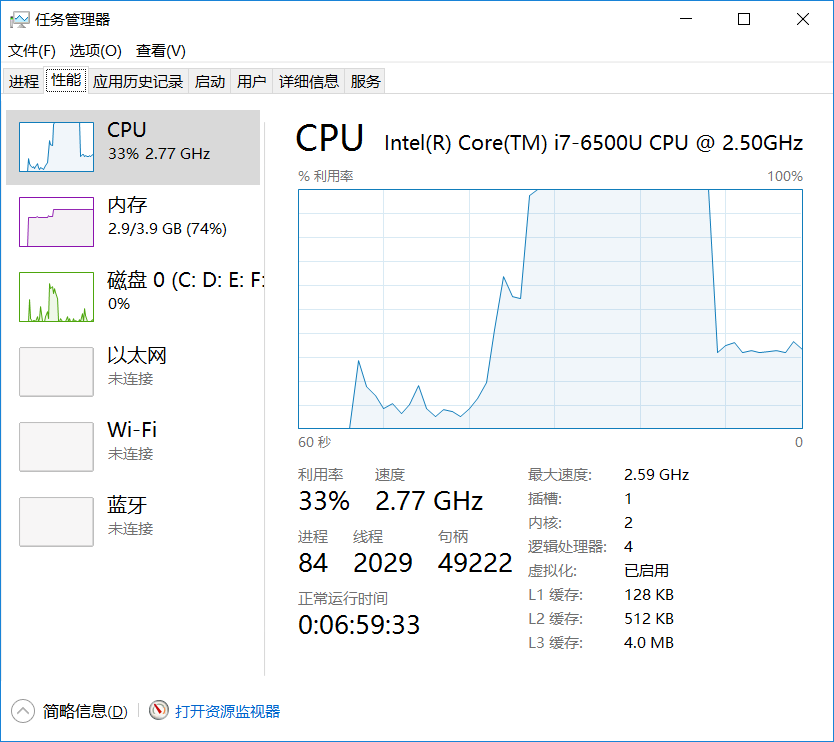
# 多线程实现矩阵乘法

1.CPU INTEL i7-6500U 内存4GB

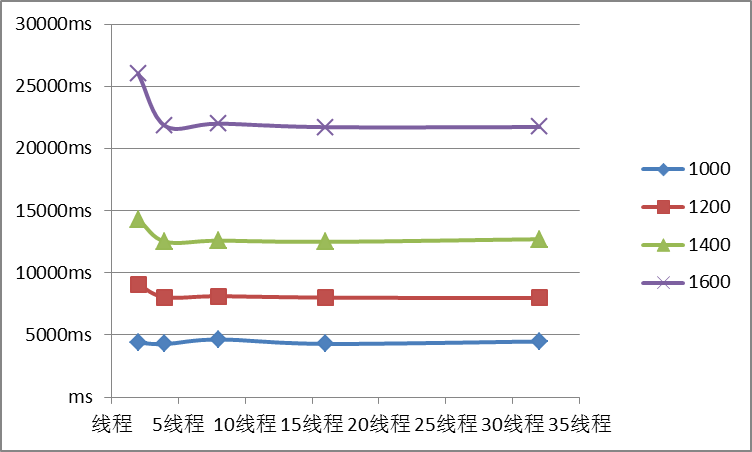


运行程序时的CPU利用率



2.测试结果：表中内容为计算所需时间，单位ms

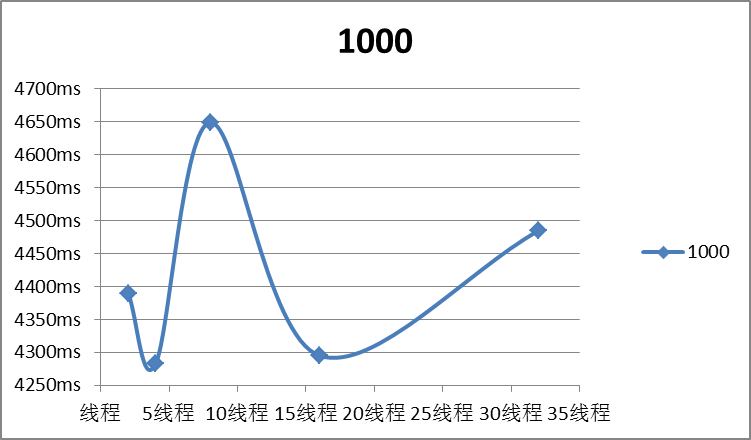
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1000\*1000 | 1200\*1200 | 1400\*1400 | 1600\*1600 |
| 2线程 | 4389 | 9045 | 14301 | 26013 |
| 4线程 | 4283 | 8052 | 12518 | 21832 |
| 8线程 | 4649 | 8125 | 12594 | 22007 |
| 16线程 | 4295 | 8018 | 12498 | 21724 |
| 32线程 | 4485 | 7982 | 12710 | 21735 |
| 直接运算 | 7331 | 15244 | 24372 | 45555 |



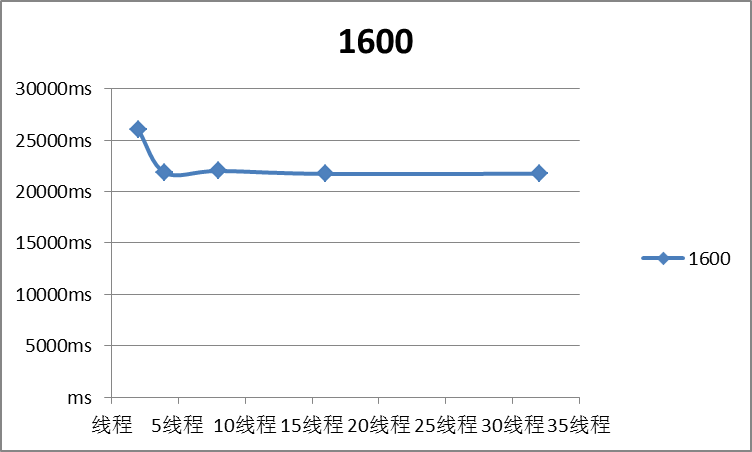
观察图表可以发现：

1. 矩阵越大，计算时间越长
2. 矩阵大小一定的情况下，多线程计算比直接计算快，差不多快了一倍
3. 不论矩阵多大，在4线程时，曲线陡降，效率最高，这应该与CPU的型号有关

3.结果比较



1000\*1000的矩阵测试结果



1600\*1600的矩阵测试结果

（1）将1000\*1000矩阵和1600\*1600矩阵的测试结果对比来看，发现随着线程的增加，1000\*1000的矩阵前期的计算速度并没有加快，反而减慢；后期到16线程时，速度有明显提升。

（2）线程增加后，1600\*1600矩阵的总体计算速度是加快的。

综上可以看出，并不是线程越多计算速度越快，开辟线程也需要时间。而且需要硬件的配合，如果只有一个核，多线程也会变成串行。似乎线程数=核数\*2时的效率最高，从本人和室友的测试结果中可以见得，不过是否为真命题还需大量验证，当然这些都是后话了。