实验5：Linux环境多线程编程

1. 实验目的

（1）掌握线程的概念、pthread线程库的使用

（2）掌握多线程实现DGEMM

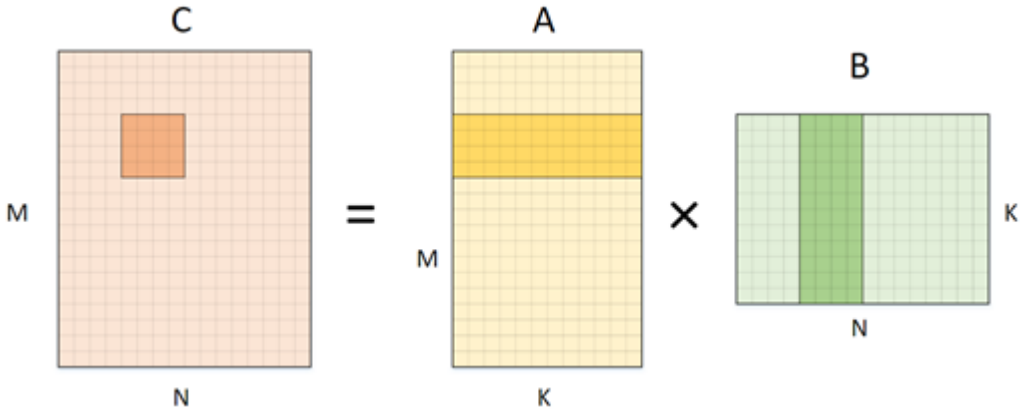
2. 实验内容

（1）在框架代码how-to-optimize-gemm中集成多线程实现DGEMM的版本，并记录相关数据。

3. 实验原理

3.1 矩阵分块

多线程实现DGEMM，要实现加速的效果，则每个线程均分计算量，即对矩阵进行分块，每个线程计算一小块。分块是针对计算结果矩阵C分块，矩阵AB对应做分块处理。



分块有多种方式：

1. 方式一：把矩阵C分为固定数目的块，比如横竖各切一刀即4块，块的数目固定，每个块大小不固定，与输入矩阵大小有关。分块的数目要等于cpu物理核的数才能充分利用cpu的性能，且假定矩阵大小m,n,k均能被p整除，其中p为CPU个数。
2. 方式二：把矩阵C分为固定大小的块，比如4*4、16*16，分块大小固定，分块的数目不固定，不仅能够提升cpu利用率，还能提高缓存利用率，达到很好的加速效果。具体实现上可以for循环嵌套，也可以递归实现。最优的分块大小是跟cpu的缓存有关。
3. 方式三：组合两种方式，先按照线程数分成固定数目的块，每个线程内部再划分成固定大小的块。

3.2 多线程实现

如果直接使用多线程实现DGEMM，可能出现

1. segmentation fault
2. 结果算的不对

且多线程调试难度稍大，建议先单线程实现分块，再改为多线程实现。

多线程实现时，先初始化矩阵A、B、C，主线程创建子线程并分配任务，确保不同线程写矩阵C的不同位置，不会造成数据覆盖。子线程执行任务，直接读取A、B、C的数据，并修改矩阵C的数据。

CPU有多少物理核就设置多少线程！！

需要修改makefile使得能够链接pthread库！！

4. 实验报告及要求

该实验暂无需写实验报告，在lab6优化完成后做统一的汇总分析，本实验需要记录：

1. 实验过程中自己认为值得记录的问题。
2. 测试较大规模的矩阵时cpu利用率和能体现多线程的运行截图，多线程查看使用top、ps、pstree等不限。