Lab2

Infomations

- 课程名:高性能计算
- 学期:2024秋
- 姓名:凌建杰
- 学号:2023311C08

Environment

- OS版本:5.15.153.1-microsoft-standard-WSL2
- gcc版本: 11.4.0
- CPU:
  - 型号:AMD Ryzen 7 6800H with Radeon Graphics
  - 频率:3.2 GHz
  - 物理核数:8
- 内存:7717216

`test_cblas_dgemm.c`

- 修改代码:

```
    int lda = K;
    int ldb = N;
    int ldc = N;
    cblas_dgemm(CblasRowMajor, CblasNoTrans, CblasNoTrans, M, N, K, alpha, A, lda, B,
ldb, beta, C, ldc);
```

- 改为行主序后,输出: `-4.000000 11.000000 0.000000 11.000000 -9.000000 5.000000 8.000000 5.000000 6.000000`
- 原列主序输出: `-4.000000 11.000000 0.000000 11.000000 -9.000000 5.000000 8.000000 5.000000 6.000000`
- 没有变化

time_dgemm.c

Values	256	1024	4096	8192
cblas_dgemm duration	0.077820 s	0.102401 s	0.616526 s	4.347744 s
naive_dgemm duration	0.045752 s	0.151559 s	3.907243 s	36.796382 s
cblas_dgemm gflops	0.862360 GFLOPS	41.942630 GFLOPS	445.849659 GFLOPS	505.784898 GFLOPS
naive_dgemm gflops	1.466796 GFLOPS	28.338583 GFLOPS	70.350860 GFLOPS	59.761942 GFLOPS

- 分析:
  - 矩阵规模较小时,Native算法和Cblas不相上下,甚至稍微优化后的Native算法优于Cblas(仅在M,N,K<=512验证)
  - 矩阵规模较大时,Native算法运行速度远低于Cblas,同时gflops远小于Cblas
  - 可能的原因:
    - Cblas内部做了储存优化,将数据暂存到主存,提高调用速度
    - 向量化计算

Problems

- Native算法在MNK数值较大时运行速度过慢
  - 解决方案:
    - 调用OpenMP做并行计算
    - 分块计算(4*4)
    - 使计算时间稍微能被接受