中山大學
SUN YAT-SEN UNIVERSITY

并行程序设计与算法实验

Lab5-基于 OpenMP 的并行矩阵乘法

姓 名 王志杰

学 号 22331095

学 院 计算机学院

专 业 计算机科学与技术

2025 年 4 月 23 日

1 实验目的

- 掌握 OpenMP 编程的基本流程
- 掌握常见的 OpenMP 编译指令和运行库函数
- 分析调度方式对多线程程序的影响

2 实验内容

- 使用 OpenMP 实现并行通用矩阵乘法
- 设置线程数量（1-16）、矩阵规模（128-2048）、调度方式
  - 调度方式包括默认调度、静态调度、动态调度
- 根据运行时间，分析程序并行性能
- 选做：根据运行时间，对比使用 OpenMP 实现并行矩阵乘法与使用 Pthreads 实现并行矩阵乘法的性能差异，并讨论分析。

3 实验结果

表 1: 默认调度

进程数	矩阵规模				
	128	256	512	1024	2048
1	0.00564401	0.046831	0.927868	7.55587	53.7882
2	0.00292381	0.0294496	0.46195	3.78493	25.0139
4	0.00163956	0.0201091	0.238161	2.00279	14.2377
8	0.00108729	0.0119222	0.123429	0.977024	7.92955
16	0.000922101	0.00652498	0.0673296	0.503261	4.21961

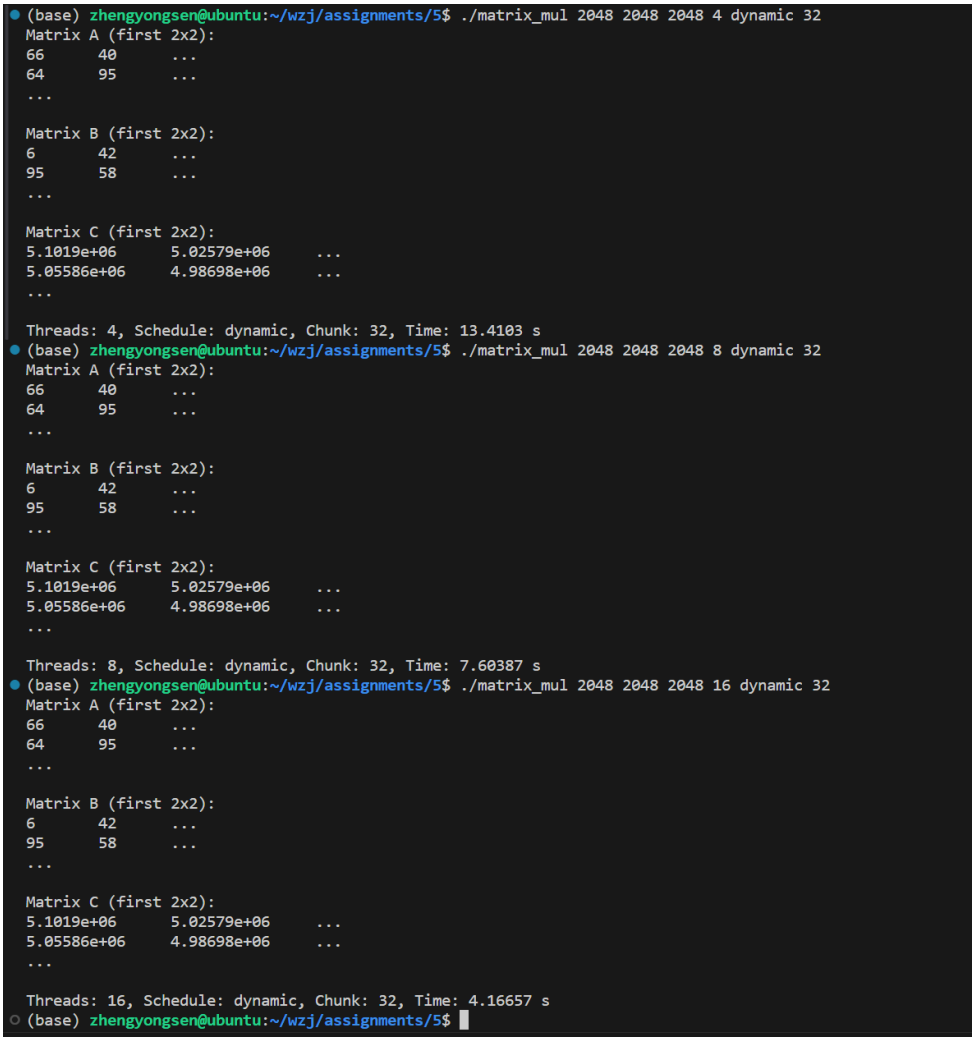

图 1: openmp 多线程结果正确性分析

表 2: 静态调度

进程数	矩阵规模				
	128	256	512	1024	2048
1	0.00412169	0.0453031	0.923607	7.47267	54.5692
2	0.00305276	0.0285643	0.465226	3.8028	25.8445
4	0.00172041	0.0160321	0.236393	2.0382	15.8064
8	0.00107711	0.0123473	0.122511	0.99355	7.78238
16	0.00106917	0.006391	0.0673578	0.49784	4.14464

表 3: 动态调度

进程数	矩阵规模				
	128	256	512	1024	2048
1	0.00558997	0.0465365	0.926299	7.55273	54.7765
2	0.00293234	0.0299083	0.468237	3.71085	25.1622
4	0.00168568	0.0199562	0.236772	1.99734	13.4103
8	0.00174055	0.0118168	0.123794	0.986725	7.60387
16	0.00202099	0.0125517	0.0667741	0.507405	4.16657

4 实验分析

- 根据运行时间，分析程序并行性能
- 回答：从表中可见，随着线程数从 1 增加到 16，所有三种调度模式的运行时间均显著下降。对于小规模矩阵（如 128×128、256×256），由于线程调度和同步开销占比高，8 及以上线程时加速比提升有限；而对于大规模矩阵（512×512 及以上），多线程能够充分发挥并行优势，8 ~ 16 线程下的加速比已接近理想值。默认（default）和静态（static）调度由于无额外运行时决策开销，其性能略优于动态（dynamic）调度；动态调度在小规模下开销明显，但在大规模时也能获得与静态调度接近的加速效果。
- 选做题：根据运行时间，对比使用 OpenMP 实现并行矩阵乘法与使用 Pthreads 实现并行矩阵乘法的性能差异，并讨论分析。
- 回答：结合之前实验三做的 pthreads 实验结果分析，在大规模矩阵（1024×1024 及以上）和高线程数（8 ~ 16）条件下，OpenMP 与 Pthreads 的运行时间相差不大，均能实现接近线性的加速；Pthreads 动态负载均衡能力稍优，但编程和维护复杂度较高；OpenMP 则凭借编译器优化和简洁的指令集，能够在保证高性能的

同时大幅降低开发成本。另外，openmp 适合多级并行（与 MPI、CUDA 混合使用），而 pthreads 只能处理用户空间线程，跨节点扩展需要额外通信库。

注：实验报告格式参考本模板，可在此基础上进行修改；实验代码以 zip 格式另提交；最终提交内容包括实验报告 (pdf 格式) 和实验代码 (zip 压缩包格式)