

电子科技大学
计算机科学与工程学院

标准实验报告

(实验) 课程名称 分布式并行计算

电子科技大学教务处制表

电子科技大学

实验报告

学生姓名：朱若愚 学号：2022150501027 指导教师：李可欣

实验地点：A2-412

实验时间：2025.4.19

一、实验室名称： 计算机学院实验中心

二、实验项目名称：N-Body 问题并程序设计

三、实验学时：4 学时

四、实验原理：

1. 研究 N 个质点相互之间在万有引力作用下的运动规律，对其中每个质点的质量和初始位置、初始速度都不加任何限制。
2. 当 $N=2$ 时，即为二体问题，已完全解决。 $N=3$ 即成为著名的三体问题。
3. 对于 $N>3$ 的 N 体问题，根本无法求出分析解。现在主要是采用数值方法和定性方法来进行研究。

五、实验目的：

1. 使用 CUDA 编程环境实现 N-Body 并行算法。
2. 掌握 CUDA 程序进行性能分析以及调优方法。

六、实验内容：

1. 学习和使用集群及 CUDA 编译环境
2. 基于 CUDA 实现 N-Body 程序并行化
3. N-Body 并行程序的性能优化

七、实验环境：

Intel(R) Xeon(R) Gold 6342 x2

2.80GHz – 3.50GHz

24C48T x2

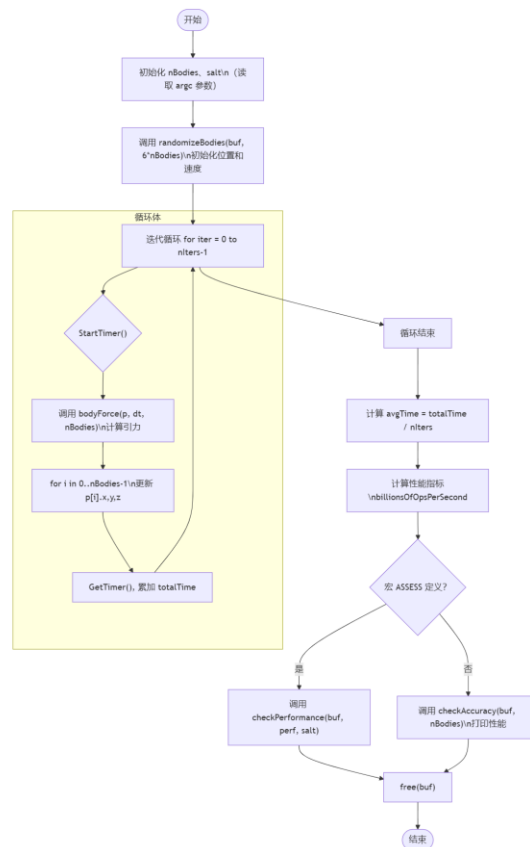
NUMA node0 CPU(s): 0-23,48-71

NUMA node1 CPU(s): 24-47,72-95

512G 内存

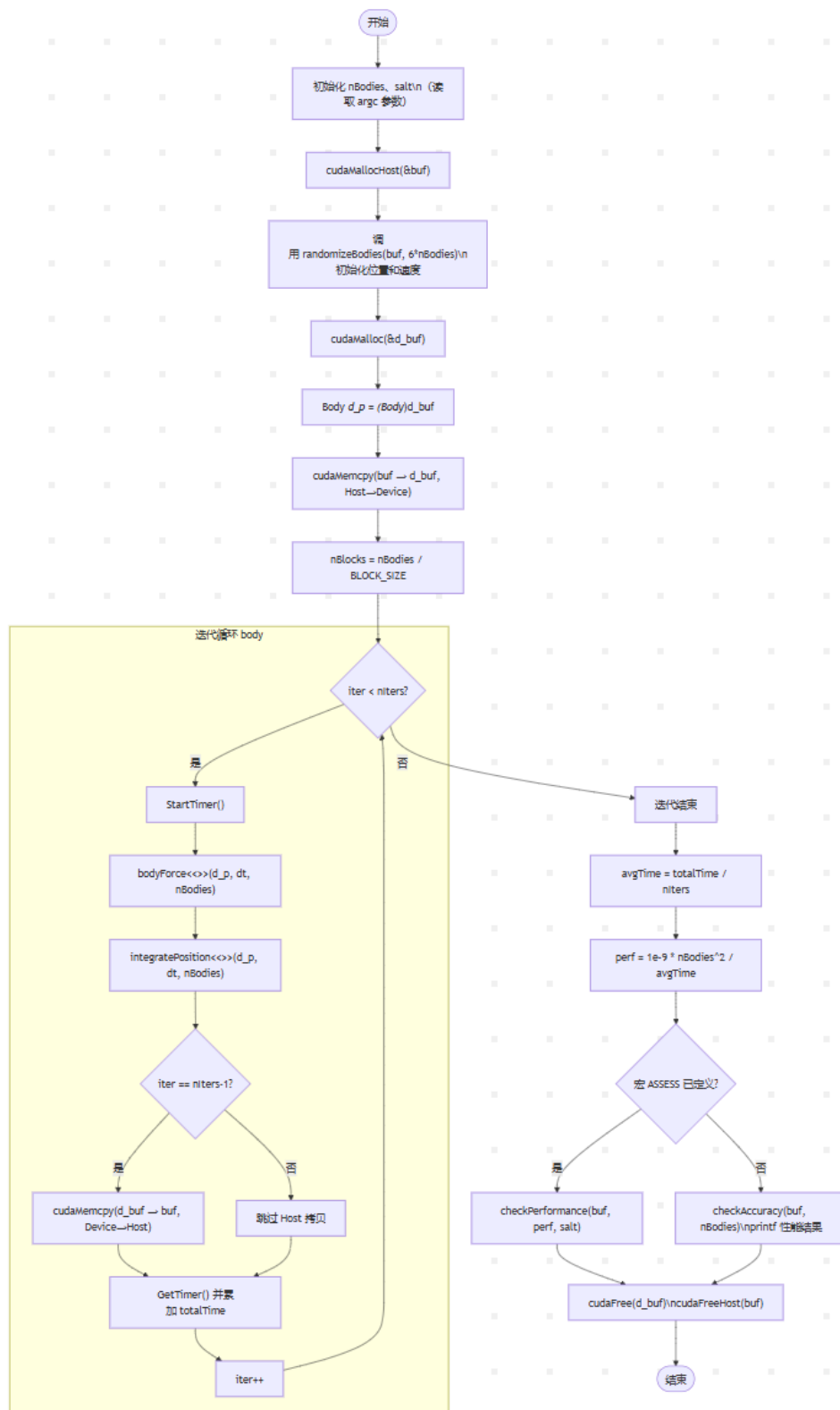
八、实验步骤：

1.基准代码 base.cu



- (1) 在 cmd 中输入 `scp -P 14004 "E:\VS\CUDA\base.cu" a2022150501027@121.48.170.1:/home/a2022150501027/` 输入密码并完成代码上传。
- (2) 在 Xshell 中连接到服务器，输入 `nvcc -o base base.cu` 进行编译。
- (3) 继续输入 `./base` 测试三次。

2. 并行修改 nbody1.cu



(1)在 cmd 中输入 `scp -P 14004 "E:\VS\CUDA\nbody1.cu" a2022150501027@121.48.170.1:/home/a2022150501027/` 输入密码并完成代码上传。

(2)在 Xshell 中连接到服务器，输入 `nvcc -o nbody1 nbody1.cu` 进行编译。

(3)继续输入 `./nbody1` 测试三次。

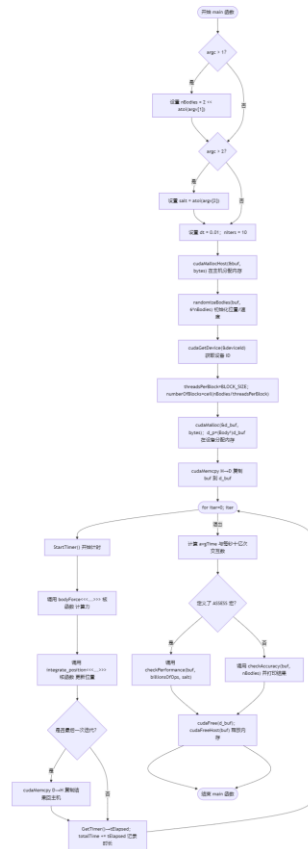
3. 性能优化 nbody2.cu

首先对原始代码进行并行化处理，每个线程负责处理一个位置的 body（可参考 `nbody_parallel.cu` 文件）。

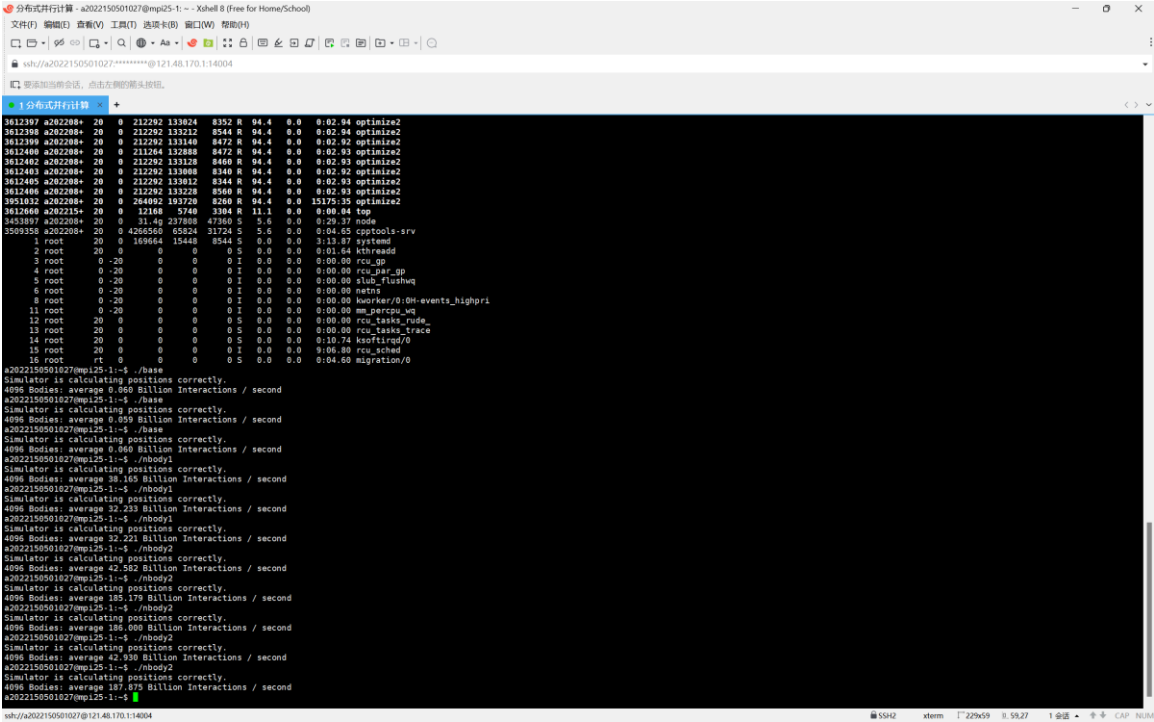
自行管理内存的拷贝、申请与释放，避免出现缺页异常等问题（例如将 `cudaMallocManaged` 替换为 `cudaMalloc` 和 `cudaMallocHost`）。

通过调整 `BLOCK_SIZE` 参数，找到相对最优的值（在测试环境中，该值为 32）。利用共享内存（`shared_memory`）进行优化，使一个线程块共享一块共享内存，每个线程负责处理部分数据，从而提高数据访问效率。

在观察 `body_force` 函数时发现，其主要运算为加法。因此，对原有的每个线程处理一个 body 的方式进行了改进，改为不同线程块中的多个线程共同处理一个 body 的数据信息，以此进一步提升并行效率。



九、数据分析：



	base	nbody1	nbody2
1	0.06	38.165	185.179
2	0.059	32.233	186
3	0.06	32.221	187.875
平均	0.059667	34.20633	186.3513
性能	1	573.2905	3123.207

十、实验结论：

所有代码都能够正确计算出坐标。在并行优化后，平均每秒交互 34.21（十亿次），性能变为原来的 573.29 倍；在进一步优化后，平均每秒交互 186.35（十亿次），性能变为原来的 3123.21 倍。

十一、总结及心得体会：

通过本次实验，我初步掌握了 CUDA C 程序的编写方法，学会了利用 CUDA API 进行内存管理以及原子操作。同时，我对 CUDA 的核心概念，尤其是共享内存，有了更深入的理解。目前，我能够通过分析程序代码，准确找到关键的优化点，并实施有效且高性能的优化措施。

朱若久