第三次作业说明

琚锡廷

2023 年 3 月 6 日

1 作业背景

GPU 具有计算性能高、线程多的特点，适合于大规模并行计算，可以有效提高程序的性能。上手实操 GPU 编程，可以让我们了解 GPU 的使用方式和优化策略，方便我们将 GPU 用在实际任务中。

模板计算是程序中常见的循环运算模式，比如熟知的热传导问题，有限差分问题的显式求解等，都可以归结为模板计算问题。模板计算问题有天然的并行性，计算相对规则，访存优化和指令级并行，以及寻找算法级的时间并行性是其主要的优化手段。熟悉模板计算的优化可以让我们更容易的将并行优化技术应用到实际当中。

2 作业描述

2.1 作业目标

掌握 GPU 上基本的并行程序实现和优化手段。

将课上所学的并行优化技术应用到常见的模板计算问题中。

2.2 作业要求

1. 独立完成代码实现与优化。

2. 提交文件夹命名格式为学号 + 作业编号 + 姓名，如第三次作业：2022000000_h3_name，其中包含文件夹 stencil_GPU，和报告 report.pdf。

3. 注意 DDL，以网络学堂发布的为准。

2.3 作业任务：GPU 模板计算并行优化

2.3.1 任务描述

在GPU 单卡上对模板计算进行并行实现和性能优化。

在本次作业中需要实现的模板计算是 27 点模板计算，和第二次作业中的算例相同。大家可以比较一下相同算例下，CPU 双处理器 (即 CPU 单节点) 和 GPU 单卡的计算性能差异。

基础代码可以从网络学堂上获取。

解压 homework3.zip 后，有 stencil_GPU 一个子目录，是模板计算在 GPU 上的实现。

stencil 文件夹中，stencil-naïve 是基础的实现，供同学们熟悉。而 stencil-optimized 是需要自行实现的部分。其中 create_dist_grid 函数可以根据进程划分自行改动。函数中的 halo_size 可以理解为了方便计算，人为在最外层加了一层网格点，以避免在边界进行额外的判断与反复通信。若有同学使用的优化算法对边界有特殊要求，则可以自行修改 halo_size 的值。

stencil_27 为需要实现的模板计算。函数的前两个输入参数为两个指针，指向两个数组，其中第一个数组 grid 在输入时存储的是初值，即初始输入的数据；aux 是用于存储迭代过程中间值的另一个数组，在迭代过程中，grid 和 aux 这两个数组会作为滚动数组循环使用。两个数组外层均有额外添加 halo，且 halo 层已初始化为 0。依据迭代步数，最终的计算结果会存在其中一个数组中，请返回对应数组的指针。第三个参数是模板计算的网格维度以及 halo 区的信息。第四个参数是迭代的步数。

编译可以通过提供的 Makefile 来实现。

```
1 make benchmark-optimized
```

test.sh 是正确性验证脚本，只有通过了 test.sh 才会被判断为有效的作业。test.sh 的命令行参数以及其含义和 benchmark.sh 相同。

2.3.2 任务要求

请统一使用双精度浮点数据类型进行计算，运算参考结果由自带的 benchmark 给出。

2.4 作业评分

2.4.1 模板计算 100%

1. 评测 stencil 的性能结果，按照提交后的多个固定计算规模的 stencil_27 程序平均性能排序，以及代码质量进行打分 (60%)。

2. **详细描述**采取的优化手段，代码对应的部分，以及对应的实验方案 (例如测试次数，测试性能取值方式等) 与结果，可以采用性能工具或者模型来解释目前取得的性能结果 (30%)。

3. 给出一张完整的实验结果图，描述当前算法的性能，横坐标为数据规模，纵坐标为 $Gflop/s$(10%)。

3 参考资料

基础的编程参考资料：CUDA 文档，GPU 官方样例。

stencil 计算可以参考stencilProbe。还有一些文献介绍了在 GPU 上的优化方法 [1], [2]，[3], [4]。大家可以选择性的阅读，也可以在网上自行查找其他的相关工作。

参考文献

[1] Marcin Krotkiewski and Marcin Dabrowski. Efficient 3d stencil computations using cuda. *Parallel Computing*, 39(10):533–548, 2013.

[2] Anamaria Vizitiu, Lucian Itu, Cosmin Niţă, and Constantin Suciu. Optimized three-dimensional stencil computation on fermi and kepler gpus. In *2014 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–6. IEEE, 2014.

[3] Uday Bondhugula, Vinayaka Bandishti, and Irshad Pananilath. Diamond tiling: Tiling techniques to maximize parallelism for stencil computations. *IEEE Transactions on Parallel and Distributed Systems*, 28(5):1285–1298, 2016.

[4] Prashant Singh Rawat, Miheer Vaidya, Aravind Sukumaran-Rajam, Atanas Rountev, Louis-Noël Pouchet, and P Sadayappan. On opti-

mizing complex stencils on gpus. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 641–652. IEEE, 2019.