并行程序设计与算法第二次作业

学号	姓名	邮件	专业
21307172	刘俊杰	liujj255@mail2.sysu.edu.cn	计算机科学与技术

习题1

CUDA中的__syncthreads()函数如何确保同一个线程块内的线程协调它们的活动？请解释其工作原理并说明其作用。

工作原理

CUDA中的`__syncthreads()`函数确保同一个线程块内的线程协调它们的活动，起到一个屏障同步的作用。具体而言，当一个线程块中的所有线程执行到`__syncthreads()`时，它们都会在此处等待，直到该线程块中的所有其他线程也到达这一点。只有当线程块内的所有线程都到达屏障点时，它们才会继续执行后续的代码。这确保了在`__syncthreads()`之前的所有操作在整个线程块中都完成之后，才开始执行`__syncthreads()`之后的操作。

作用

其作用在于保证数据的一致性和同步性，尤其是在需要多个线程协作处理数据的情境下非常重要。

习题2

(1)

当同一个warp中的线程遵循不同的执行路径时，我们称这些线程表现出什么现象？请解释这一现象对CUDA程序执行的影响。

现象

当同一个warp中的线程遵循不同的执行路径时，会出现“分支发散”。分支发散发生在warp中的不同线程在执行条件语句（如if-else）时，选择了不同的执行路径。

影响

这种现象会导致warp内的线程不能同时执行，必须分别执行不同路径的指令，从而降低并行效率和性能。

(2)

在实现一个向量加法核函数（addVecKernel）时，当向量长度不是线程块大小的整数倍时，会出现(1)中的现象。假设向量长度为1003，线程块大小为64，一个warp有32个线程，请简述出现此现象的原因，并分析向量长度分别为100、10000时此现象对性能的影响。

一个warp包含32个线程，这个条件在讨论CUDA程序中的分支发散和性能影响时是非常重要的。它直接影响了线程的调度和执行效率。让我们重新分析一下，考虑到warp的特性，如何影响不同向量长度下的性能。

原因

由于向量长度不是线程块大小的整数倍时，最后一个线程块可能会包含不足一个warp的线程，或者一个warp内部分线程没有有效工作。这会导致分支发散现象。

向量长度为100时此现象对性能的影响

- **线程块和warp划分**：
  - 向量长度100，需要 $\lceil \frac{100}{64} \rceil = 2$ 个线程块。
  - 第1个线程块处理64个元素，包含2个warp。
  - 第2个线程块处理36个元素，包含2个warp（第一个warp处理32个元素，第二个warp处理4个元素）。
- **分支发散**：

- 第2个线程块的第一个warp处理32个元素，没有分支发散。
- 第2个线程块的第二个warp中，只有4个线程需要工作，其他28个线程无有效工作，造成分支发散。

性能影响:
由于总线程数较少，尽管第二个线程块的第二个warp存在较大比例的无效工作，但总体影响相对有限。

向量长度为10000时此现象对性能的影响

- **线程块和warp划分**:
  - 向量长度10000，需要 $\lceil \frac{10000}{64} \rceil = 157$ 个线程块。
  - 前156个线程块处理9984个元素，每个线程块包含2个warp。
  - 第157个线程块处理16个元素，包含1个warp和另1个warp的16个线程。
- **分支发散**:
  - 前156个线程块（9984个元素）完全填满，64个线程全部工作，无分支发散。
  - 第157个线程块，只有1个warp处理16个元素，其余16个线程无效工作，造成分支发散。

性能影响:
尽管最后一个线程块存在分支发散，但对总共10000个元素的计算来说，仅仅最后32个线程中的16个线程无效工作的比例非常小。因此，分支发散对总体影响相对有限。

习题3

CUDA中的存储层次结构主要包括哪些类型的内存？请分别指出每种内存的作用域、生存周期以及硬件所在位置（片上/片外）。

1. **全局内存（Global Memory）**
   - 作用域：所有线程
   - 生存周期：程序运行期间
   - 硬件所在位置：片外
2. **共享内存（Shared Memory）**
   - 作用域：线程块内的所有线程
   - 生存周期：线程块的生命周期
   - 硬件所在位置：片上
3. **常量内存（Constant Memory）**
   - 作用域：所有线程
   - 生存周期：程序运行期间
   - 硬件所在位置：片外（但有片上缓存）
4. **纹理内存（Texture Memory）**
   - 作用域：所有线程
   - 生存周期：程序运行期间
   - 硬件所在位置：片外（但有片上缓存）
5. **本地内存（Local Memory）**
   - 作用域：单个线程
   - 生存周期：线程的生命周期
   - 硬件所在位置：片外
6. **寄存器（Registers）**
   - 作用域：单个线程
   - 生存周期：线程的生命周期
   - 硬件所在位置：片上

习题4

请简要解释CUDA全局内存上的统一内存寻址技术，并说明其优缺点。
统一内存寻址技术:
使用相同的内存地址（指针）在主机和设备上进行访问,允许CPU和GPU共享一块内存地址空间，而无需显式的内存拷贝操作,简化了内存管理。

优点:

- 统一内存中创建托管内存池,底层系统在统一内存空间中自动在主机和设备间进行传输,简化程序员视角中的内存管理
- 减少了因内存拷贝错误导致的bug，提供更直观的编程模型。

缺点:

- 自动数据传输机制可能导致性能损失，特别是在大量数据传输或频繁访问内存时。
- 统一内存可能引入更高的访问延迟，特别是在CPU和GPU频繁切换访问时。

习题5

在CUDA编程中，什么是共享内存的存储体冲突？请解释共享内存存储体冲突的产生原因。

共享内存的存储体冲突
在CUDA编程中，共享内存的存储体冲突是指多个线程同时访问共享内存中的不同地址，但这些地址映射到相同的内存存储体。这会导致访问顺序执行，从而降低并行性能。

产生原因:
共享内存被分成多个存储体（通常是32个或64个），每个存储体可以同时服务一个线程的请求。如果多个线程同时访问不同地址但这些地址位于同一个存储体，就会发生存储体冲突。

习题6

现有一个GPU，其硬件限制如下：每个SM（流式多处理器）最多2048个线程，最多32个线程块（block），以及最多64K（65,536）个寄存器。对于以下kernel特性，判断该kernel是否可以达到满占用（full occupancy）。如果不能，说明限制因素是什么。

a. kernel的每个block有128个线程，每个线程有30个寄存器。

- 每个线程块的寄存器需求：128 * 30 = 3840
- 每个SM的寄存器限制：65536
- 最大线程块数：65536 / 3840 ≈ 17
- 硬件限制：每个SM最多32个线程块

因为寄存器数量限制，最多可以有17个线程块，无法达到满占用。

b. kernel的每个block有32个线程，每个线程有29个寄存器。

- 每个线程块的寄存器需求：32 * 29 = 928
- 每个SM的寄存器限制：65536
- 最大线程块数：65536 / 928 ≈ 70
- 硬件限制：每个SM最多32个线程块

由于线程块数量限制，最多可以有32个线程块，可以达到满占用。

c. kernel的每个block有256个线程，每个线程有34个寄存器。

- 每个线程块的寄存器需求：256 * 34 = 8704
- 每个SM的寄存器限制：65536
- 最大线程块数：65536 / 8704 ≈ 7
- 硬件限制：每个SM最多32个线程块

因为寄存器数量限制，最多可以有7个线程块，无法达到满占用。