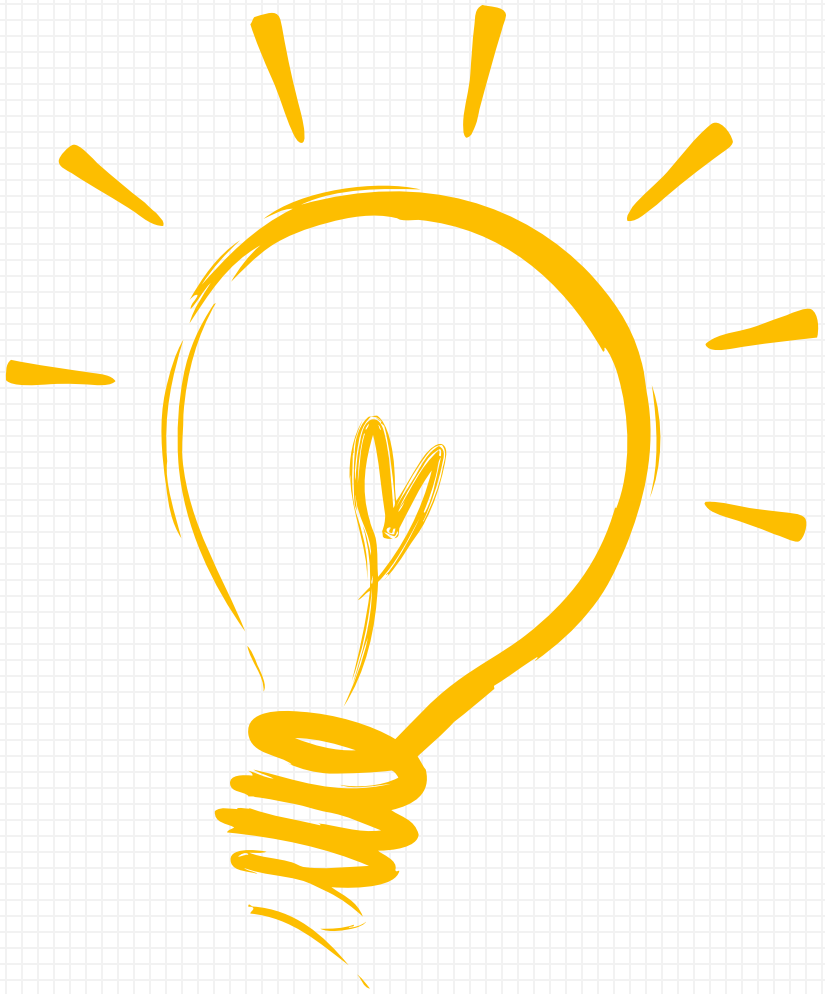
避免线程束分化

CUDA并行编程系列课程

主讲：权双

CONTENTS

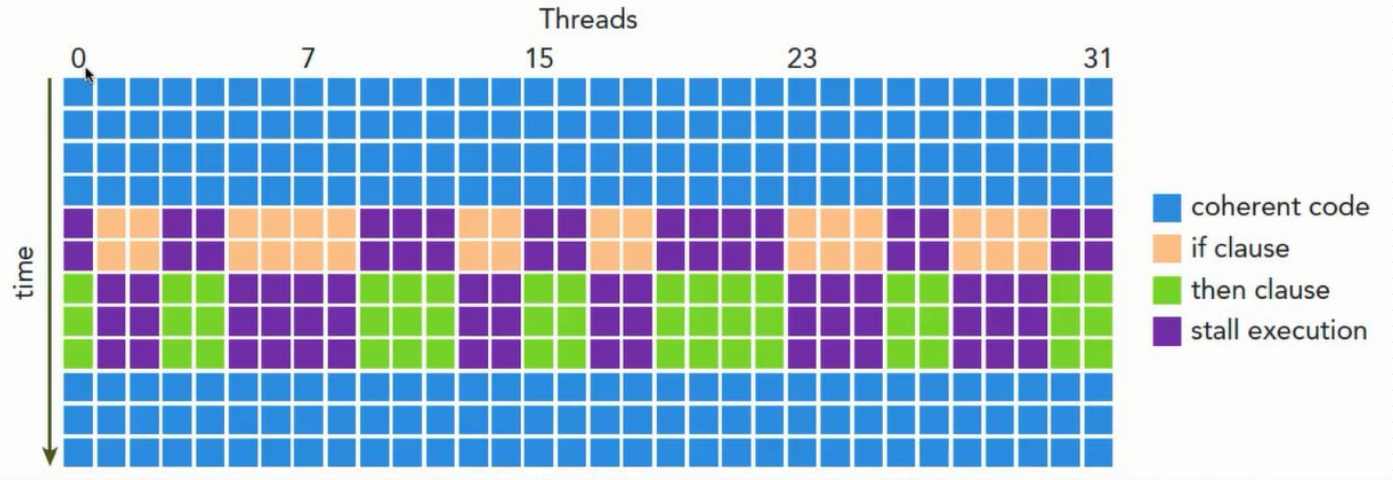


线程束分支

- GPU支持传统的、C/C++风格的显式控制流结构，如if…then…else for和while；
- GPU是相对简单的设备，没有复杂的分支预测机制；
- 一个线程束中的所有线程在同一个周期中必须执行相同的指令；
- 如果同一个线程束中的线程执行不同分支的指令，则会造成线程束分支；

```
__global__ void kernel(float* A)
{
    int tid = blockIdx.x * blockDim.x + threadIdx.x;
    float a = 0.0f;
    float b = 0.0f;
    if (tid % 2 == 0)          { a = 10.0f;}
    else                       {b = 20.0f;}

    A[tid] = a + b;
}
```



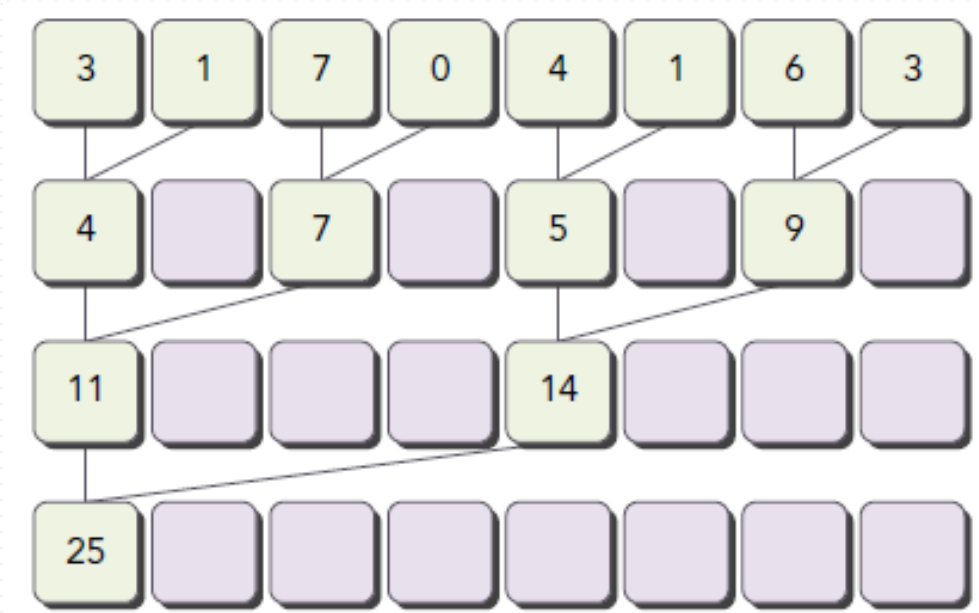
线程束分支

- ★ 线程束分支会降低GPU的并行计算能力，条件分支越多，并行性削弱越严重；
- ★ 线程束分支只发生在同一个线程束中，不同线程束不会发生线程束分化；
- ★ 为获取最佳性能，应避免在同一个线程束中有不同的执行路径；

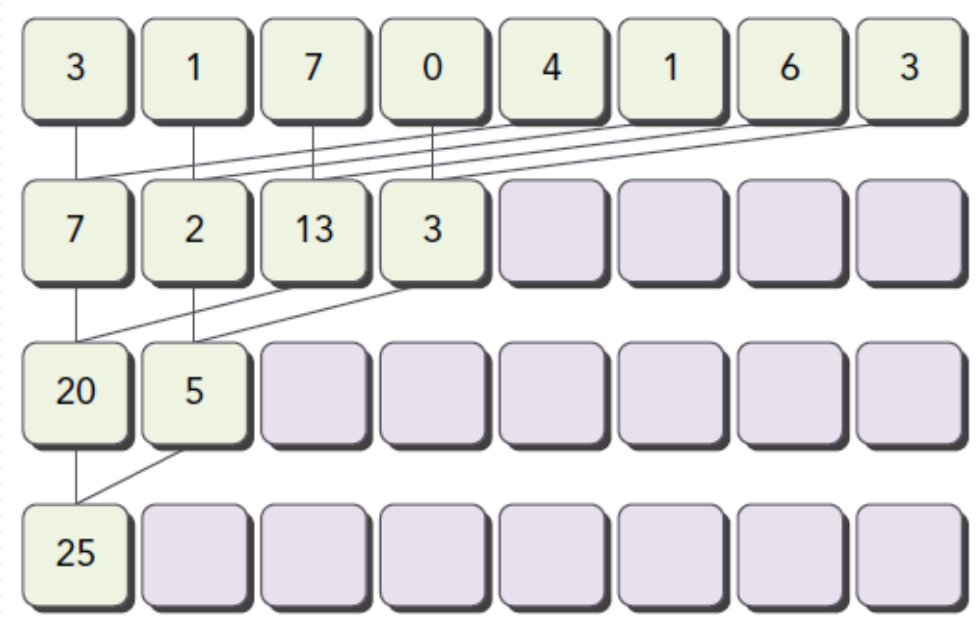
```
if ((tid / 32) % 2 == 0)        { a = 10.0f;}
    else                        {b = 20.0f;}
```

并行规约计算

★ 在向量中满足交换律和结合律的运算，称为规约问题，并行执行的规约计算称为并行规约计算；



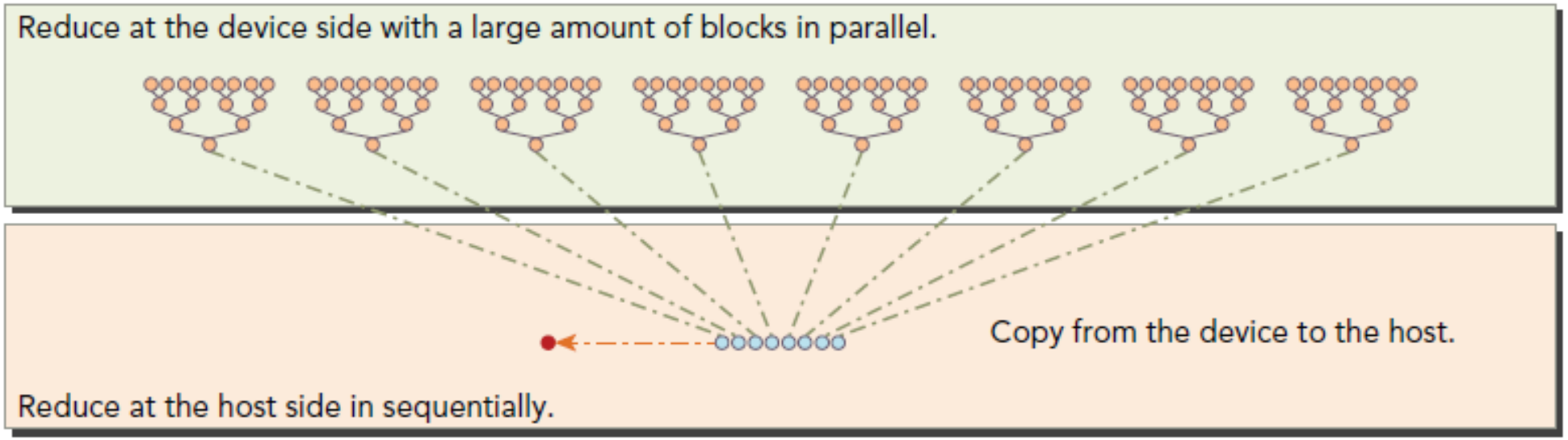
邻域并行计算



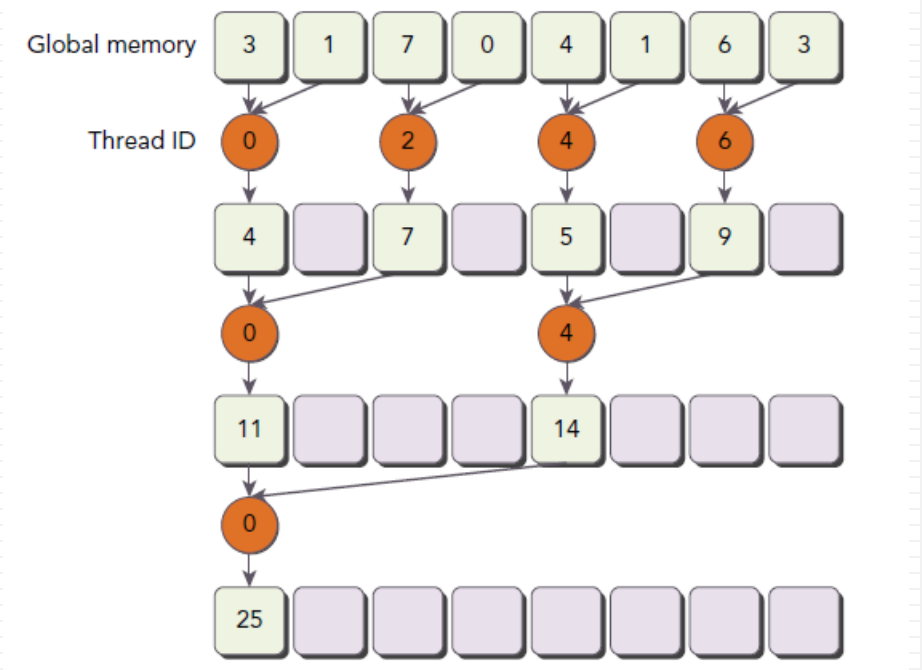
间域并行计算

并行规约计算

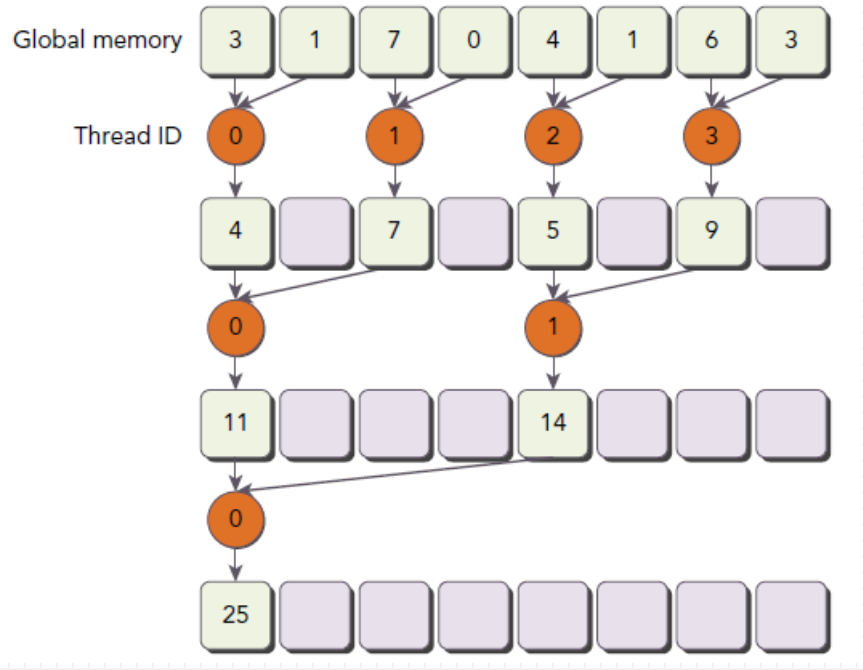
★ 假设要计算4096个元素求和，设计线程块大小为512，每个线程负责一个数据元素，共需8个线程块。



并行规约计算



线程束分支



无线程束分支

★ 严重的线程束分化，例如设置512个线程的线程块，第一轮归约，16个线程束都参与计算，每个线程束只有16个线程参与计算。接下来的计算依旧线程束分化。。。

★ 无线程束分支，例如设置512个线程的线程块，前8个线程束进行第一轮归约，剩下8个线程束什么也不做；第二轮，前4个线程束执行归约，剩下剩下12个线程束什么也不做；在最后5轮，数据的数量少于线程束大小时，同样会产生线程束分化。

THANKS

谢谢聆听