week9实验记录

zxp

April 27, 2024

1 environment

cpu:Intel(R) Xeon(R) Gold 6330 CPU @ 2.00GHz (使用时申请了56个核心并且使用独占指令)

gpu:rtx3090(使用时申请了一块)

System:CentOS7

Compiler:9.5

2 code

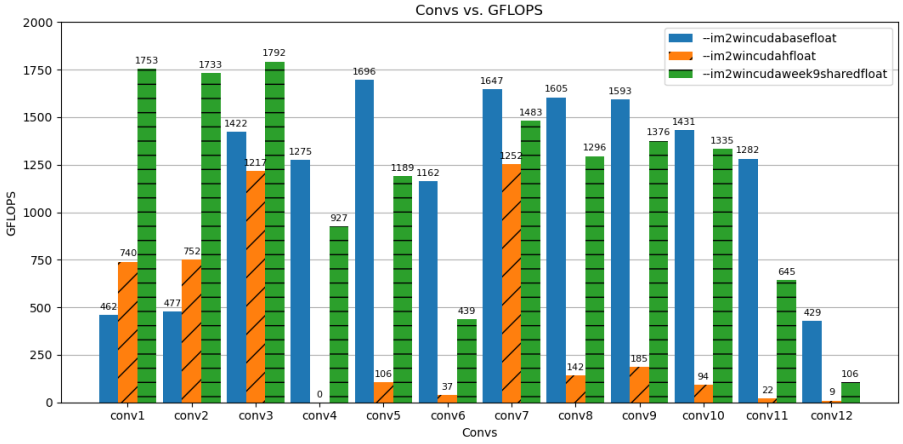

Figure 1: GPU

3 Experiment

之前在GPU上实现了没有优化的im2win卷积（蓝色柱子），上周尝试使用共享内存去优化（橙色柱子）发现效果很差，这周还是使用共享内存，但换了分块的方式。这周的分块的方式是按input张量分块，因为input张量远比fliter大，于是在每个线程块内的共享内存都放入所有批次的filter，这样每个核函数不止计算一个output的元素，而是能将output张量所有的通道计算出来（output的通道用到的input元素一样，但用到fliter不一样不同，output的通道靠不同fliter的批次决定），将每个input分成不同的大小放入不同的线程块（按连续性最好复用最多的纬度宽分开），并且尽可能地填满共享内存（用input的高和通道的一部分），这样只需要将input放入共享内存一次，因为将input和filter的全部通道放入共享内存会太大，还需要在核函数外面用个循环历遍所有的input和filter的通道（绿色柱子）。

3.1 Analysis

效果要比上周好很多，并且conv1-3是正优化，并且conv1-2还比未优化快了很多。但在其他conv还是负优化，特别是conv12，表现不好的conv大部分都是通道特别大。不能只考虑填满共享内存，感觉cuda优化的核心就是怎么将运算分块，后续的优化都依赖分块和共享内存，不同conv大概率是需要不同的分块策略，感觉这几周都过分在意im2win张量中宽这个纬度复用的部分，但效果不好。不过将更小的filter全部放进去，计算的时候不止计算一个output的通道而是计算多个通道以此复用input张量是可行的，这周主要是分块策略还不好。

4 Experiment2

尝试使用向量化加载，CUDA中有特殊变量float4，一次加载4个float元素，好处是加载指令减少了一个时钟周期一个加载指令可以加载4个float，需要的带宽还是不变（卢帅师兄说这个优化提升很小，大概率是带宽限制了），但失败了，float4需要内存按16位对齐，但之前使用共享内存的时候并没有考虑这点，所以没法直接用，还需要考虑其他使用共享内存的方式。

4.1 Analysis

CUDA后面的优化高度依赖前面的分块和共享内存，后面还能做的优化并不多吗，还有数据预取和双缓冲区，主要还是需要合适的分块策略啊。