Abstract

随着硬件的专门化，计算核心的增幅远高于内存，一些计算密集型的操作也受到内存延迟的影响，再结合上其前后的其他内存密集型操作，多个计算密集型算子之间的融合内核变的受关注起来。典型例子就是flash attention。单独看矩阵乘法是计算受限的，而其特殊的尺寸和其中的softmax操作，使得GEMM的融合变得相当有价值。然而这种实现却没有被推广到更多GEMM的场景下，如QKV-生成，attention后的concat，以及FFN。之前没有实现的主要困难在于不同block间的数据交换只能通过global memory，而DSM的出现能够很好的缓解这个问题。此外，Block可以互通的场景也进一步扩展了WASP的概念，以往的WASP是不同warp不同的工作，而当不同block做不同工作时，数据只能通过global交换，这限制了可能的收益。

在本文中，我们提出不同的计算密集型算子的融合策略XXX，一个利用计算核心间互联特性，并针对不同场景选择合适融合策略的优化框架。提出了分布式数据流动抽象loc-dim-exchange(LDE)，LDE通过将每个block的任务位置，发生数据交换的维度映射到每个核心上，实现了核间通信开销，片上内存开销，计算通信重叠，多操作符可融合性的多因素权衡。

Discussion

* 四个权衡是什么，怎样选择？

核间通信开销（如果cluster很大，开销很大。），片上内存消耗（使用核间通信则需要消耗DSM，否则就是走global，但是occupancy可以提高。或者不占用DSM，而直接用寄存器的ld指令移动，速度慢一些，但是可以提高occupancy），计算通信重叠（pipeline的重叠更好，因为可以选择GEMM0和GEMM1分配的CTA数量。有可能更细粒度的调整。策略1在传输时没法计算，好吧也可以用tensorcore做累加，然后边传边算，策略2在传输的时候可以选择同时计算。），中间结果交换方式是否有利于后续计算（stationary的不同。）

* LDE是什么，三种策略下怎么表示？

数据依赖要考虑不同维度。比如在GEMM0的M维度上交换，那就是上下两个矩阵交换B矩阵的内容。在GEMM1的N维度交换，那就是不同block间做reduce。这种数据交换的抽象表达，有助于统一表示不同的融合策略。GEMM0的N维度和GEMM1的N维度相关联。（或者叫GEMM1的K维度），如果在GEMM0的N维度上发生数据交换，则GEMM1的N维度也要发生交换。MNK中，K是累加维，而MN是平等的。对于grid特别小，可以用DSM-reduce来做K的block交互。T10的场景就仅考虑M或者N。而我要考虑的就加入了GEMM0-N-GEMM1。我觉得应该以block负责的位置来表示。结合上哪个维度的数据交换。

比如策略1就是，cluster负责GEMM0的一行，和GEMM1的一块，然后MNKT中只有N发生了数据交换。（也可以增加M）。但是考虑到我们不希望在GEMM1的一行里安排多个block，所以T维度就不发生数据交换。

策略2就是，cluster负责GEMM0的一行，和GEMM1的一行。然后MNKT中N发生了数据交换，T也发生了数据交换。

策略3就是，cluster部分负责GEMM0的一行，部分GEMM1的一行。如果GEMM1分配了多个block，也可以在T上发生数据交换。

我的优化是基于GEMM的，这样就有MNKL四个维度。子汉他们是GEMV，这样就没有M维度。我可以不强调专门针对于大模型，我甚至可以做做conv，splitK。子汉他们可以更针对推理场景一些。