CANN
6.3.RC2

图融合和 UB 融合规则参考

文档版本 01

发布日期 2023-07-26

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 简介

概述

算子融合是整网性能提升的一种关键手段，包括图融合和UB融合。

系统内置了一些图融合和UB融合规则，均为默认开启（如有默认关闭的，会特殊说明）。当前文档仅对部分融合规则进行介绍。

图融合

客户在使用图方式描述网络时，采用跟硬件无关的融合优化，实现算子性能提升。

图融合是FE根据融合规则进行改图的过程。图融合用融合后算子替换图中融合前算子，提升计算效率。图融合的场景如下：

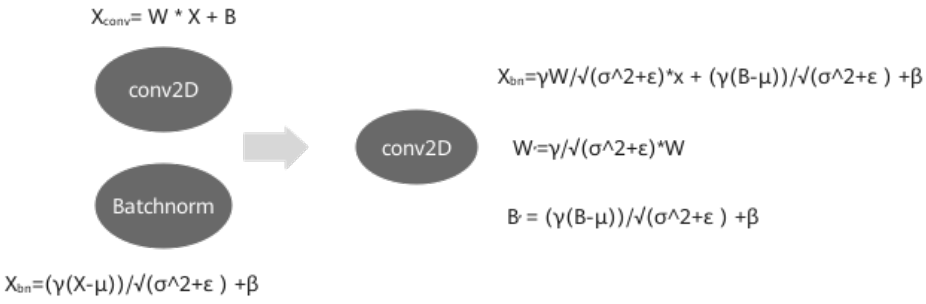
- 在某一些算子的数学计算量可以进行优化的情况下，可以进行图融合，融合后可以节省计算时间。例如：conv+biasAdd，可以融合成一个算子，直接在l0c中完成累加，从而省去add的计算过程。
- 在融合后的计算过程中可以通过硬件指令加速的情况下，可以进行图融合，融合后能够加速。例如：conv+biasAdd的累加过程，就是通过l0c中的累加功能进行加速的，可以通过图融合完成。

图融合包括：图融合、图拆分融合。

- 图融合：对图上算子进行数学相关的融合，将多个算子融合成一个或者几个算子，该融合跟硬件无关。

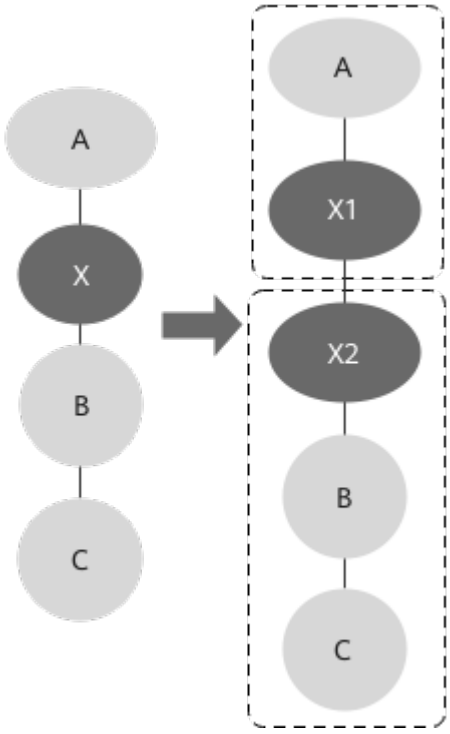
  如图1-1所示，conv2D和BatchNorm算子做融合，经过数学公式的推导，将batchNorm作用到conv2D上，融合成了conv2D算子。

  图 1-1 原图融合示例



- 拆分融合：将一个算子拆分成多个算子的融合。
  如图1-2所示，算子X被拆分成X1和X2两个算子。

  图 1-2 拆分融合



UB 融合

客户在使用图方式描述网络时，经过图编译对图进行UB融合优化，实现硬件相关的融合优化，提升算子执行性能。

UB即昇腾AI处理器上的Unified Buffer，UB融合是对图上算子进行硬件UB相关的融合。例如两个算子单独运行时，算子1的计算结果在UB上，需要搬移到DDR。算子2再执行时，需要将算子1的输出由DDR再搬移到UB，进行算子2的计算逻辑，计算完之后，又从UB搬移回DDR。

从这个过程会发现1的结果从UB->DDR->UB->DDR。这个经过DDR进行数据搬移的过程是浪费的，因此将1和2算子合并成一个算子，融合后算子1的数据直接保留在UB，算子2从UB直接获取数据进行算子2的计算，节省了一次输出DDR和一次输入DDR，省去了数据搬移的时间，提高运算效率，有效降低带宽。

图 1-3 UB 融合

如何关闭/开启融合规则

用户可以在模型编译时，提前识别是否需要关闭/开启某些融合规则，便于提升编译性能，但并不会提升计算性能。方法如下：

- ATC模型转换时，通过--fusion_switch_file配置融合开关配置文件路径以及文件名。示例：
  ```
  --fusion_switch_file=/home/fusion_switch.cfg
  ```
- IR模型构建时，通过FUSION_SWITCH_FILE配置融合开关配置文件路径以及文件名。示例：
  ```
  std::map<Ascendstring, Ascendstring> global_options = {
        {ge::ir_option::FUSION_SWITCH_FILE, "/home/fusion_switch.cfg"},
    };
  auto status = aclgrphBuildInitialize(global_options);
  ```
- 模型训练和在线推理时，通过fusion_switch_file配置融合开关配置文件路径以及文件名。示例：
  ```
  custom_op.parameter_map["fusion_switch_file"].s = tf.compat.as_bytes("/home/fusion_switch.cfg")
  ```

其中，传入的*fusion_switch.cfg*文件需要用户自己创建，文件名自定义，文件内容示例如下，on表示开启，off表示关闭。

```
{
    "Switch":{
        "GraphFusion":{
            "ConvToFullyConnectionFusionPass":"on",
            "SoftmaxFusionPass":"on",
            "ConvConcatFusionPass":"on",
            "MatMulBiasAddFusionPass":"on",
            "PoolingFusionPass":"on",
            "ZConcatv2dFusionPass":"on",
            "ZConcatExt2FusionPass":"on",
            "TfMergeSubFusionPass":"on"
        },
        "UBFusion":{
            "FusionVirtualOpSetSwitch":"on"
        }
    }
}
```

同时支持用户一键关闭/开启融合规则，如下以关闭为例。

```
{
    "Switch":{
```



```
    "GraphFusion":{
        "ALL":"off"
    },
    "UBFusion":{
        "ALL":"off"
    }
  }
}
```

需要注意的是：

1. 以上一键式关闭融合规则仅是关闭系统部分融合规则，而不是全部融合规则，原因是关闭某些融合规则可能会导致功能问题。
2. 一键式关闭融合规则的同时，可以开启部分融合规则：

   ```
   {
       "Switch":{
           "GraphFusion":{
               "ALL":"off",
               "SoftmaxFusionPass":"on"
           },
           "UBFusion":{
               "ALL":"off",
               "FusionVirtualOpSetSwitch":"on"
           }
       }
   }
   ```

3. 一键式开启融合规则的同时，可以关闭部分融合规则：

   ```
   {
       "Switch":{
           "GraphFusion":{
               "ALL":"on",
               "SoftmaxFusionPass":"off"
           },
           "UBFusion":{
               "ALL":"on",
               "FusionVirtualOpSetSwitch":"off"
           }
       }
   }
   ```


2 图融合规则说明

MatmulCastFusionPass

MatMulBiasAddFusionPass

DeconvWeightTransFusionPass

Conv2DbpInputBiasAddFusionPass

BatchMatMulV2ReshapeFusionPass

BatchMatmulFusionPass

SwapMergeCastFusionPass

PSROIPoolingFusionPass

ConvWeightCompressFusionPass

ConcatQuantFusionPass

BatchMatmulV2ReduceFusionPass

BatchMatmulNonAlignedFusionPass

Conv3DbpInputBiasAddFusionPass

FullyConnectionPowerPass

AFullyConnectionReshapePass

GemmTransFusionPass

MatmulTransdataFusionPass

Matmulv2FusionPass

Resnet50DbnDwFusionPass

AAMatMulNzToNdFusionPass

CastReluCastFusionPass

StrideHoistingPass

PadConv2dFusionPass

Conv2DTransposeBatchnormFusionPass

AvgPoolV2GradFusionPass

DropOutDoMaskFusionPass

TransposeSigmoidMulAddPowConcatFusionPass

ConvCastFusionPass

YoloxBoundingBoxDecodeONNXFusionPass

MatMulUnsqueezeSqueezeFusionPass

LayerNormSpecialTrainingFusionPass

StridedSliceConcatFusionPass

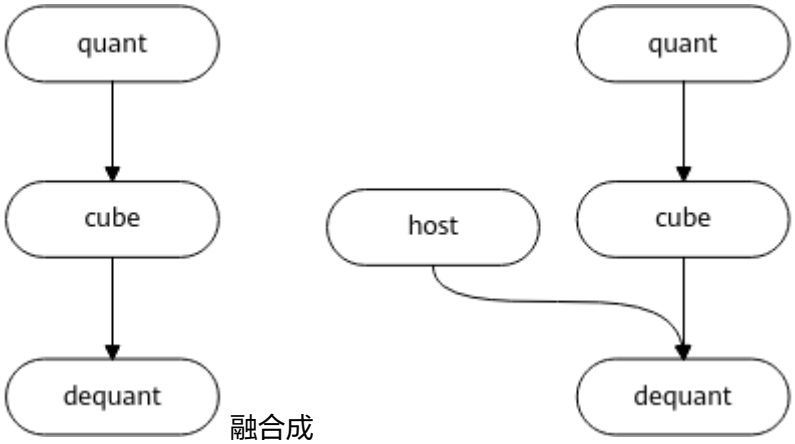
ZSplitVDFusionPassV2

2.1 V100RequantFusionPass

融合说明

该融合在推理场景下对量化节点进行优化。

在dequant的另一个输入插入host算子，同时在融合时，根据算法对相应算子参数进行调整。



使用约束

该融合规则主要用于推理网络量化模型时，对反量化算子融合处理。

模型小型化工具对原始框架模型进行量化时，会插入量化和反量化算子，而使用ATC工具进行模型转换过程中，会对插入的量化和反量化算子进行融合，此情况下就无法进行量化后模型dump结果与原始模型dump结果的比对，因此如果用户想使用通过模型小型化工具量化后的模型进行精度比对，则必须通过该配置文件，关闭融合功能。

配置文件模板如下所示，当前仅支持如下Pass规则的配置，使用该配置文件时，如下规则需要同时关闭。

```
V100RequantFusionPass:off                  //量化场景下，满足反量化(dequant)和量化(quant)相关pattern时，进行部署优化，提升推理性能
TbePool2dQuantFusionPass:off               //量化场景下，对Pool2d-quant连续的节点，标记UB融合，提升推理性能
```

支持的芯片型号

昇腾310 AI处理器

昇腾910 AI处理器

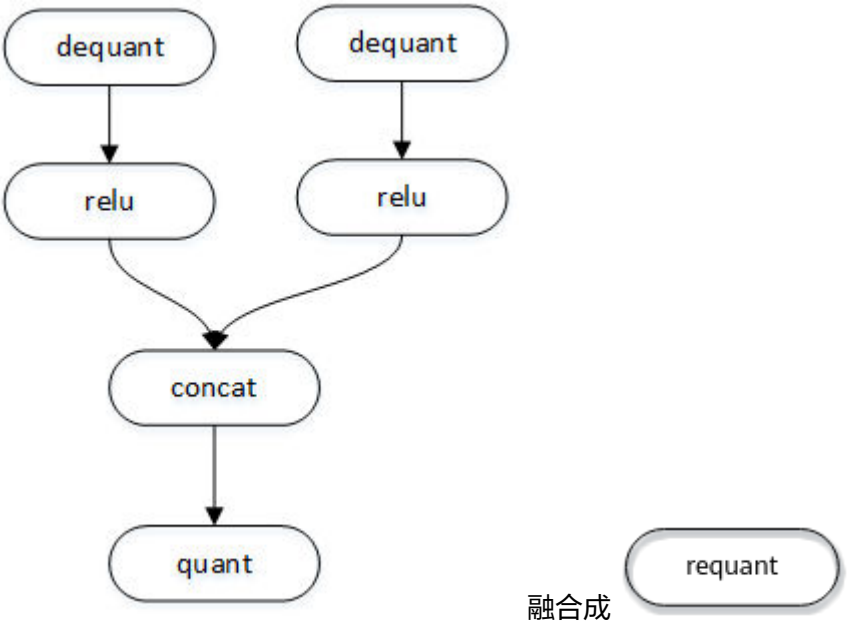
昇腾910B AI处理器

2.2 V200RequantFusionPass

融合说明

该融合在推理场景下对量化节点进行优化。

匹配dequant+relu+concat（可选）+quant，融合成requant，同时在融合时，根据算法对相应算子参数进行调整。



使用约束

该融合pass主要用于推理网络量化模型时，对反量化算子融合处理。

模型小型化工具对原始框架模型进行量化时，会插入量化和反量化算子，而使用ATC工具进行模型转换过程中，会对插入的量化和反量化算子进行融合，此情况下就无法进行量化后模型dump结果与原始模型dump结果的比对，因此如果用户想使用通过模型小型化工具量化后的模型进行精度比对，则必须通过该配置文件，关闭融合功能。

配置文件模板如下所示，当前仅支持如下Pass规则的配置，使用该配置文件时，如下规则需要同时关闭。

```
V200RequantFusionPass:off                  //量化场景下，满足反量化(dequant)和量化(quant)相关pattern时，进行部署优化，提升推理性能
TbePool2dQuantFusionPass:off               //量化场景下，对Pool2d-quant连续的节点，标记UB融合，提升推理性能
```

支持的芯片型号

昇腾310P AI处理器

2.3 ConvToFullyConnectionFusionPass

融合模式

该融合规则将Conv卷积算子融合修改为FullyConnection算子，提高计算性能：

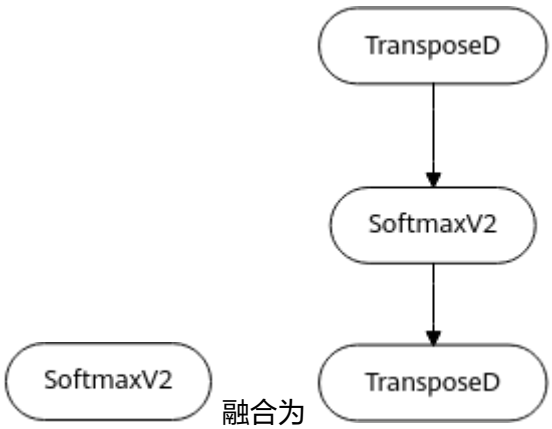
使用约束

无

2.4 SoftmaxFusionPass

融合模式

该融合规则将SoftmaxV2输入和输出添加TransposeD算子。



使用约束

无

2.5 V100NotRequantFusionPass

融合模式

对dequant、quant单算子处理，修改参数，在dequant的输入插入host算子。

使用约束

无

支持的芯片型号

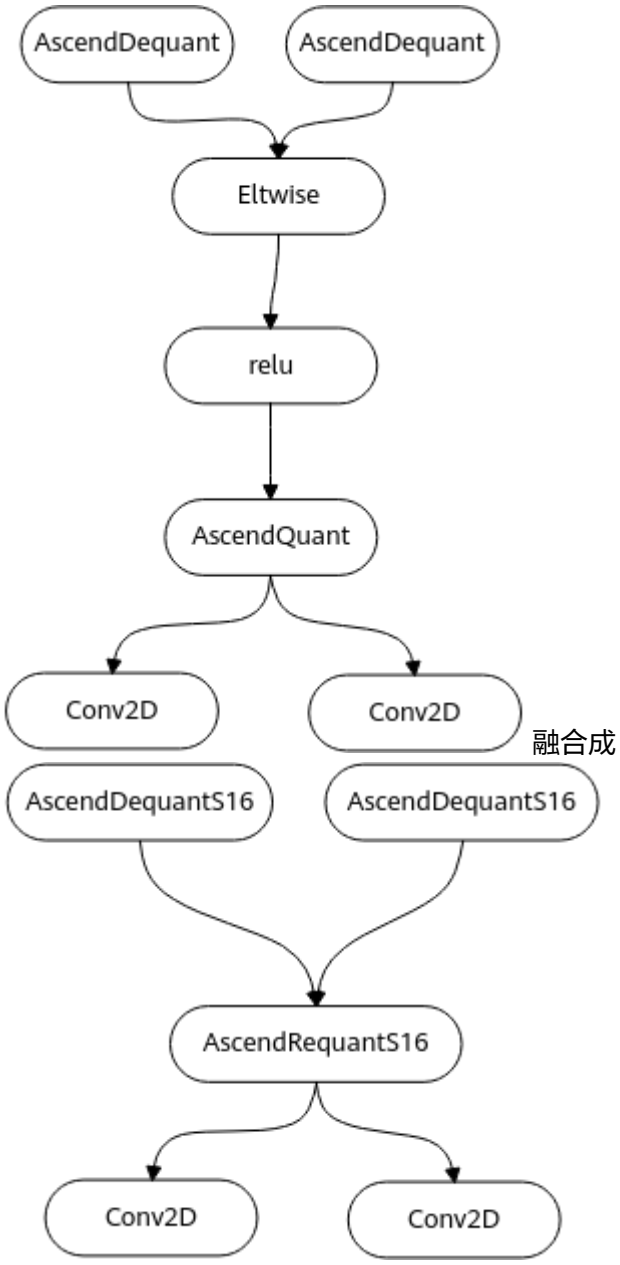
昇腾310 AI处理器

昇腾910 AI处理器

2.6 V200NotRequantFusionPass

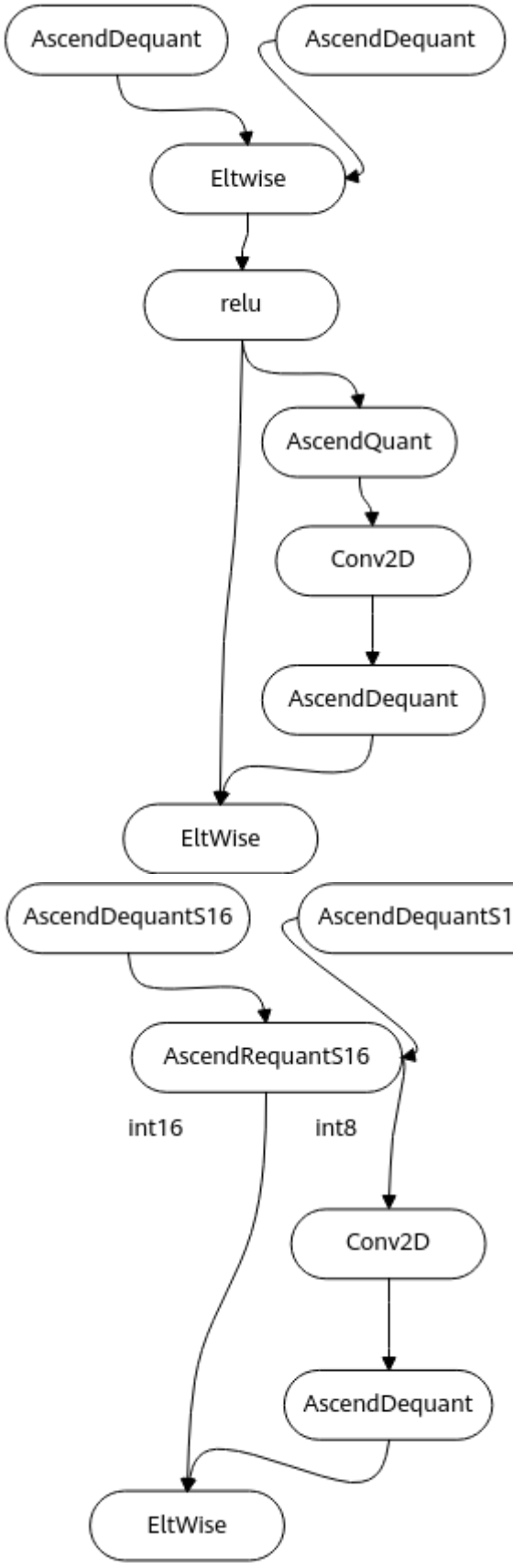
融合模式

融合场景如下：

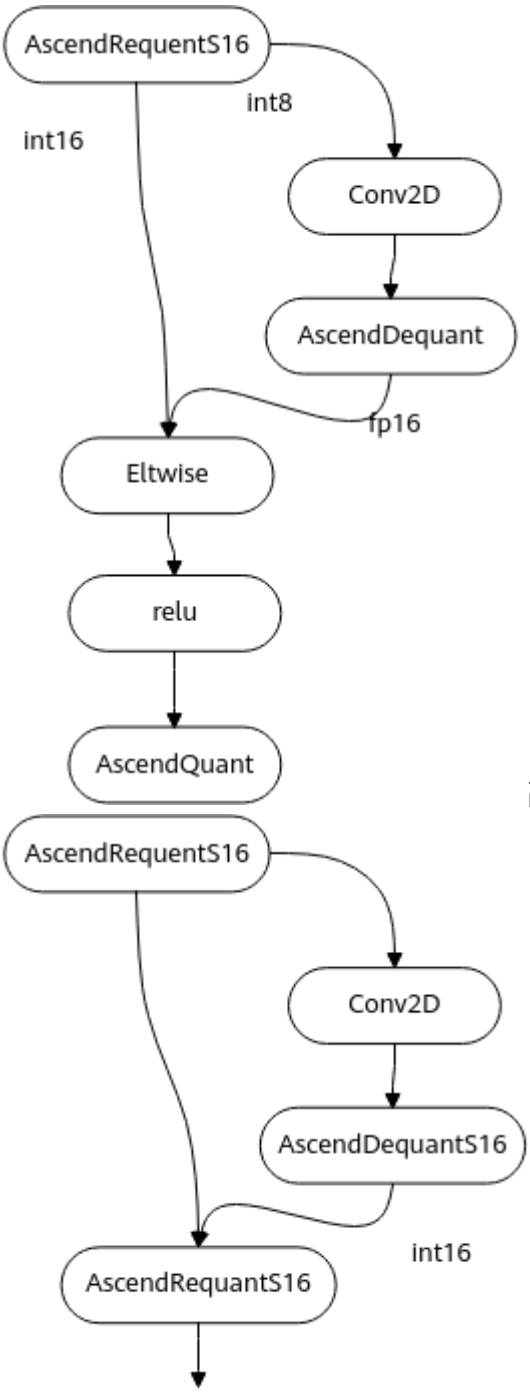

AscendDequant
AscendDequant
Eltwise
relu
AscendQuant
Conv2D
Conv2D
AscendDequantS16
AscendDequantS16
AscendRequantS16
Conv2D
Conv2D

融合成

或



或



使用约束

无

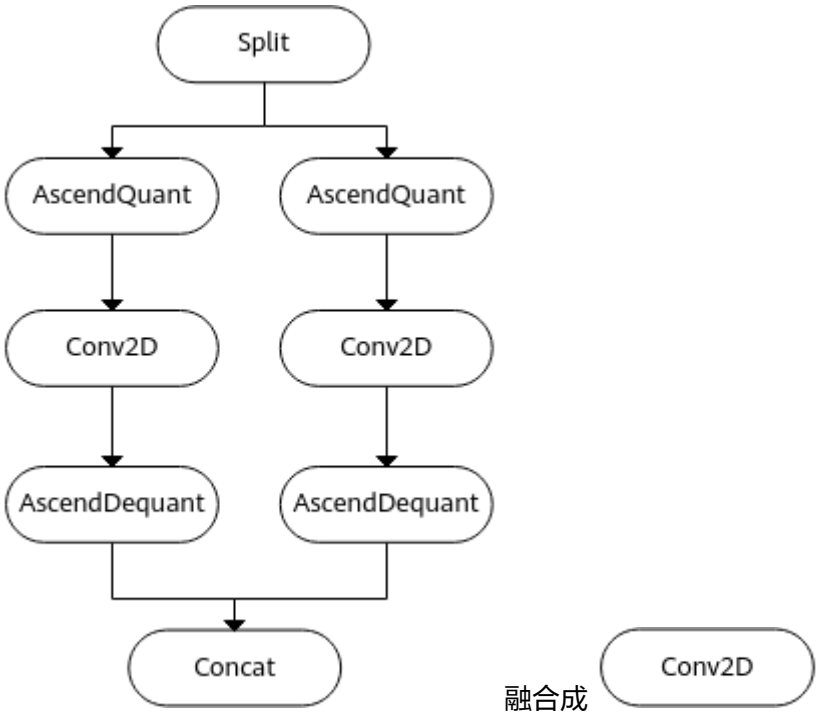
支持的芯片型号

昇腾310P AI处理器

2.7 SplitConvConcatFusionPass

融合模式

对包含Split、AscendQuant、Conv2D、AscendDequant算子相关场景进行融合：



使用约束

- 该融合结构会优先匹配ASplitConv2dConcatPass，不满足ASplitConv2dConcatPass约束条件才会匹配此融合pass。
- Split连接接的AscendQuant算子的scale、offset、sqrt_mode等属性需要一致。

支持的芯片型号

昇腾310 AI处理器

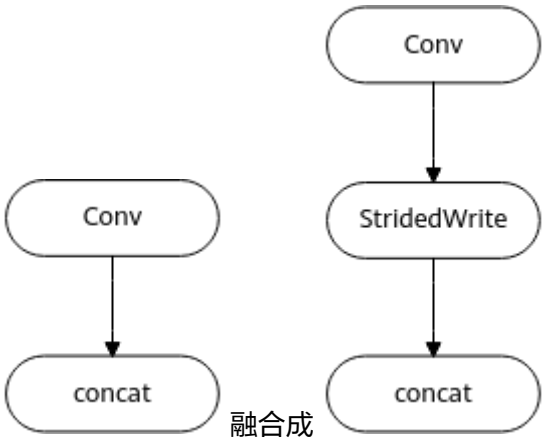
昇腾910 AI处理器

昇腾910B AI处理器

2.8 ConvConcatFusionPass

融合模式

在concat算子前插入跳写算子，将原先通过concat拼接多个Conv内存的方式，修改成通过StridedWrite算子进行Conv内存拼接，以消除concat算子任务执行带来的性能消耗。

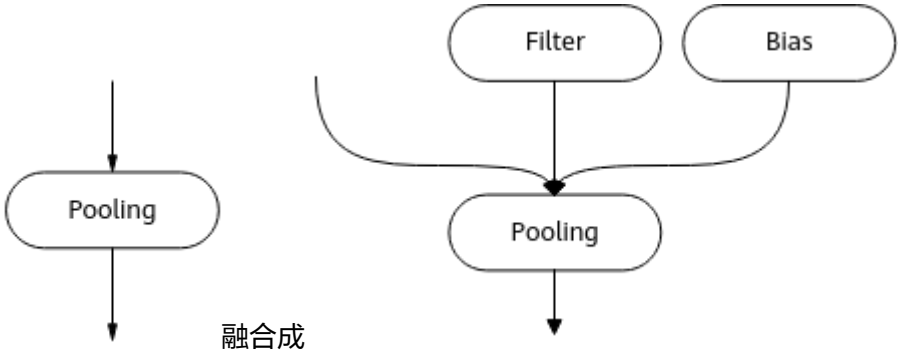


使用约束

量化场景下该融合规则必须打开，否则会导致transdata输出的dtype不支持。

2.9 PoolingFusionPass

融合模式



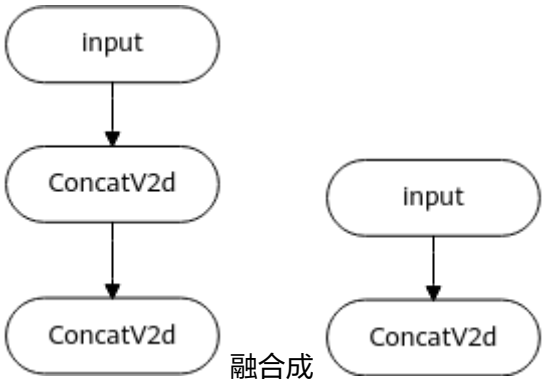
使用约束

在AVG Pooling下融合规则生效，如果网络模型中涉及AVG Pooling int8量化情况下，即prototxt文件中pooling属性参数pool为AVE时，该融合规则必须打开。

2.10 ZConcatv2dFusionPass

融合模式

该融合规则将多级Concat算子合并。



使用约束

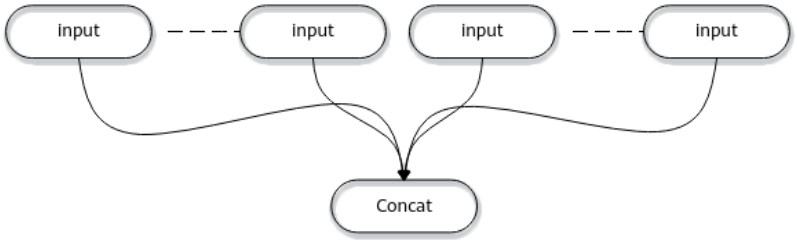
当存在某一ConcatV2d算子的入参tensor个数大于63个的时候，需要将该融合规则关闭：

```
ZConcatv2dFusionPass: off
```

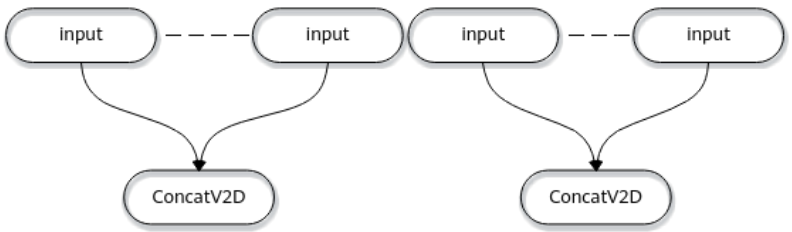
2.11 ZConcatExt2FusionPass

融合模式

该融合规则将Concat算子的多个输入拆分成多个ConcatV2D。



融合成



融合后ConcatV2D个数根据实际输入个数按照一定计算规则确定。

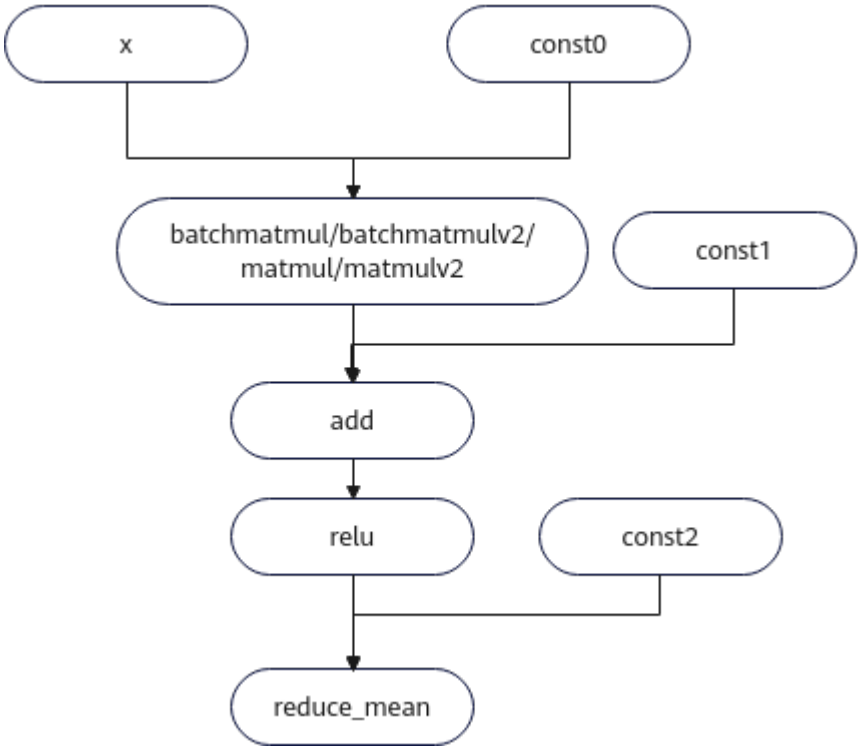
使用约束

无

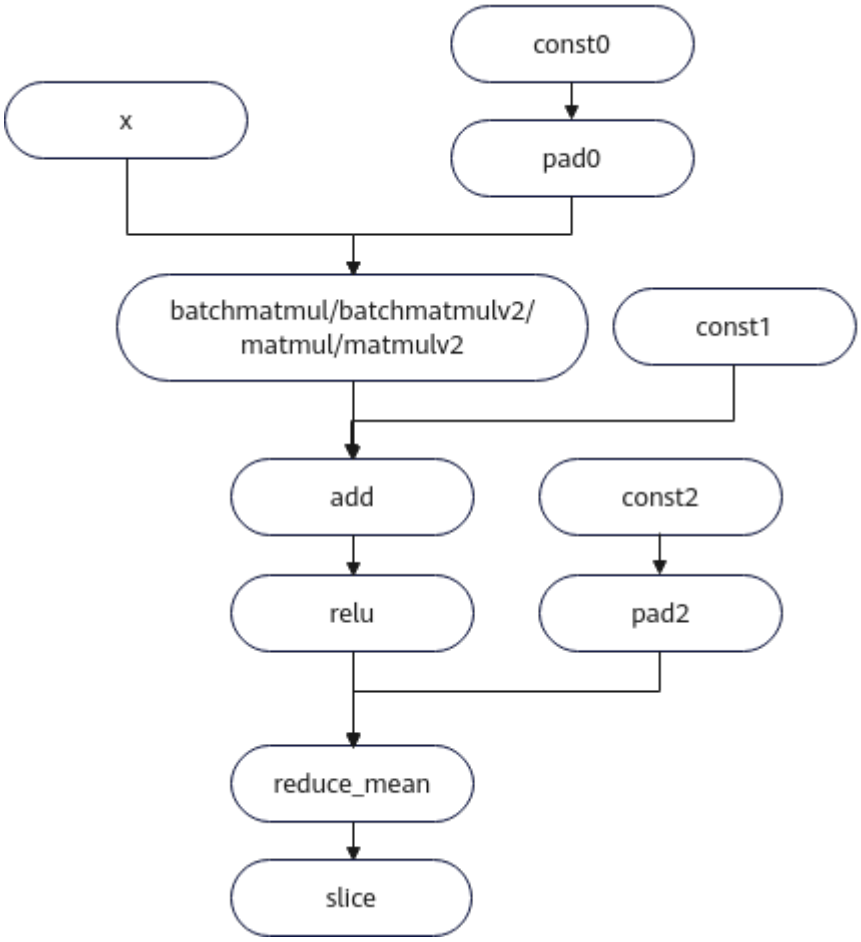
2.12 BatchMatMulReduceMeanFusionPass

融合模式

为batchmatmul、batchmatmulv2、matmul、matmulv2和reducemean算子节点的常量输入添加pad算子节点，提高计算性能。



融合为



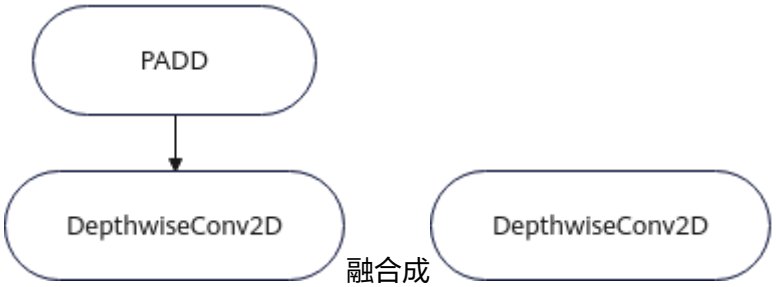
使用约束

无

2.13 PadDepthwiseConv2dFusionPass

融合模式

该融合规则将PadD+DepthwiseConv2D算子融合为 DepthwiseConv2D算子。



使用约束

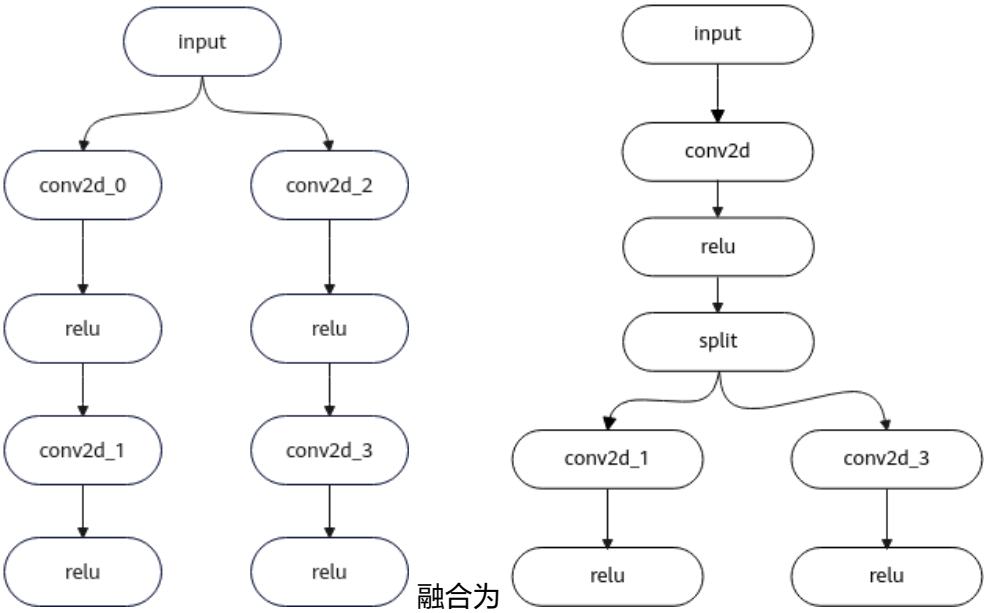
- 融合前DepthwiseConv2d的paddingMode必须为VALID。

- PADD算子不可指向多个DepthwiseConv2D算子。

2.14 SameInputConv2dPass

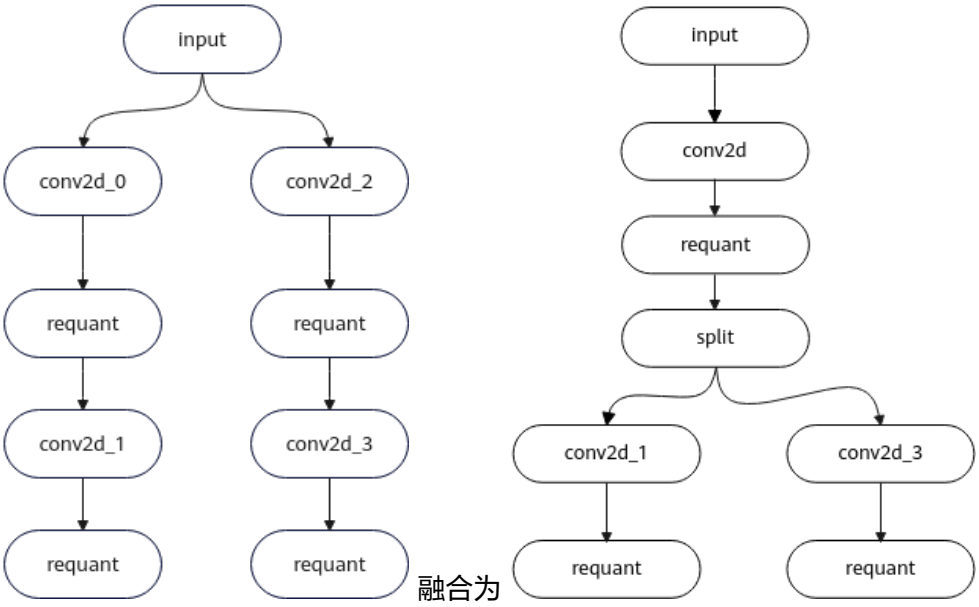
融合模式

非量化场景：该融合将符合约束条件的多路Conv2D+Relu算子融合成一路Conv2D+Relu+Spilt融合算子，如下图所示。

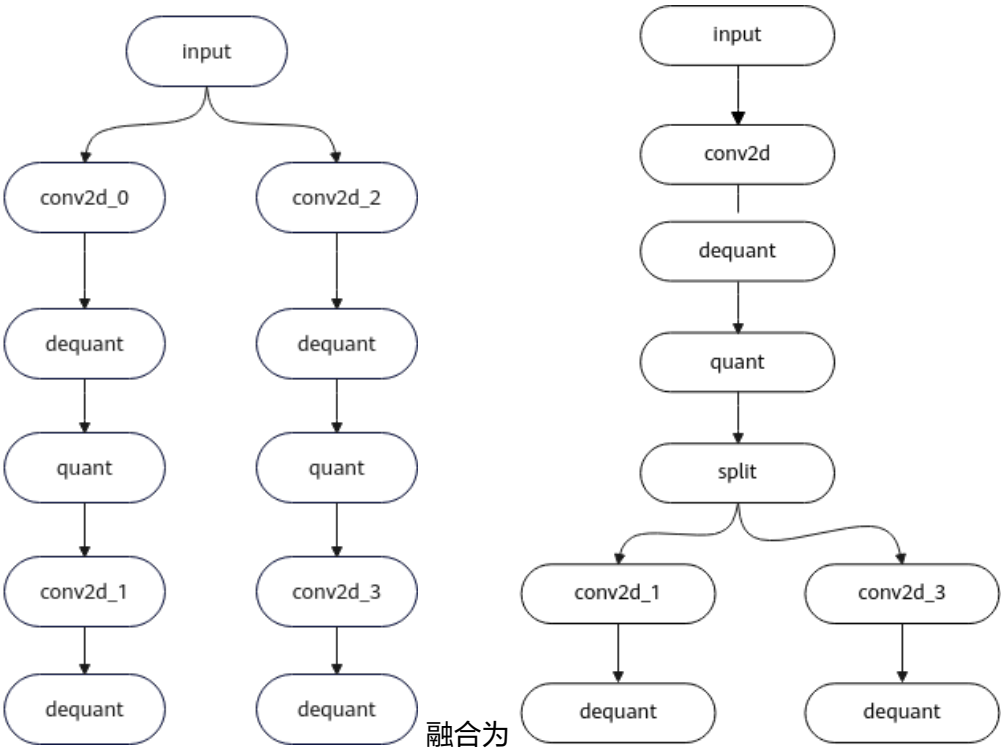


量化场景：

该融合将符合约束条件的多路Conv2D+AscendRequant算子融合成一路Conv2D+AscendRequant+Split融合算子，如下图所示。



该融合将符合约束条件的多路Conv2D+AscendDequant+AscendQuant算子融合成一路Conv2D+AscendDequant+AscendQuant+Split融合算子，如下图所示。



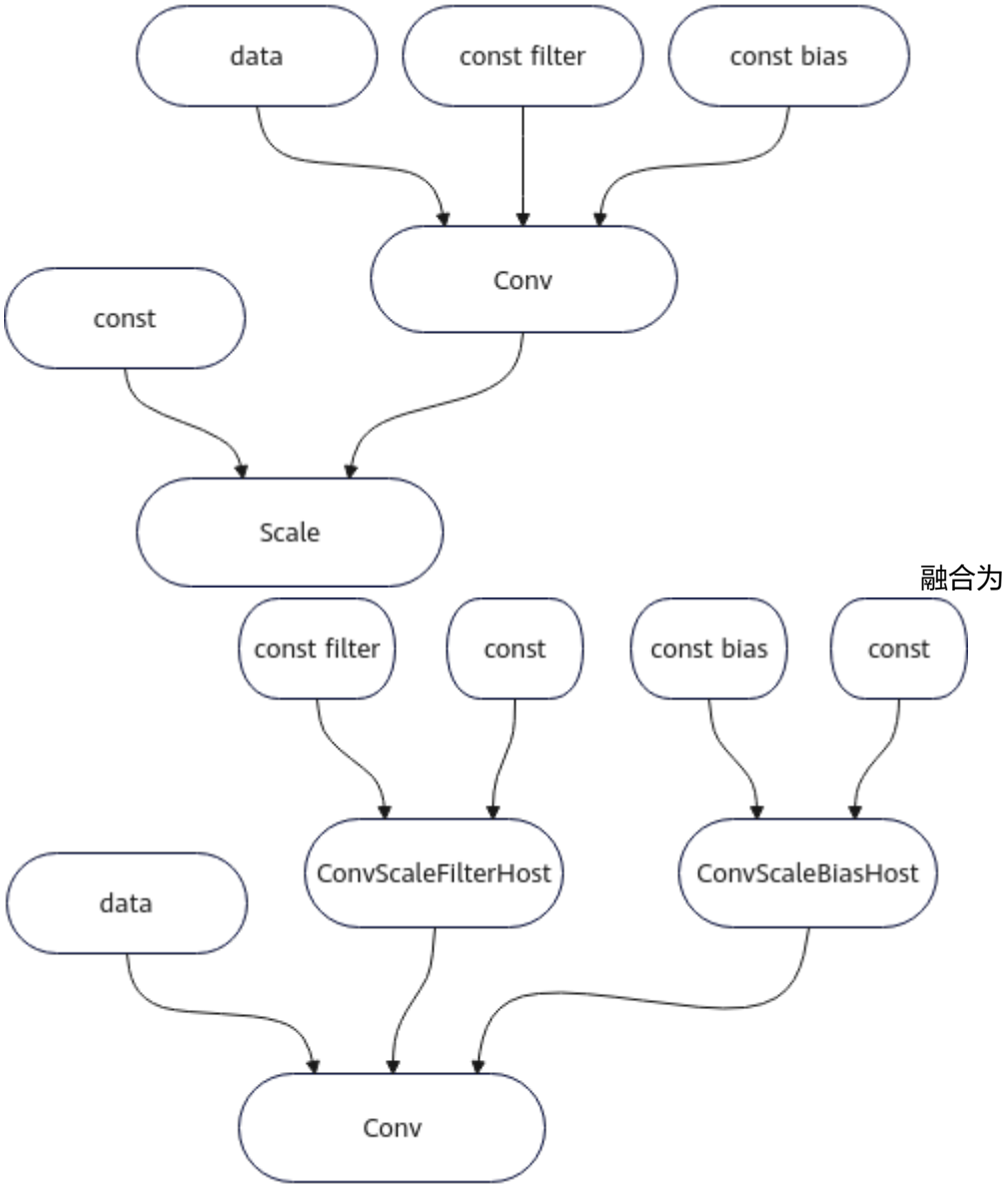
使用约束

- 仅支持静态输入shape（ fmap, filter, bias ）。
- 仅支持Conv2D groups为1。
- 每路第一个Conv2D（ 图示中conv2d_0与conv2d_2 ）的filter batch之和需为16的倍数，如果fmap的data type为int8，filter batch之和需为32的倍数。
- filter仅支持const、RequantHostCpuOp、ConvBnFilterHost、AscendWeightQuant节点。
- 非量化场景下，仅支持最后的节点为relu或conv2d。
- 量化场景下，仅支持以dequant/requant/conv2d节点结尾。

2.15 ConvScaleFusionPass

融合模式

该融合规则将conv卷积算子和scale算子融合为conv卷积算子，filter输入替换为ConvScaleFilterHost算子，bias输入替换为ConvScaleBiasHost算子，提高计算性能：



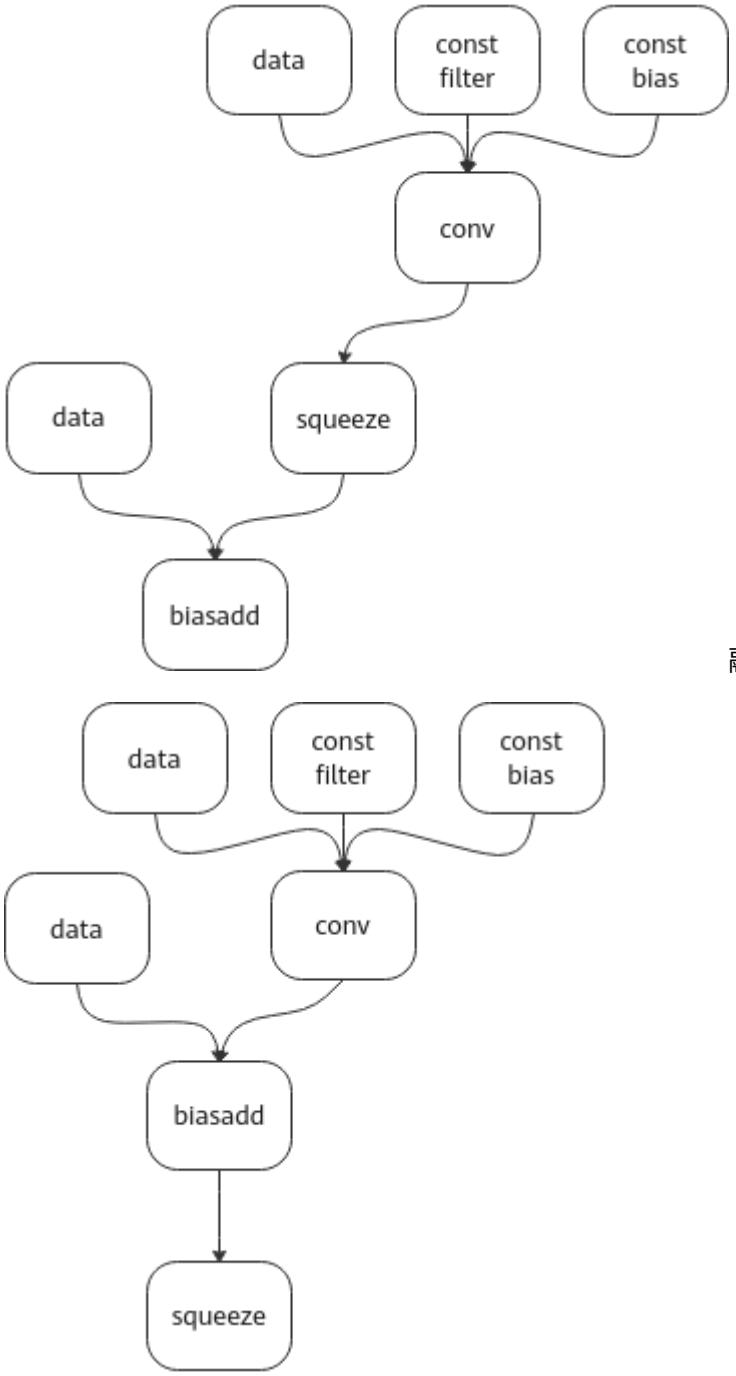
使用约束

- 不支持conv卷积算子的filter输入为QuantWeightRollBack。
- 不支持conv卷积算子多输出场景。
- filter，bias和scale节点的另一路输入必须为const。

2.16 Conv2DSqueezeBiasaddFusionPass

融合模式

该融合将Conv2d+squeeze+biasadd转换成Conv2d+biasadd+squeeze的结构。



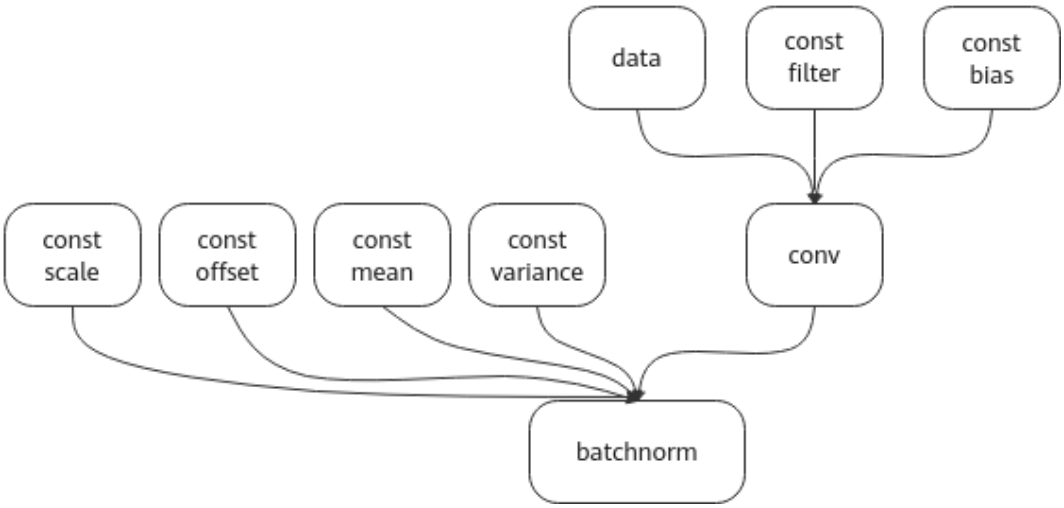
使用约束

- biasadd节点的另一路输入data的维度必须是1，否则报错。
- biasadd节点的另一路输入data，如果来自Variable节点，则不融合。

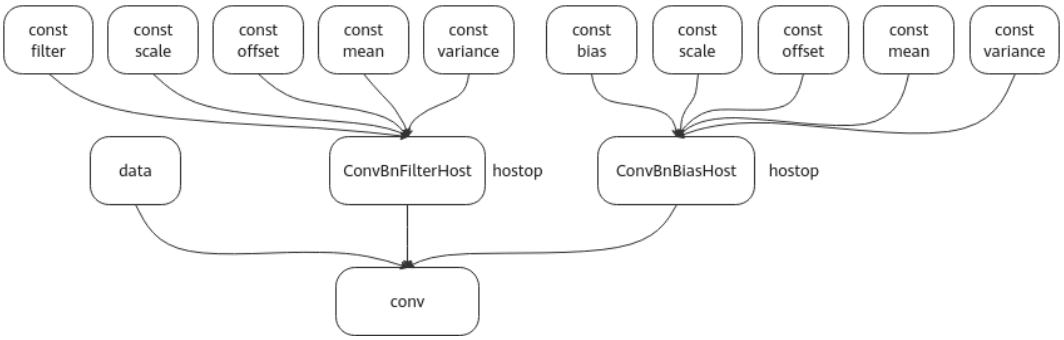
2.17 ConvBatchnormFusionPass

融合模式

该融合将Conv2d+batchnorm或者Conv3d+batchnorm融合为1个融合算子；当batchnorm的输入为2输入时，也可以将DepthwiseConv2d+batchnorm融合为1个融合算子。



融合为



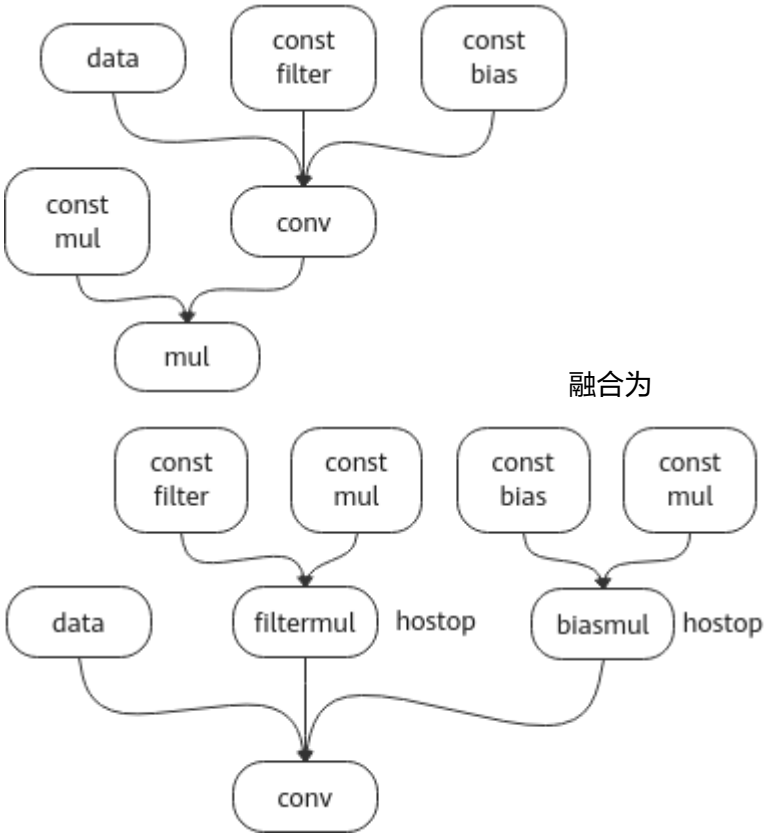
使用约束

- conv节点可以是Conv2D，也可以是 Conv3D。当batchnorm节点只有2个输入（ mean，variance ）时，conv节点也可以是DepthwiseConv2D。
- batchnorm节点可以是batchnorm，也可以是BNInference。
- filter、bias必须是const，否则不融合；如果filter是QuantWeightRollBack，不融合。
- batchnorm节点的输入，可以是2个输入（ mean，variance ），也可以是上述4个输入，所有输入必须是const，否则不融合。
- data输入为动态时，支持融合。

2.18 AConv2dMulFusion

融合模式

该融合将Conv2d+mul或Conv3d+mul融合为1个融合算子Conv。



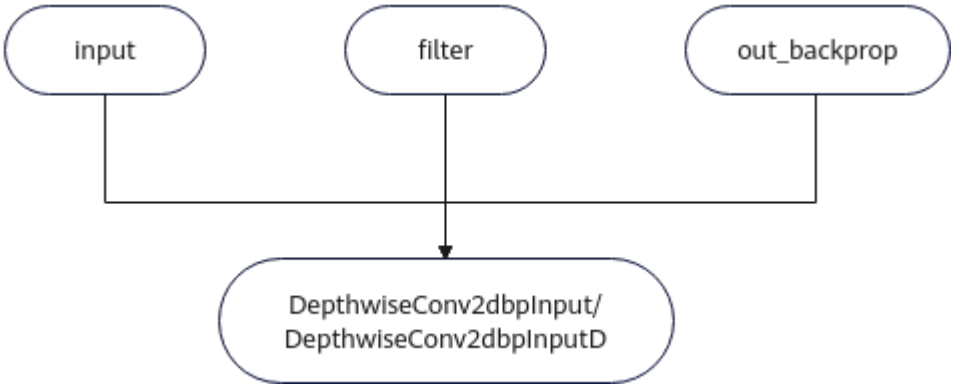
使用约束

- conv节点可以是Conv2D，也可以是 Conv3D。
- data输入为动态时，支持融合。
- filter、bias和mul的另一路，三个输入均为const时，支持融合，如果是动态输入，则不支持。

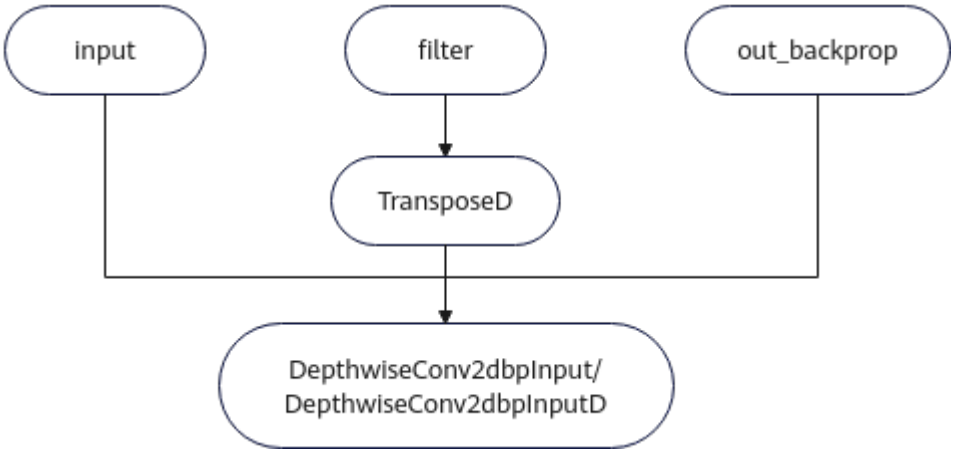
2.19 DepthwiseDfFusionPass

融合模式

该融合规则为DepthwiseConv2DbackpropInput卷积算子添加TransposeD算子，提高计算性能。



融合为



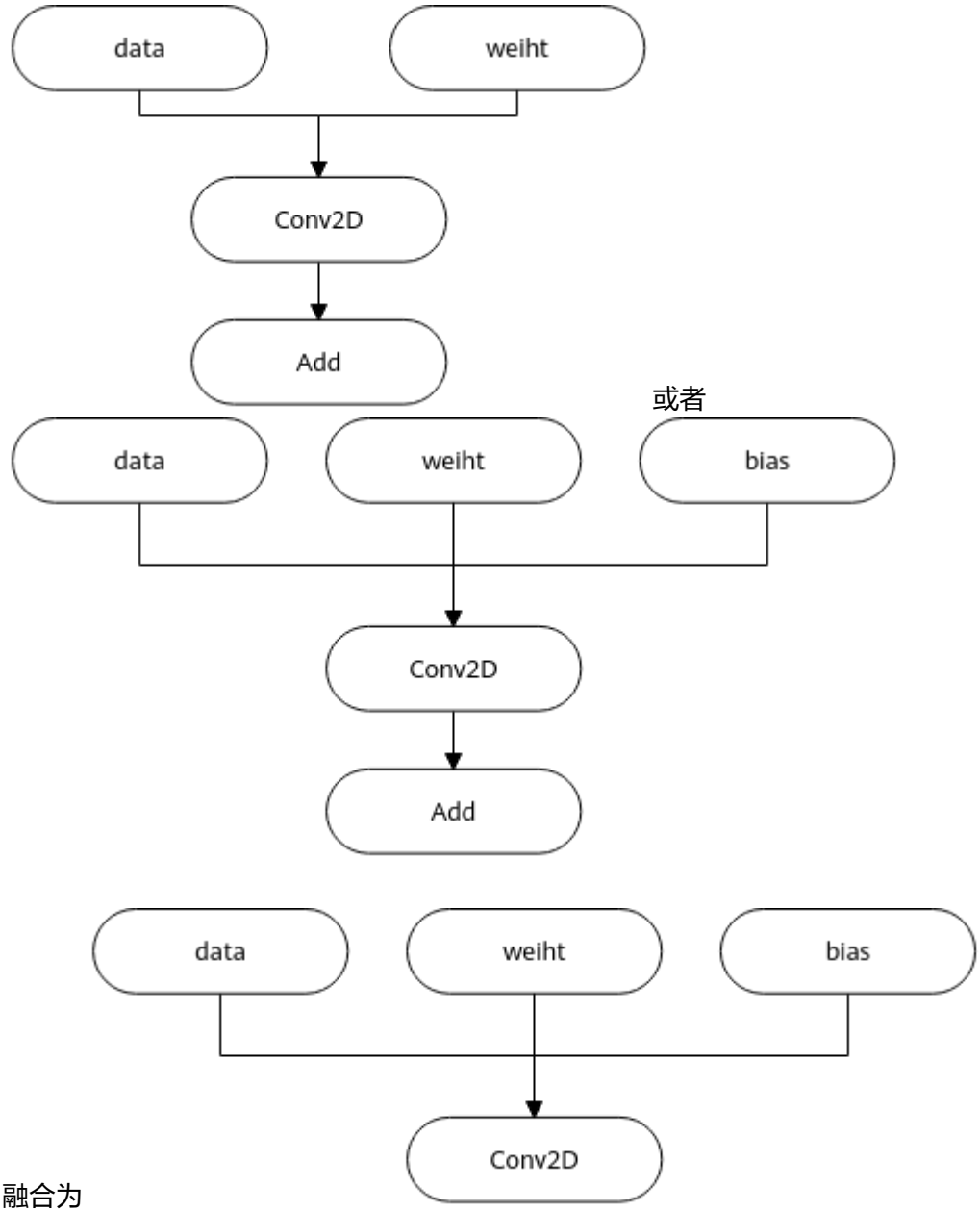
使用约束

该融合只在filter为NCHW时生效。

2.20 TBEConvAddFusion

融合模式

该融合规则将Conv + Add 两个算子融合为一个 Conv 算子，提高计算性能：



融合为

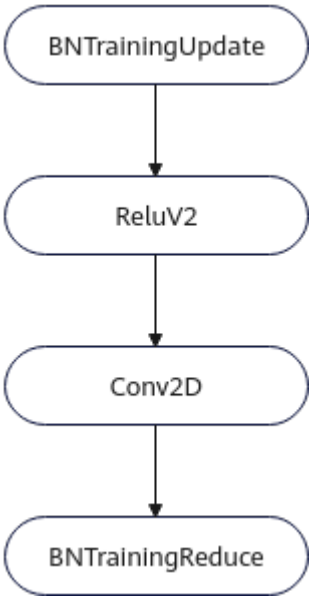
使用约束

- Conv 只能单输出。
- 如果是Conv3d的话 Add 只能单输出。
- Conv 的 Bias 必须是常量。
- Add 权重大小必须为1。
- Add 的另一输入必须为常量。
- Add 的另一输入大小不能小于或者等于0。

2.21 ZBNupdateReluV2Conv2DBNreducePass

融合模式

该融合规则将BNTrainingUpdate+ReluV2+Conv2D+BNTrainingReduce四个算子融合成一个Conv2D算子，提高性能。



使用约束

- 只支持昇腾910 AI处理器系列芯片。
- 不支持Conv2D节点带bias场景。

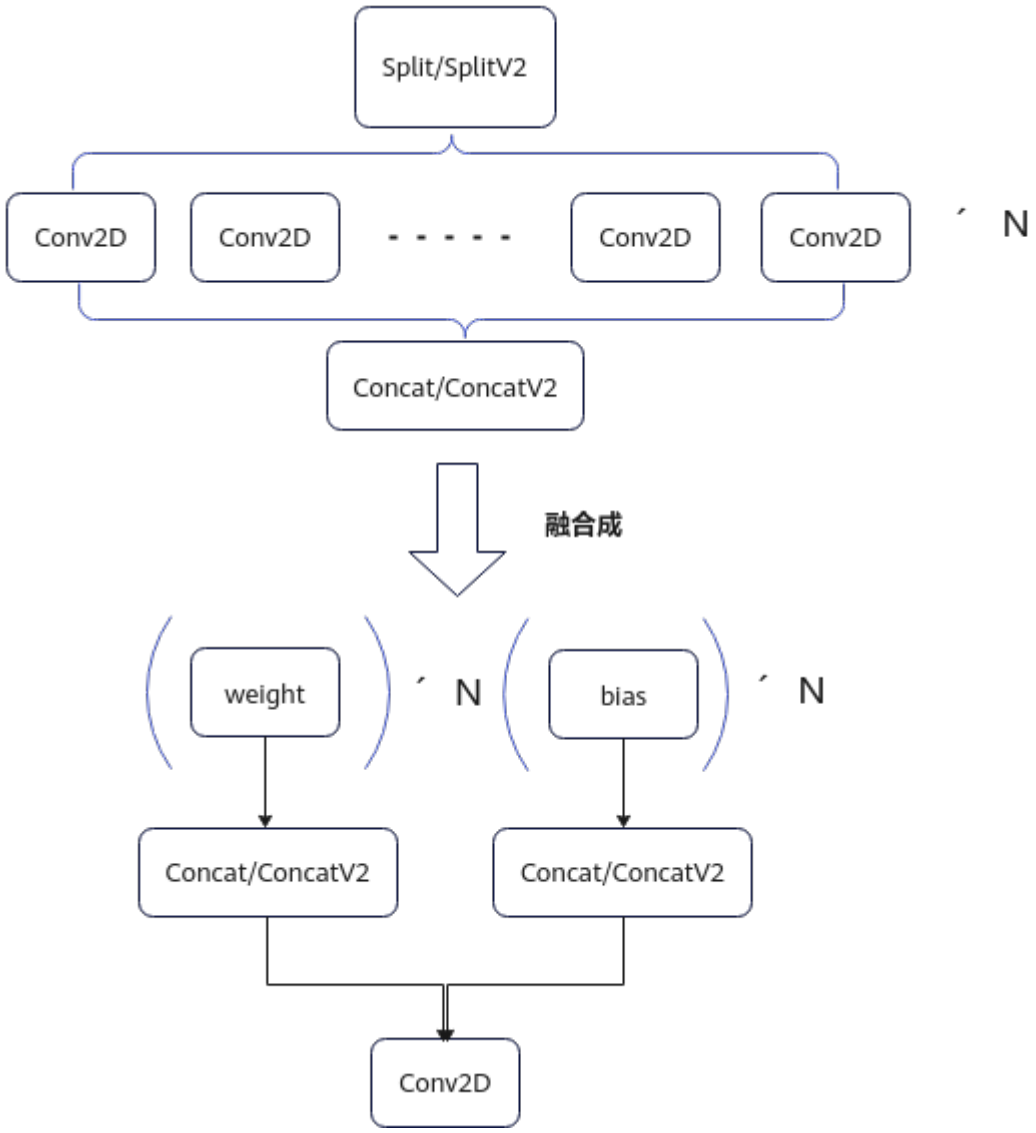
支持的芯片型号

昇腾910 AI处理器

2.22 ASplitConv2dConcatPass

融合模式

将Split/SplitV2 + Conv2D *N + Concat/ConcatV2 融合成一个组卷积，简化图结构。



使用约束

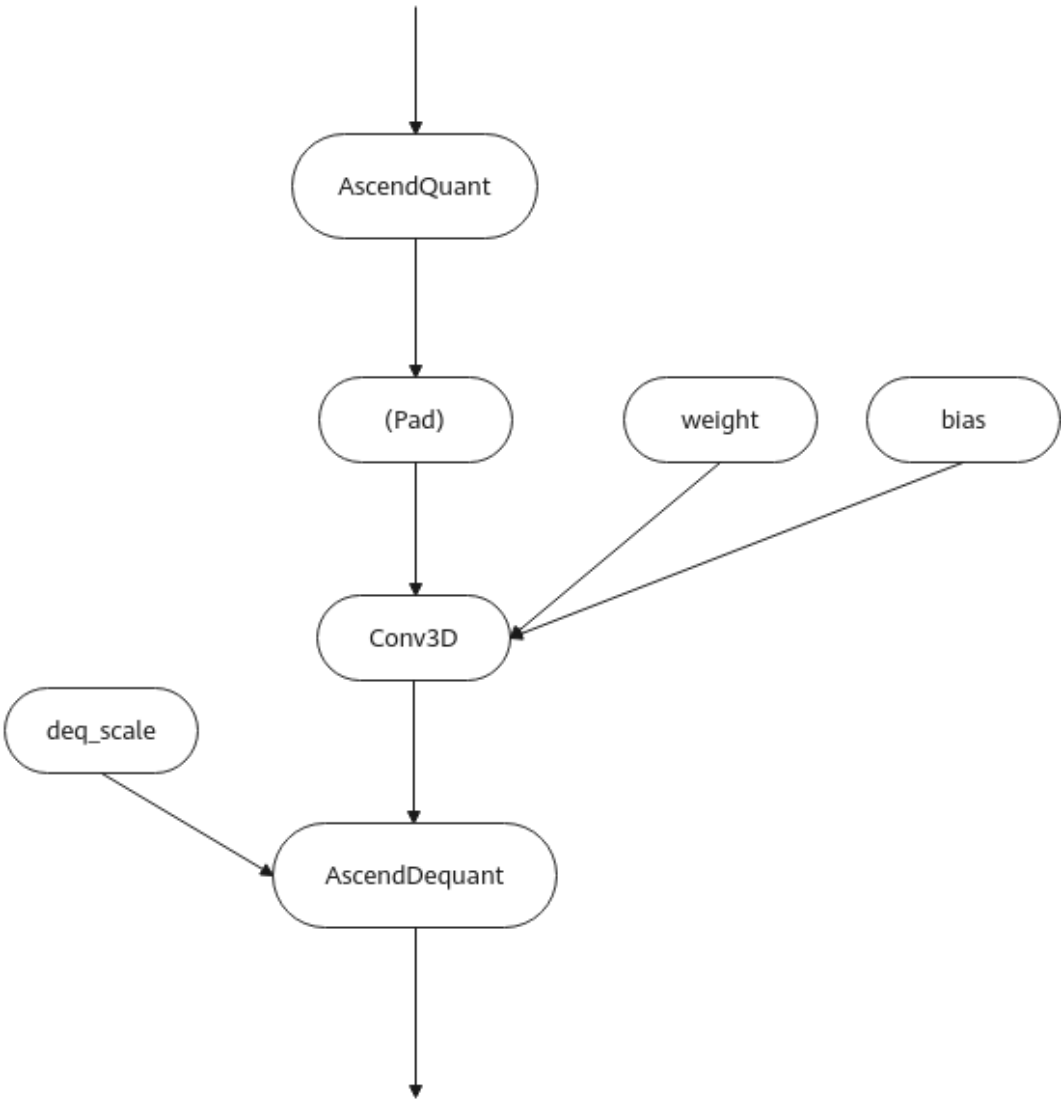
- 每个Conv2D的输入个数必须相同，且大于等于2个。
- Conv2D输入的类型只支持：float、float16、int8和int32。
- 不支持Conv2D动态输入。
- 每个Conv2D的filter的shape和format要相同（只支持NWCN 或 NCHW）。
- 每个Conv2D只能单输出。
- Conv2D的filter和bias必须是类型{"Const", "Constant", "QuantBiasOptimization", "QuantWeightRollBack", "QuantBiasRollBack", "AscendWeightQuant"}中的一个。
- Split/SplitV2节点和Concat/ConcatV2节点的另一路输入必须是const类型，且Split/SplitV2节点的输出个数和Concat/ConcatV2的输入个数必须相同。
- Split/SplitV2节点的切分轴和Concat/ConcatV2节点的组合轴必须都为channel方向。

2.23 Conv3DQuantProcessFusionPass

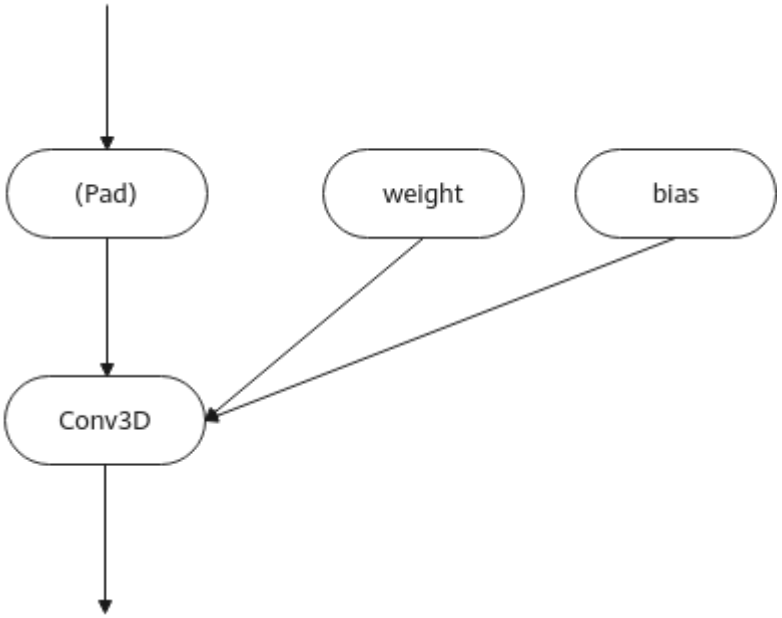
融合模式

当匹配到如下图结构时，可以进行量化回退或者bias优化。

- 量化回退，则会插入常量折叠算子，用来把AscendQuant和AscendDequant算子消除。具体请参见下图。
- bias优化，则不涉及图结构修改。



量化回退场景下融合成



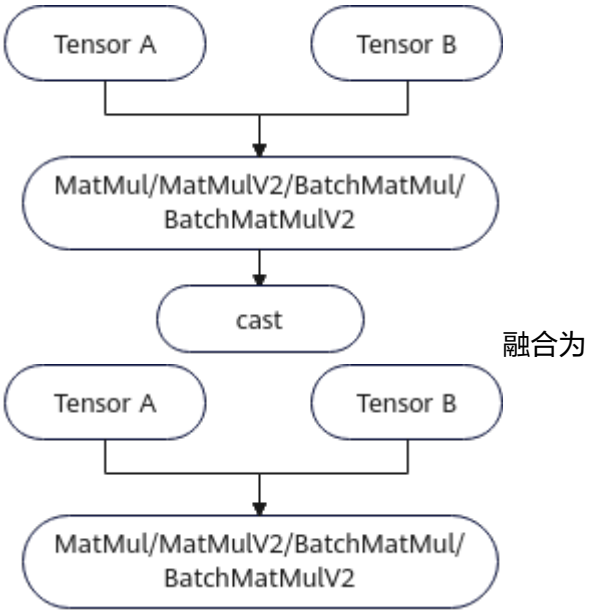
使用约束

无

2.24 MatmulCastFusionPass

融合模式

将matmul/matmulv2/BatchMatMul/BatchMatMulV2算子和cast算子融合为matmul/matmulv2/BatchMatMul/BatchMatMulV2算子。



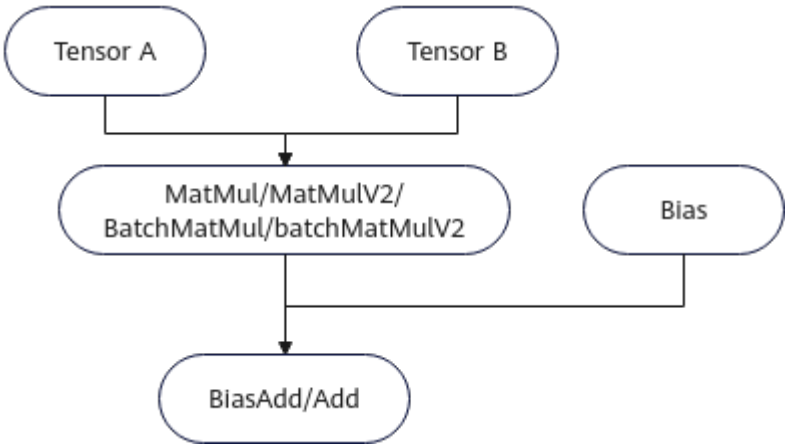
使用约束

当matmul的输入数据类型为float16,cast输出数据类型为float32时，该融合生效。

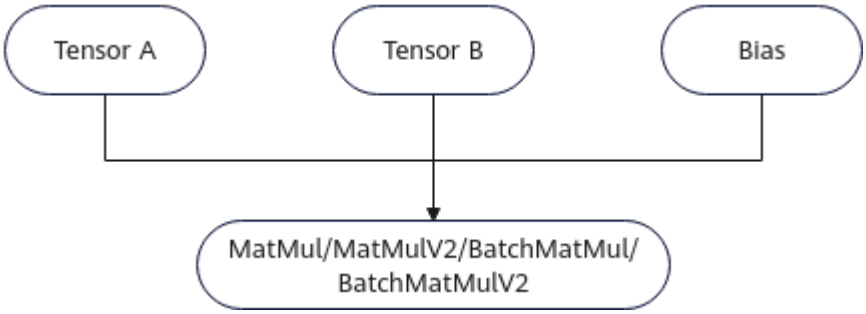
2.25 MatMulBiasAddFusionPass

融合模式

将matmul/matmulv2/batchmatmul/batchmatmulv2算子和biasadd/add算子融合为matmul/matmulv2batchmatmul/batchmatmulv2算子。



融合为



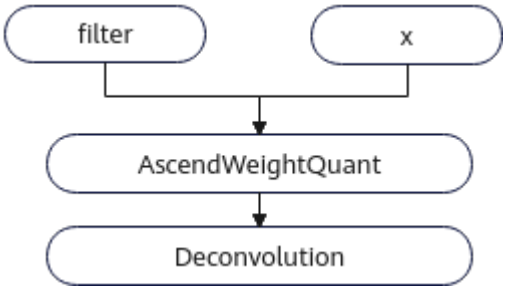
使用约束

无

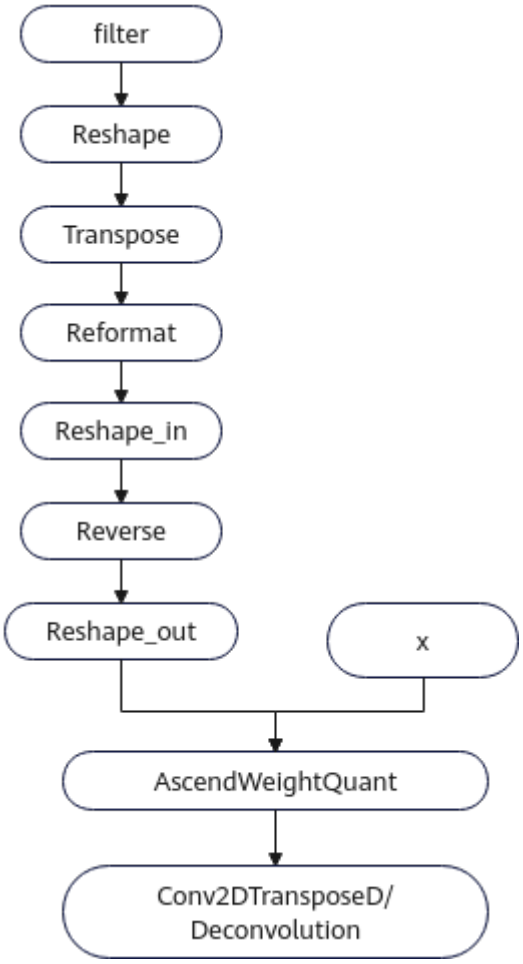
2.26 DeconvWeightTransFusionPass

融合模式

在int8量化场景下，对输入filter进行转置逆序。



融合为



- 当filter的维度不等于4时，会在filter后插入complement_dimension节点。
- 当filter的shape中H和W维度都不为1时，会在Reformat后依次插入reshape_in,reverse和reshape_out节点。
- 所有插入节点都会进行常量折叠。

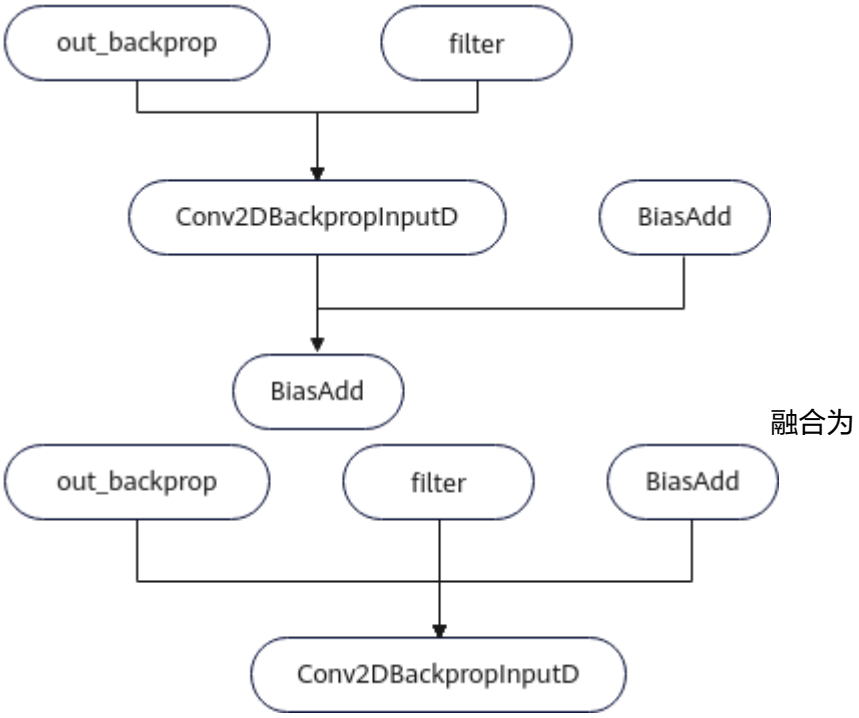
使用约束

int8量化场景下，该融合生效。

2.27 Conv2DbpInputBiasAddFusionPass

融合模式

将conv2dbackpropinput算子和biasadd算子融合为conv2dbackpropinputD算子。



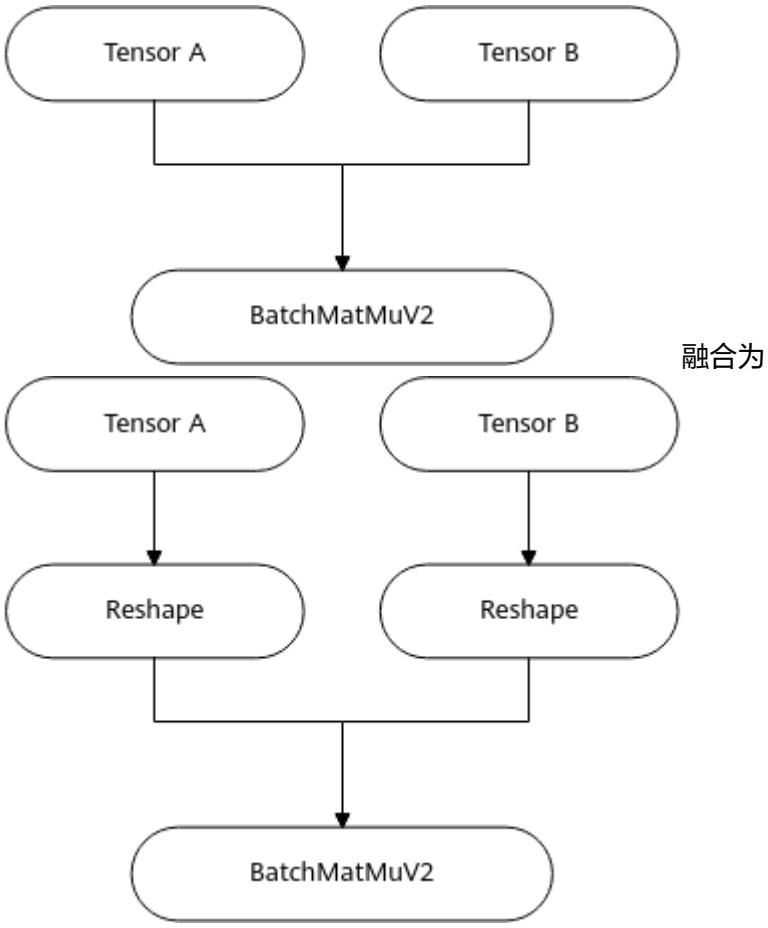
使用约束

无

2.28 BatchMatMulV2ReshapeFusionPass

融合模式

针对a或b的shape为1维场景，使用reshape将输入重置为2维。



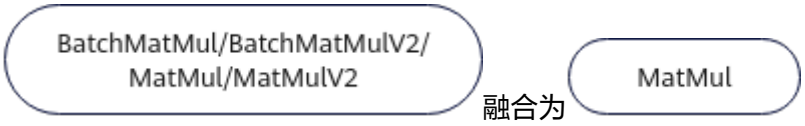
使用约束

输入tensor A或tensor B的shape为1维时，图融合生效。

2.29 BatchMatmulFusionPass

融合模式

该融合规则将BatchMatMul/BatchMatMulV2/MatMul/MatMulV2算子融合修改为MatMul算子，提高计算性能。



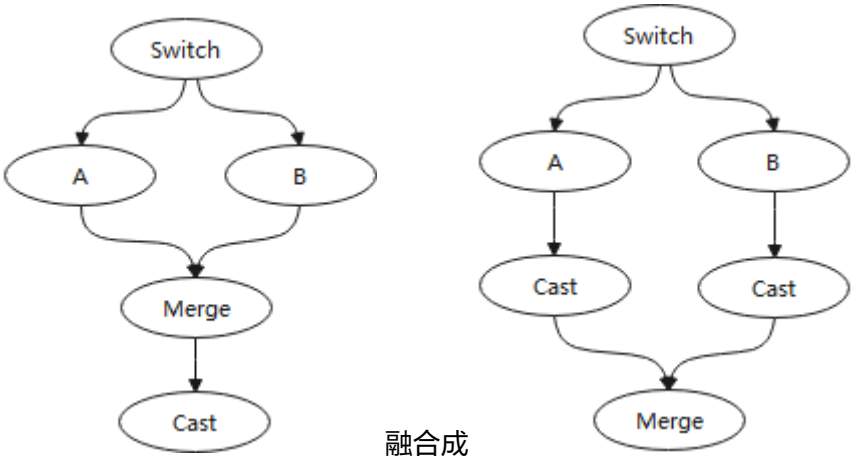
使用约束

当BatchMatMul算子的两个输入shape维度都不大于2时，该融合生效。

2.30 SwapMergeCastFusionPass

融合模式

该融合调节merge与cast节点间顺序：



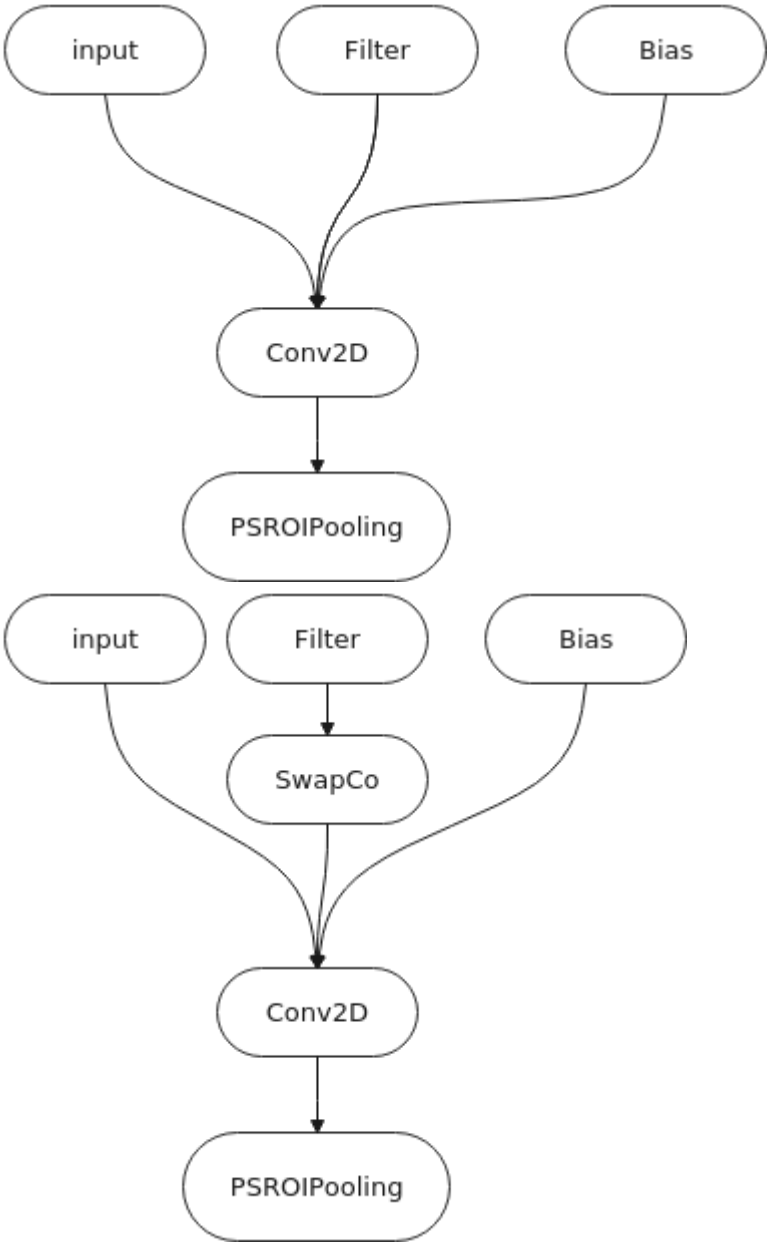
使用约束

无

2.31 PSROIPoolingFusionPass

融合模式

当PSROIPooling前驱是Conv2D时，在Conv2D的Filter和Bias输出插入SwapCo算子；如果PSROIPooling前驱不是Conv2D，则在PSROIPooling输入插入SwapCi算子。



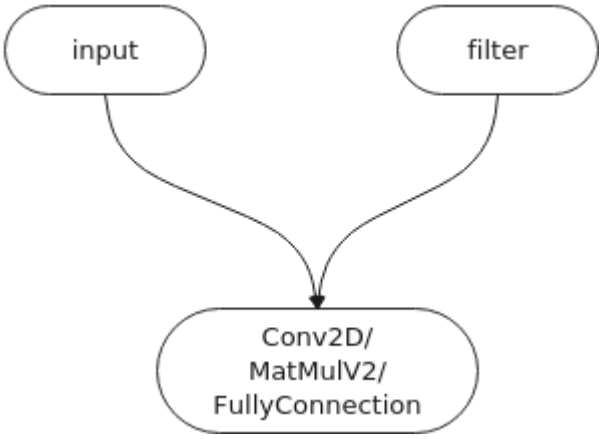
使用约束

无

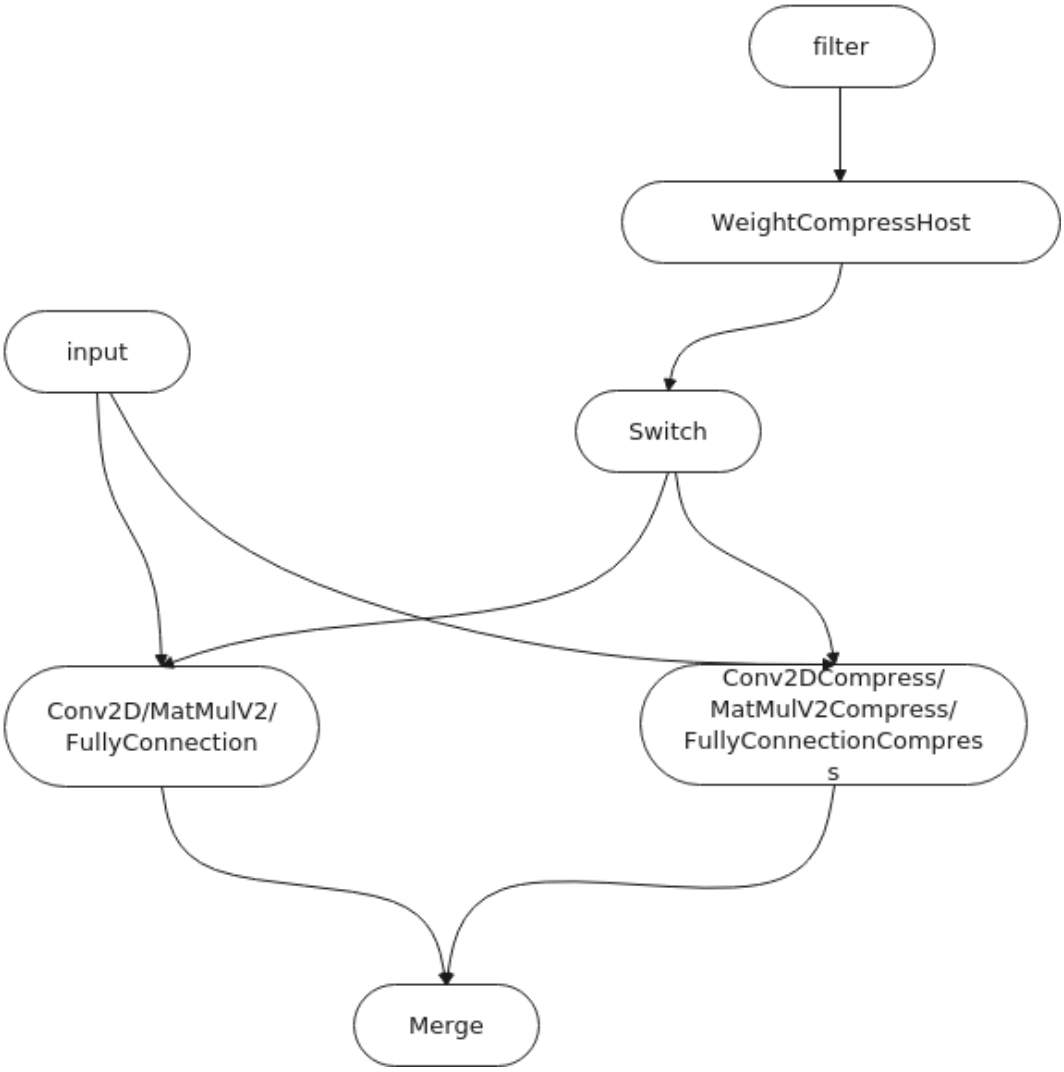
2.32 ConvWeightCompressFusionPass

融合模式

对于Cube类运算算子，将Filter通过插入压缩算子进行压缩，对应Cube算子支持Conv2D、FullyConnection、MatMulV2算子。



融合成



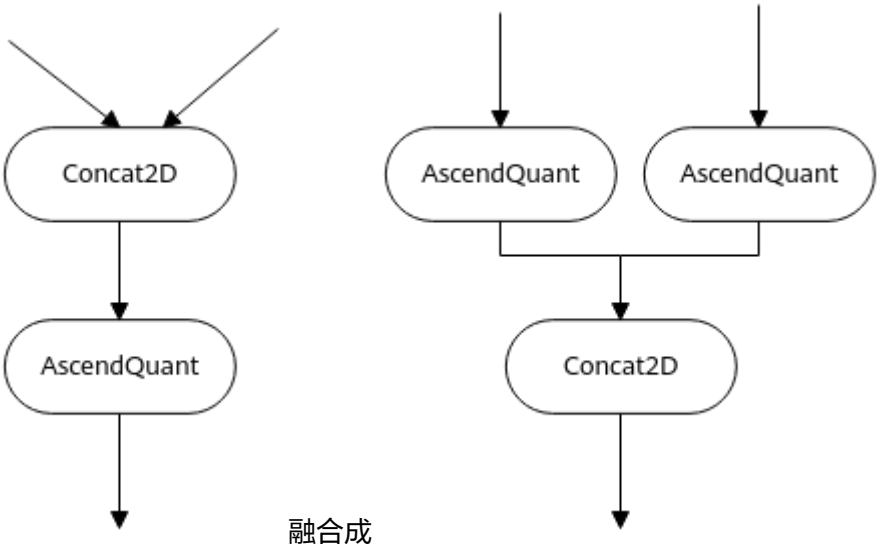
使用约束

无

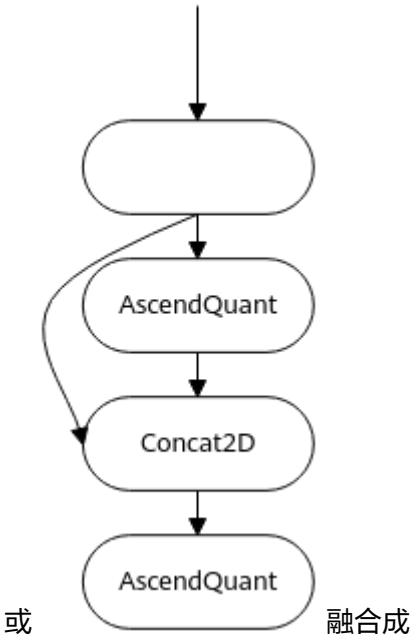
2.33 ConcatQuantFusionPass

融合模式

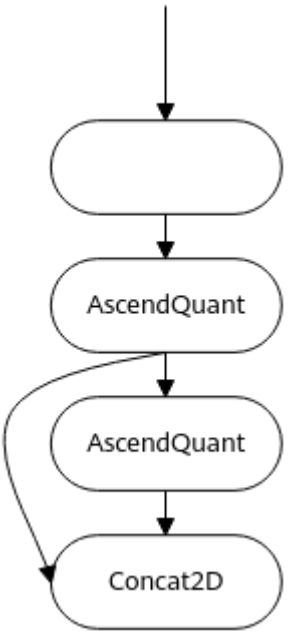
该融合规则将ConcatV2D+AscendQuant子图融合成AscendQuant+ConcatV2D子图模式。



融合成



或



融合成

使用约束

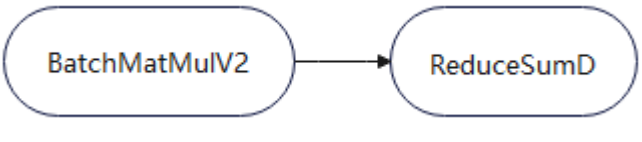
在数据比对时需要关闭对应融合规则。

2.34 BatchMatmulV2ReduceFusionPass

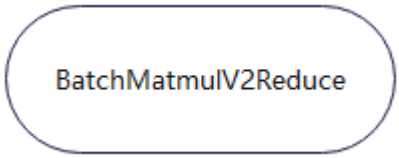
融合模式

该融合规则将BatchMatmulV2+Reduce算子融合为BatchMatmulV2Reduce算子。

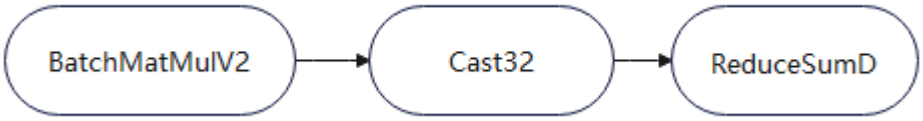
模式一：



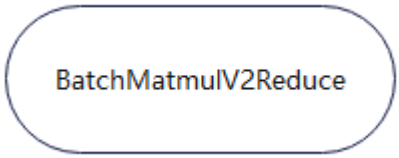
融合成



模式二：



融合成



使用约束

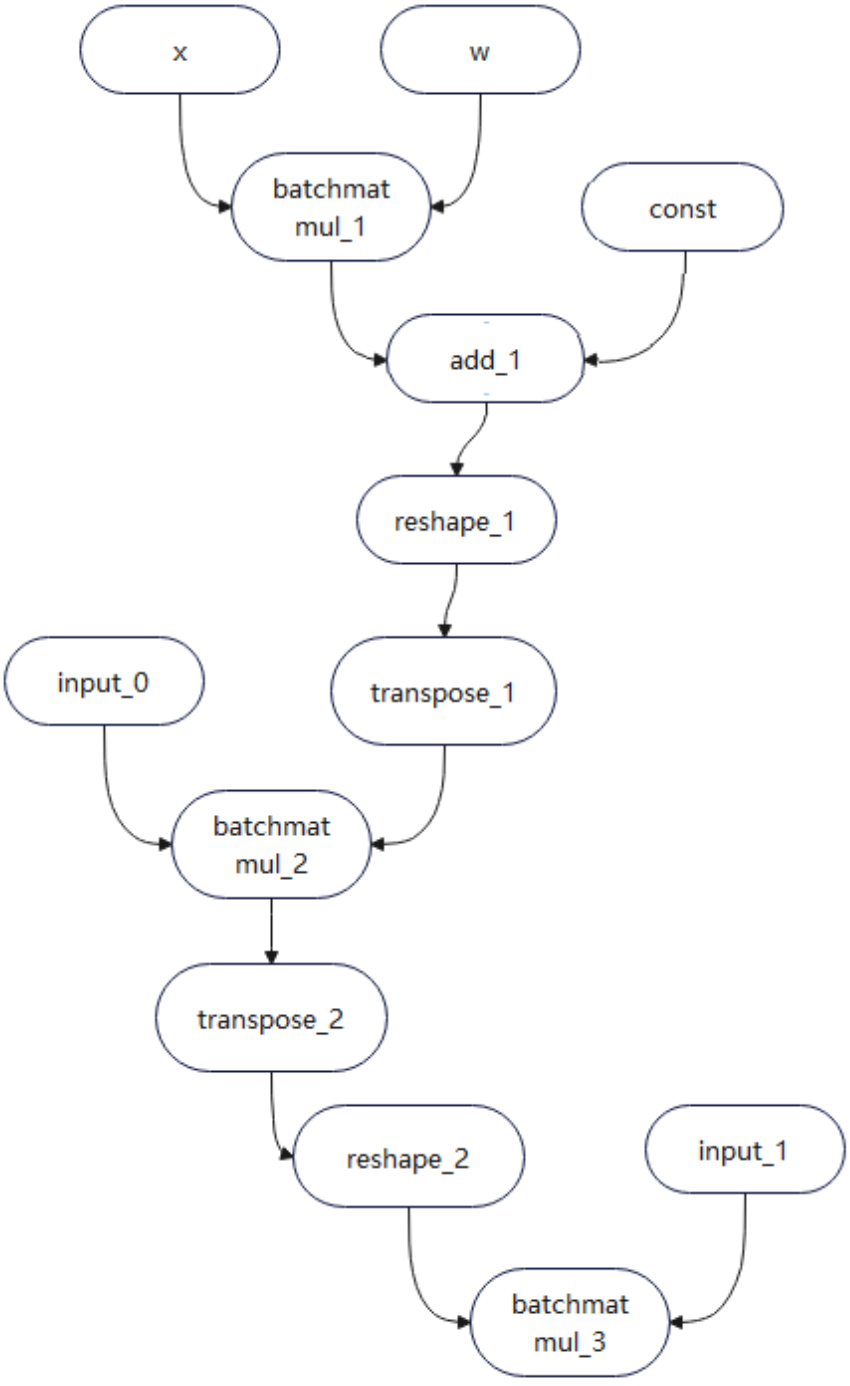
- BatchMatmul后节点若为Cast32，输出数据类型为fp16，Cast输出数据类型为fp32。
- BatchMatmul输出节点数只能为1。

2.35 BatchMatmulNonAlignedFusionPass

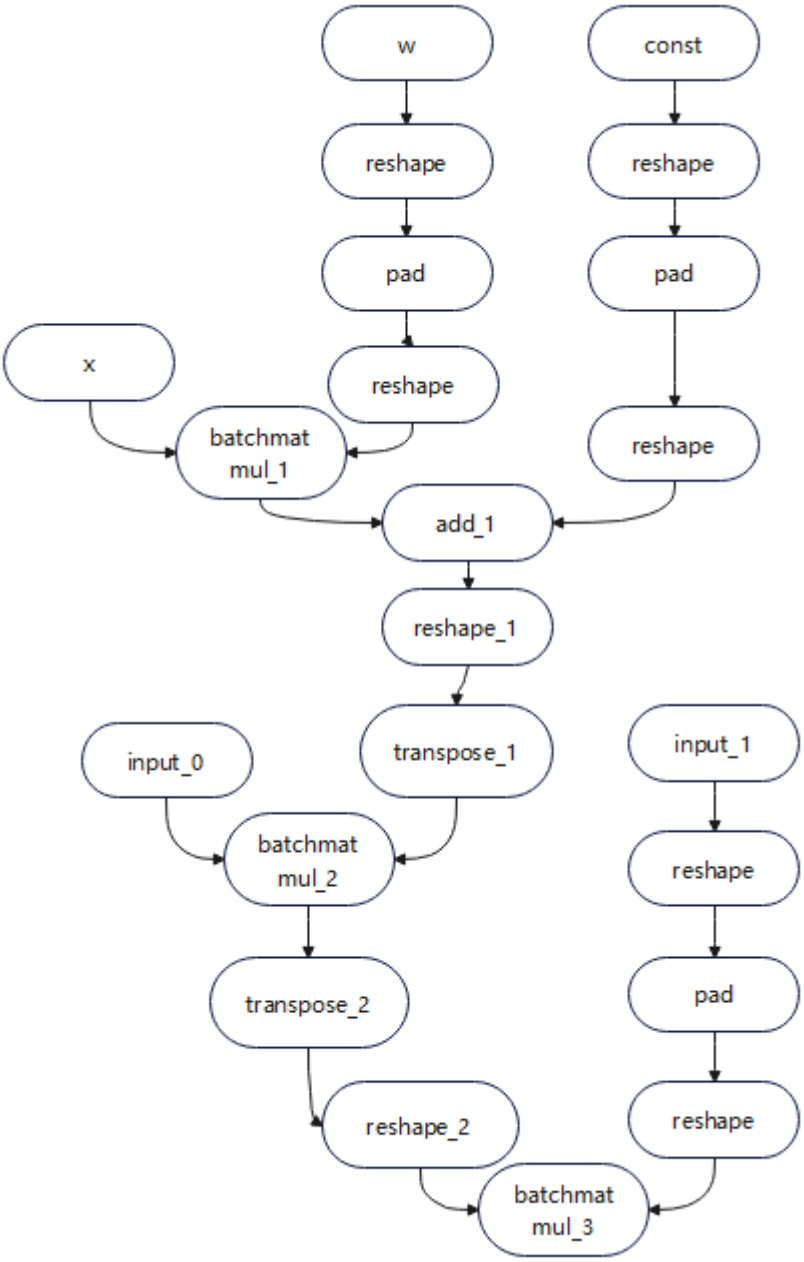
融合模式

该融合为batchmatmul非对齐场景下的融合。

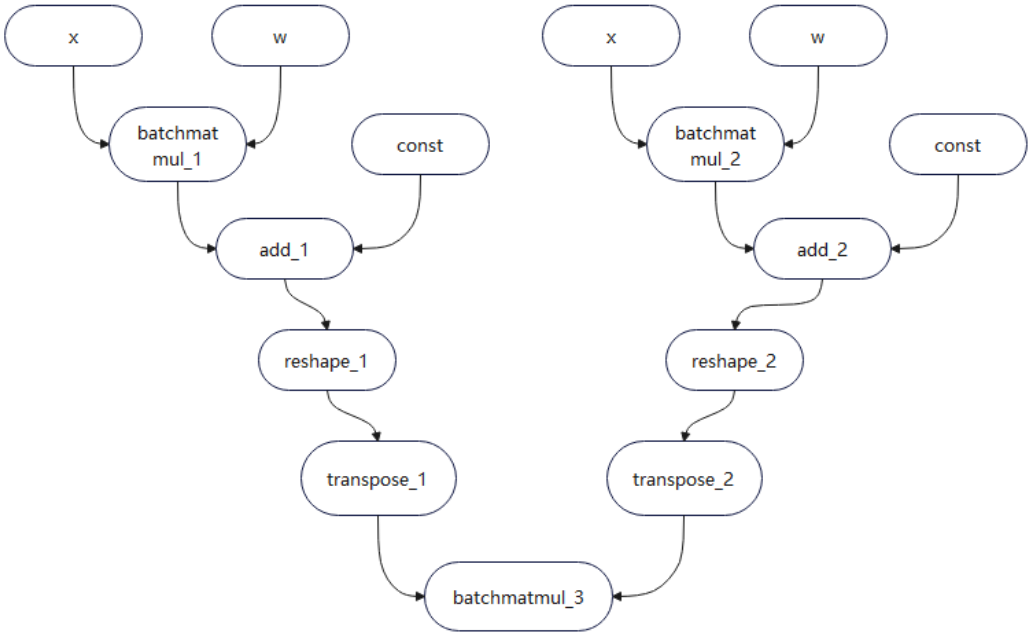
模式一：



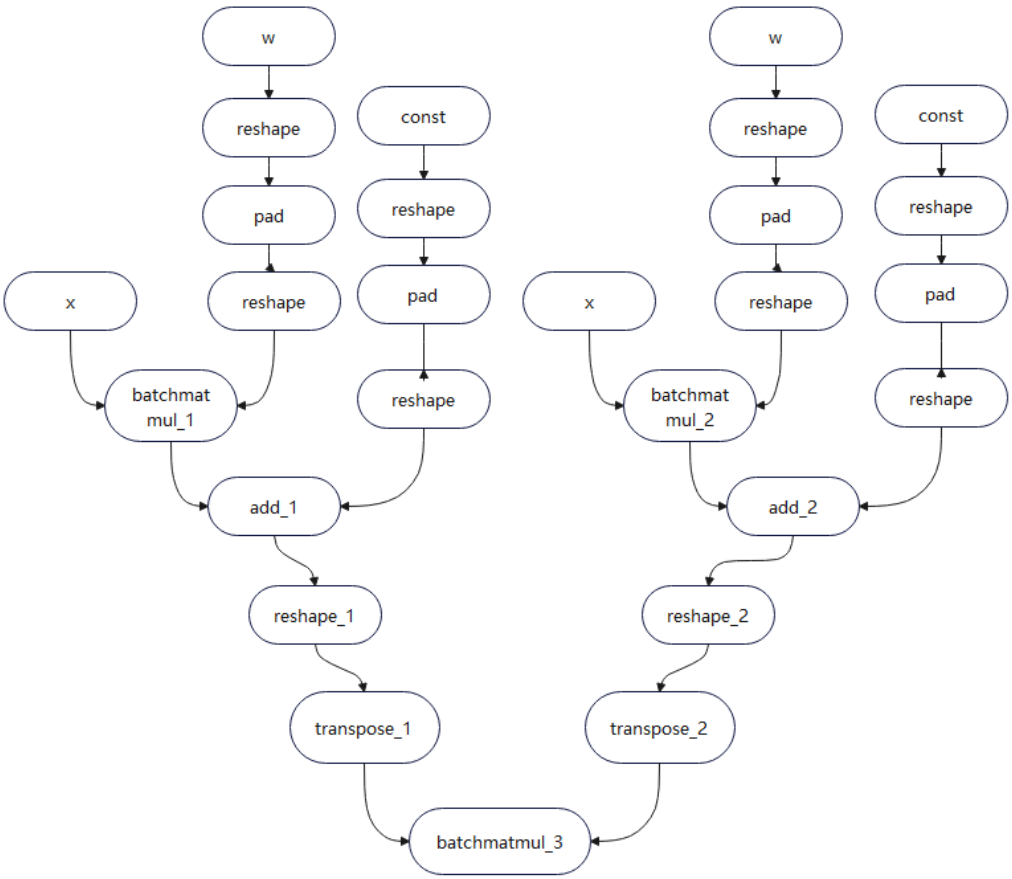
融合成



模式二：



融合成



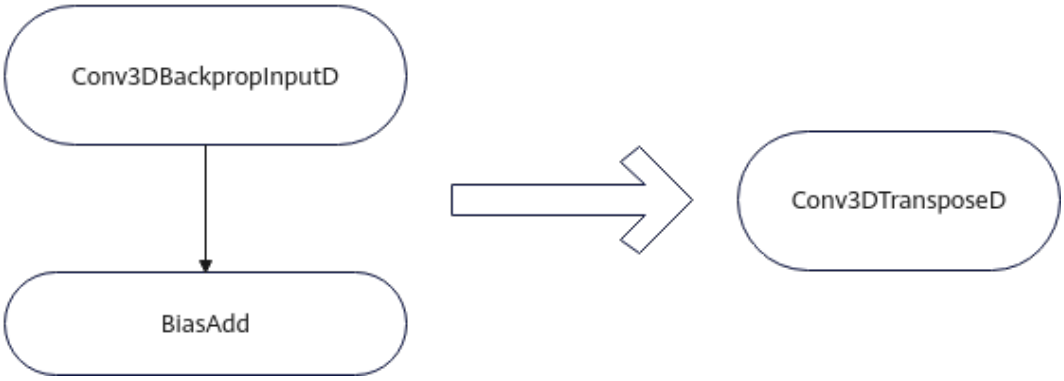
使用约束

batchmatmul输入的M需要是16的倍数，输入的K不能是16的倍数。

2.36 Conv3DbpInputBiasAddFusionPass

融合模式

该融合规则将Conv3DBackpropInputD+BiasAdd算子融合为Conv3DTransposeD算子，bias输入作为Conv3DTransposeD的输入bias。



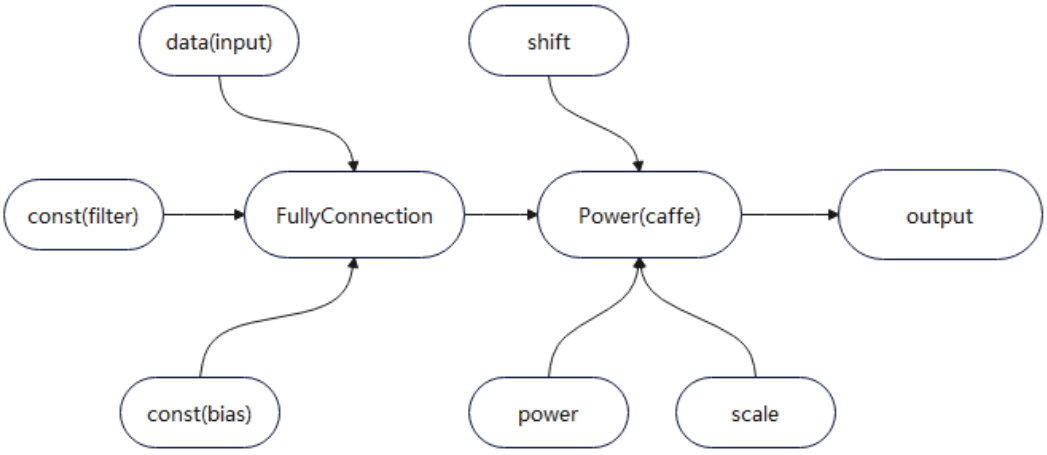
使用约束

无

2.37 FullyConnectionPowerPass

融合模式

该融合规则将FullyConnection+Power算子融合为 FullyConnectionPower算子。



融合成

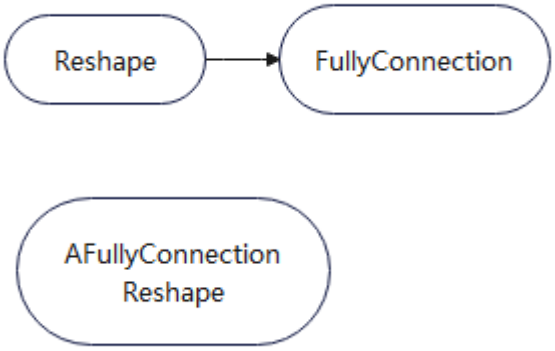
使用约束

- FC的weight不能少于2个。
- FC的输入数据类型需要为fp16。

2.38 AFullyConnectionReshapePass

融合模式

该融合规则将FullyConnection+Reshape算子融合为 AFullyConnectionReshape算子。



使用约束

- Reshape的输出节点不能多于1。
- FullyConnection的轴必须为1。
- Reshape第0根轴的输入输出维度必须相等且输出维度不能为0。

2.39 GemmTransFusionPass

融合模式

该融合规则将Transpose+Gemm算子融合为 GemmTrans算子。

使用约束

Transpose输入format需为ND且不能为非对齐场景。

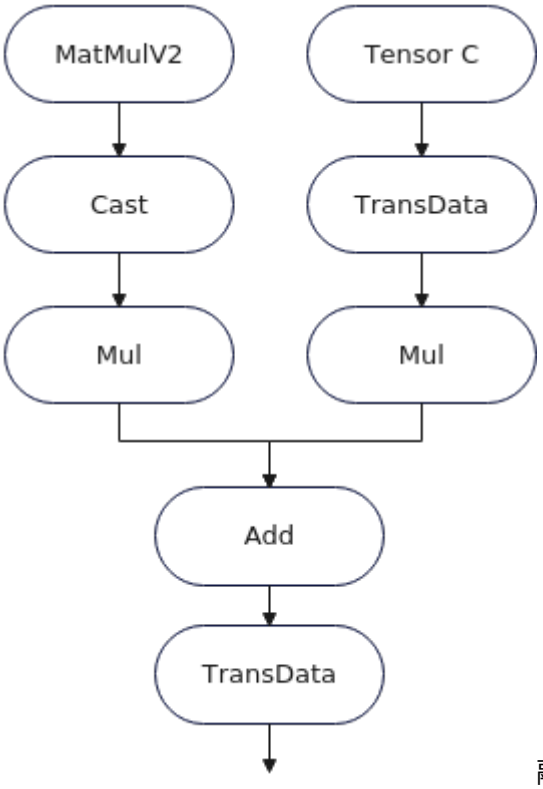
支持的芯片型号

昇腾310 AI处理器

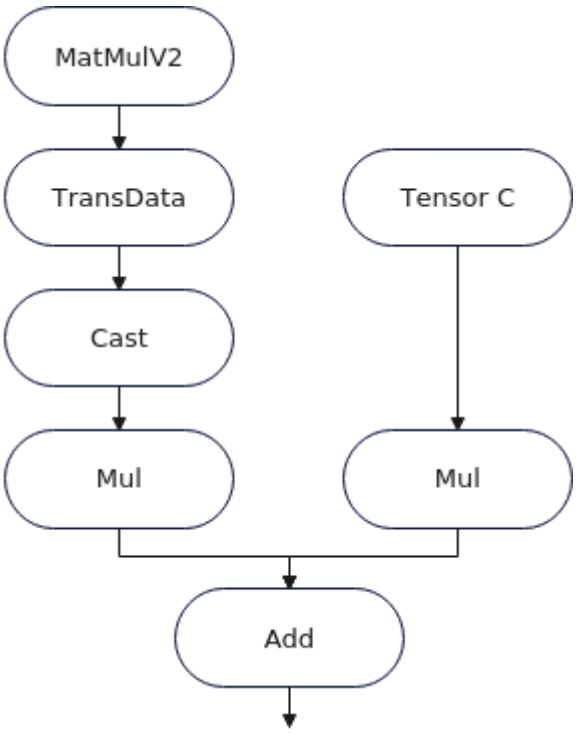
2.40 MatmulTransdataFusionPass

融合模式

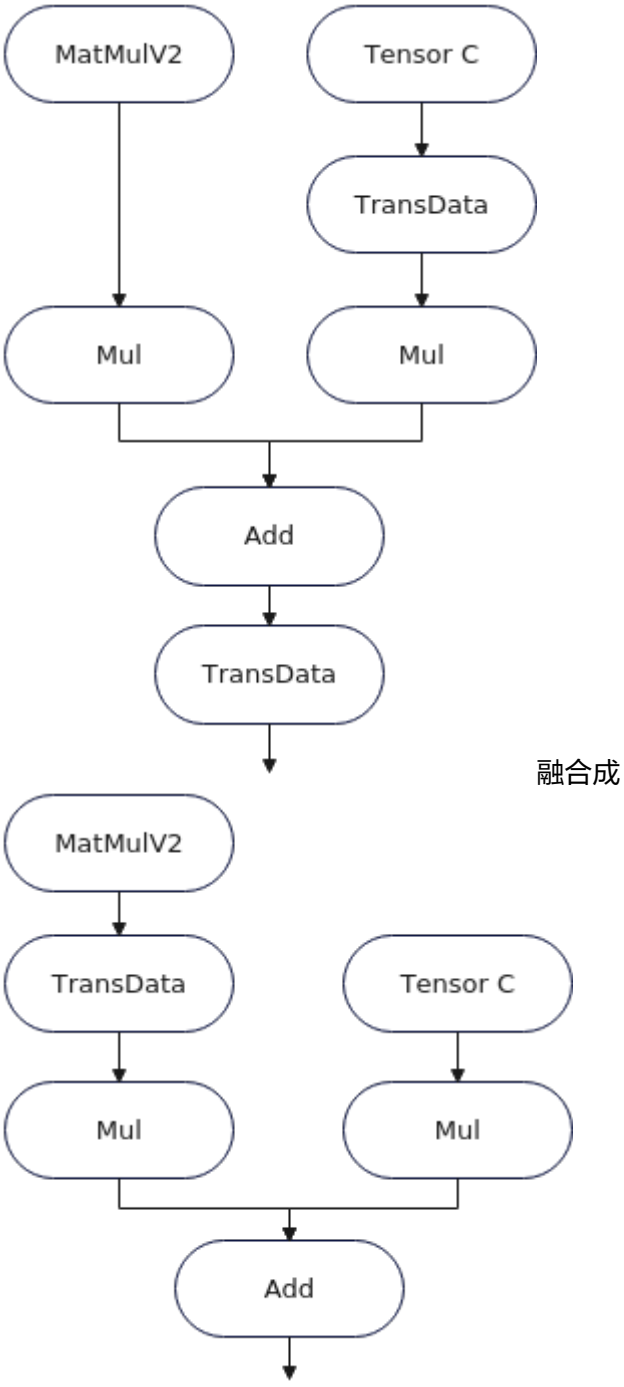
该融合规则通过等价替换将图中的Transdata算子从2个减少为1个。



融合成



或在没有Cast算子的场景



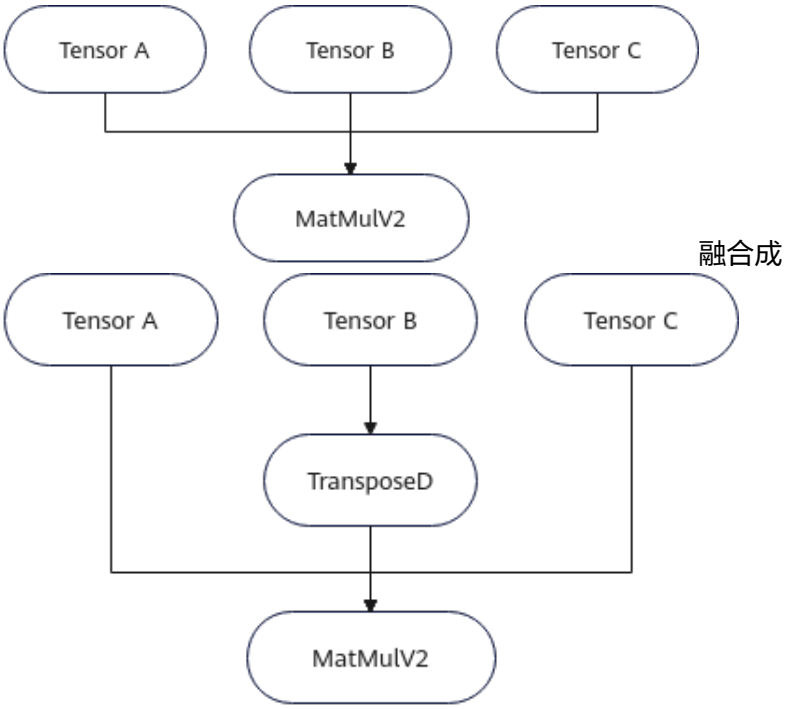
使用约束

无

2.41 Matmulv2FusionPass

融合模式

对于输入数量为3的MatMulV2算子，该融合规则在需要转置的输入TensorB之后插入TransposeD算子。



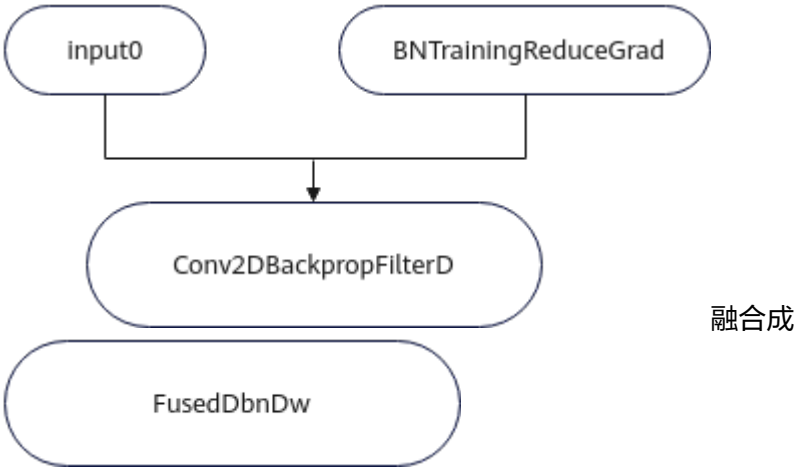
使用约束

- MatMulV2的OpDesc必须有transpose_b这个布尔值，且该布尔值为True。
- TensorB的维度数量应该为2，且数据类型为int8。

2.42 Resnet50DbnDwFusionPass

融合模式

该融合规则将Conv2DBackpropFilterD算子前添加BNTrainingReduceGrad融合成FusedDbnDw算子。



使用约束

非普通融合，仅对个别case做此融合。仅支持昇腾910 AI处理器。

支持的芯片型号

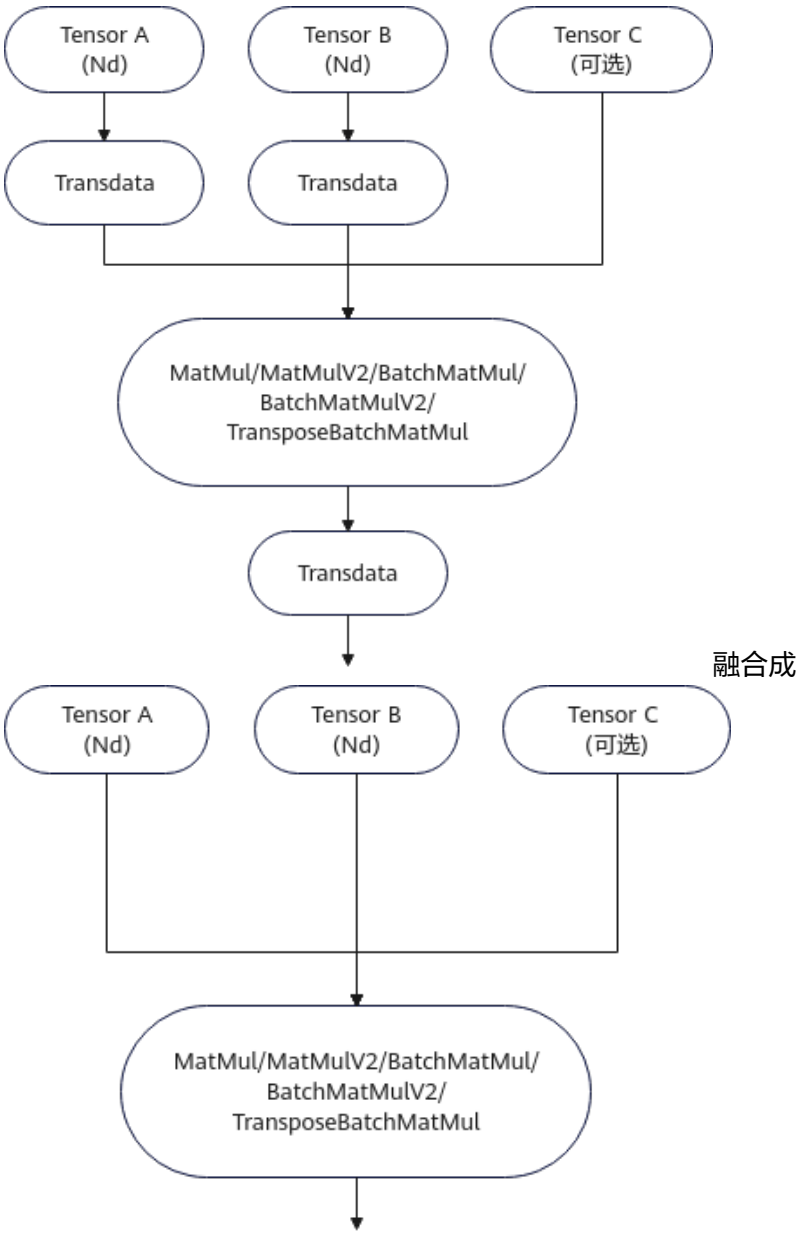
昇腾910 AI处理器

2.43 AAMatMulNzToNdFusionPass

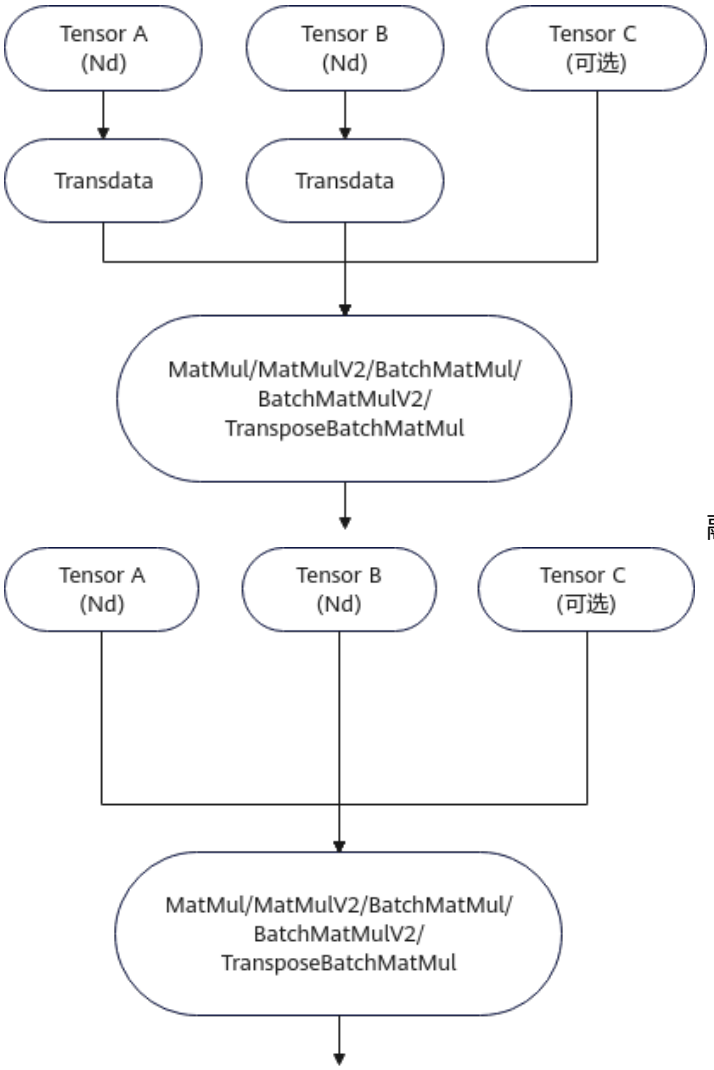
融合模式

该融合规则用于减少如下4个场景中的TransData或Cast算子。其中，TensorC节点是可选的。

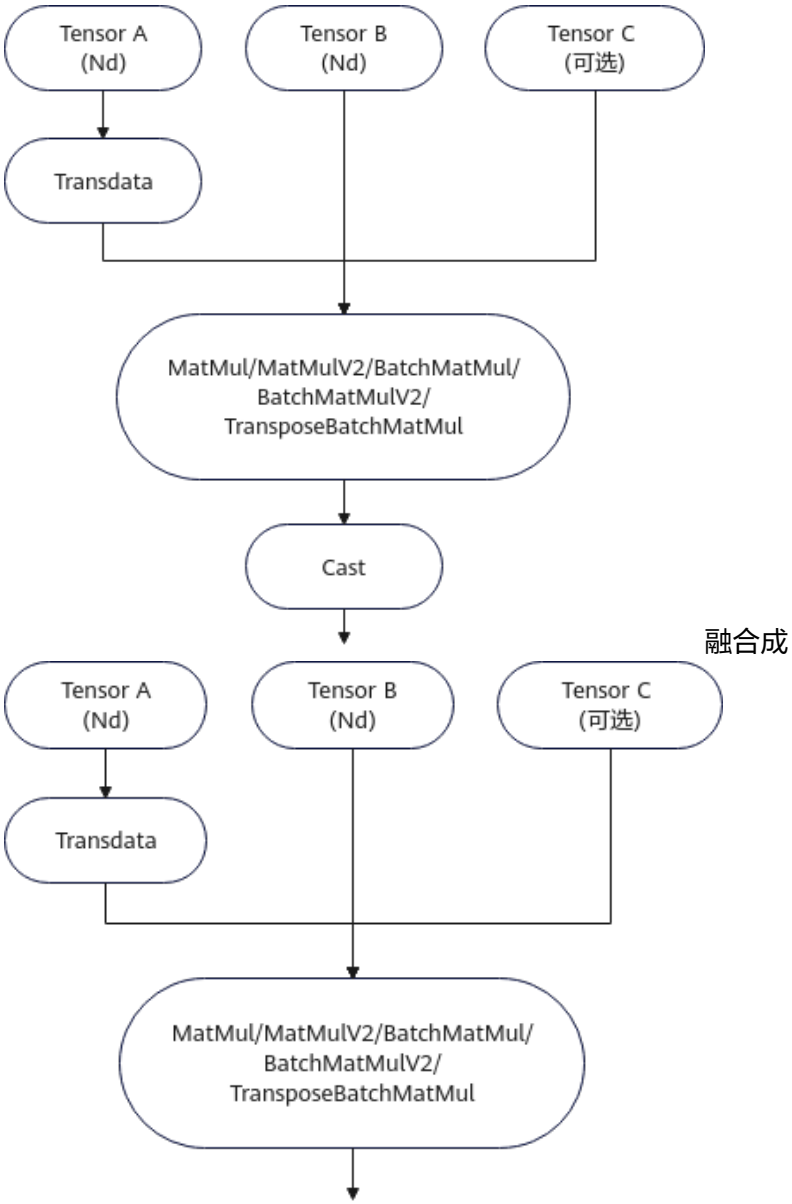
- 场景1：



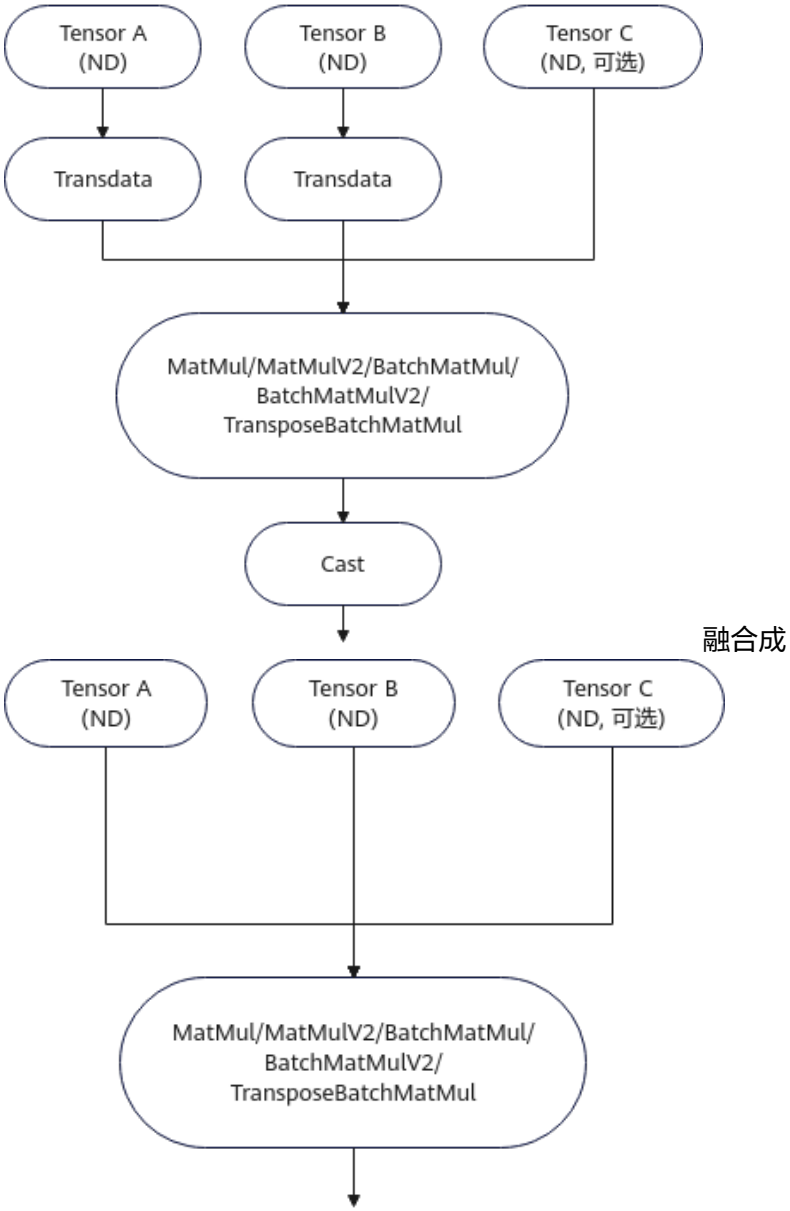
- 场景2：



- 场景3：



- 场景4：



使用约束

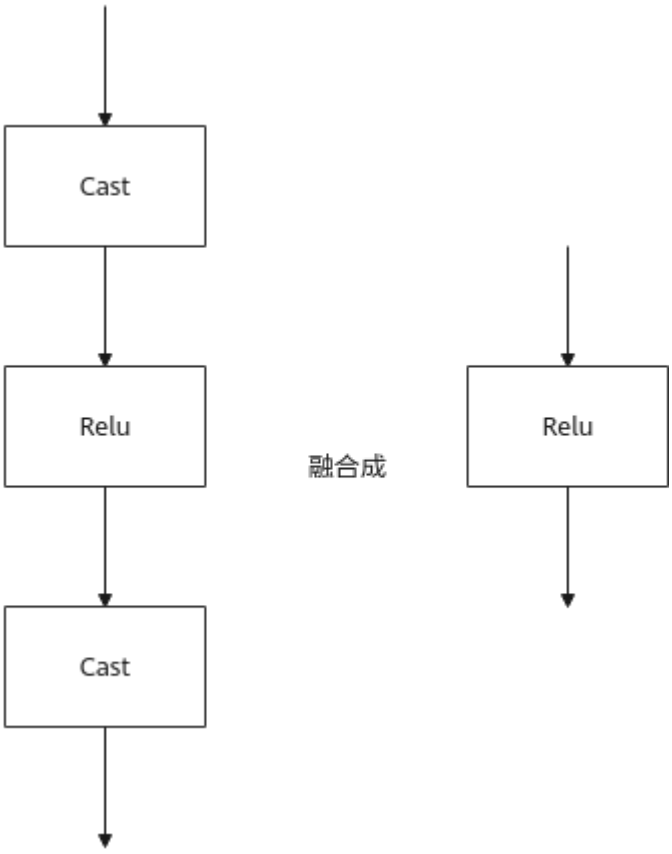
- 不支持TensorA和TensorB都为静态shape且非16对齐的场景。
- 场景1中，输出节点TransData算子的输入与输出格式应当分别为Fractal_Nz和Nd；其他场景中，MatMul算子的输出格式应为Fractal_Nz。
- 场景1、2、4中，TensorB之后的TransData算子的输入和输出数量应当都为1，并且其输入与输出格式应当分别为Nd和Fractal_Nz。
- 作为输出节点的算子，其输出数量应当为1，MatMul类算子的输出数量应当为1。
- TensorA之后的TransData算子的输入和输出数量应当都为1，并且其输入与输出格式应当分别为Nd和Fractal_Nz。
- 节点TensorA和TensorB的算子类型应为Data，输出数量应当为1。
- 输出节点的输出数据节点的算子类型应为NetOutput。
- TensorA和TensorB对应的两个MatMul类算子输入的数据类型应为float16。

- 场景2中，MatMul类算子的输出数据类型应当为float32；场景1、3、4中，MatMul类算子的输出数据类型应当为float16。
- 场景3、4中，Cast算子的输出数据类型应当为float32。

2.44 CastReluCastFusionPass

融合模式

该融合规则将Cast+Relu+Cast算子融合为 Relu算子，即消除Relu前后的Cast算子。



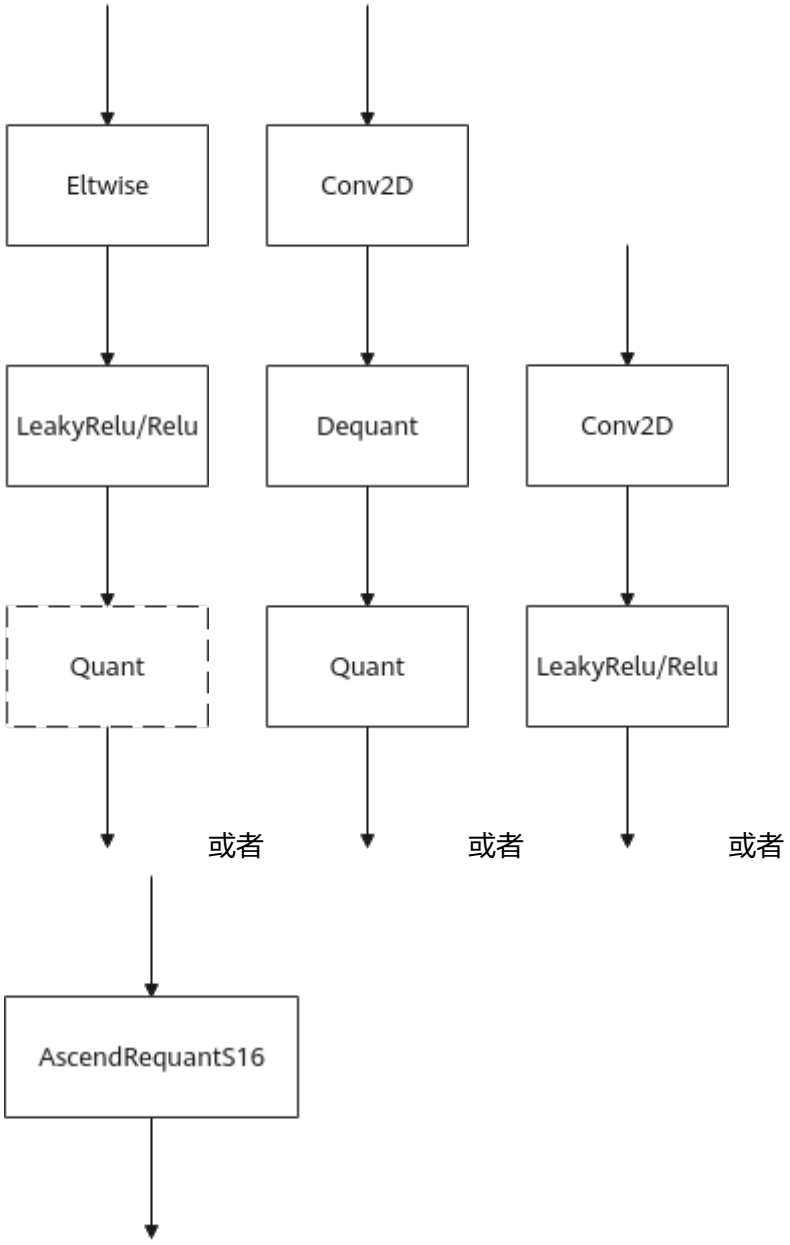
使用约束

Relu本身的数据类型是float32，前后的Cast算子分别是float16转float和float转float16。

2.45 StrideHoistingPass

融合模式

该融合主要是根据不同的图结构插入ReadSelect算子，如果匹配到Conv2D，则会修改Conv2D的shape和属性。最终目的是让计算量减半。



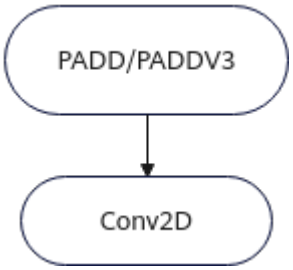
使用约束

无

2.46 PadConv2dFusionPass

融合模式

该融合将PadD/PadDV3+Conv2D算子融合成Conv2D算子。



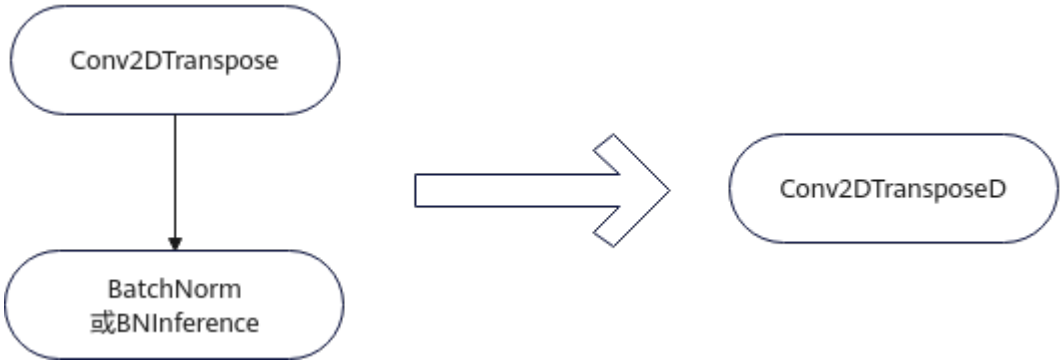
使用约束

该融合中pad算子的constant_values必须为0，即不支持补非零的pad。

2.47 Conv2DTransposeBatchnormFusionPass

融合模式

该融合规则将Conv2dTranspose+BatchNorm或BNInferenceD 算子融合为Conv2dTransposeD算子，将BatchNorm或BNInference的输入转化为Conv2dTransposeD的weight和bias输入。

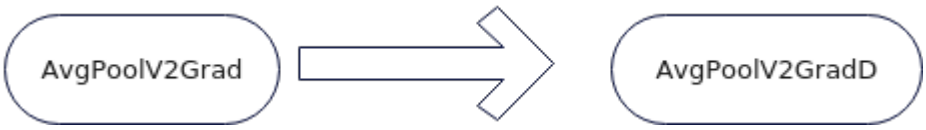


使用约束

- Conv2dTranspose的属性groups必须等于1。
- BatchNorm或BNInference的输入类型（除输入x之外）必须是const类型。

2.48 AvgPoolV2GradFusionPass

该融合规则将AvgPooV2lGrad算子融合为AvgPoolV2GradD算子。



使用约束

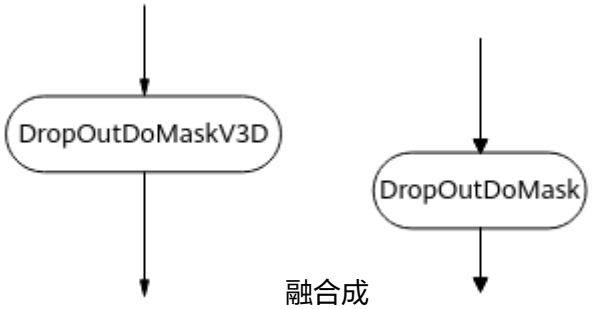
AvgPoolV2Grad的输入ori_input_shape必须是const节点或者带value值的data节点。

2.49 DropOutDoMaskFusionPass

融合模式

该融合规则替换DropOutDoMaskV3D算子为DropOutDoMask。

融合后DropOutDoMaskV3D算子替换为DropOutDoMask算子，DropOutDoMask算子可以走DSL分支通过Elewise模板实现，支持UB融合。



使用约束

在支持DSA模块时融合规则生效且必须打开，在需要DropOutDoMask算子进行UB融合时该融合规则必须打开。

支持的芯片型号

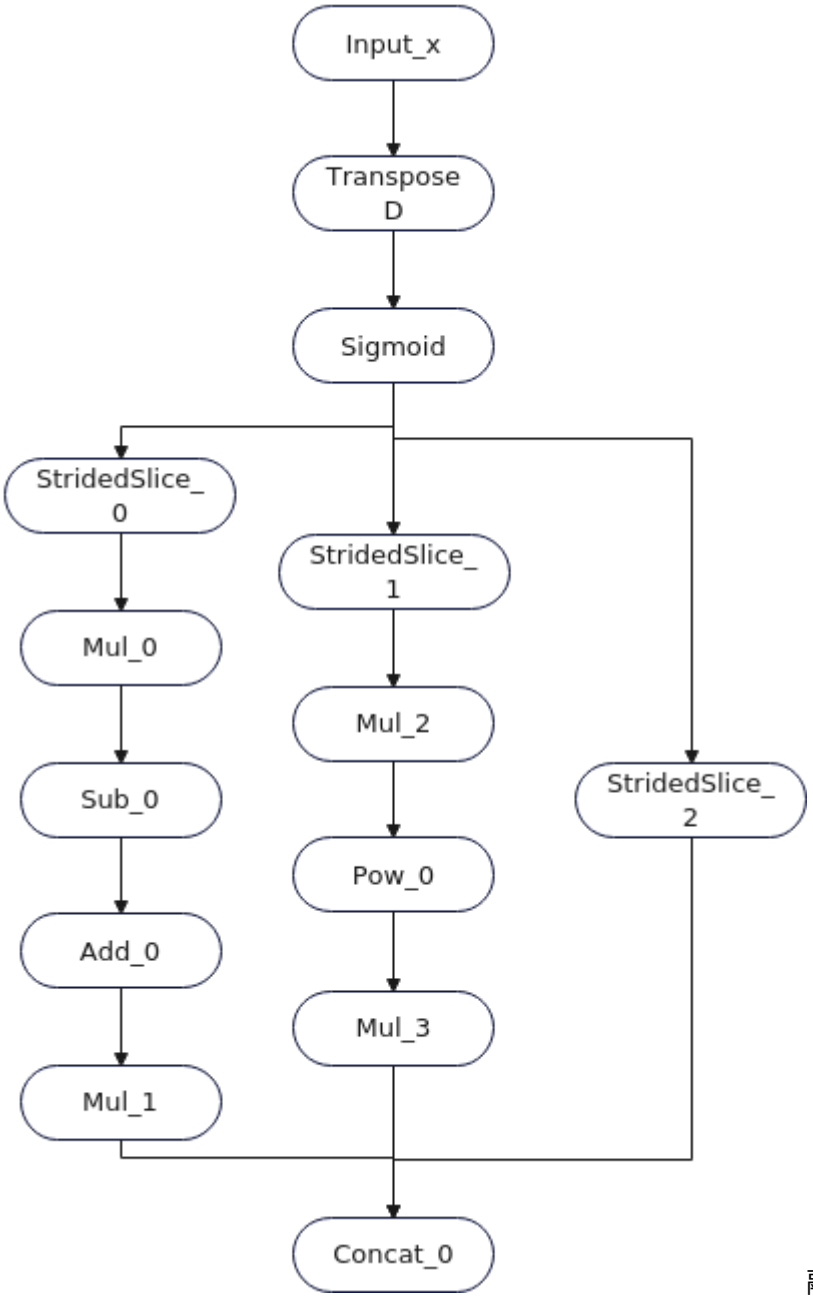
昇腾910B AI处理器

2.50 TransposeSigmoidMulAddPowConcatFusionPass

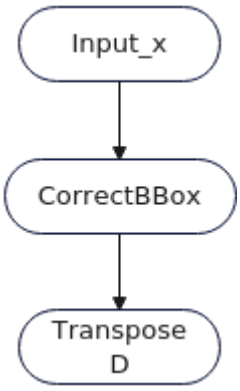
融合模式

该融合规则将yolov3，yolov5和yolov7不带nms后处理算子的模型最后的sigmoid结构融合成一个算子。

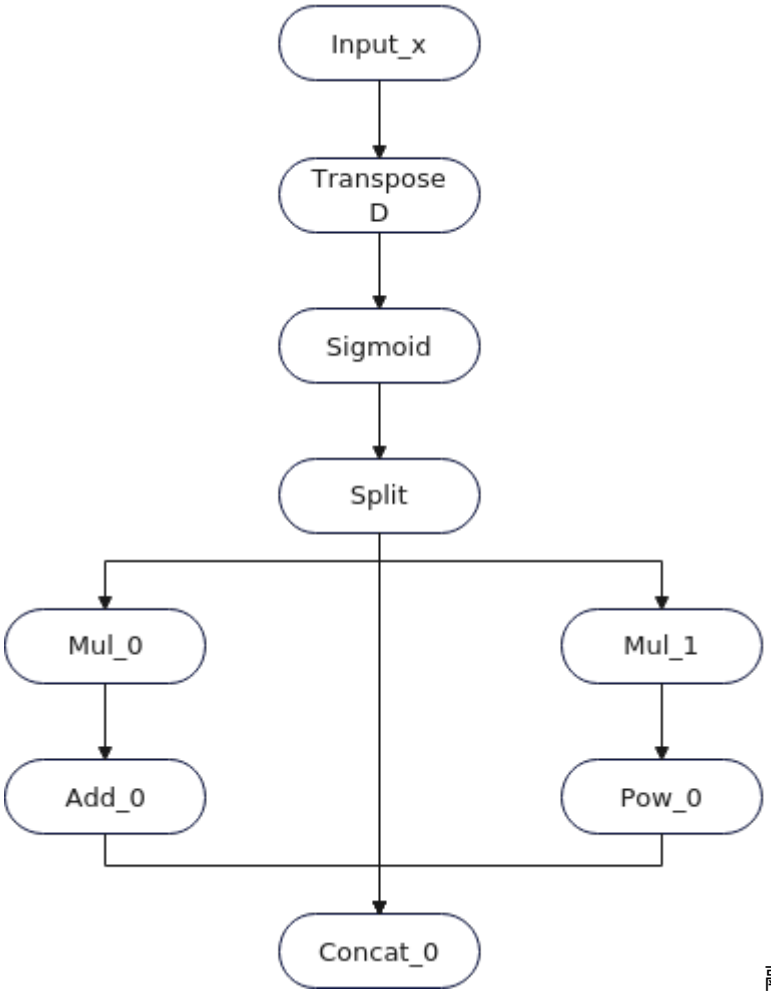
模式一：



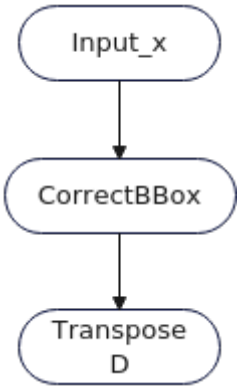
融合成



模式二：



融合成



使用约束

无

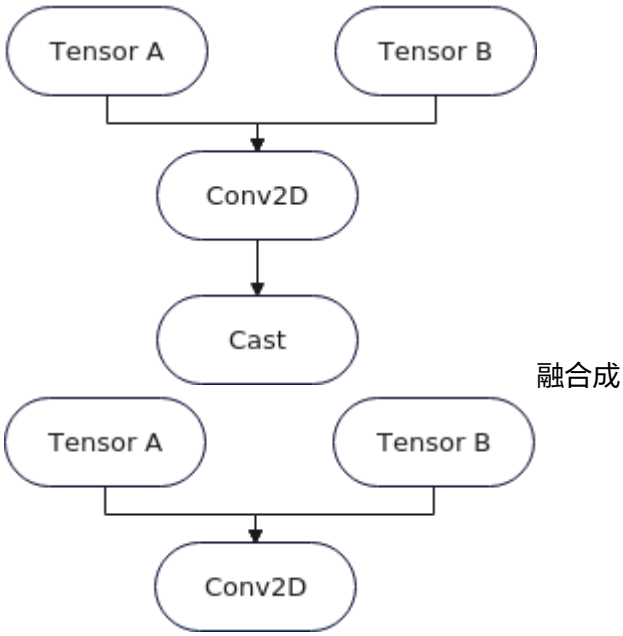
支持的芯片型号

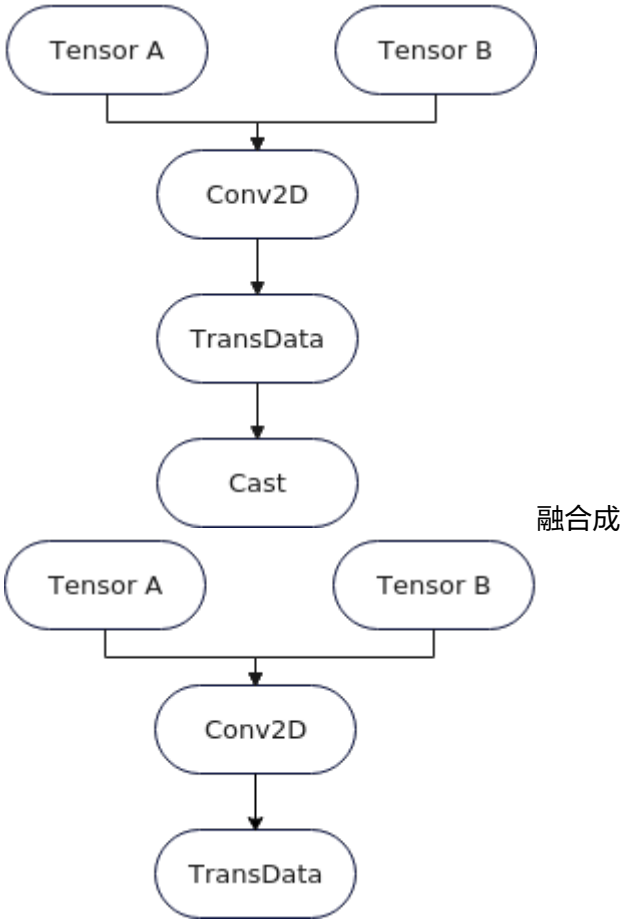
昇腾310P AI处理器

2.51 ConvCastFusionPass

融合模式

将Conv2D算子跟Cast算子融合为一个Conv2D算子





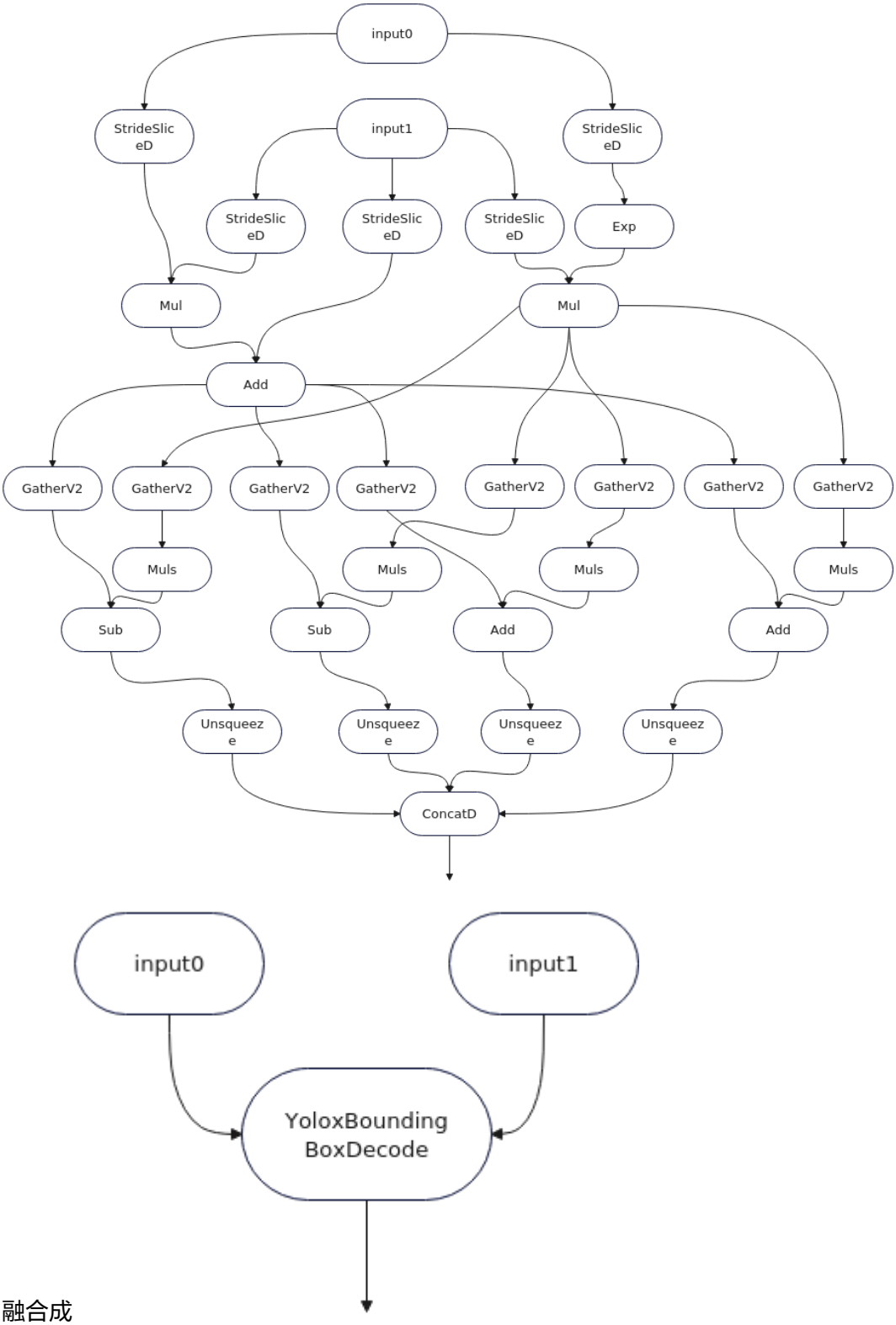
使用约束

当Conv2D的输入数据类型为float16,Cast输出数据类型为float32时，该融合生效。

2.52 YoloxBoundingBoxDecodeONNXFusionPass

融合模式

该融合规则将StridedSliceD/Mul/Exp/GatherV2/Add/Muls/Sub/Unsqueeze/ConcatD算子融合修改为YoloxBoundingBoxDecode算子，提高计算性能。



融合成

使用约束

无

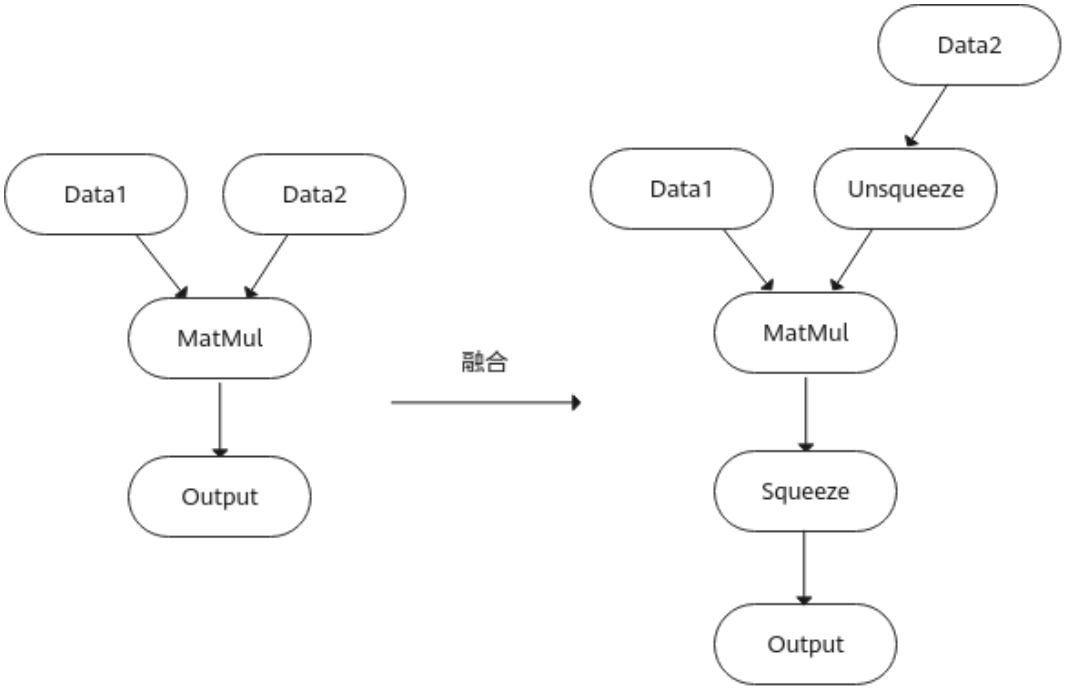
支持的芯片型号

昇腾310P AI处理器

2.53 MatMulUnsqueezeSqueezeFusionPass

融合模式

MatMul/MatMulV2/BatchMatmul/BatchMatmulV2支持一维输入场景下，需要将一维输入插入Unsqueeze算子扩成二维，输出插入Squeeze算子去掉对应的扩维轴。



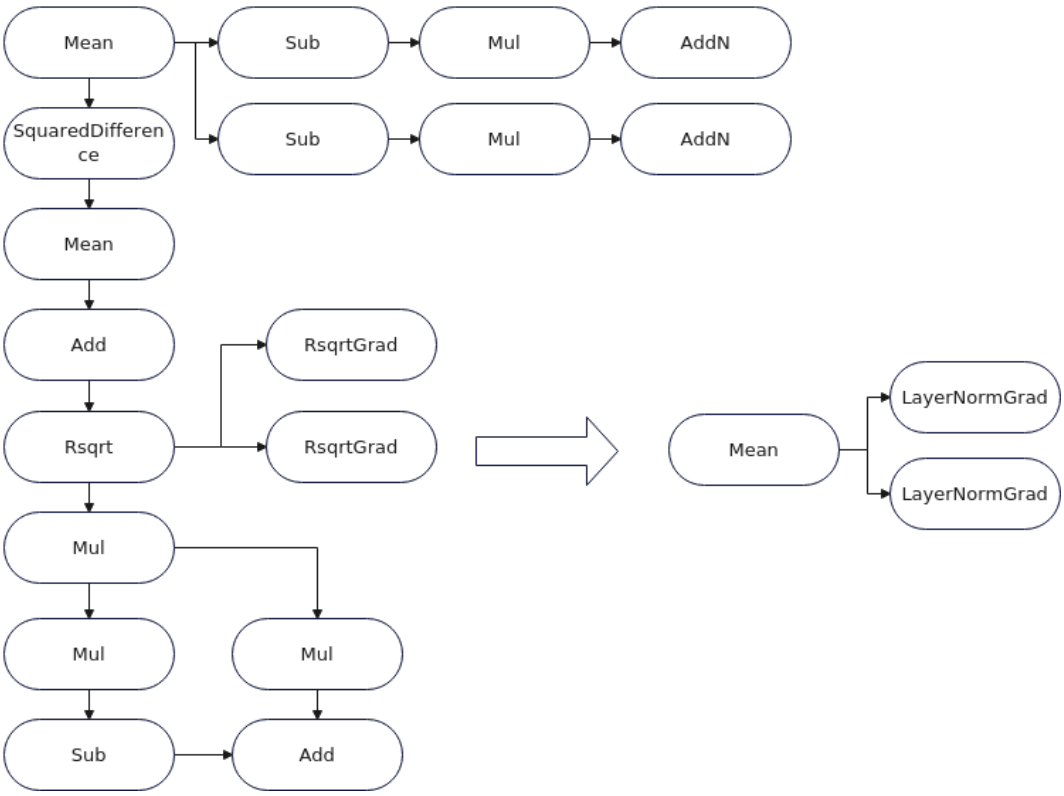
使用约束

无

2.54 LayerNormSpecialTrainingFusionPass

融合模式

该融合规则将Mean/SquaredDifference/Add/Rsqrt/RsqrtGrad/Mul/Sub/AddN算子融合成一个LayerNorm算子和两个LayerNormGrad算子，提高计算性能。如下图所示：



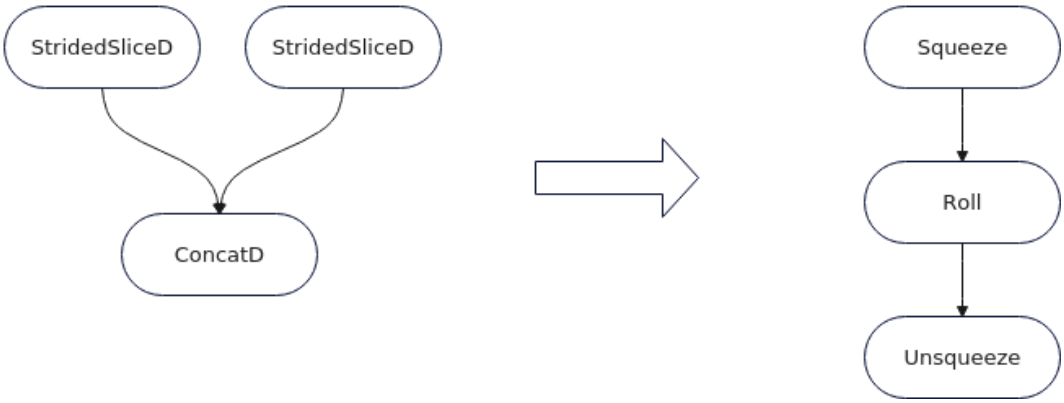
使用约束

无

2.55 StridedSliceConcatFusionPass

融合模式

该融合规则将StridedSliceD/StridedSliceD/ConcatD算子融合成Squeeze/Roll/Unsqueeze，提高计算性能。如下图所示：



使用约束

满足如下条件时，该融合规则不生效。

- concat_dim不为1。
- 两个StridedSliceD的输入不为同一个。
- 输入的shape不为4维。
- 输入shape的最后两维不是32位对齐。
- 属性strides的值不全为1。

支持的芯片型号

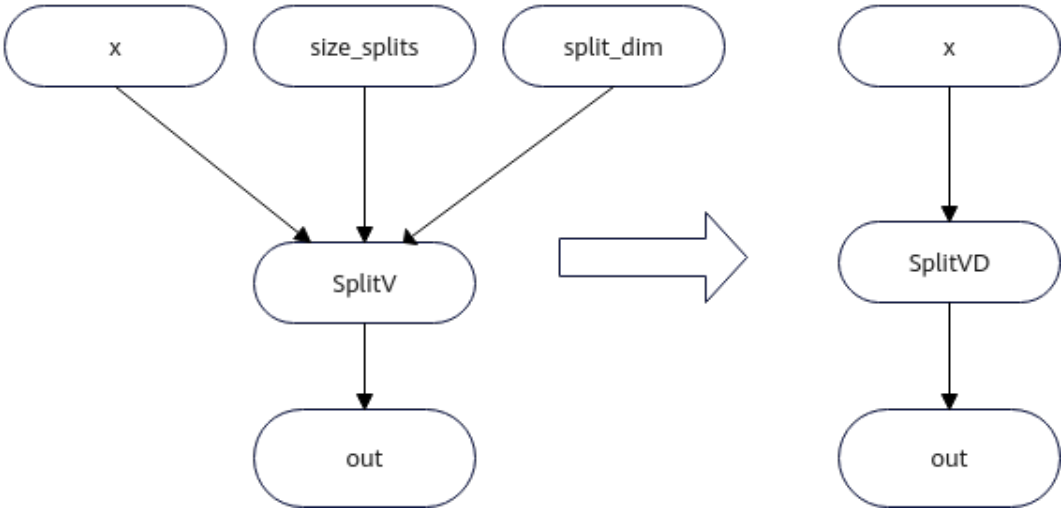
昇腾310P AI处理器

昇腾910 AI处理器

2.56 ZSplitVDFusionPassV2

融合模式

该图融合用在SplitV的size_splits和split_dim为常量时，将其融合成SplitVD，同时删除了size_splits和split_dim节点，将其变成SplitVD的属性，用于提升算子性能。



使用约束

无

3 UB 融合规则说明

Conv2DDequantClipByValueFusionPass

TbeConv2dAddClipMulDivFusionPass

TbeConv2DAddMulQuantPass

TbeConv2dAddRelu6MulMulFusionPass

ConvClipByValueFusionPass

TbeAippConvReluMaxpoolingFusion

TbeAippCommonFusionPass

AutomaticBufferFusion

TbeSegmentElemwiseFusionPass

TbeReduceElemwiseFusionPass

TbeReadSelectEltwiseFusionPass

TbeMultiOutputFusionPass

TbeEltwiseWriteSelectFusionPass

TbeEltwiseQuantFusionPass

TbeEltwiseFusionPass

TbeDynamicElemwiseReduceFusionPass

TbeDynamicElemwiseBroadcastFusionPass

TbeConvBnreduceFusionPass

TbeBnupdateEltwiseFusionPass

TbeConv2DBackpropElemwiseFusionPass

TbeDepthwiseConvElemwiseFusionPass

TbeDxDeqElemQuantPass

TbeDxElemwisePass

TbeConv2dBackpropRequantFusionPass

TbeDwTransdataFusionPass

TbeDxTransdataFusionPass

MatmulGeneralizedUbFusion

MatmulTransdataUbFusion

TbeElemwiseQuantFusionPass

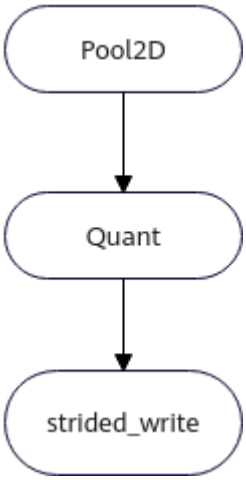
TbeEltwiseCastFusionPass

TbeConv2DBackpropDequantFusionPass

3.1 TbePool2dQuantFusionPass

融合模式

该融合将Pool2D+quant+strided_write算子融合成1个融合算子。



使用约束

该融合中strided_write会被优化，不可以单独关闭该融合规则。

3.2 FusionVirtualOpSetSwitch

融合模式

将Split、Concat类算子（SplitD/SplitVD/ConcatD/ConcatV2D这四个算子）加上no_task属性，从而可以使Split、Concat算子N轴拼接时提升性能。如果Split、Concat算子输入的内存过大，可以选择关闭该融合规则。

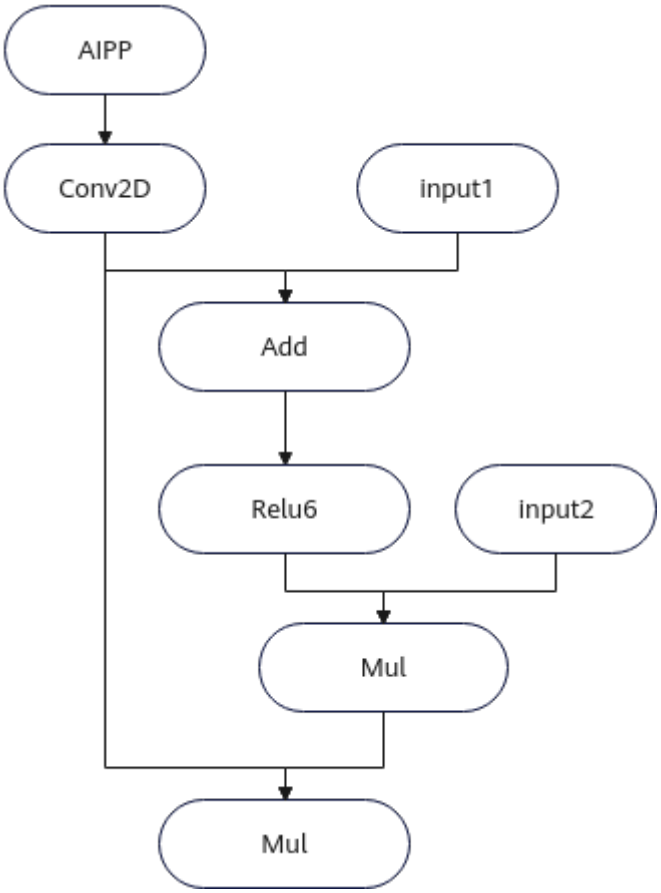
使用约束

无

3.3 TbeAippConv2dAddRelu6MulMulFusionPass

融合模式

该融合规则将满足如下Pattern的结构融合成一个融合算子。



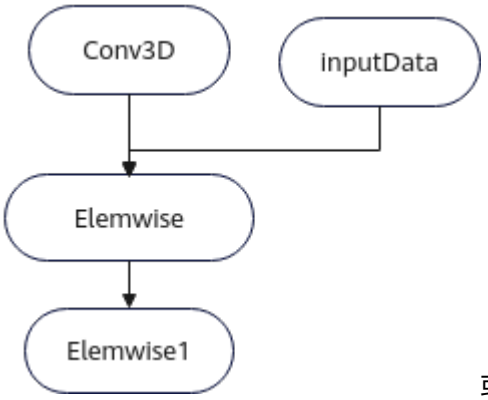
使用约束

无

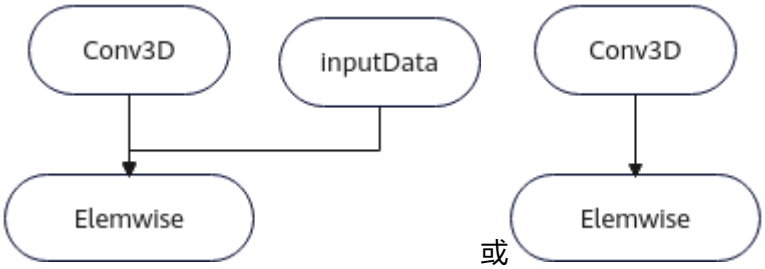
3.4 TbeConv3dElemwisePass

融合模式

该融合将满足如下Pattern关系的子图中Conv3D和Elemwise进行UB融合。



或



或

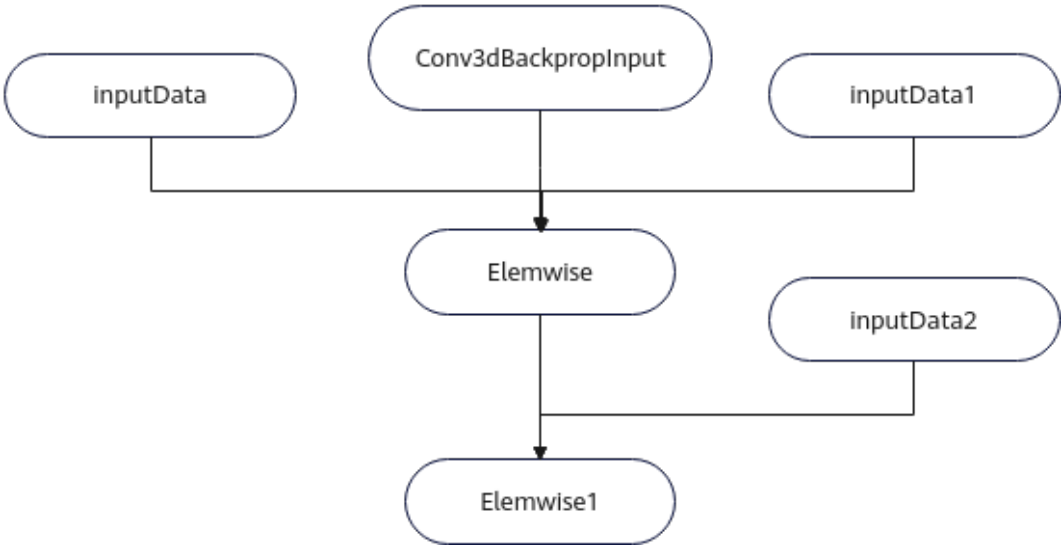
使用约束

无

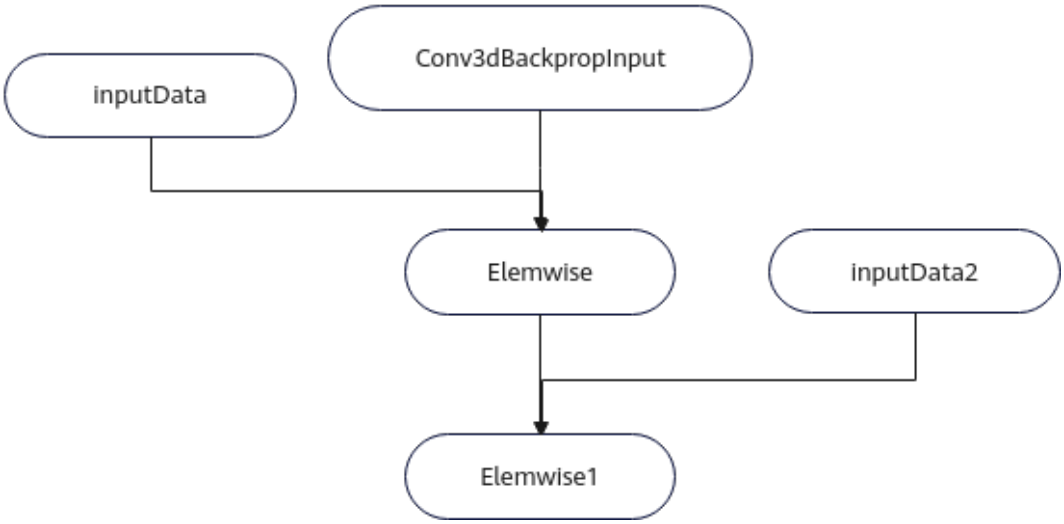
3.5 TbeConv3dDxElemwisePass

融合模式

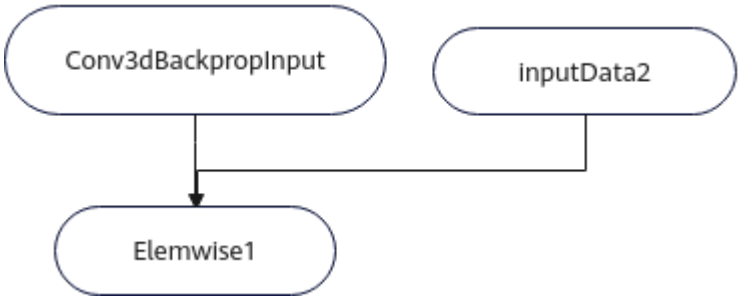
该融合将满足如下Pattern关系的子图中Conv3dBackpropInput和Elemwise进行UB融合。



或



或



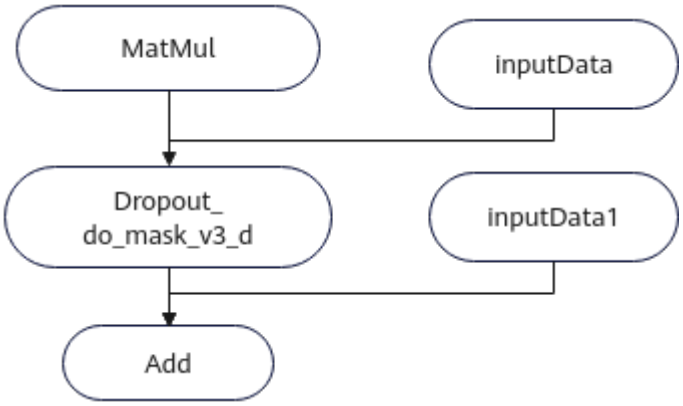
使用约束

图中elemwise只能为AddN；elemwise1只能为LeakyReluGrad。

3.6 MatMulDropoutDoMaskV3dFusionPass

融合模式

该融合将满足如下Pattern关系的子图中MatMul和Dropout_do_mask_v3_d/Add进行UB融合。



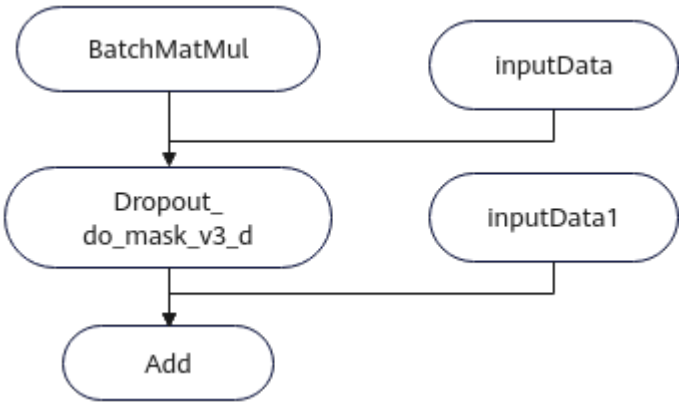
使用约束

无

3.7 BatchMatMulDropoutDoMaskV3dFusionPass

融合模式

该融合将满足如下Pattern关系的子图中BatchMatMul和Dropout_do_mask_v3_d/Add进行UB融合。



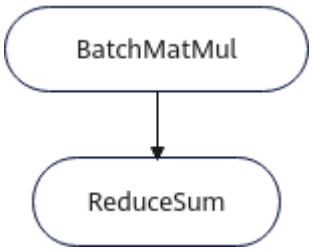
使用约束

无

3.8 MatmulReduceSumUbFusion

融合模式

该融合将满足如下Pattern关系的子图中BatchMatMul和ReduceSum进行UB融合。



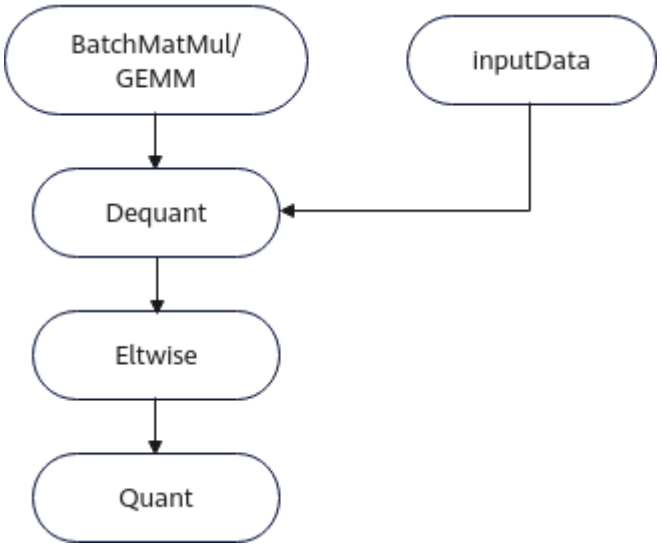
使用约束

BatchMatMul的输入都有且都不为1；输出为1维；reduce的输出为float32类型，keep_dim为false的情况下，进行UB融合。

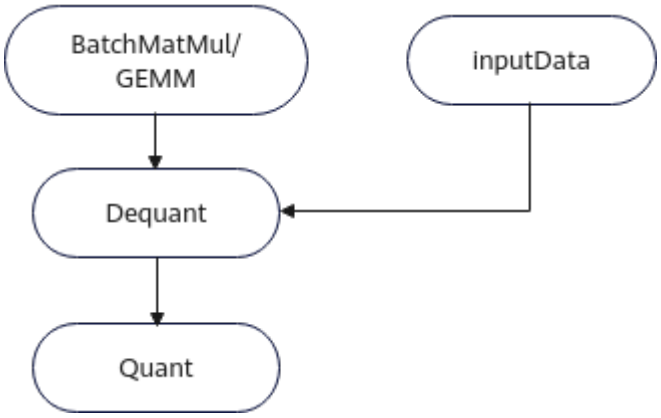
3.9 TbeBatchMatMulQuantFusionPass

融合模式

该融合将满足如下Pattern关系的子图中BatchMatMul/GEMM和Dequant/Quant进行UB融合。



或



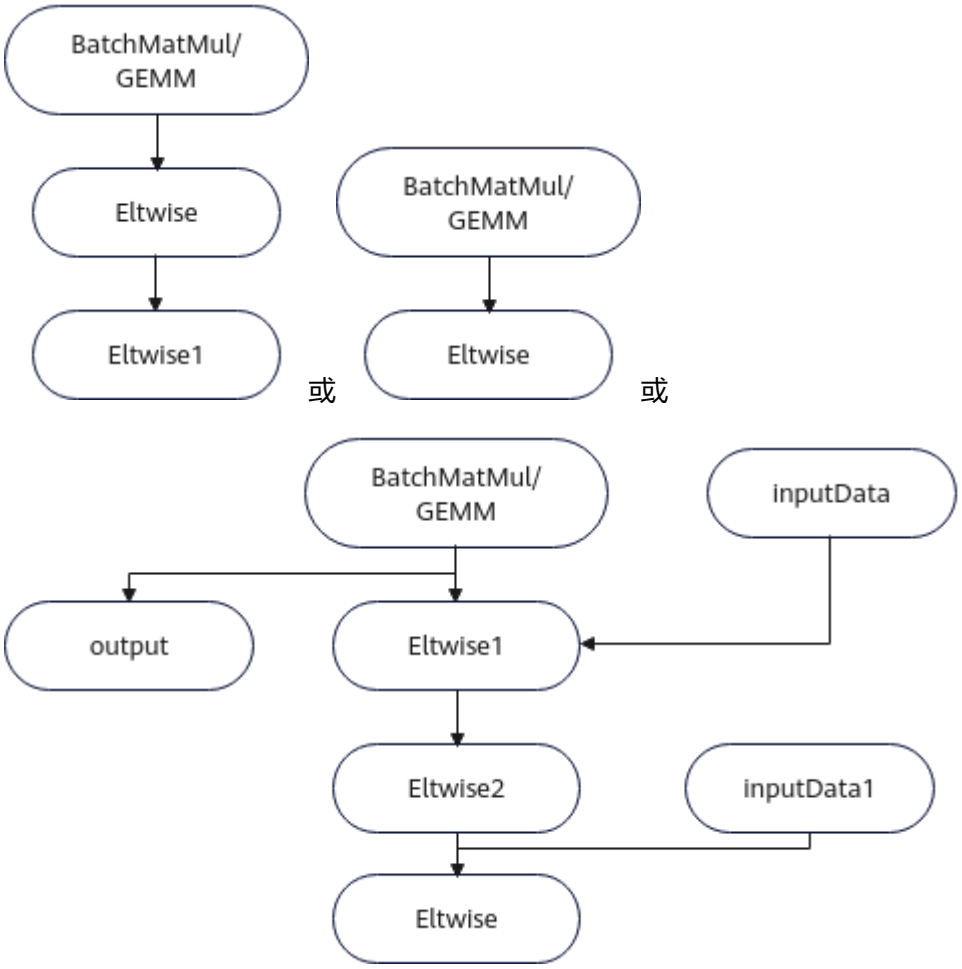
使用约束

无

3.10 TbeBatchMatMulElementWiseFusionPass

融合模式

该融合将满足如下Pattern关系的子图中BatchMatMul/GEMM和Eltwise进行UB融合。



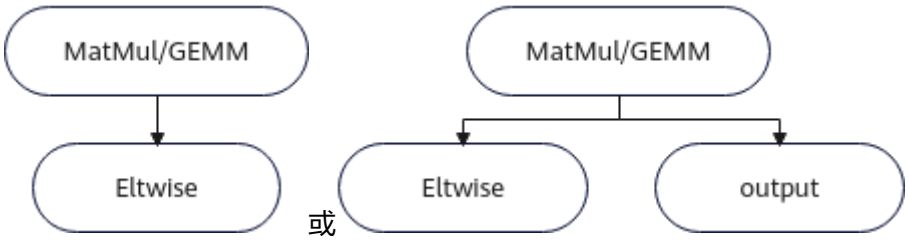
使用约束

无

3.11 ATbeMatMulElemwiseFusionPass

融合模式

该融合将满足如下Pattern关系的子图中MatMul/GEMM和Eltwise进行UB融合。



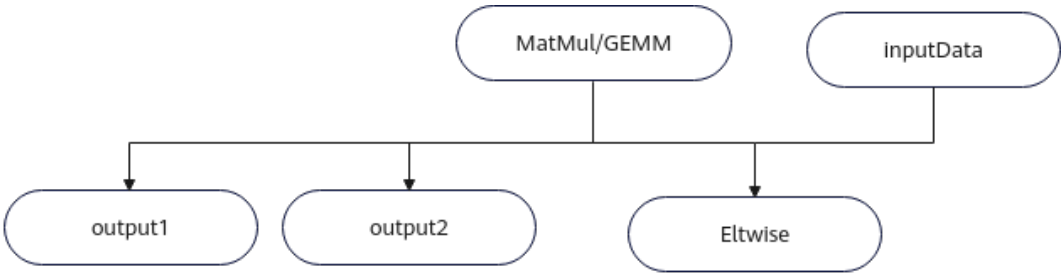
使用约束

无

3.12 MatMulFastgelugradUbFusion

融合模式

该融合将满足如下Pattern关系的子图中MatMul/GEMM和Eltwise进行UB融合。



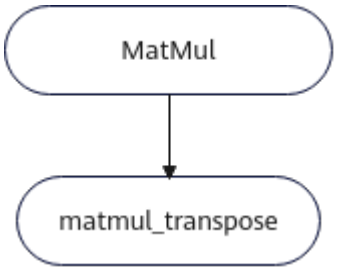
使用约束

无

3.13 MatmulConfusiontransposeUbFusion

融合模式

该融合将满足如下Pattern关系的子图中MatMul和matmul_transpose进行UB融合。



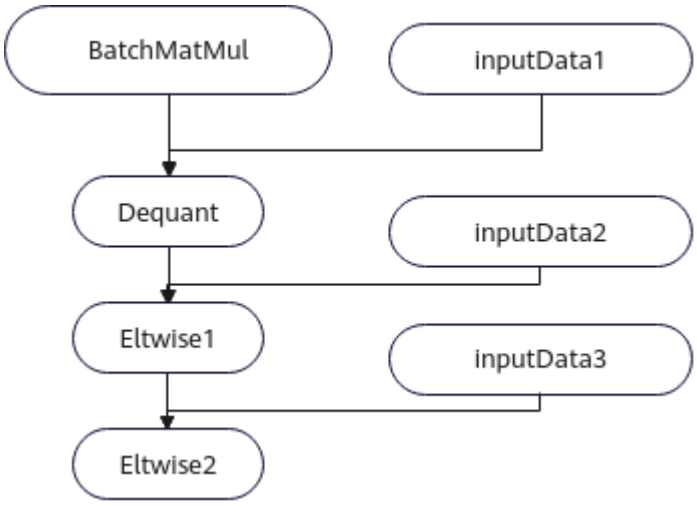
使用约束

无

3.14 BatchMatMulV2DequantMulAddFusionPass

融合模式

该融合将满足如下Pattern关系的子图中BatchMatMul+Eltwise+Dequant节点进行UB融合。



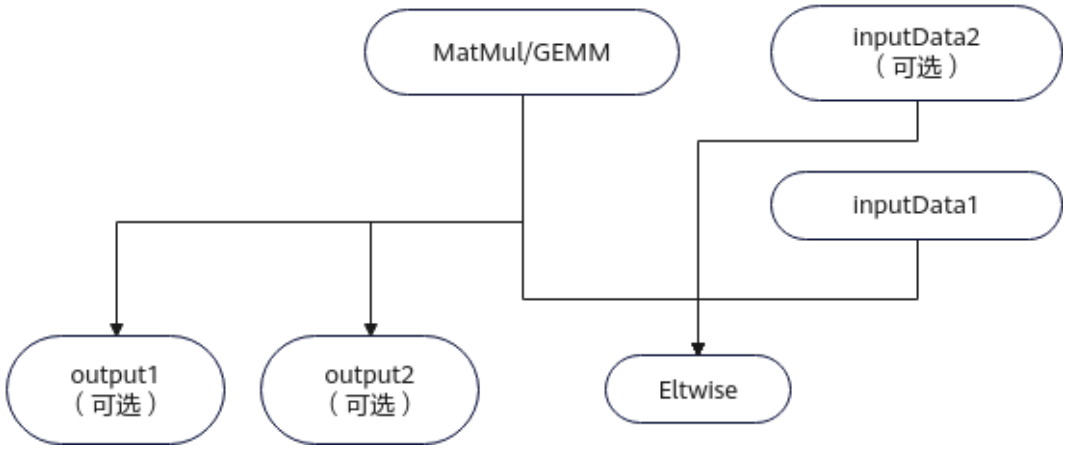
使用约束

上图中节点Eltwise1为mul，节点Eltwise2为Add。

3.15 MatMulGelugradUbFusion

融合模式

该融合将满足如下Pattern关系的子图中MatMul/GEMM和Eltwise进行UB融合。



使用约束

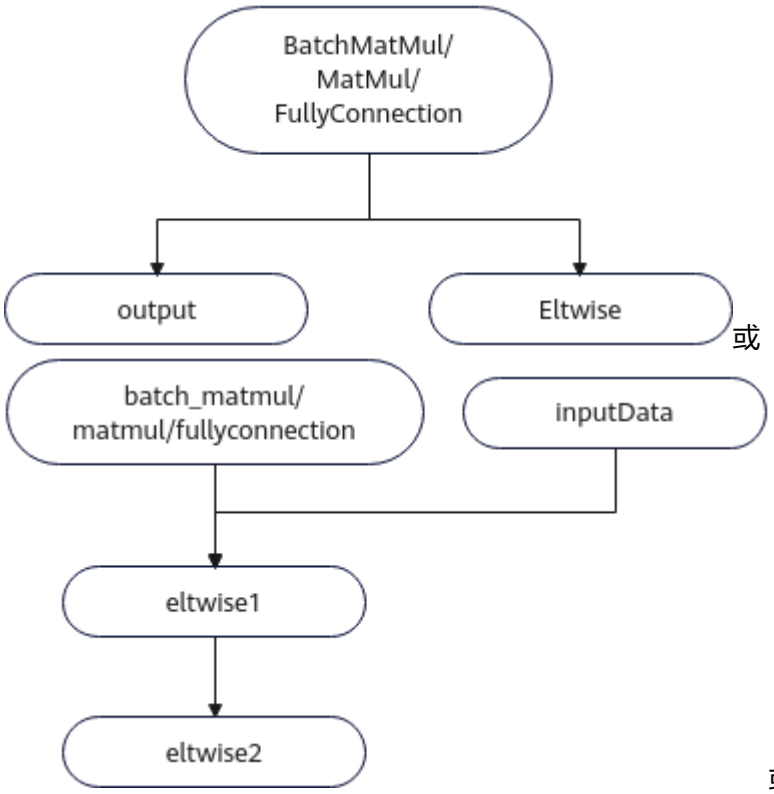
无

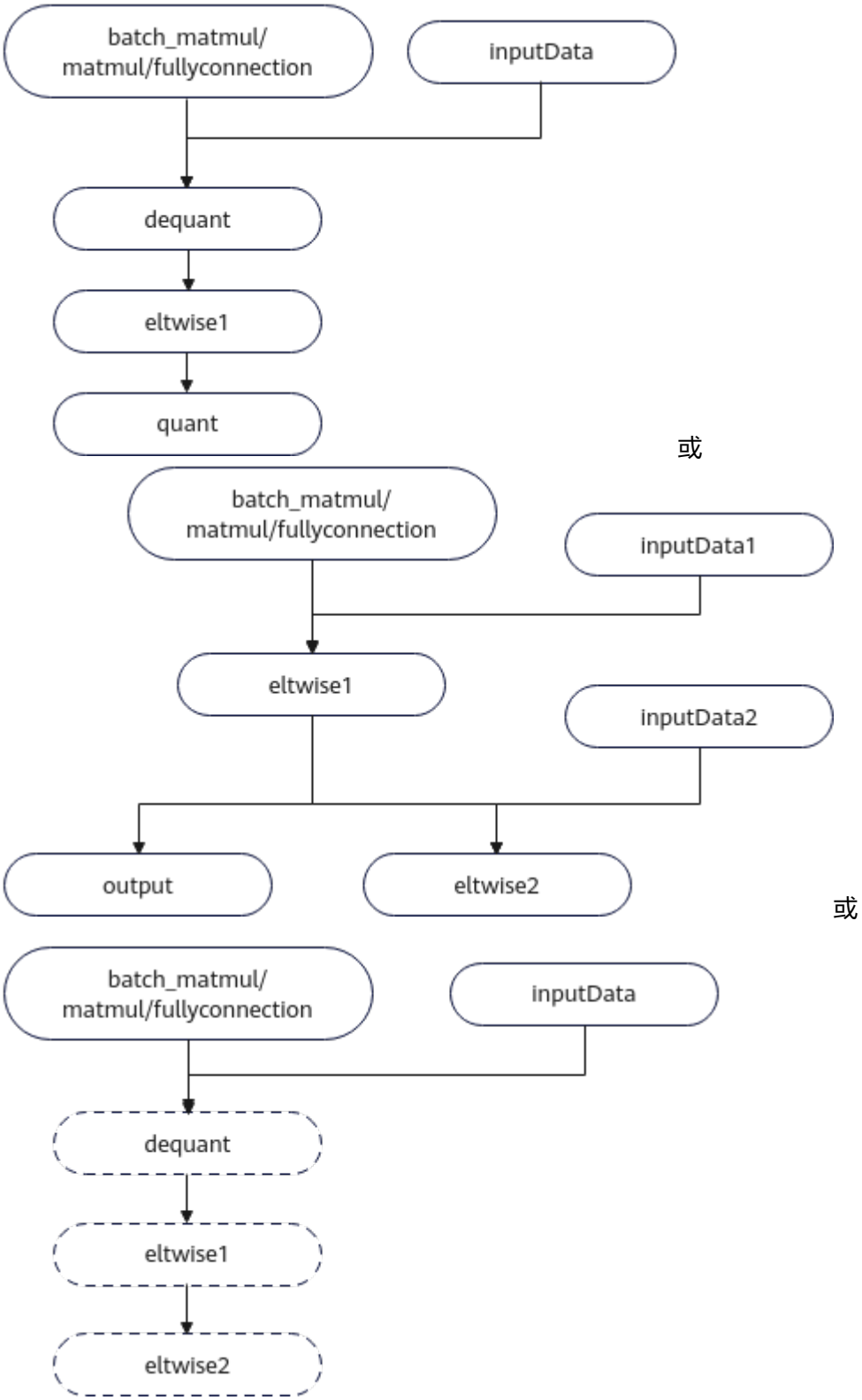
3.16 TbeFullyconnectionElemwiseDequantFusionPass

融合模式

该融合将满足如下Pattern关系的子图中BatchMatMul/MatMul/FullyConnection +Eletwise+Quant+Dequant对应节点进行UB融合。

第5个Pattern中的虚线框代表这些节点可以没有匹配。





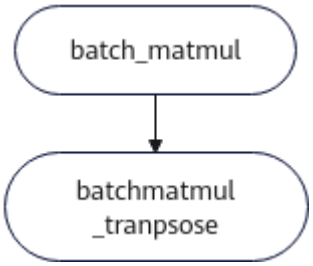
使用约束

无

3.17 BatchMatmulConfusiontransposeUbFusion

融合模式

该融合将满足如下Pattern关系的子图中batch_matmul和batchmatmul_transpose进行UB融合。



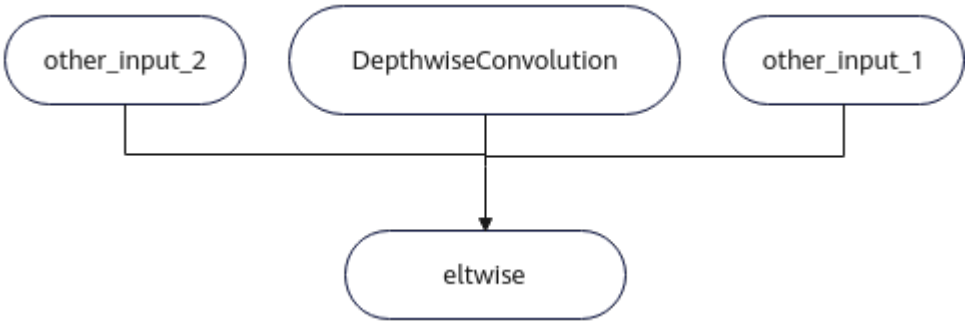
使用约束

无

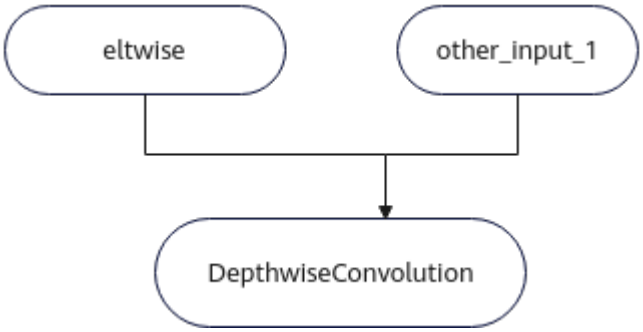
3.18 DepthwiseconvClipByValueFusionPass

融合模式

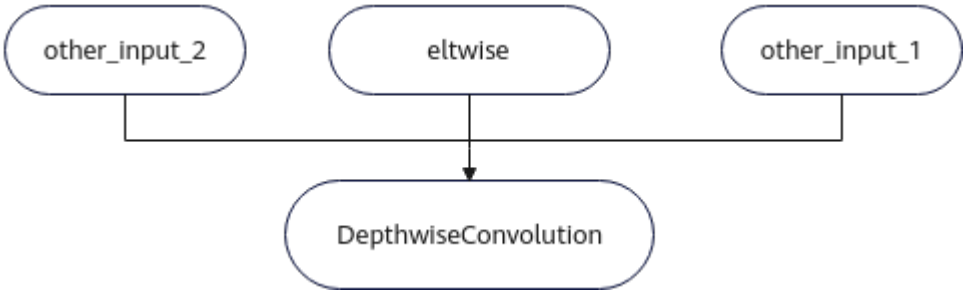
该融合将DepthwiseConvolution+eltwise或eltwise+DepthwiseConvlution算子融合成1个算子。



或



或



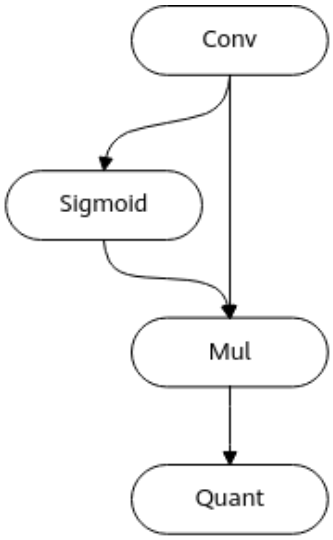
使用约束

该融合中eltwise只支持ClipbyValue。

3.19 TbeConvSigmoidMulQuantFusionPass

融合模式

该融合将Conv+Sigmoid+Mul+Quant算子融合成1个融合算子。



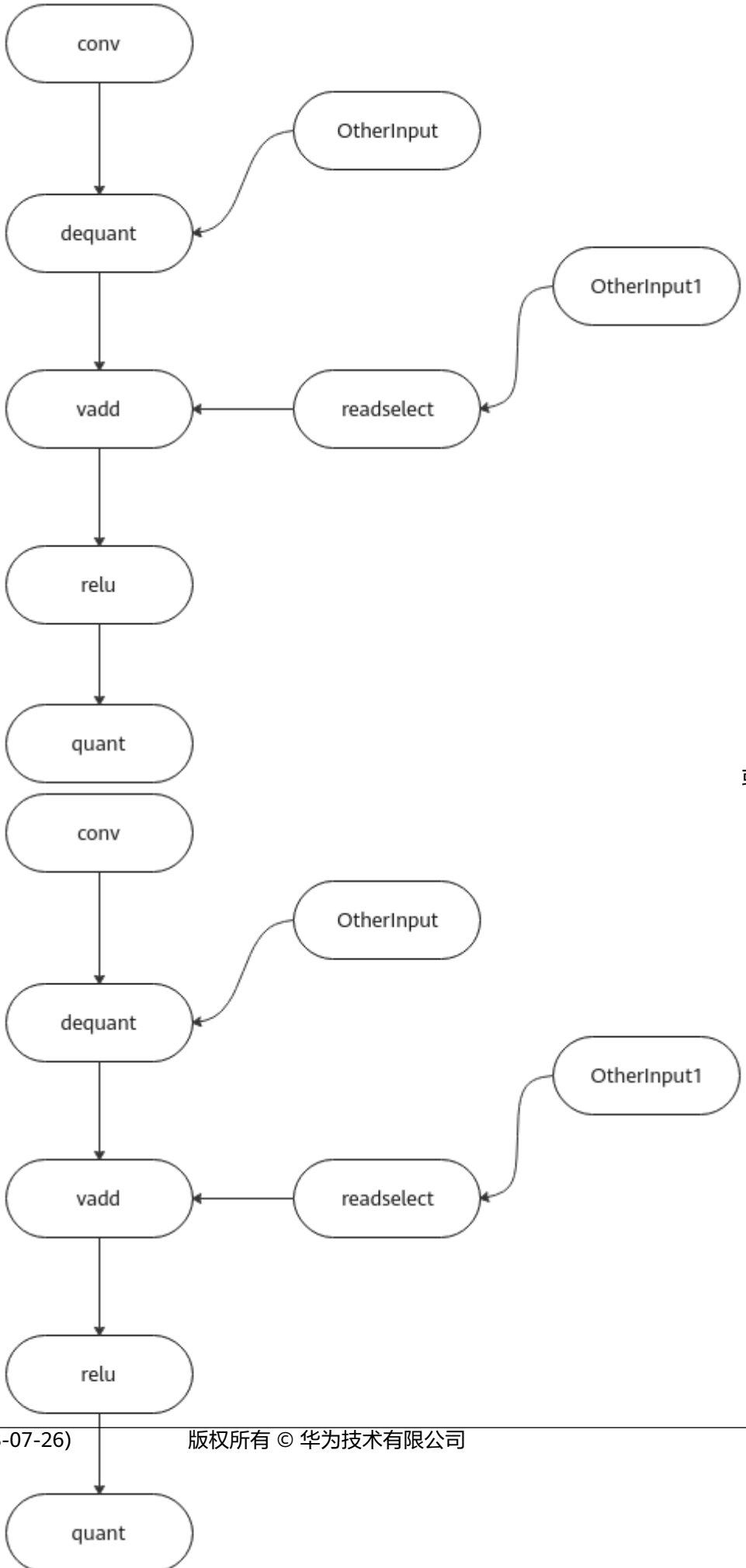
使用约束

无

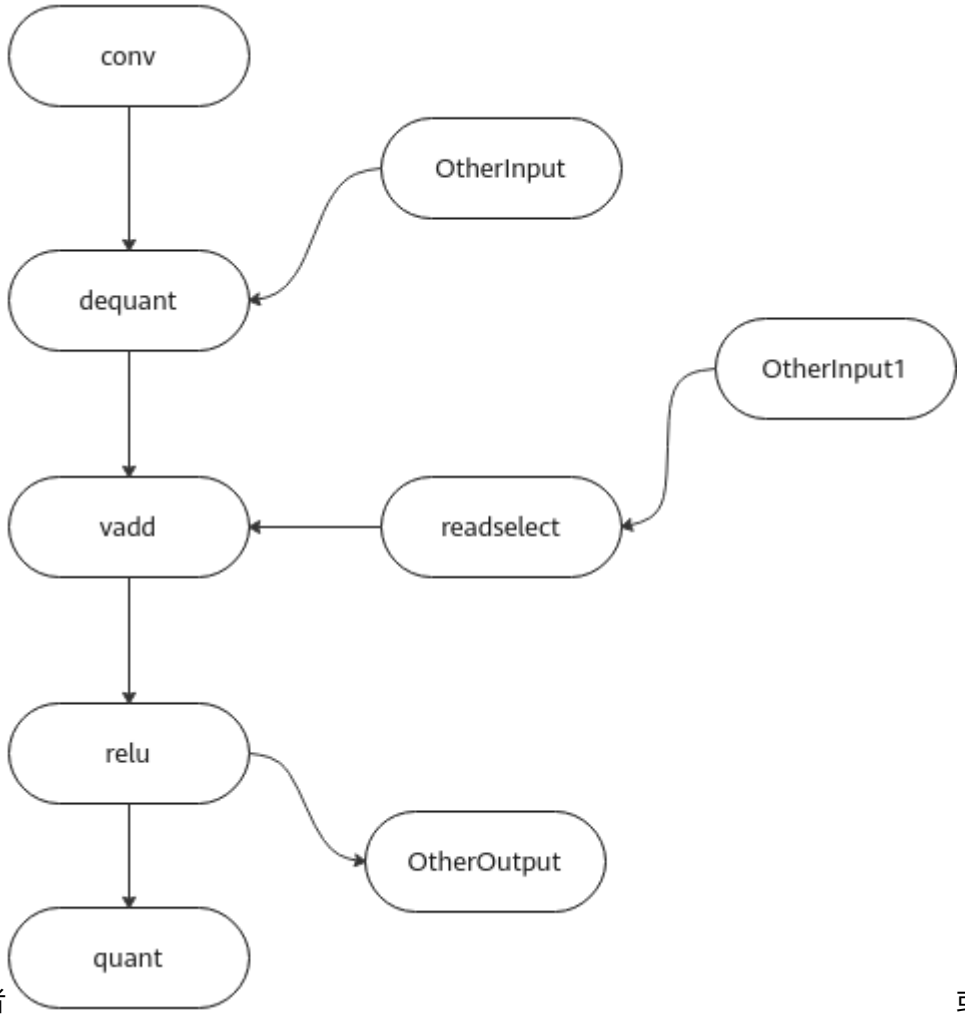
3.20 TbeConvDequantVaddReluQuantFusionPass

融合模式

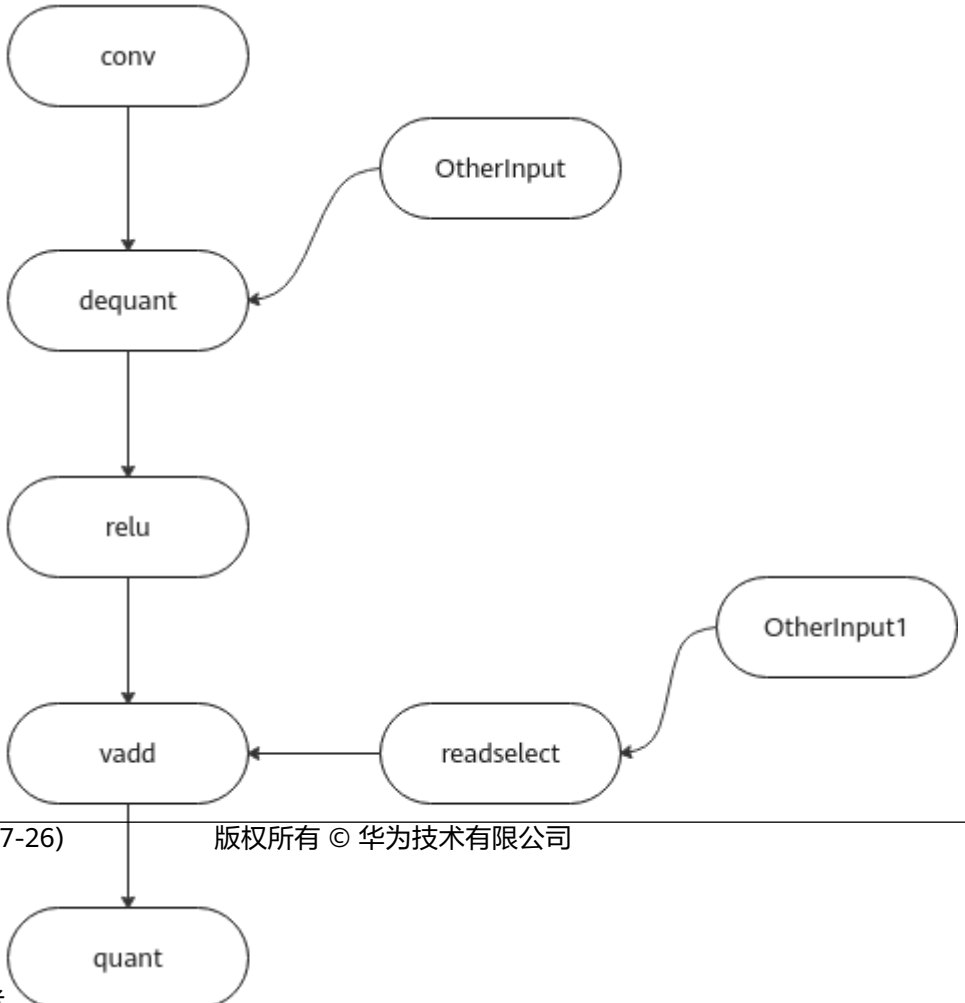
该融合支持将以下7种融合模式融合成单个conv2d融合算子。



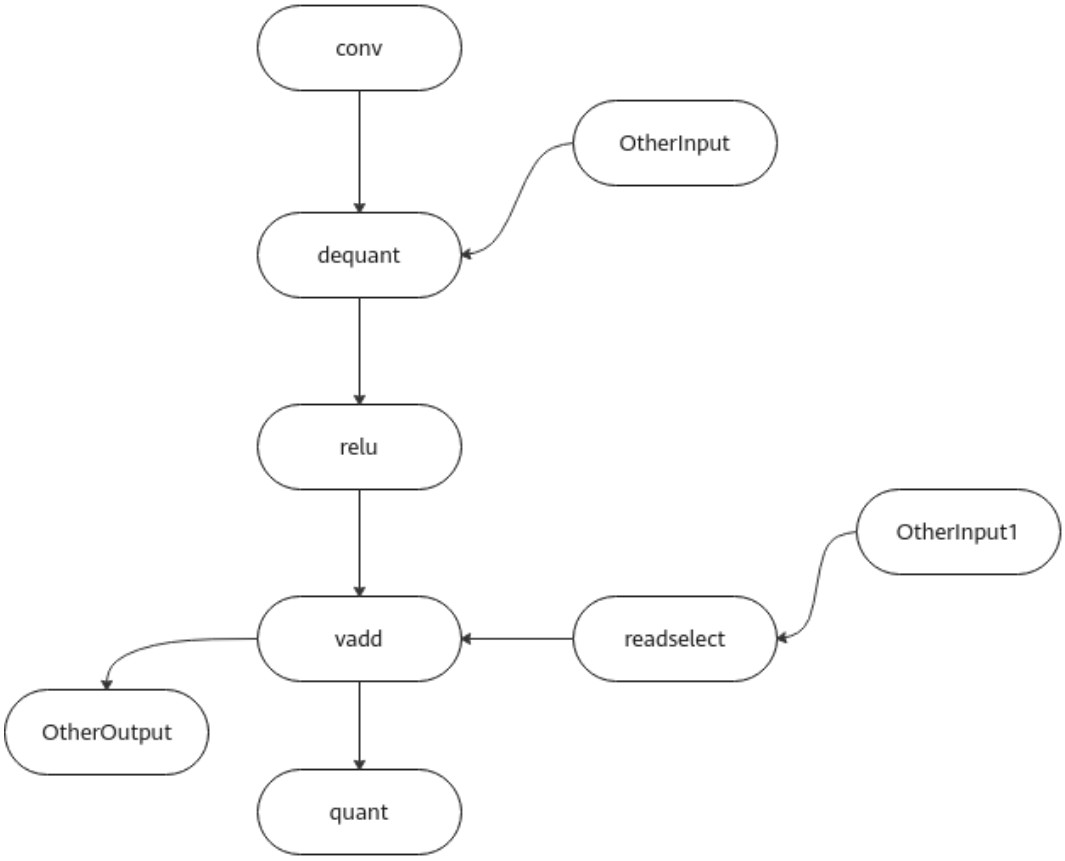
或者



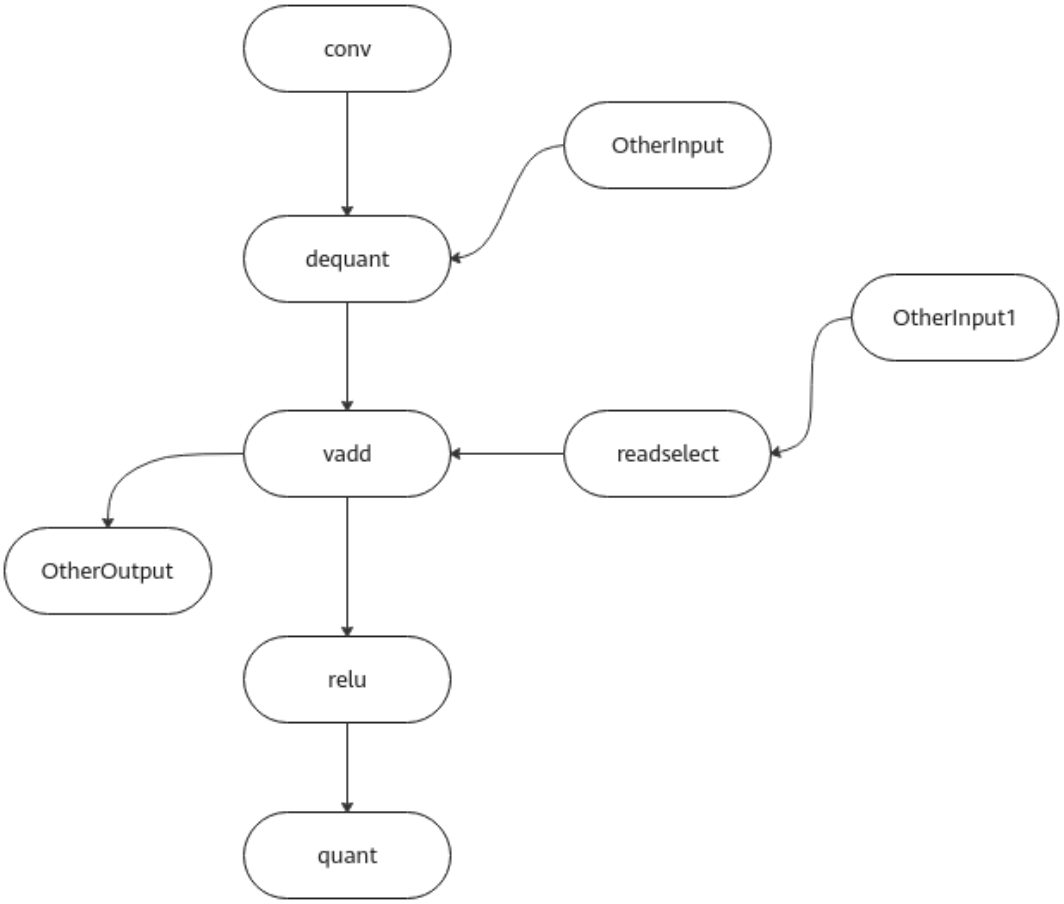
或者 或



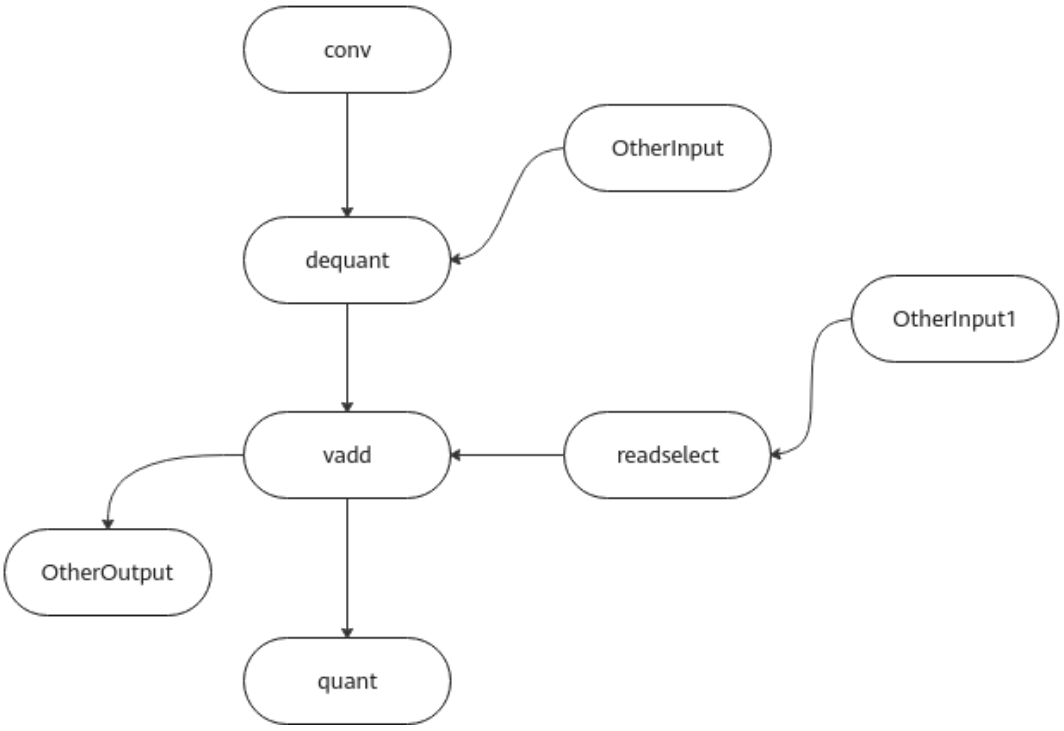
或者



或者



或者



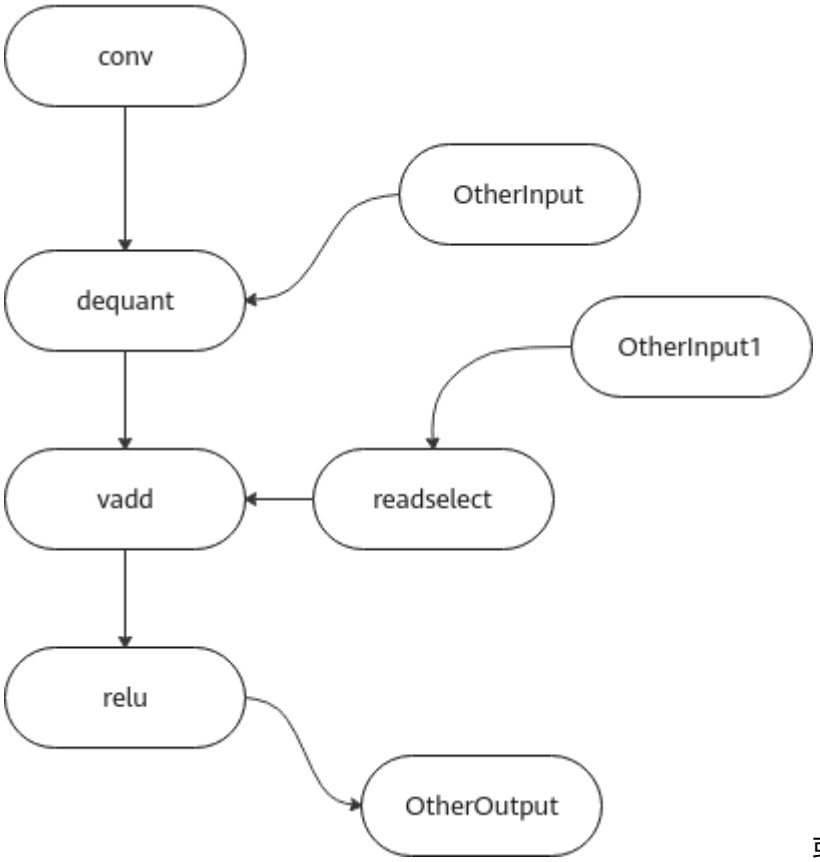
使用约束

vadd节点必须是Add算子。

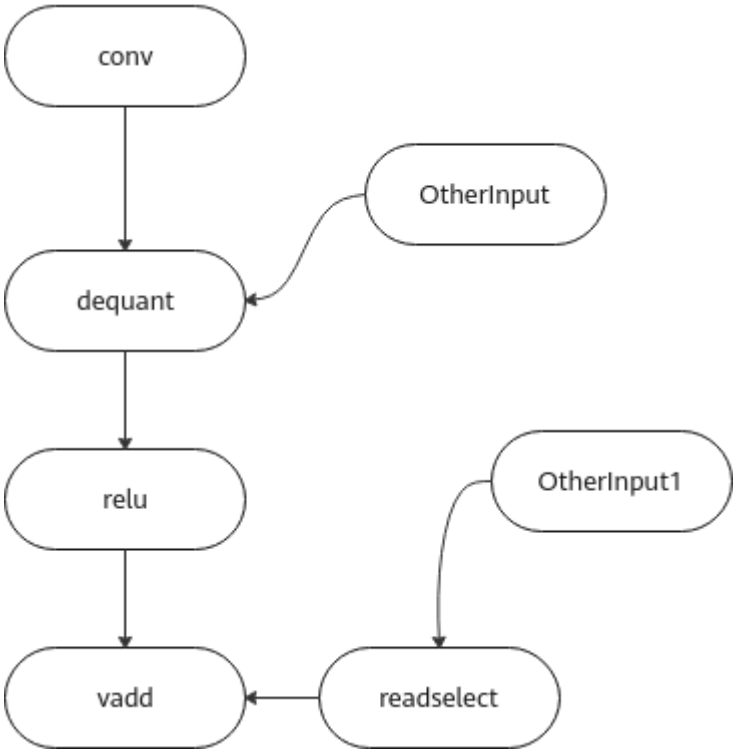
3.21 TbeConvDequantVaddReluFusionPass

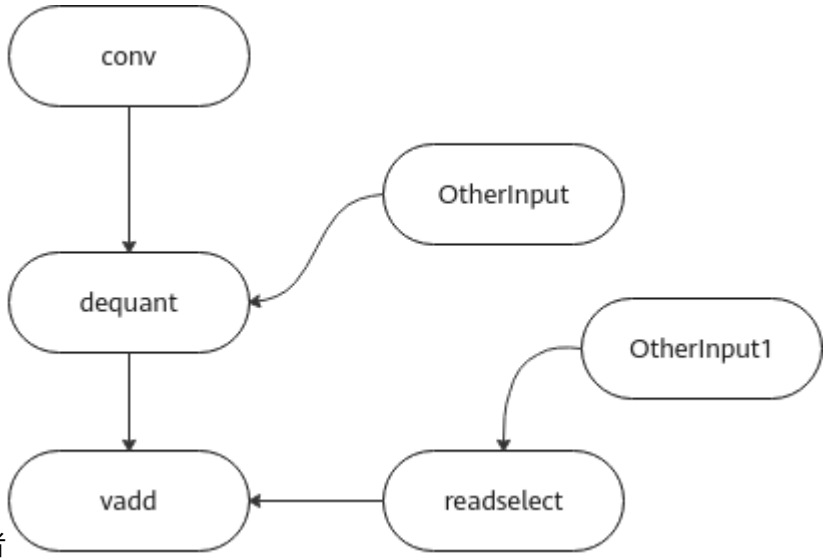
融合模式

该融合支持将以下三种融合模式融合成单个Conv2D融合算子。



或者





或者

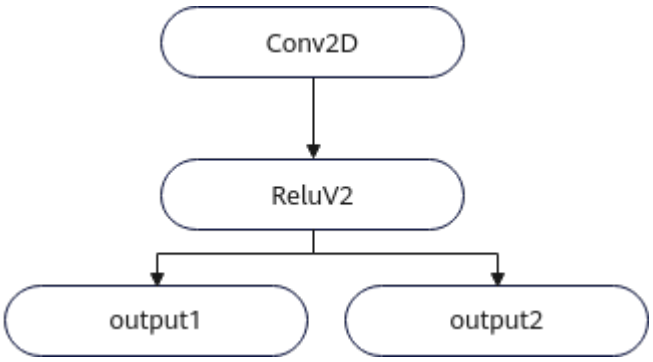
使用约束

relu节点的匹配不支持prelu。

3.22 TbeConv2DReluv2Pass

融合模式

该融合将Conv2D+ReluV2算子融合成1个Conv2D融合算子。



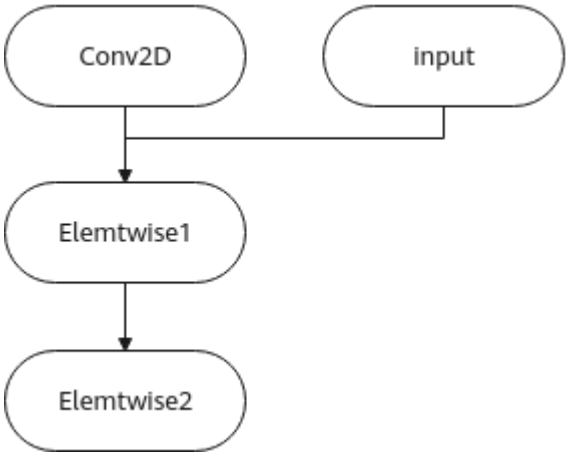
使用约束

- Conv2D的第一个输出output1的数据类型需要为float16。
- ReluV2的输出边要有两条。

3.23 TbeConvDoubleInFusionPass

融合模式

该融合将Conv2D + Elemtwise + Elemtwise算子融合成1个Conv2D融合算子。



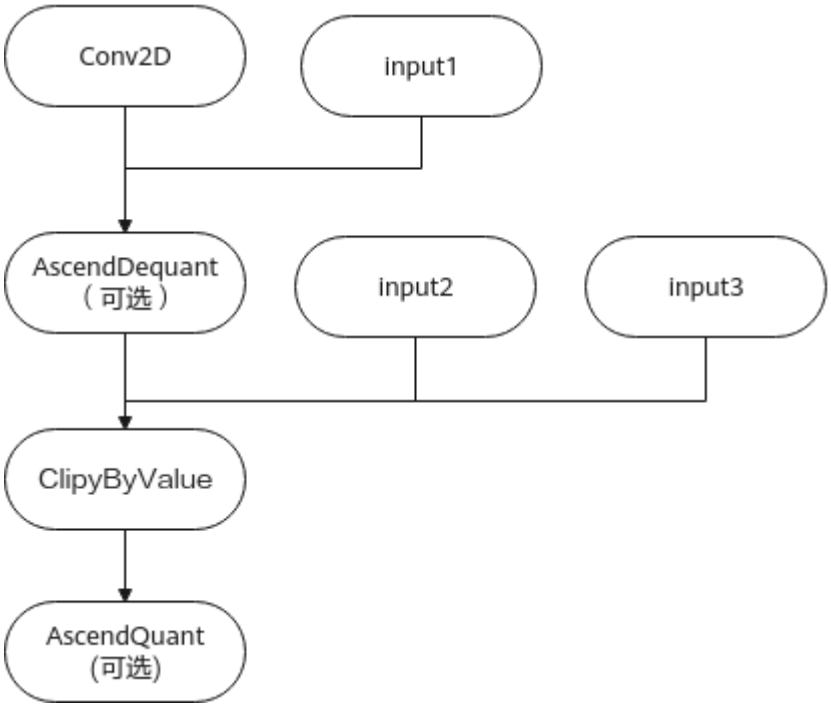
使用约束

该融合目前只支持节点格式为NCHW，NHWC和HWCN三种格式。

3.24 Conv2DDequantClipByValueFusionPass

融合模式

该融合将Conv2D + AscendDequant + ClipByValue + AscendQuant 算子融合成1个融合算子。



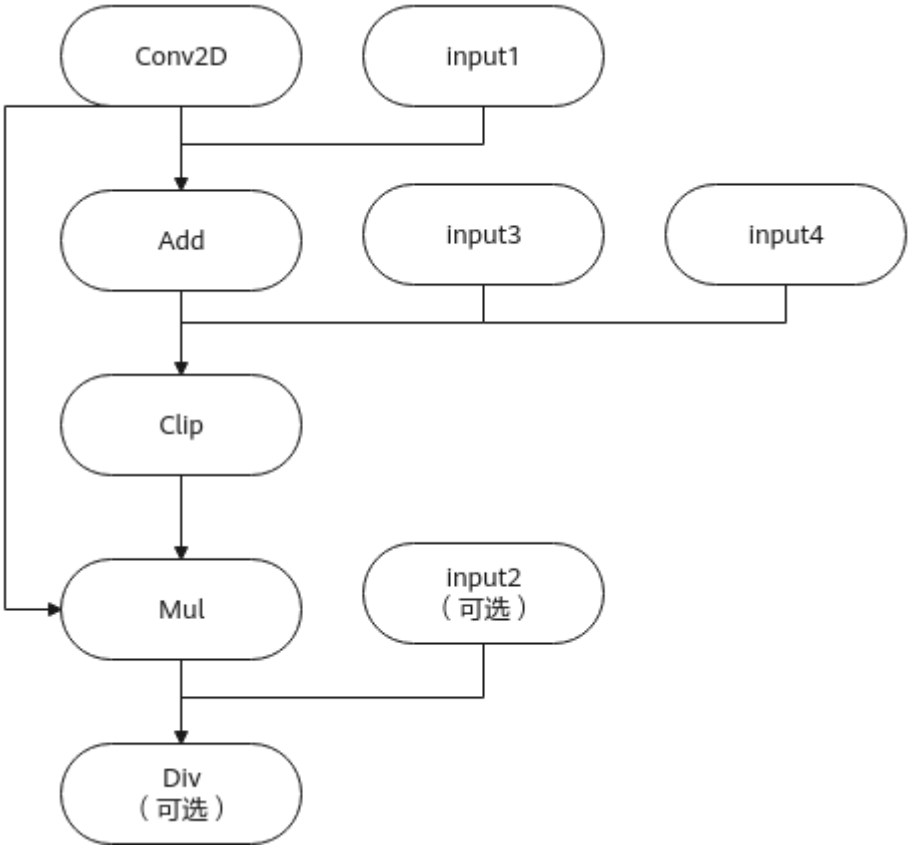
使用约束

无

3.25 TbeConv2dAddClipMulDivFusionPass

融合模式

该融合将Conv2D + Add + Clip + Mul + Div算子融合成1个融合算子。



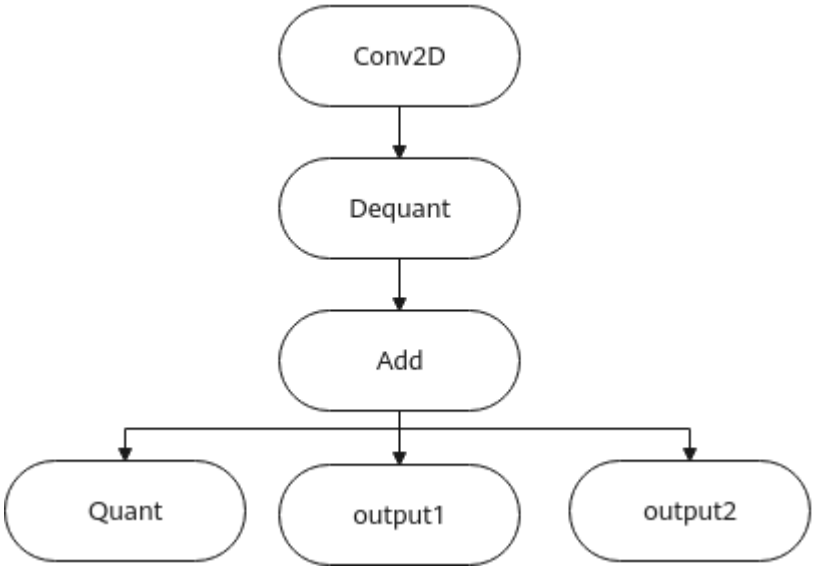
使用约束

无

3.26 TbeConv2DAddMulQuantPass

融合模式

将Conv2D + Dequant + Add + Quant 进行UB融合。



使用约束

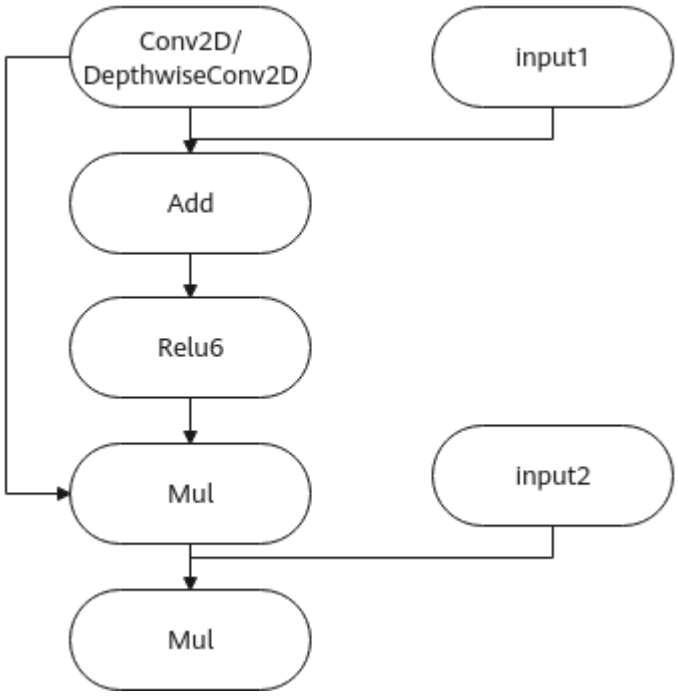
当Add算子另外两路输出节点为MaxPoolV3类型时，不支持融合。

3.27 TbeConv2dAddRelu6MulMulFusionPass

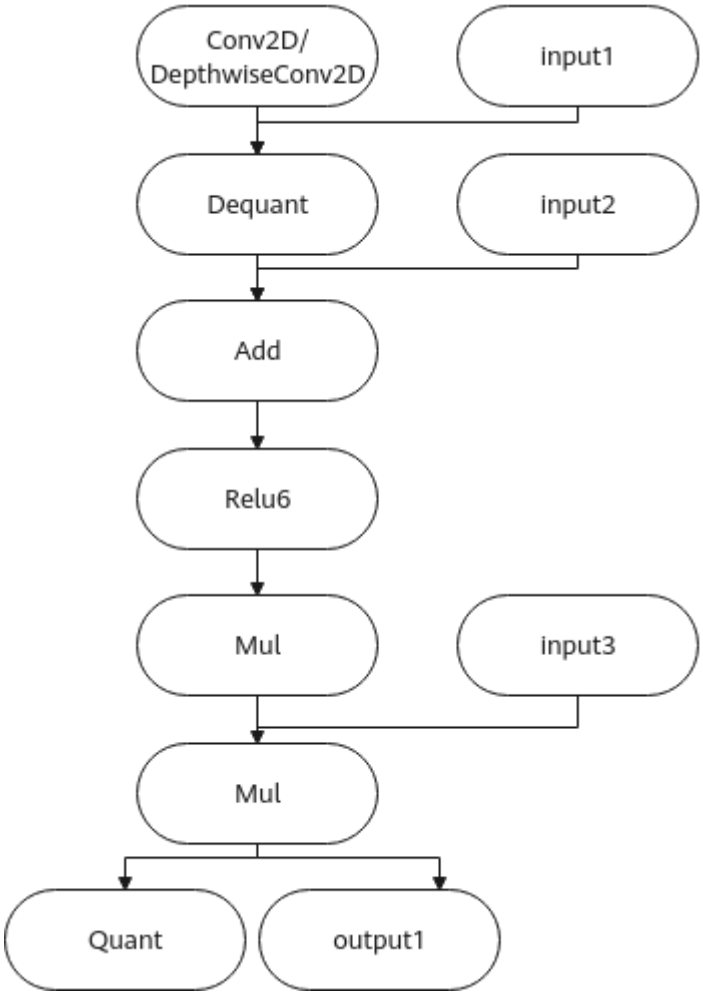
融合模式

支持以下两种融合模式

将Conv2D/DepthwiseConv2D + Add + Relu6 + Mul + Mul 融合成一个算子。



将Conv2D/DepthwiseConv2D + Dequant+ Add + Relu6 +Mul +Mul +Quant 融合成一个算子。



使用约束

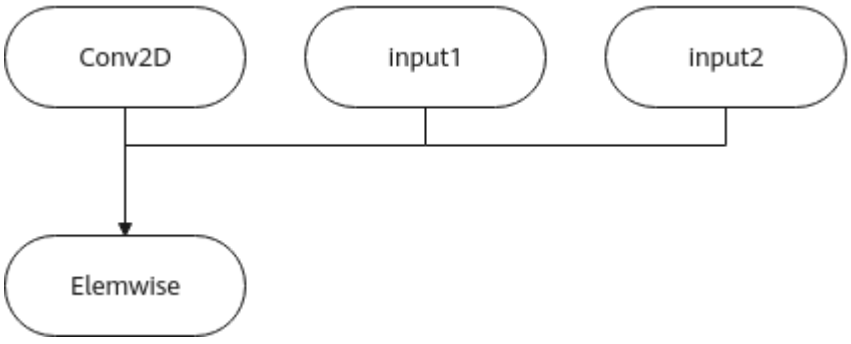
无

3.28 ConvClipByValueFusionPass

融合模式

支持以下融合模式

将Conv2D + Elemwise（ClipByValue类型）算子融合成1个融合算子。



使用约束

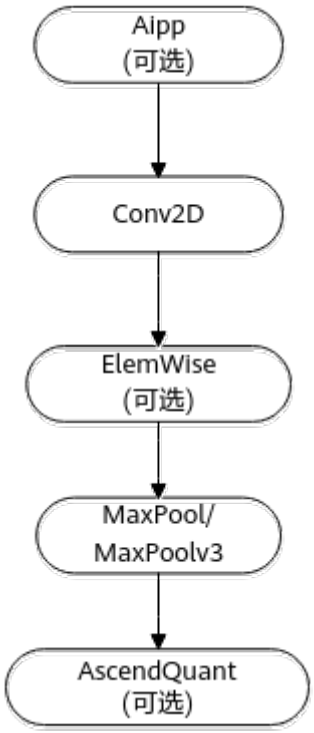
Elemwise算子必须是ClipByValue类型。

Conv2D算子的output channel需为16的整数倍大小。

3.29 TbeAippConvReluMaxpoolingFusion

融合模式

该融合将满足如下Pattern关系的子图中Aipp + Conv2D + ElemWise + MaxPool（MaxPool/Pooling/MaxPoolv3）+ AscendQuant算子融合成1个融合算子。



使用约束

满足如下条件才可以融合:

- Conv2D：small channel 使能, kernel需为3*3，5*5或7*7，strides需为[1, 1]或[2, 2]，cout ≤ 64。
- MaxPool：strides = [2, 2], ksize = [2, 2]/[3, 3]。
- 当MaxPool ksize = [2, 2]时，Conv2D input width超过1000不启用该融合。
- 当MaxPool ksize = [3, 3]时，Conv2D input width超过800不启用该融合。
- Pooling：strides = [2, 2], window= [2, 2]/[3, 3]。
- 当Pooling window = [2, 2]时，Conv2D input width超过1000不启用该融合。
- 当Pooling window = [3, 3]时，Conv2D input width超过800不启用该融合。

Maxpoolv3满足如下条件才可以融合:

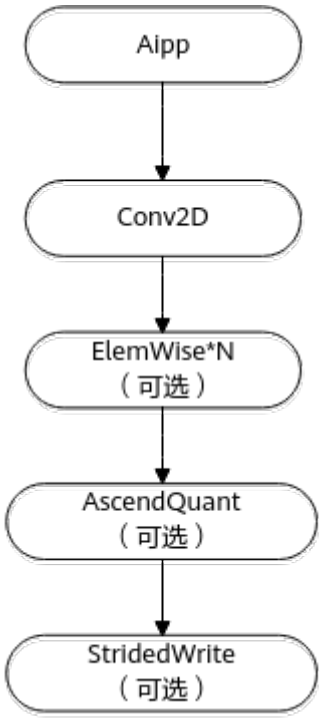
- soc：Ascend310P3
- conv2d：
  （1）输入参数的format为NCHW
  （2）fmap的shape为[N,3,224,224]，N为任意合法输入
  （3）filter的shape为[N,3,7,7]，N为1~96
  （3）pad的shape为[3,3,3,3]
  （4）strdie的shape为[N,N,2,2]，N为任意合法输入
  （5）dilation的shape为[N,N,1,1]，N为任意合法输入
  （6）group为1
- maxpoolv3:
  （1）输入参数的format为NCHW
  （2）strdie的shape为[N,N,2,2]，N为任意合法输入
  （3）ksize的shape为[N,N,3,3]，N为任意合法输入
  （4）padding_mode为CALCULATED
  （5）pad的shape为[1,1,1,1]
  （6）global_pooling为false
  （7）ceil_mode为false
- aipp:使能C04

3.30 TbeAippCommonFusionPass

融合模式

该融合将满足如下Pattern关系的子图中Aipp + Conv2D + ElemWise + AscendQuant + StridedWrite对应节点进行UB融合。

ElemWise可以有多个串连，最多支持5个。



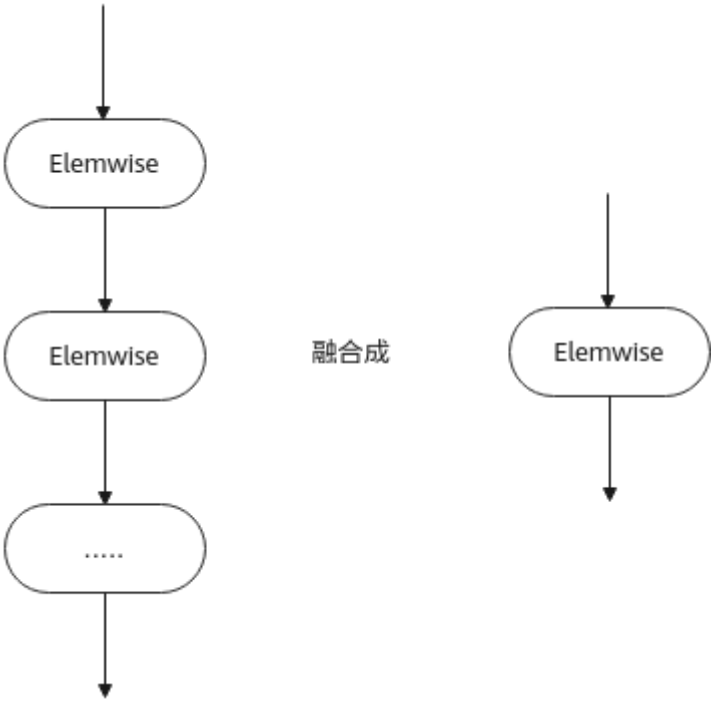
使用约束

- Conv2D算子， strides = [1, 1], pad = [0, 0, 0, 0], kernel 1*1 的场景不开启融合。
- AIPP开启resize，不开启融合。
- AIPP mode为dynamic，不开启融合。
- AIPP的input format设置为["RGB16", "RGB20", "RGB24","RGB8_IR", "RGB16_IR","RGB24_IR"]中的一种时，不开启融合。
- AIPP开启padding时，不开启融合。
- 若Conv2D算子使能DMA，不支持融合。
- 若给出最小Tiling，L1仍无法容纳Aipp处理结果，则放弃融合。
- 若卷积的kernal H小于或者等于上下任意一方向的Conv2d pad与AIPP pad之和，则放弃融合。

3.31 AutomaticBufferFusion

融合模式

经过其他UB融合之后，将未参与UB融合且相连的Elemwise类算子做UB融合。该融合优先级最低。



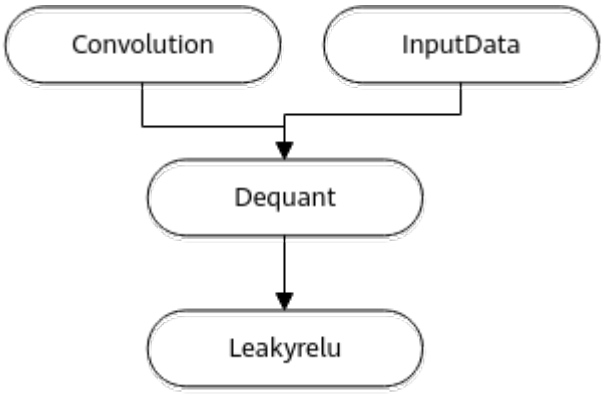
使用约束

- 只支持静态shape网络。
- 最多融合28个Elemwise类算子。
- 已经被其他UB融合规则匹配的算子不参与融合。
- Relu6Grad算子不参与融合。
- 输入数量超过6个的AddN算子不参与融合。

3.32 TbeSegmentElemwiseFusionPass

融合模式

该融合将ElemWise + Segment类算子做UB融合。



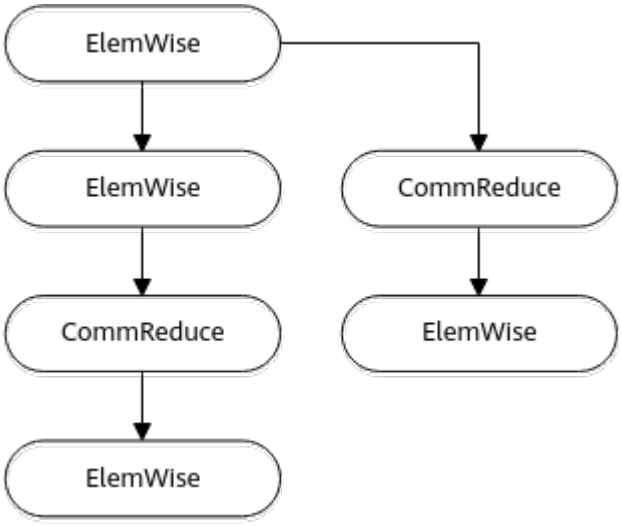
使用约束

无

3.33 TbeReduceElemwiseFusionPass

融合模式

该融合将ElemWise + CommReduce类算子做UB融合。



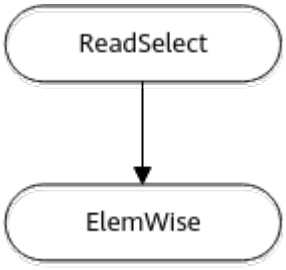
使用约束

无

3.34 TbeReadSelectEltwiseFusionPass

融合模式

该融合将ReadSelect + ElemWise算子做UB融合。



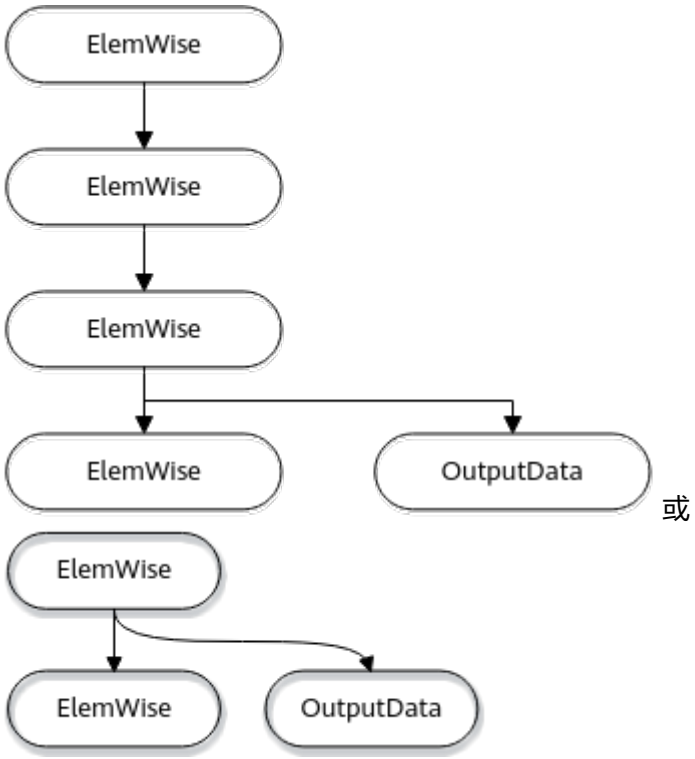
使用约束

无

3.35 TbeMultiOutputFusionPass

融合模式

该融合将满足如下Pattern关系的子图中ElemWise对应节点进行UB融合。



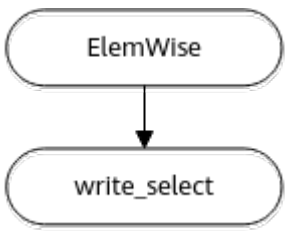
使用约束

无

3.36 TbeEltwiseWriteSelectFusionPass

融合模式

该融合将满足如下Pattern关系的子图中ElemWise + write_select对应节点进行UB融合。



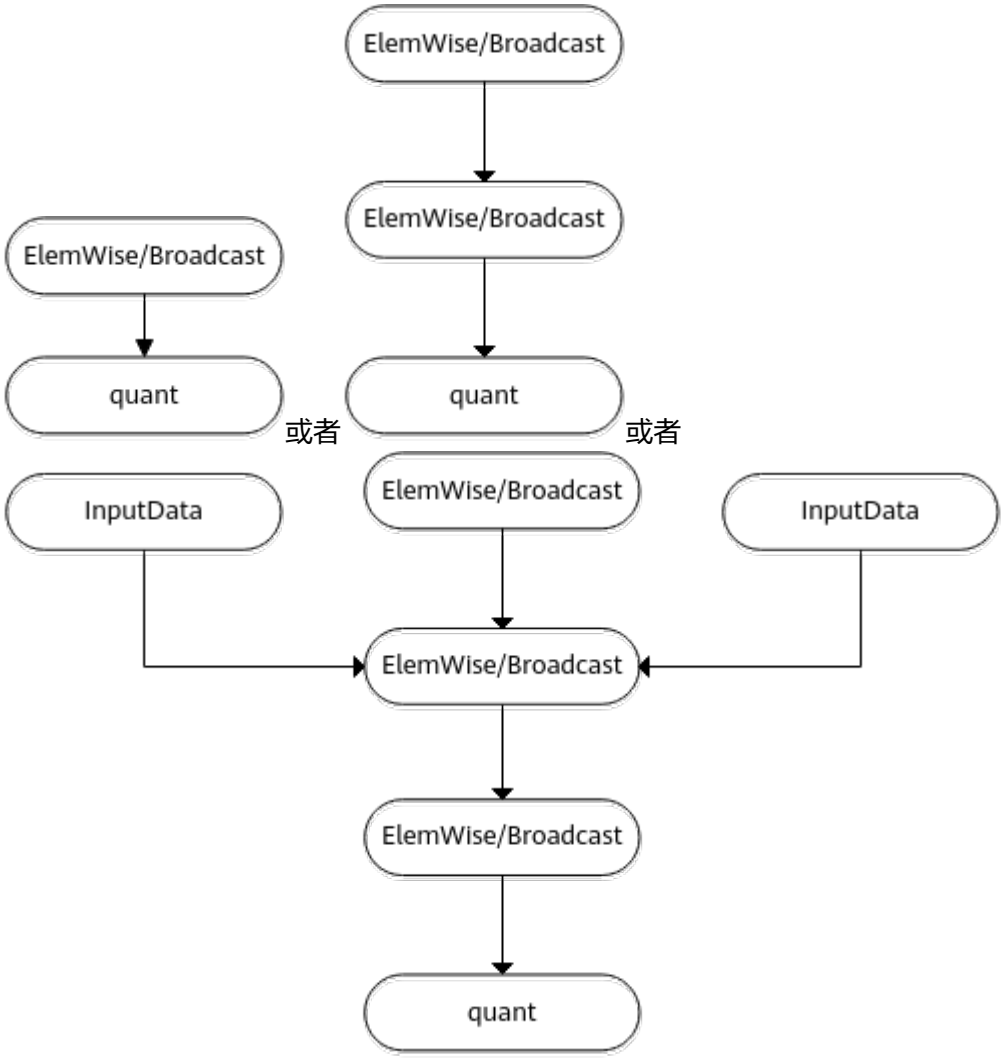
使用约束

无

3.37 TbeEltwiseQuantFusionPass

融合模式

该融合将满足如下Pattern关系的子图中ElemWise/Broadcast类和quant类节点进行UB融合。



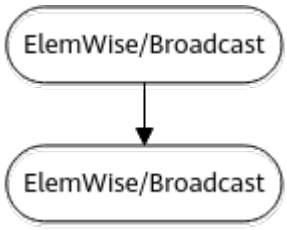
使用约束

ElemWise/Broadcast+ElemWise/Broadcast+quant场景中，ElemWise/Broadcast最多支持5个。

3.38 TbeEltwiseFusionPass

融合模式

该融合将满足如下Pattern关系的子图中ElemWise对应节点进行UB融合，ElemWise节点可以连续多个，最多支持5个



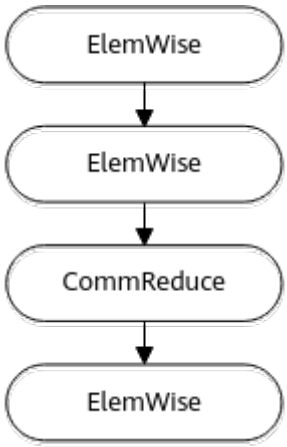
使用约束

无

3.39 TbeDynamicElemwiseReduceFusionPass

融合模式

该融合规则在动态shape场景下将如下pattern子图进行UB融合。



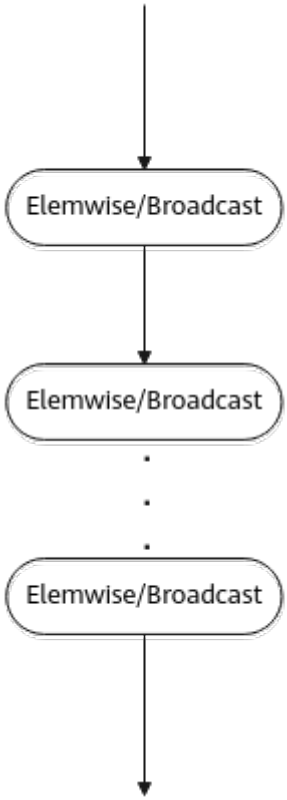
使用约束

无

3.40 TbeDynamicElemwiseBroadcastFusionPass

融合模式

该融合在动态shape场景下Elemwise/Broadcast类算子的UB融合，最多融合6个算子。



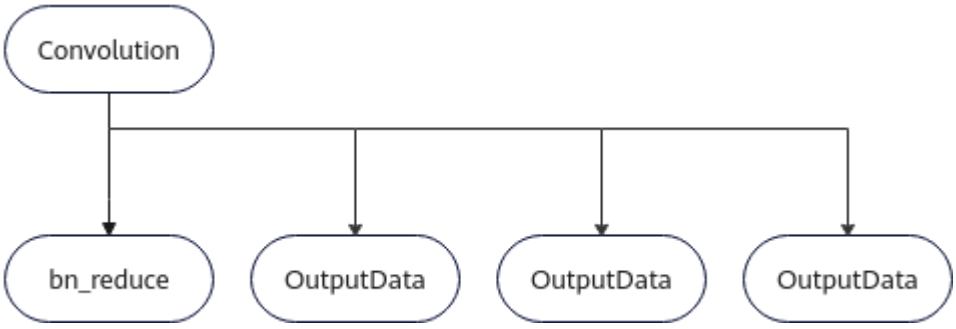
使用约束

无

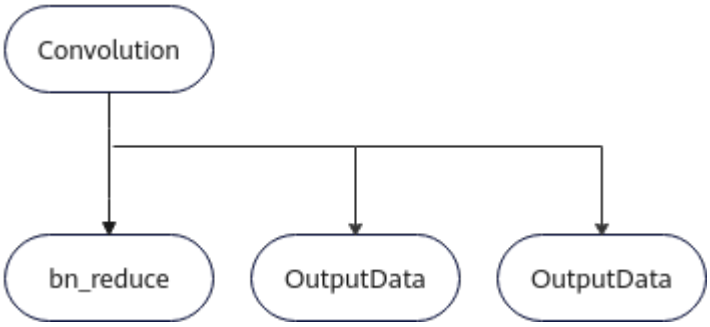
3.41 TbeConvBnreduceFusionPass

融合模式

该融合将满足如下Pattern关系的子图中Convolution + bn_reduce对应节点进行UB融合。



或者



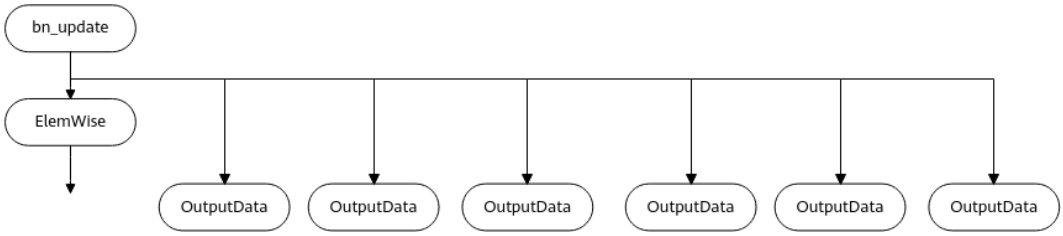
使用约束

无

3.42 TbeBnupdateEltwiseFusionPass

融合模式

该融合将满足如下Pattern关系的子图中bn_update + ElemWise对应节点进行UB融合。



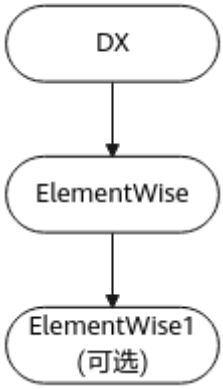
使用约束

无

3.43 TbeConv2DBackpropElemwiseFusionPass

融合模式

该融合将满足DX+ElementWise+ElementWise1进行UB融合。



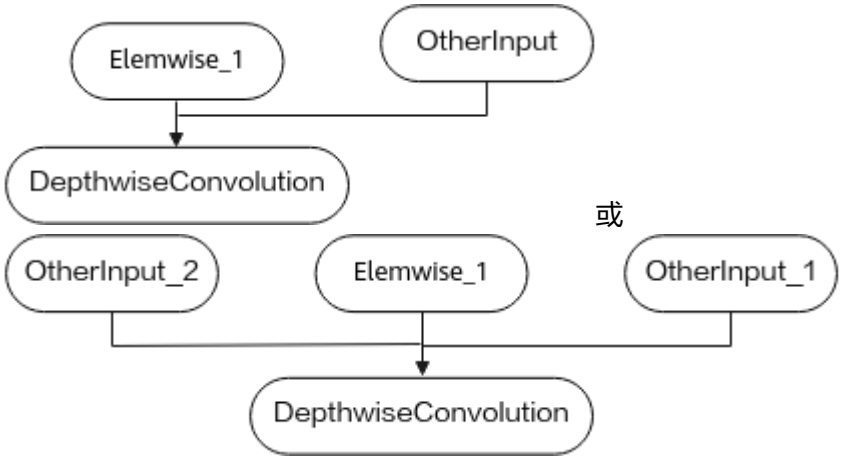
使用约束

- 仅ElementWise存在时，支持Add和ReluGradV2；当ElementWise1存在时，ElementWise支持AddN和Add，ElementWise1支持ReluGradV2。
- DX支持Conv2DBackpropInputD、Conv2DTransposeD和Deconolution。

3.44 TbeDepthwiseConvElemwiseFusionPass

融合模式

该融合将满足Transdata+Cube+AscendDequant/AscendQuant/AscendRequant+Elemwise+Transdata进行UB融合。



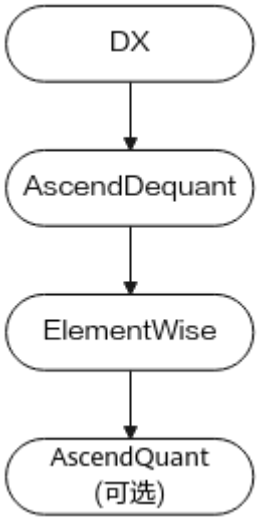
使用约束

Elemwsie支持LeakyRelu和Relu6；Elemwise_1仅支持系数为0的LeakyRelu。

3.45 TbeDxDeqElemQuantPass

融合模式

该融合将满足DX+AscendDequant + ElementWise + AscendQuant进行UB融合。



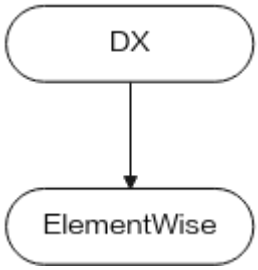
使用约束

- ElementWise支持LeakyRelu和Prelu。
- DX支持Conv2DBackpropInputD、Conv2DTransposeD和Deconolution。
- 当AscendQuant存在时，还支持ElementWise和AscendQuant双输出。

3.46 TbeDxElemwisePass

融合模式

该融合将满足DX+ElementWise进行UB融合。



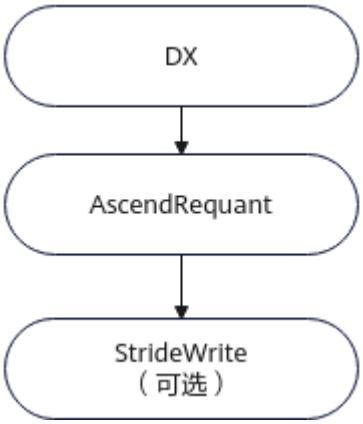
使用约束

- ElementWise支持LeakyRelu,Prelu。
- DX支持Conv2DBackpropInputD,Conv2DTransposeD,Deconolution。

3.47 TbeConv2dBackpropRequantFusionPass

融合模式

该融合将满足DX+AscendRequant+StrideWrite进行UB融合。



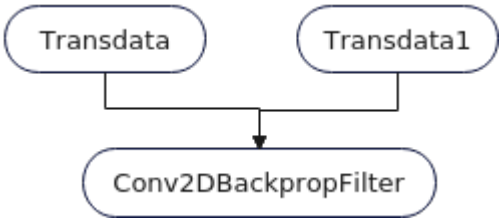
使用约束

DX支持Conv2DBackpropInputD、Conv2DTransposeD和Deconolution。

3.48 TbeDwTransdataFusionPass

融合模式

该融合在input输入到Conv2DBackpropFilter之前插入Transdata算子。



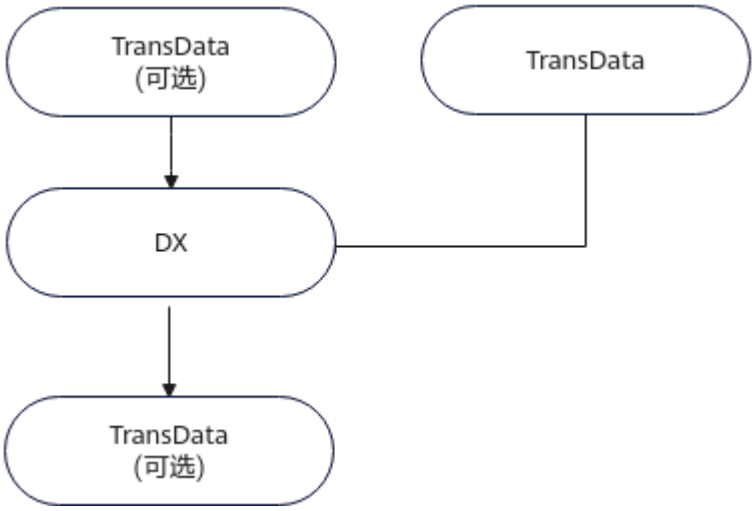
使用约束

- Transdata算子仅支持NCHW/NHWC到NC1HWC0。
- 仅在group=1, dilations=1,1,1,1时进行UB融合。

3.49 TbeDxTransdataFusionPass

融合模式

该融合将满足TransData+TransData+DX+TransData进行UB融合。



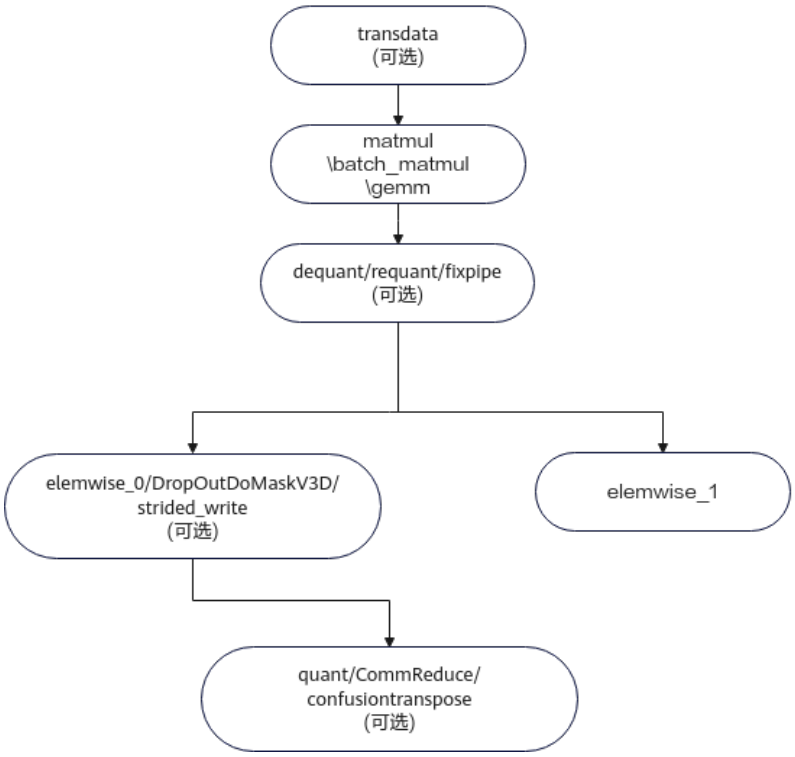
使用约束

只支持DX为Conv2DBackpropInput，并且属性groups=1, dilations={1,1,1,1}的动态场景。

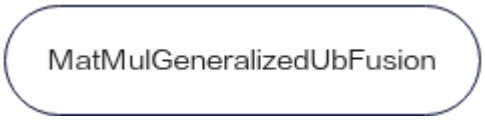
3.50 MatmulGeneralizedUbFusion

融合模式

该融合将满足Transdata+Matmul+ Dequant/Fixpipe/Requant +Elemwise+ Quant/CommReduce/Confusiontranspose 进行UB融合。



融合成



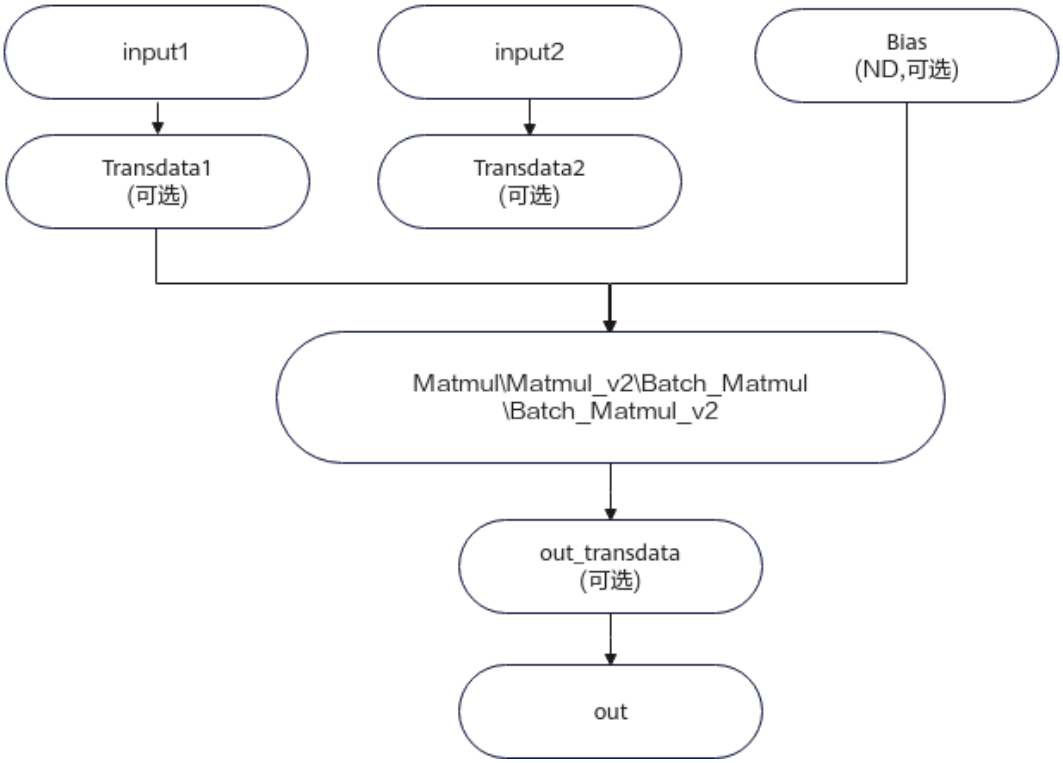
使用约束

无

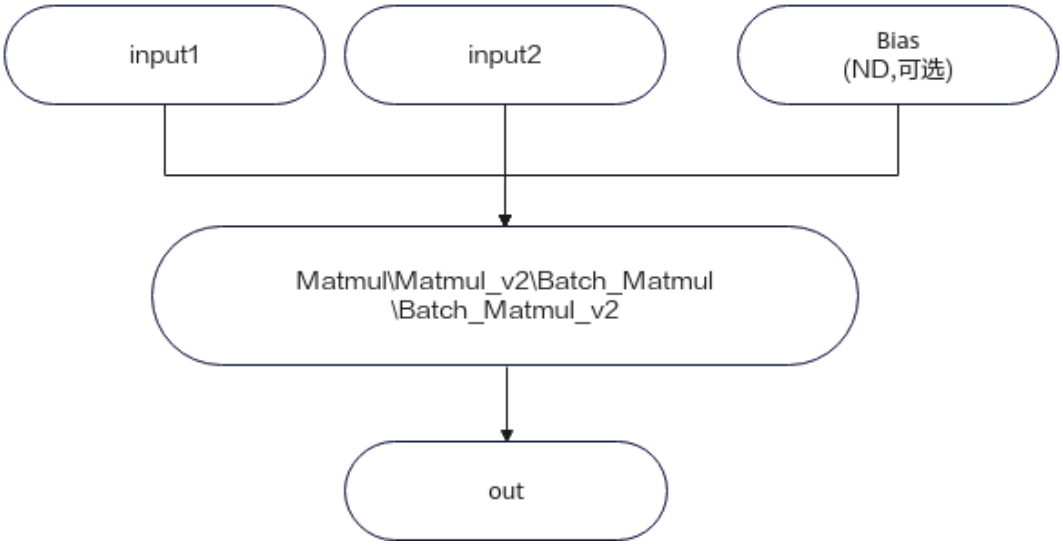
3.51 MatmulTransdataUbFusion

融合模式

该融合将Transdata+Matmul+Transdata进行UB融合。



融合成



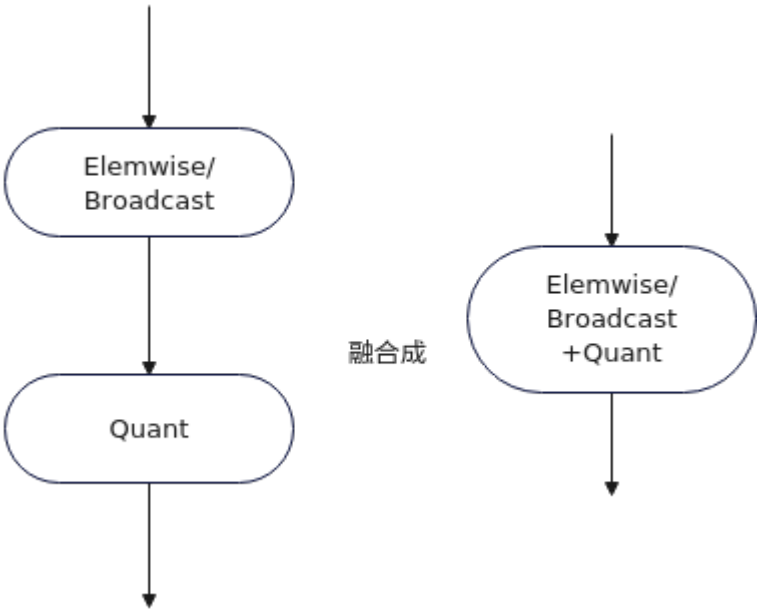
使用约束

- 至少两个输入，bias可选，必须为ND格式。
- Transdata1和Transdata2输入为ND，输出为NZ。out_transdata输入为NZ，输出为ND。
- 不支持静态和非对齐场景。
- 仅支持fp16进，fp16出。

3.52 TbeElemwiseQuantFusionPass

融合模式

该融合将Elemwise/Broadcast类算子和Quant算子进行UB融合。



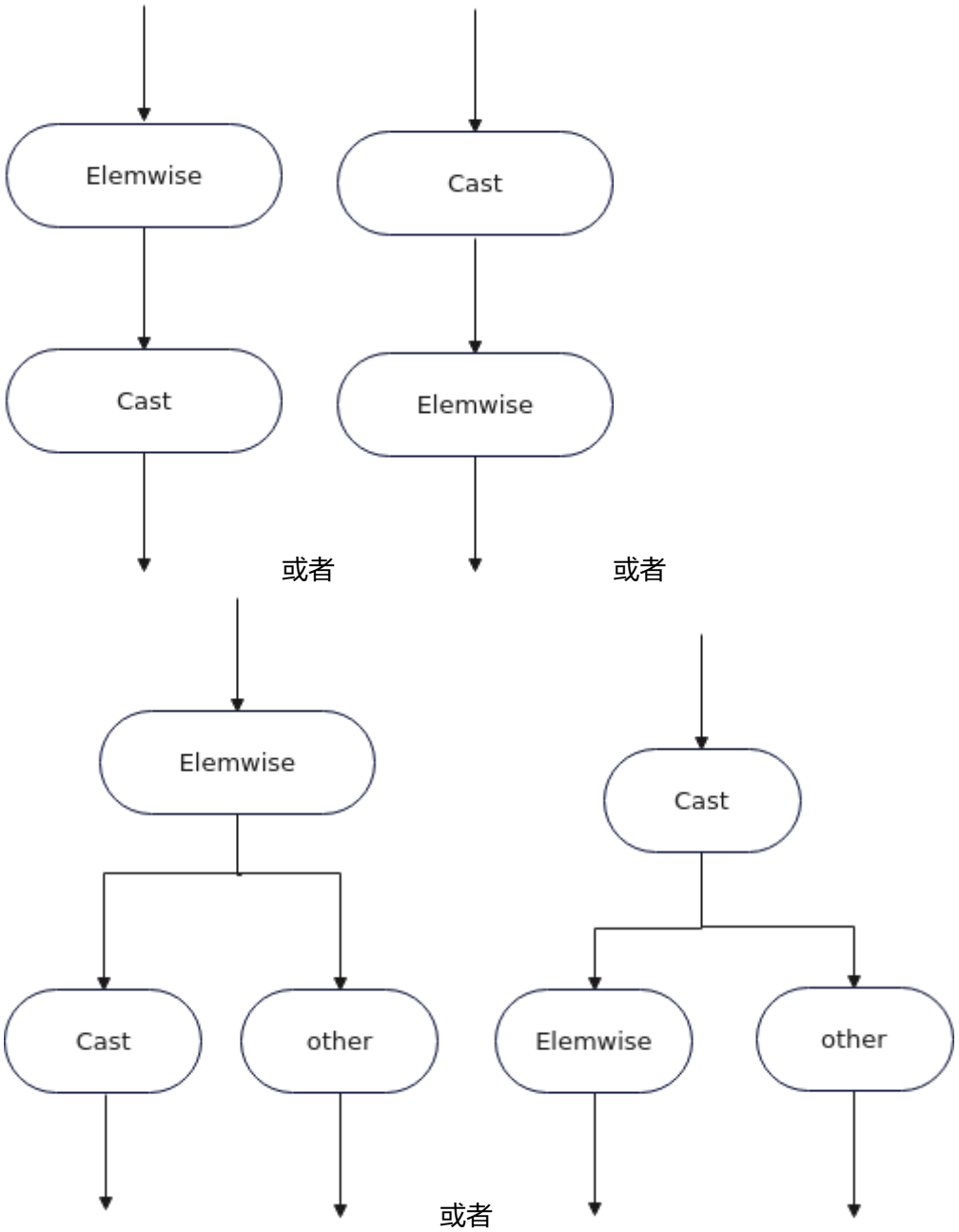
使用约束

- Elemwise/Broadcast算子类型不能是Eltwise。
- Elemwise/Broadcast算子必须是双输入。

3.53 TbeEltwiseCastFusionPass

融合模式

该融合将Elemwise类算子和Cast算子做融合，具体可以分为如下四种场景。



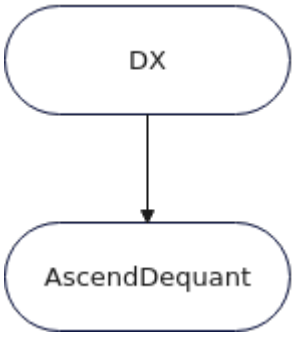
使用约束

- Cast算子必须是float16转float32或者float32转float16。
- Elemwise类算子类型是"Relu", "Add", "Mul", "Sqrt"中的一种。
- other算子不参与融合，只参与匹配。

3.54 TbeConv2DBackpropDequantFusionPass

融合模式

该融合将满足DX+AscendDequant进行UB融合。



使用约束

- DX类、AscendDequant为必选。
- DX支持Conv2DBackpropInputD、Conv2DTransposeD和Deconolution。

