tvm节点类型问题

在hw的fuse_op具体问题中所处理的NodeType为以下几种：
- FunctionNode
- CallNode
- VarNode
- ConstantNode

暂且不管用以辅助的例如OpNode或没有使用到的例如TupleNode。
其中FunctionNode和CallNode的关系没有怎么厘清，借此笔记边整理边思考。

CallNode class如下：

```
class CallNode : public ExprNode {
 public:
  /*!
   * \brief The operator(function) being invoked 被function调用的operator
   *
   *  - It can be relay::Op which corresponds to the primitive operators.
   *  - It can also be user defined functions (Function, GlobalVar, Var).
   *  - 可以是元算子，也可以是用户自定义functions
   */
  Expr op;

  /*! \brief The arguments(inputs) of the call */
  //该call的输入参数，见如下图
  tvm::Array<relay::Expr> args;

  /*! \brief The additional attributes */
  Attrs attrs;
  ...
```

FunctionNode class如下：

```
class FunctionNode : public BaseFuncNode {
 public:
  /*! \brief Function parameters */
  tvm::Array<Var> params;
  /*!
   * \brief
   * The expression which represents the computation of the function,
   * the expression may reference the parameters, and the type of it
   * or sub-expressions may reference the type variables.
   * 表示函数计算的表达式，表达式可以引用参数，它或子表达式的类型可以引用类型变量。
   */
  Expr body;
  /*! \brief User annotated return type of the function. */
  Type ret_type;
  /*!
   * \brief Type parameters of the function.
   *  Enables the function to vary its type based on these.
   *  This corresponds to template paramaters in c++'s terminology.
   *
   * \note This can be usually empty for non-polymorphic functions.
   */
  tvm::Array<TypeVar> type_params;

  /*!
   * \brief The attributes which store metadata about functions.
   */
  tvm::Attrs attrs;
```

```
GROUP[6] ROOT:0x6ce9ff0 Op(nn.leaky_relu)
7 6:1
new_args are: Var(p0, ty=TensorType([1, 1024, 13, 13], float32))
7 7:0
7 7:0
new_args are: CallNode(Op(nn.pad), [Var(p0, ty=TensorType([1, 1024, 13, 13], float32))], relay.attrs.PadAttrs(0x6c35768), [TensorType([1, 1024, 13, 13], float32)])
new_args are: Constant([[[[ 3.19110951e-03  8.65325145e-03 -4.04290743e-02]
   [-6.63737580e-02 -6.08754754e-02 -5.59197441e-02]
   [-1.82669330e-02 -1.84959767e-03 -2.14269683e-02]]

  [[ 2.11132839e-02  4.70295101e-02  6.11791052e-02]
   [ 7.82777444e-02  4.97723222e-02  5.27830832e-02]
   [-1.67056941e-03  1.59923211e-02  1.78829078e-02]]

  [[ 6.27283975e-02 -9.14163068e-02 -5.42042404e-02]
   [ 2.04739477e-02  1.16414472e-01  3.87375355e-02]
   [ 7.95943514e-02  2.34684777e-02  6.18486665e-02]]

  ...

  [[ 2.97288410e-02 -2.78714821e-02  1.83622371e-02]
   [ 2.91654915e-02 -2.40816623e-02 -4.40047728e-03]
   [-1.41656790e-02 -3.18748020e-02 -1.72210820e-02]]

  [[ 5.22858044e-03  2.81296000e-02 -1.29548004e-02]
   [ 4.23780158e-02 -9.78771481e-04  1.58469018e-03]
   [-6.74006045e-02 -7.52917752e-02 -5.20187579e-02]]

  [[-4.33363812e-03 -4.05620262e-02  3.14398222e-02]
   [-9.78523679e-03 -2.80521289e-02  3.39381397e-03]
   [-6.44844249e-02 -5.86274406e-03 -7.98369721e-02]]]
```

对fuse_op patition的补充： group->rootref是整个group最后一个node，后序访问。

> GROUP[1] ROOT:0x9bac430 Op(nn.max_pool2d)

对每个节点的访问，找到其所在的group，如果内部arg所在的group与该节点group不同，则将该arg group中所有节点的args表线性合并，替换该节点

明天再说 看晕了

图形学部分

```
Intersect(Ray ray, BVH node){
    if (ray misses node.bbox) return;

    if (node is a leaf node){
        test intersection with all objs;
        return closest intersection;
    }

    hit1 = Intersect(ray, node.child1);
    hit2 = Intersect(ray, node.child2);

    return the closer of hit1, hit2;
}
```