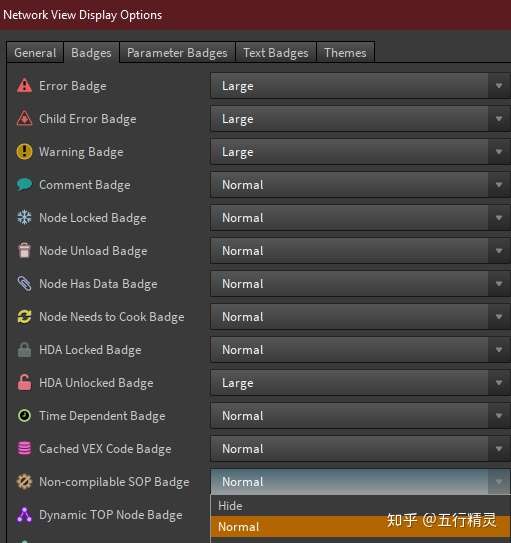
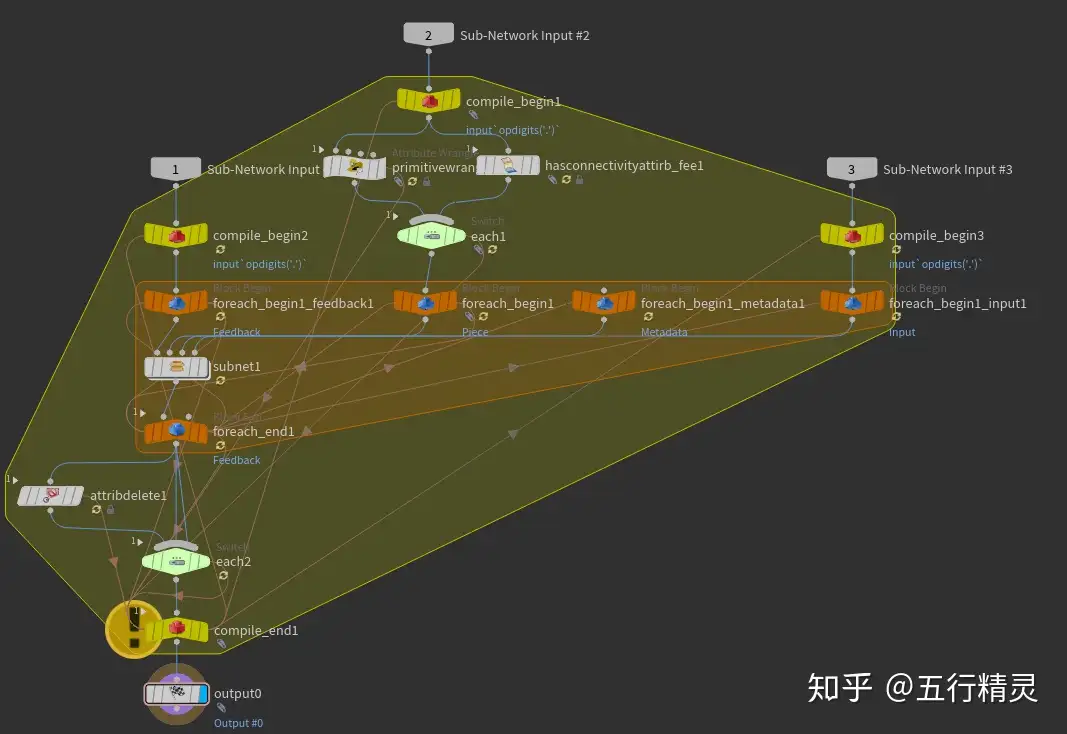
## **一：首先我们要看一个节点能否compile**

鼠标放在network editor的SOP模块里，默认[快捷键](https://zhida.zhihu.com/search?q=%E5%BF%AB%E6%8D%B7%E9%94%AE&zhida_source=entity&is_preview=1" \t "/Users/wakikoera/Documents\\x/_blank)按D，然后选择下面这个

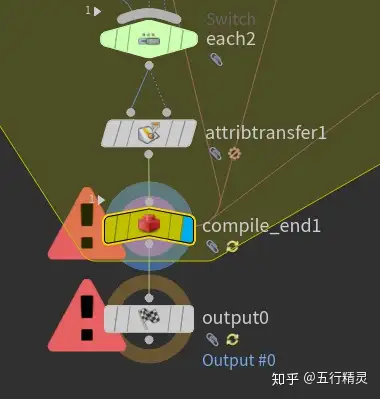


然后以FeE For Each Filter Piece SOP为例

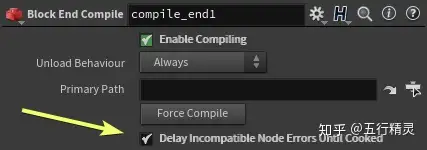
compile block内不能有non compilable节点，如下，没有一个节点带小齿轮



如下，Attribute Transfer有小齿轮，放在compile block的**计算流程**内就会报错

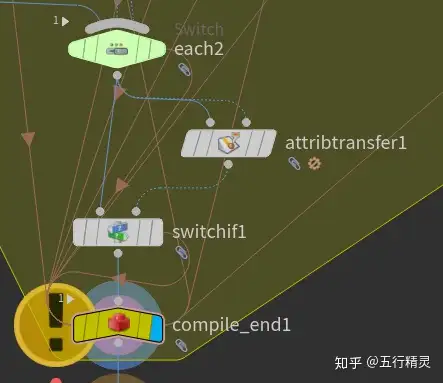


注意是计算流程内，H16.5后，如果勾上这个选项



然后整个计算中用不到这个Attribute Transfer，如下，也可以compile

注：虚线在houdini一般代表reference，switch SOP中意味着ref不会被cook



## **二：涉及属性的连线（蓝线和紫线）都不能越过区块**

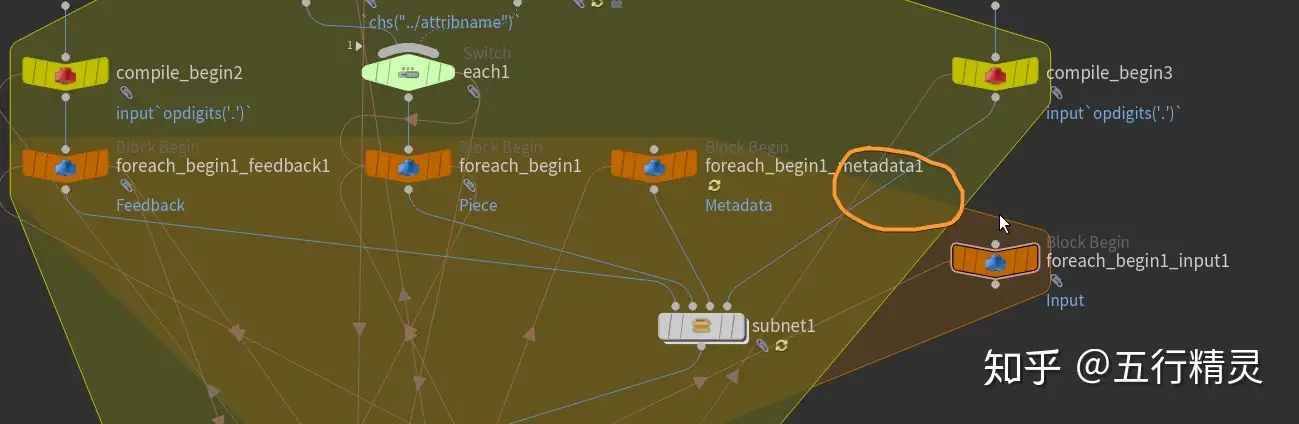
**离题技巧：**

**在视图中显示橙色连线（代表这两个节点间的参数有关联**

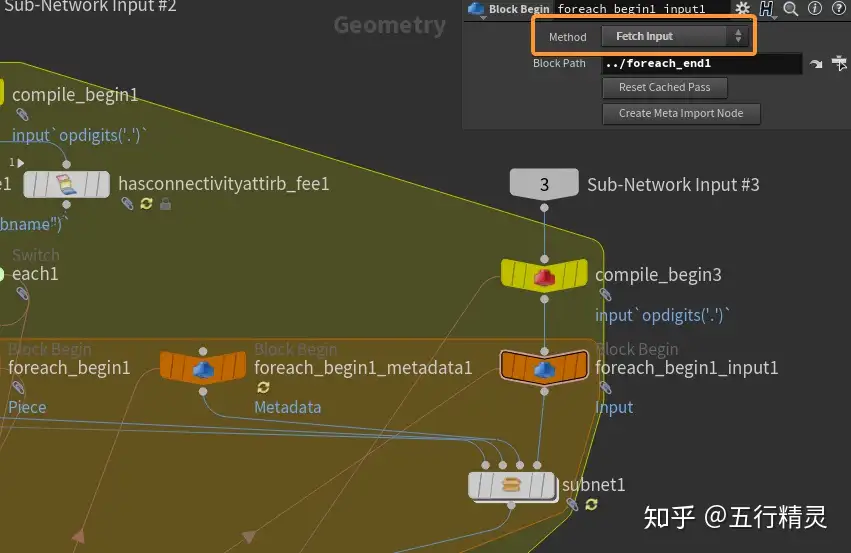


**回到题目**

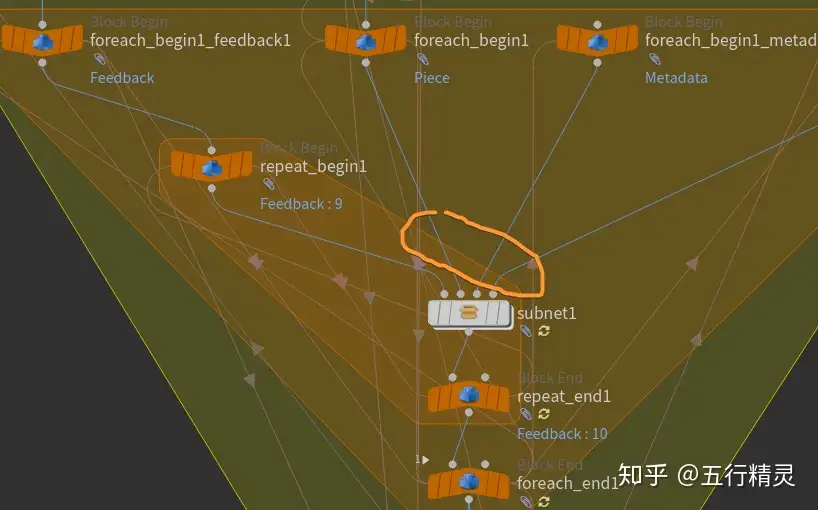
**例如蓝线，这里就跨区块**了



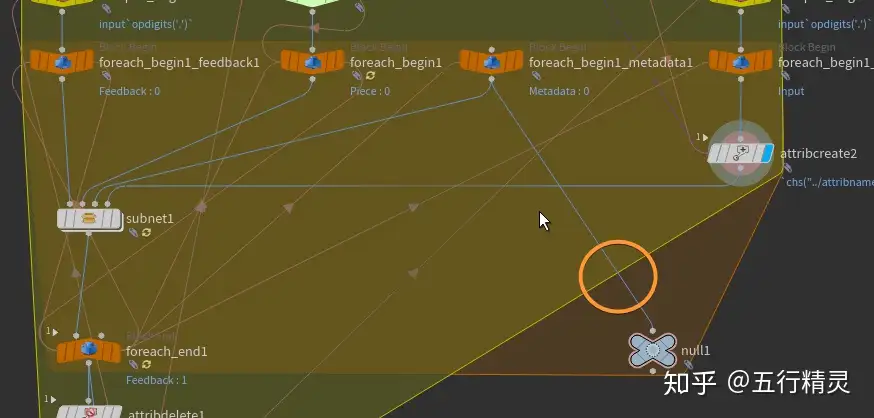
**所以我们要这样**



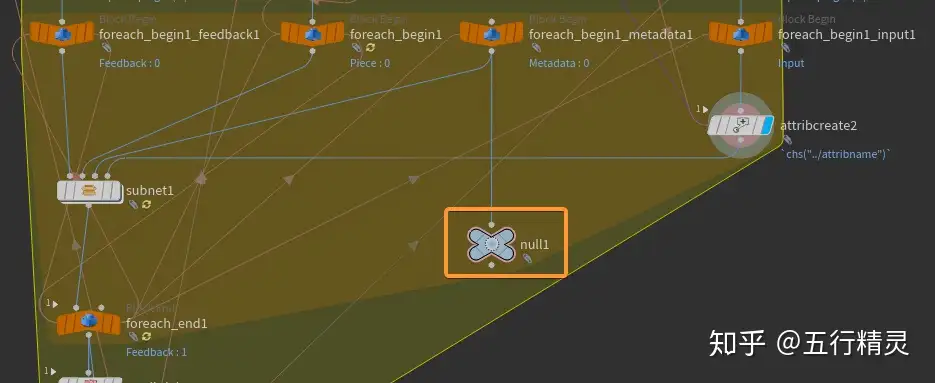
这样也算跨区块



这样不算跨区块



否则难道像下面这样这样移动一下节点，就能变成不算跨区？就可以compile了？



## **然后就是属性引用**

**我们要引用别的节点的属性时，一般是这样写**

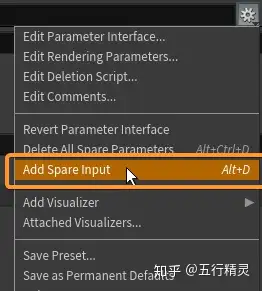
IMG_269

如果我们有两个参数都引用了这个box1，之后我们要改成box2，要改多次就很麻烦

IMG_270

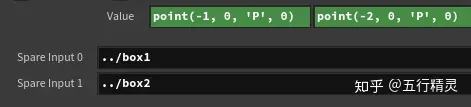
**所以我们应该这样**

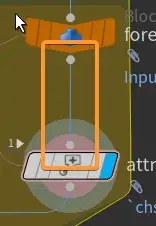
**点参数面板右上角的齿轮（**建议添加快捷键



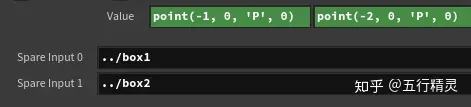
然后spare input的性质，其实就相当于蓝线那样，直接把box1 SOP和box2 SOP连在这个节点上，然后序号按照-1， -2， -3这样排下去，和蓝线连接的排序0， 1， 2相反

注：蓝线连接就是下下图的那条线

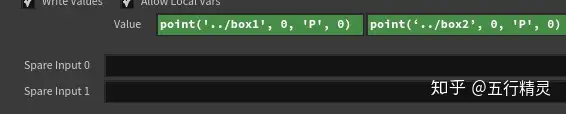




**spare input这样写**



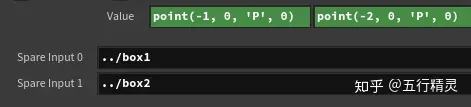
**其实就相当于**



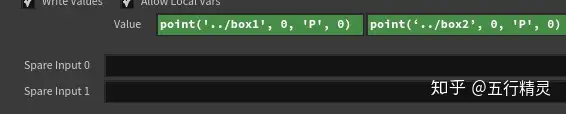
## **然后就是compile的规则**

## **所有属性引用都必须要用spare input**

**也就是都得这样写**



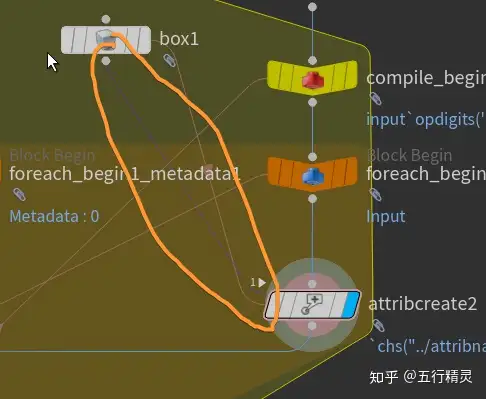
**不能这样写**



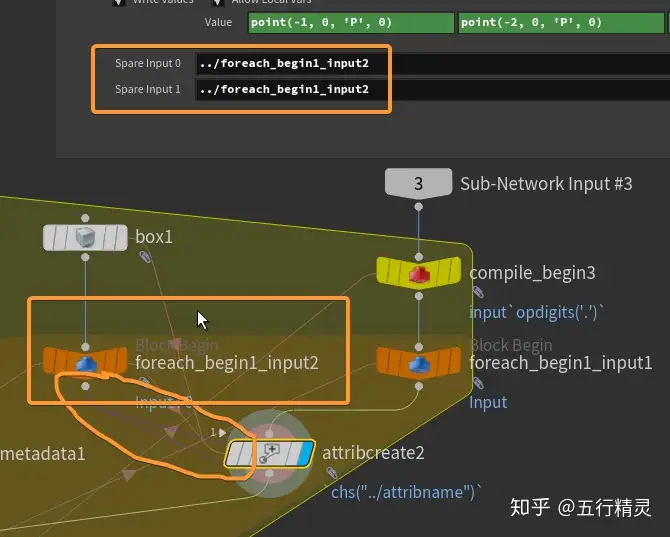
**也不能这样写**

IMG_278

**然后紫线（其实就是spare input的引用线）也不能跨区块，如下就是错误**



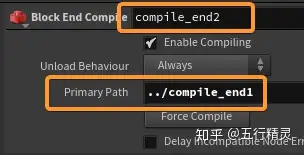
**得这样**



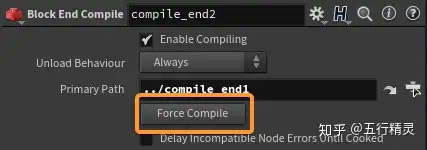
## **这样做看似麻烦，其实也是限定了引用不会写得乱七八糟，对可读性提升也好**

## **很多新手引用乱七八糟（尤其是copy stamp时期）可读性非常差**

## **compile是可以有多个输出口的，建立多个block end SOP**

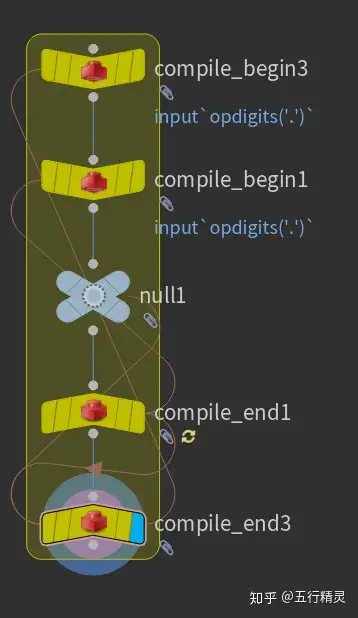


## **compile的输出结果好像没更新？点这儿~**



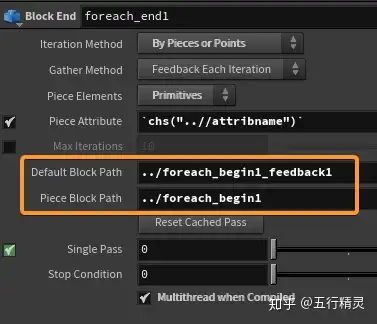
**注：如果compile block在锁定的HDA里，是不会出现不更新问题的**

**compile可以嵌套**



## **最后就是loop block内，如果同时有fetch feedback和fetch pieces**

block end得这么写



最后最后是一个小tips，可以用invoke SOP调用compile blcok计算，这个作用不大，以后无聊可能会写文章吧

