

索引文件的读取（三）（Lucene 8.4.0）

本文承接[索引文件的读取（二）](#)之dim&&dii继续介绍剩余的内容，下面先给出读取索引文件.dim&&dii的流程图：

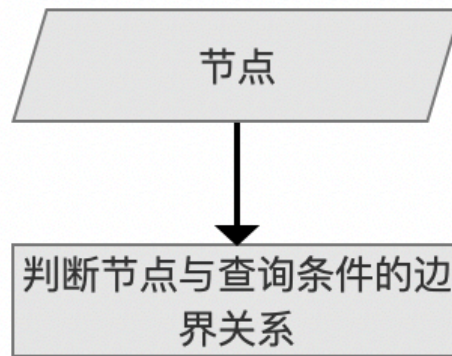
图1：

[点击查看大图](#)

读取索引文件.dim&&dii

判断节点与查询条件的边界关系

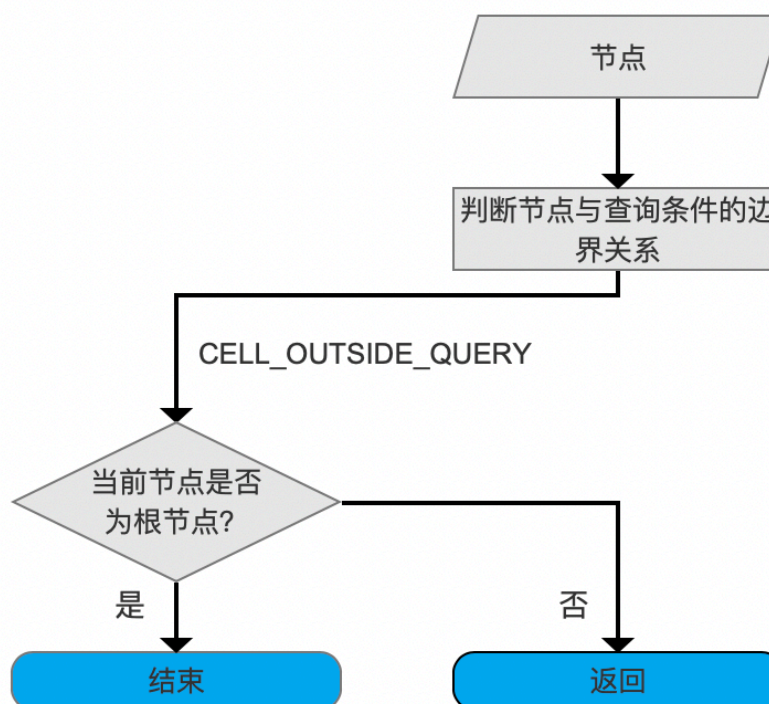
图2：



当前节点与查询条件的边界关系即CELL_INSIDE_QUERY、CELL_OUTSIDE_QUERY、CELL_CROSSES_QUERY共三种关系（见文章[索引文件的读取（一）之dim&&dii](#)）。

CELL_OUTSIDE_QUERY

图3:



如果当前节点为根节点，并且根节点中的点数据的数值范围跟查询条件的数值范围的边界关系为CELL_OUTSIDE_QUERY，意味着BKD树中没有任何点数据满足查询条件，此时就可以直接跳过（skip）当前点数据域的查询，故可以结束BKD树的读取。

如果当前节点不是根节点，不管它是内部节点（非叶节点）还是叶子节点，我们都不用继续读取这个节点的信息以及以这个节点为祖先节点的所有子节点，故返回到上一层递归继续处理。

CELL_INSIDE_QUERY

图4:

如果当前节点为根节点，并且根节点中的点数据的数值范围跟查询条件的数值范围的边界关系为 CELL_INSIDE_QUERY，意味着BKD树中所有的点数据都满足查询条件，那么直接通过深度遍历统计BKD树中的所有叶子节点中的文档信息，注意的是在深度遍历的过程中当遍历到一个内部节点时，只需要获取该节点的左右子树在索引文件.dim中位置即可，直到遍历到叶子节点，最后统计所有叶子节点中点数据对应的文档号，最后退出。

另外要提下的是，在图1的流程点 是否段中的文档都满足查询条件? 中，满足的条件之一是 BKD树中的点数据数量跟段中的文档数量一致，当这个前提不满足时并且根节点中的点数据的数值范围跟查询条件的数值范围的边界关系为CELL_INSIDE_QUERY，就会执行图4的流程。

如果当前节点是内部节点，处理的方式跟根节点的是一样的，差别在于处理完当前节点后是返回到上一层递归。

如果当前节点是叶子节点，那么读取该节点中的文档信息后返回到上一层递归。

在上面的描述中，我们至少会有以下两个疑问：

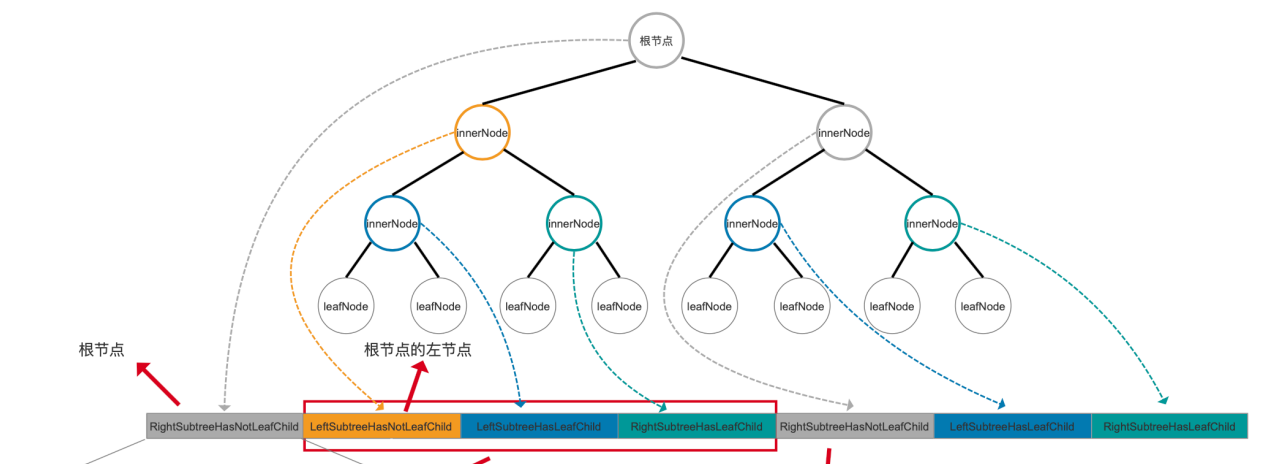
- 问题一：如何获得当前节点的左右子树节点在索引文件.dim中的起始读取位置
- 问题二：如何读取叶子节点中的文档信息

如果你没有看过文章[索引文件之dim&&dii](#)，建议跳过这一段。

问题一：如何获得当前节点的左右子树节点在索引文件.dim中的起始读取位置

由于BKD树的遍历方式为深度遍历，故我们以根节点为例来作介绍：

图5：



[点击查看大图](#)

在文章[索引文件的生成（十四）之dim&&dii](#)中我们介绍了描述节点（内部节点/叶子节点）的信息在索引文件.dim中的位置，由于生成BKD树也是深度遍历的方式，那么读取完根节点的信息后，索引文件.dim的下一个字节就是左子树（左节点）的起始读取位置，对于右节点，描述根节点信息的RightSubtreeHasNotLeafChild字段中的leftNumBytes值，它描述的是根节点的所有子孙节点（不包括叶子节点）在索引文件.dim占用的字节数，那么通过leftNumBytes的值，就可以找到根节点的右子树在索引文件.dim中的起始读取位置。

在图5中，读取完根节点的信息，即RightSubtreeHasNotLeafChild之后，索引文件.dim的下一个字节就是根节点的左子树信息的起始读取位置，随后跳过leftNumBytes个字节，就是根节点的右子树信息的起始读取位置。

问题二：如何读取叶子节点中的文档信息

在文章[索引文件的读取（一）之dim&&dii](#)中我们了解到，通过读取索引文件.dii的IndexFP字段，获取的信息是内部节点（非叶节点）信息在索引文件.dim中的起始读取位置，在索引文件.dii中并没有其他的提供叶子节点信息在索引文件.dim中的位置，如下所示：

图6：

[点击查看大图](#)

实际上叶子节点信息在索引文件.dim中的起始读取位置是通过叶子节点的父节点获得的，在文章[索引文件之dim&&dii](#)中，我们介绍了存储内部节点（非叶节点）信息一共有四种数据结构：

- 非叶节点的子树是叶子节点，并且它是父节点的左子树：LeftSubtreeHasLeafChild
- 非叶节点的子树是叶子节点，并且它是父节点的右子树：RightSubtreeHasLeafChild
- 非叶节点的子树不是叶子节点，并且它是父节点的左子树：LeftSubtreeHasNotLeafChild
- 非叶节点的子树不是叶子节点，并且它是父节点的右子树：RightSubtreeHasNotLeafChild

RightSubtreeHasLeafChild

RightSubtreeHasLeafChild描述的是当前内部节点的左右子树都是叶子节点，并且该节点是父节点的右子树，RightSubtreeHasLeafChild中通过LeftLeafBlockFP、RightLeafBlockFP分别存放了它的左右节点信息在索引文件.dim中的起始读取位置，如下图所示：

图7：

[点击查看大图](#)

LeftSubtreeHasLeafChild

LeftSubtreeHasLeafChild描述的是当前内部节点的左右子树都是叶子节点，并且该节点是父节点的左子树，通过LeftSubtreeHasLeafChild中的RightLeafBlockFP字段获得右子树在索引文件.dim中的起始读取位置，并且通过该节点的某个祖先节点传递左子树在索引文件.dim中的起始读取位置。

哪个祖先节点传递左子树在索引文件.dim中的起始读取位置

该祖先节点满足的要求条件是：该节点是它的父节点的右子树（右节点），即节点的数据结构为RightSubtreeHasNotLeafChild，并且是最近的。例如下图中，传递左子树在索引文件.dim中的起始读取位置的祖先节点就是根节点，存储根节点的数据结构正是RightSubtreeHasNotLeafChild，在深度遍历的过程中，根节点对应的RightSubtreeHasNotLeafChild中的LeftLeafBlockFP一层一层往下传递，如下图所示：

图8：

[点击查看大图](#)

为了便于理解，我们再给出另一个数据结构类型为LeftSubtreeHasLeafChild的节点：

图9：

[点击查看大图](#)

在介绍了如何获得叶子节点的信息在索引文件.dim中的起始读取位置之后，我们就可以开始读取叶子节点中的文档号信息了，基于篇幅，读取文档号的逻辑将在介绍图1的流程点 收集叶子节点中满足查询条件的文档号 时展开。

结语

无

[点击下载附件](#)