

doc文件

索引文件doc中按块（block）的方式存放了每一个term的文档号、词频，并且保存了skip data来实现块之间的快速跳转。

doc文件的数据结构

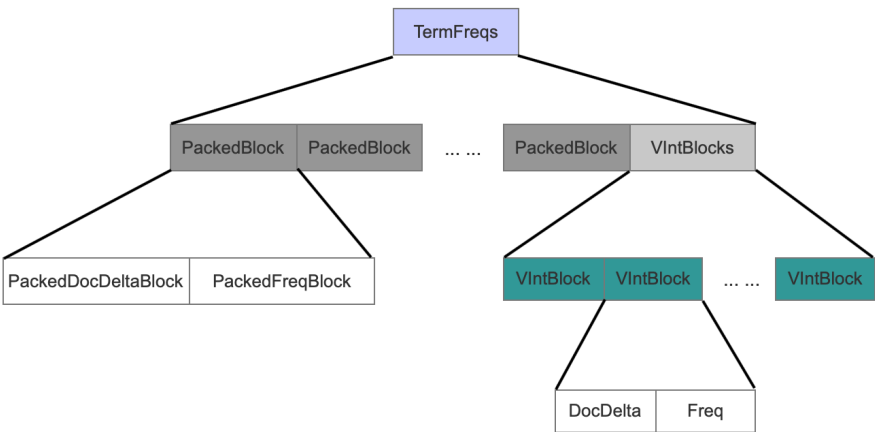
图1：



TermFreqs保存了term的所有文档号、词频信息，TermFreqs中按块存储，所以SkipData实现这些块之间的快速跳转。

TermFreqs

图2：



PackedBlock

每处理包含term的128篇文档，就将这些文档的信息处理为一个PackedBlock。

PackedDocDeltaBlock

PackedDocDeltaBlock存放了128篇文档的文档号，计算相邻两个文档号的差值后，利用PackedInts压缩存储。

PackedFreqBlock

PackedFreqBlock存放了term分别在128文档中的词频，利用PackedInts压缩存储。

这里注意是由于在每篇文档中的词频值无法保证递增，而在使用PackedInts时会用统一大小的字节数来存放每一个词频值，并且这个字节数取决于词频最大的那个。

VIntBlocks & VIntBlock

如果包含term的文档号不足128个，那么将这些文档的信息处理为一个VIntBlocks。(比如包含term的文档数量有200，那么前128篇文档的信息被处理为一个PackedBlock，剩余的72篇文档处理为72个VIntBlock，72个VIntBlock为一个VIntBlocks)

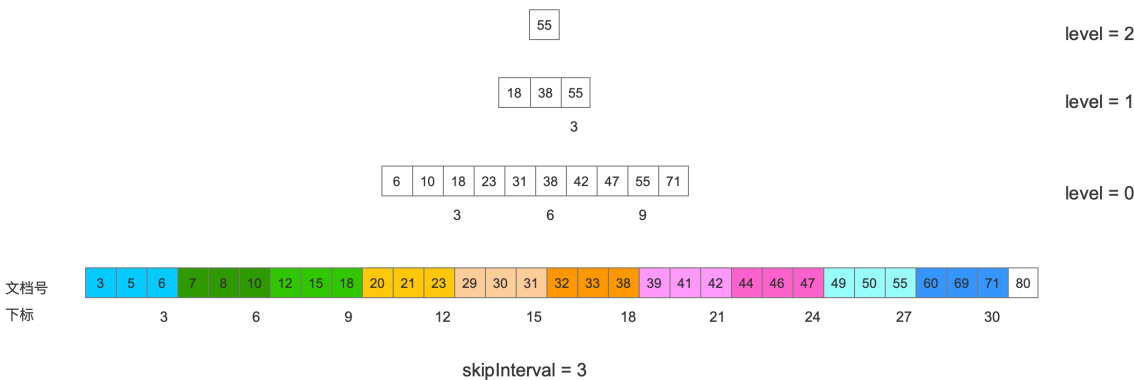
DocDelta

当前文档号跟上一个文档号的差值。

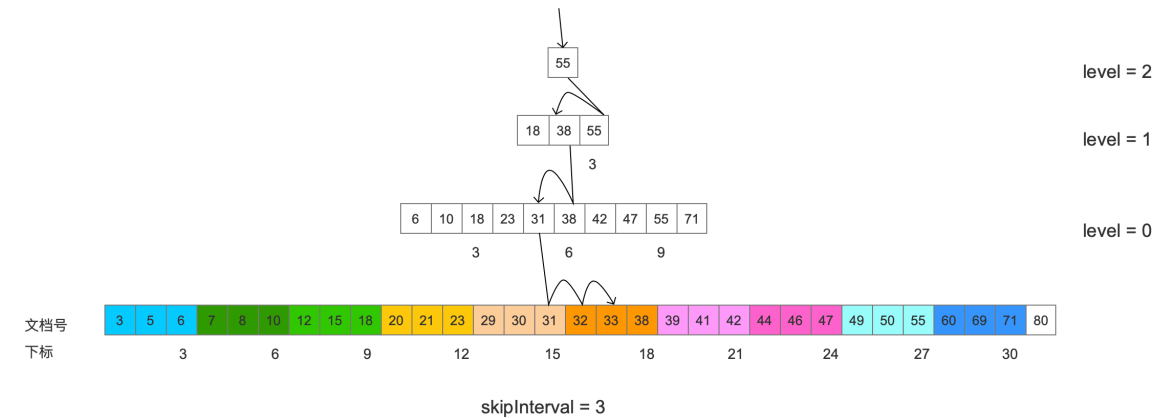
Freq

term在当前文档中的词频。

在介绍SkipData前先介绍下跳表（SkipList）的概念 图3：

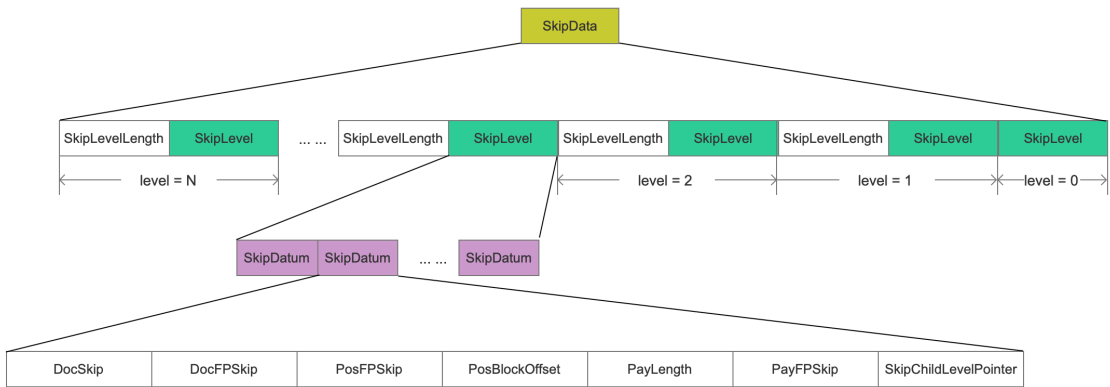


在每一层中，每skipInterval个数据块就会在上一层中添加一个索引，实现了对数级别的时间复杂度。比如我们要找文档号为33的值。那么查找的过程如下图。图4：



SkipData

图5:



SkipLevelLength

当前层的跳表（skipList）数据长度，在读取的时候用来确定往后读取的一段数据区间。

SkipLevel

SkipLevel描述了当前层中的所有跳表真实数据

SkipDatum

当前层中每一个跳表信息按块处理为一个SkipDatum。

DocSkip

描述了当前SkipDatum指向的文档号，不过DocSkip的实际值是当前文档号与上一个SkipDatum的文档号差值，还是使用了差值存储。

DocFPSkip

每当处理128篇文档，在level = 0的跳表中就会生成一个SkipDatum，而DocFPSkip指向的就是存储这128篇文档的PackedBlock的起始位置。

PosFPSkip

PosFPSkip指向了.pos文件中一个位置。这个位置是PackedPosBlock(每128个position信息处理为一个PackedPosBlock)的起始位置

PosBlockOffset

PosBlockOffset描述的是上一条说的PackedPosBlock中的一个偏移位置。

PayLength

PayLength描述的是在.pay文件中的payload的信息，这段payload的信息跟上一条中位置信息是对应的。

PayFPSkip

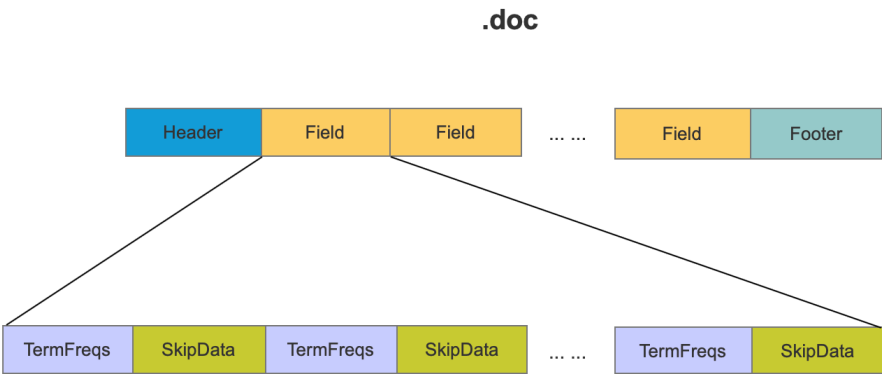
PayFPSkip指向了.pay文件中一个位置。这个位置是PackedPayBlock(每128个offset信息处理为一个PackedPayBlock)的起始位置。

SkipChildLevelPointer

如果当前SkipDatum属于大于level = 0的较高层，那么SkipChildLevelPointer指向了下一层的某个位置。

多个域的doc文件的数据结构

图6：



结语

.pos、.pay、.doc、.tim、.tip文件都是通过读取内存倒排表的过程中一起生成的，在处理完每个term的信息并写入.pos、.pay、.doc文件后，开始生成.tim、.tip文件，在最后的文章中会更新这部分内容。

[点击下载](#)Markdown文件