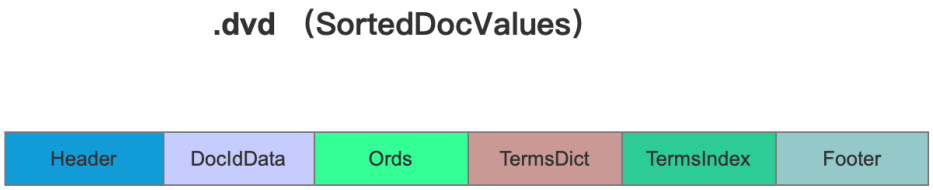
SortedSetDocValues

SortedNumericDocValues的索引结构跟SortedDocValues几乎是一致的，所以本文不会赘述跟SortedDocValues相同部分的内容，只介绍不同的部分数据结构。两种DocValue的最常用的使用场景就是对搜索结果进行排序，使用SortedSetDocValues相比较SortedDocValues的优点在于，一篇文档中可以设置多个相同域名不同域值的SortedSetDocValuesField，而SortedDocValues在一篇文档中只允许有一个相同域名的域。因此我们可以在不更改现有索引的情况下，只修改搜索的条件（更改Sort对象）就可以获得不同的排序结果，在后面介绍facet的文章中会详细介绍SortedSetDocValues的应用。

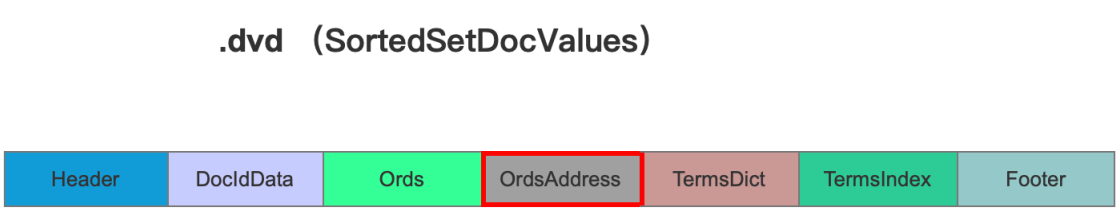
数据结构

dvd

先给出SortedDocValues的.dvm文件的数据结构。 图1：



再给出SortedSetDocValues的.dvm文件的数据结构。 图2：



两个DocValues的TermsDict、TermsIndex部分的数据结构是一模一样的，因为源码中他们实际调用的是同一个方法来写入这两块的数据，另外DocIdData的数据结构也是一样的。下面介绍不同之处OrdsAddress。

OrdsAddress

在上文中提到，索引阶段使用SortedSetDocValues的话，一篇文档中可以有多个相同域名不同域值的SortedSetDocValuesField，而SortedDocValues只能有一个相同域名的SortedDocValuesFIeld，如下图所示。 图3：

```
Document doc ;
// docId = 0
doc = new Document();
doc.add(new SortedDocValuesField( name: "superStar", new BytesRef( text: "aa")));
indexWriter.addDocument(doc);
```

图4：

```
Document doc ;
// docId = 0
doc = new Document();
doc.add(new SortedSetDocValuesField( name: "superStar", new BytesRef( text: "aa")));
doc.add(new SortedSetDocValuesField( name: "superStar", new BytesRef( text: "aaa")));
indexWriter.addDocument(doc);
```

SortedDocValues中的Ords字段，一个ord信息对应一篇文档的域值，而在SortedSetDocValues中多个ord信息对应一篇文档的多个域值，由于所有的ord都存放在Ords中，所以需要OrdsAddress，使得在读取阶段能使每一篇文档获得对应的所有ord值。

> OrdsAddress的值使用了DirectMonotonicWriter类进行了 趋势分解操作，然后使用PackedInts进行了压缩存储。DirectMonotonicWriter中的趋势分解的目的是尽可能减少空间的使用，它用来将 单调递增的整数序列(monotonically-increasing sequences of integers)进行平缓操作，使得在使用PackedInts进行压缩存储时，每一个数值能使用最少的固定bit位存储。

这里不赘述DirectMonotonicWriter中的趋势分解过程，可以看我的源码注释来理解这个过程：[https://github.com/luxugang/Lucene-7.5.0/blob/master/solr-7.5.0/lucene/core/src/java/org/apache/lucene/util/packed/DirectMonotonicWriter.java]。 **如果在索引阶段，每一篇文档的最多只有一个SortedSetDocValuesField，那么生成的.dvd索引结构跟SortedDocValues就是一样。可以理解为SortedSetDocValuesField退化成了SortedDocValuesField。**

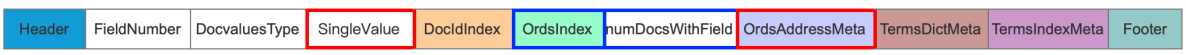
dvm

先给出SortedDocValues的.dvm文件的数据结构。 图5：

.dvm （SortedDocValues）

再给出SortedSetDocValues的.dvm文件的数据结构。 图6：

.dvm （SortedSetDocValues）

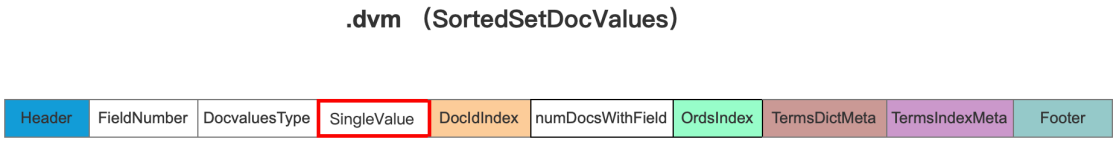


从上面两张图可以看出，SortedSetDocValues的.dvm数据结构多出了SingleValue、OrdsAddressMeta两块数据，用红色框标出，另外OrdsIndex跟numDocsWithField两块的数据位置互换了一下。

SingleValue

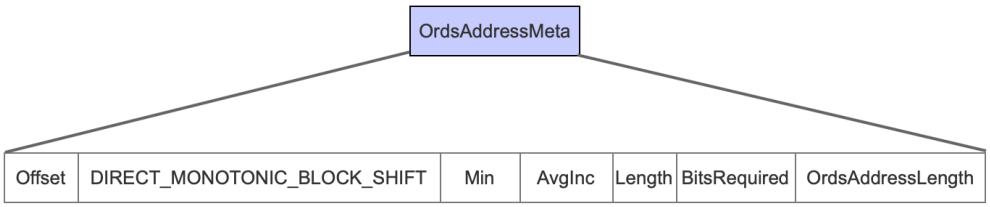
SingleValue描述了是否在索引阶段，每篇文档是否最多包含一个SortedSetDocValuesField，如果是的话，那么.dvm的剩余数据结构跟SortedDocValues是一致的

图7：



OrdsAddressMeta

图8：



Offset

OrdsAddress数据段在.dvd文件的开始位置。

DIRECT_MONOTONIC_BLOCK_SHIFT

DIRECT_MONOTONIC_BLOCK_SHIFT用来在初始化byte buffer[]的大小，buffer数组用来存放每一篇文档OrdsAddress。

Min

记录一个最小值，在读取阶段用于解码。Min的含义请看我的源码注释。

AvgInc

记录一个AvgInc，在读取阶段用于解码。AvgInc的含义请看我的源码注释。

Length

在SortedSetDocValues使用DirectMonotonicWriter的场景中，该值永远为0，不解释。

BitsRequired

```
经过DirectMonotonicWriter的数据平缓操作后，每个数据需要的固定bit位个数。
```

OrdsAddressLength

```
OrdsAddress数据段在.dvd文件的数据长度。
```

OrdsAddressLength结合Offset，就可以确定OrdsAddress在.dvd文件中的数据区间。

结语

由于SortedSetDocValues与SortedDocValues的索引文件数据结构非常类似，所以本篇介绍篇幅很小。所以先了解SortedDocValues的数据结构后，那么SortedSetDocValues的数据结构就是一目了然的。

点击下载Markdown文件