Apache Zookeeper 磁盘空间预分配策略

我们知道，Zookeeper
会将所有事务操作的数据记录到日志文件中，这个文件的存储路径可以通过 dataLogDir
参数配置。在写数据之前，Zookeeper
会采用磁盘空间预分配策略；磁盘空间预分配策略主要有以下几点好处：

- 可以让文件尽可能的占用连续的磁盘扇区，减少后续写入和读取文件时的磁盘寻道开销；
- 迅速占用磁盘空间，防止使用过程中所需空间不足。

通过这种策略 Zookeeper 避免磁盘的频繁 Seek 操作。代码层面上的实现如下（具体参见 FileTxnLog.java 类）：

```
/**
 * append an entry to the transaction log
 * @param hdr the header of the transaction
 * @param txn the transaction part of the entry
 * returns true iff something appended, otw false
 */
public synchronized boolean append(TxnHeader hdr, Record txn)
    throws IOException
{
    if (hdr == null) {
        return false;
    }
    if (hdr.getZxid() <= lastZxidSeen) {
        LOG.warn("Current zxid " + hdr.getZxid()
                + " is <= " + lastZxidSeen + " for "
                + hdr.getType());
    } else {
        lastZxidSeen = hdr.getZxid();
    }
```

```
    if (logStream==null) {
       if(LOG.isInfoEnabled()){
            LOG.info("Creating new log file: " + Util.makeLogName(hdr.getZxid()));
       }

       logFileWrite = new File(logDir, Util.makeLogName(hdr.getZxid()));
       fos = new FileOutputStream(logFileWrite);
       logStream=new BufferedOutputStream(fos);
       oa = BinaryOutputArchive.getArchive(logStream);
       FileHeader fhdr = new FileHeader(TXNLOG_MAGIC,VERSION, dbId);
       fhdr.serialize(oa, "fileheader");
       // Make sure that the magic number is written before padding.
       logStream.flush();
       currentSize = fos.getChannel().position();
       streamsToFlush.add(fos);
    }
    currentSize = padFile(fos.getChannel());
    byte[] buf = Util.marshallTxnEntry(hdr, txn);
    if (buf == null || buf.length == 0) {
        throw new IOException("Faulty serialization for header " +
                "and txn");
    }
    Checksum crc = makeChecksumAlgorithm();
    crc.update(buf, 0, buf.length);
    oa.writeLong(crc.getValue(), "txnEntryCRC");
    Util.writeTxnBytes(oa, buf);

    return true;
}

/**
 * pad the current file to increase its size to the next multiple of preAllocSize greater than the c
urrent size and position
 * @param fileChannel the fileChannel of the file to be padded
 * @throws IOException
 */
private long padFile(FileChannel fileChannel) throws IOException {
    long newFileSize = calculateFileSizeWithPadding(fileChannel.position(), currentSize, preAlloc
Size);
    if (currentSize != newFileSize) {
        fileChannel.write((ByteBuffer) fill.position(0), newFileSize - fill.remaining());
        currentSize = newFileSize;
    }
    return currentSize;
}
```

```
/**
 * Calculates a new file size with padding. We only return a new size if
 * the current file position is sufficiently close (less than 4K) to end of
 * file and preAllocSize is > 0.
 *
 * @param position the point in the file we have written to
 * @param fileSize application keeps track of the current file size
 * @param preAllocSize how many bytes to pad
 * @return the new file size. It can be the same as fileSize if no
 * padding was done.
 * @throws IOException
 */
// VisibleForTesting
public static long calculateFileSizeWithPadding(long position, long fileSize, long preAllocSize) {
   // If preAllocSize is positive and we are within 4KB of the known end of the file calculate a n
ew file size
   if (preAllocSize > 0 && position + 4096 >= fileSize) {
      // If we have written more than we have previously preallocated we need to make sure th
e new
      // file size is larger than what we already have
      if (position > fileSize){
         fileSize = position + preAllocSize;
         fileSize -= fileSize % preAllocSize;
      } else {
         fileSize += preAllocSize;
      }
   }

   return fileSize;
}
```

从上面代码可以看出，事务文件是以写入的第一条事务 zxid
为名，这种命名方式方便后面的查找。在将 Record 写入事务文件中之前，首先会调用 padFile
函数预先分配磁盘空间；如果 logStream
为空（第一次调用或者事务日志文件达到指定的条数被切割）则会直接预分配 preAllocSize
大小的空间，这个参数默认值为 64MB，可以通过参数 zookeeper.preAllocSize
进行配置，如下：

```
static long preAllocSize =  65536 * 1024; // 64MB

String size = System.getProperty("zookeeper.preAllocSize");
if (size != null) {
```

```
    try {
        preAllocSize = Long.parseLong(size) * 1024;
    } catch (NumberFormatException e) {
        LOG.warn(size + " is not a valid value for preAllocSize");
    }
}
```

其他时刻会检测事务日志文件剩余空间是不是不足 4096
字节，是的话就会开始进行文件空间扩容，即在现有文件大小上，将文件增加 preAllocSize
大小的空间。不管是第一次预分配还是后面空间不足预分配，申请到的空间都是使用0
进行填充。