**[ext4文件系统翻译之第二篇](http://blog.csdn.net/sfz200809242719/article/details/11980061)**

Extents tree

扩展树

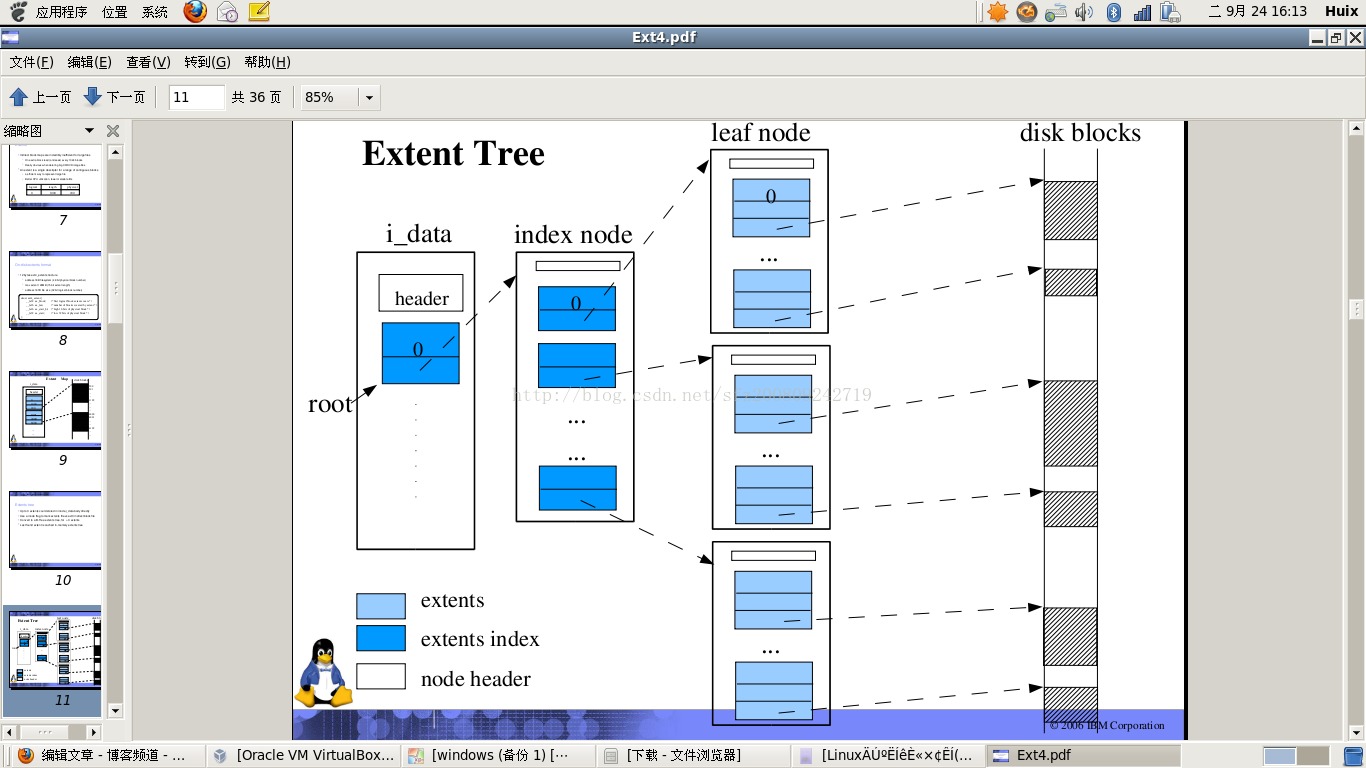
Up to 3 extents could stored in inode i\_data body directly  
最多可以有3个扩展能被直接存储在inode i\_数据区躯干（这个i\_data我估计是数据区吧）

Use a inode flag to mark extents file vs ext3 indirect block file

用一个inode标志去标记扩展文件比较ext3间接映射文件块

Convert to a B­Tree extents tree, for  > 3 extents  
转换为b树扩展当一个inode中扩展大于3的时候（我这里进行了补充）

Last found extent is cached in­memory extents tree  
上次发现的扩展被缓存到内存扩展树里



如上图，我们可以看到，一个index\_node可以有多个扩展树，根据颜色莱看，首先，颜色深的是扩展树，第一个的起点是在I\_DATA里面（就是在indode的数据区里），然后那个扩展树采用B树的方式，将枝条往外扩展，直到叶子（extent），叶子在映射连续的数据块。

48­bit block numbers  
48位块数量

Part of the extents changes  
改变扩展的区域（翻译不好啊，这句）

32bit ee\_start and 16 bit ee\_start\_hi in ext4 extent struct  
32位 的EE\_START结构体和16位的EE\_START结构体 在EXT4中

Why not 64­bit  
为什么不是64位呢（这个64位是相比48位块数量）

48­bit is enough for a 2\*\*60 (or 1EB) filesystem  
48位是足够应付一个1EB的文件系统了

Original lustre extent patches provide 48­bit block numbers  
提供48位数量块原始的平滑的扩展补丁（哎，这句希望能得到大神的翻译）

More packed meta data, less disk IO  
更多包装的元数据，更少的磁盘io

Extent generation flag allow adapt to 64­bit block number easily  
扩展上一待标志允许轻松适应64位块数量

64­bit meta data changes  
64位元数据改变

In kernel block variables to address >32 bit block number  
在内核块变量处理地址>32位块数量（翻译不好，自己也理解不好）

Super block fields: 32 bit ­> 64 bit  
超级块字段32位>64位

Larger block group descriptors (required doubling their size)  
更大的块组描述（需要双倍的他们长度）

extended attributes block number (32 bit ­> 48 bit)  
扩展属性的快号（32位>64位）