论文动机

1. 压缩SStable Block Index的存储空间，因为这些Block Index或者布隆过滤器通常都常驻缓存，所以对提高缓存空间利用率，进一步提高缓存命中率有价值, 如论文图3所示，Block Index部分。

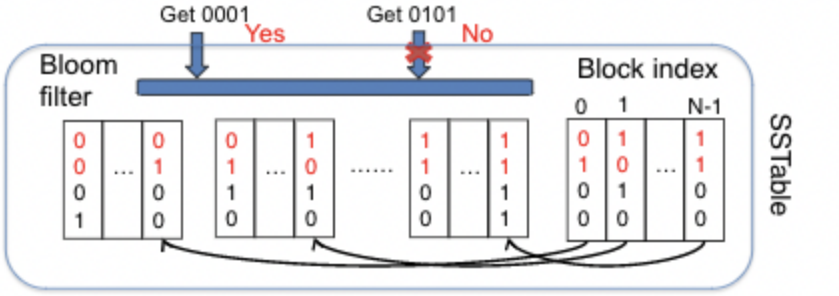

Figure 3: *Illustration of the basic index format of LevelDB's SSTable and its read path. The keys follow a semi-sorted order, so each key has two parts: the prefix (red) and the suffix (black).*

2. 在Step-merge结构中（即LSM-tree中允许某一层有多个交集的结构，典型的如LevelDB中的L0层）快速确定层数的方法

方法

1. Block Index的压缩： 如图6所示，把key分成前缀和后缀两部分。 然后把每个Block的第一个和最后一个key取出来，构成中间的vanilla index。主要是对这个vanilla index进行压缩。压缩后的结构包括三部分：

   a. 前缀去除重复

   b. offset 记录a中每一个前缀在vanilla index中最后出现的位置，有了a和b，其实很容易确定前缀是在哪几个block中

   c. 后缀数组，记录每个block最后一个元素的后缀。 这部分细节不是特别清楚。通过例子可以看出来，在a, b确定前缀在哪个block之后，要匹配block，这时候其实是可以做二分搜索的，因为同一个前缀的后缀是单调递增的，虽然整体的后缀不是递增的。所以论文中对000做二分搜索，仅仅是搜索010，110这个子数组，因为000比010小，所以一定在Block 0中。

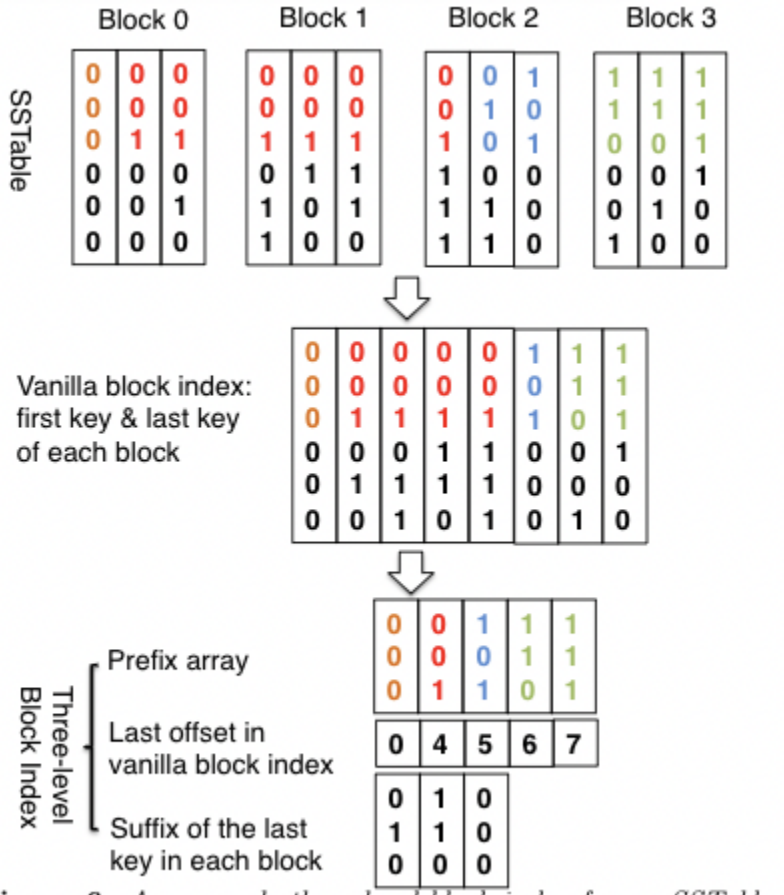

Figure 6: *An example three-level block index for an SSTable.*

压缩： 上面结构的三部分都可以进行压缩，a和c可以用ECT-Trie树，

*问题：技术能否对变长有效（前缀部分使用了数组）