Leaper——基于机器学习的高速缓存预取

Leaper: A Learned Prefetcher for Cache Invalidation in LSM-tree based Storage Engines

Lei Yang[1]*, Hong Wu[2], Tieying Zhang[2], Xuntao Cheng[2], Feifei Li[2], Lei Zou[1],
Yujie Wang[2], Rongyao Chen[2], Jianying Wang[2], and Gui Huang[2]
{yang_lei, zoulei}@pku.edu.cn[1]
{hong.wu, tieying.zhang, xuntao.cxt, lifeifei, zhencheng.wyj, rongyao.cry, beilou.wjy,
qushan}@alibaba-inc.com[2]
Peking University[1] Alibaba Group[2]

汇报人：习钟　汇报日期：2022.5.11

背景

1. 数据库缓存（对于常规的数据库设计）
    a. 缓存淘汰算法：LRU，LFU...
2. LSM-tree
    a. Log-Structured Merge-Tree(日志结构合并树)
    b. 对比B树，将随机读取磁盘变成了顺序读写磁盘
3. 缓存失效（LSM-tree）：
    a. This(flush , compaction）invalidates their corresponding entries and statistics in the cache and leads to cache miss for their lookups.（LSM-tree中的追加写和写时复制技术需要引入可变的存储数据块和频繁的后台操作来重新组织数据块。这些后台操作会使得数据块对应的缓存失效，从而导致缓存命中率骤降和访问延迟的急剧升高。）
    b. frequency of compactions decreased cache hit rates

设计思路

“Fundamentally, the cache invalidation problem can be addressed by minimizing |C and Mi| during each compaction (or flush). It requires us to predict which records are accessed in the near future, and then swap them into the cache in advance.”

> It is a prediction problem that predicts the future access of each record in the set |C and Mi|.

观点分析：

提出了使用在整体数据流中不会被影响的粒度，也就是key range的粒度作为整体的统计信息采集粒度，并且根据这些统计信息在compaction和flush的过程中使用机器学习方法去预测哪些key range（key range可以理解为用来存储的一系列硬件）会被访问。

故问题转换为了一个二分类问题——**是否会被访问**

binary classification problem：given the access of a record in the latest x · t minutes, predict whether this record is accessed in the following T minutes, where t, x, and T refer to a statistical time interval, number of intervals, and the prediction duration.

系统设计

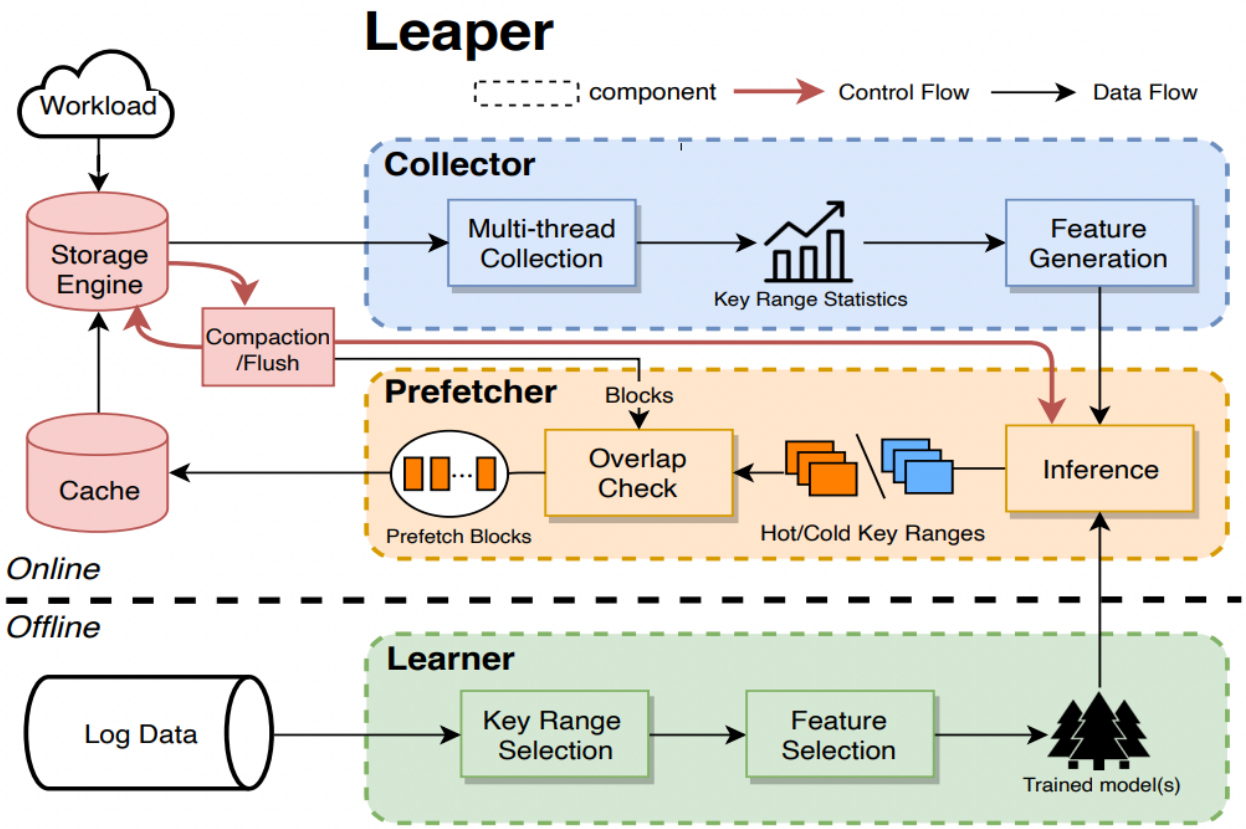

Figure 3: Workflow of LSM-tree storage engine with Leaper.

- Learner：
  - 机器学习的底层模型训练的部分，论文中使用的是梯度增强的决策树

- 将训练结果用于预测在Compaction/Flush过程中的Range位置
- 模型可以进行离线训练

- Prefetcher：
  - 对缓存直接操作的部分，通过机器学习的预测来决定缓存的命中同时对数据检查
- Collector：
  - 收集到实际工程的结果，然后对数据进行统计并进行特征提取

实现细节

Algorithm 1: Key Range Selection

Input: Total key range T, initial size A , access information and decline threshold α

Output: Most suitable granularity A^*

Initialize access matrix M ($N \times \frac{T}{A}$ bits for N time intervals and $\frac{T}{A}$ key ranges) for key range size A;

Define the number of zeroes in M as $Z(A)$;

while $2 \cdot Z(2A) > \alpha \cdot Z(A)$ **do**

  $A \leftarrow 2A$;

Binary search to find the maximum value A^* satisfying $A^* \cdot Z(A^*) > \alpha A \cdot Z(A)$ from A to $2A$;

return A^*

- 主要解决了对于Key Range选择的过程

模型

- 模型选择：
  - **Gradient Boosting Decision Tree (GDBT)**
- 输入特征：
  - a feature vector with 18 dimensions (i.e., 6 read arrival rates, 6 write arrival rates, 3 timestamp features, and 3 precursor arrival rates).
- 输出特征：
  - a binary digit (i.e., 1 or 0) indicating whether this key range would be accessed soon
  - 输出直接是二值序列来判断在选择的range中哪些部分是可以被访问到的
- 损失函数：

- Square Loss and Logarithmic Loss.

实验结论

在实际生产过程中，在天猫买家库订单表、钉钉以及sysbench生成的zipf分布的workload下比较了论文的方法和已有方法。实验结果表明，论文中的方法相比于传统方法可以多消除70%以上的cache invalidation，同时能消除几乎所有的latency抖动。

结论

1. 定义了cache invalidation问题，并且识别出了在LSM-tree问题中产生这个问题的根本原因。同时我们提出了基于机器学习的高速缓存预取策略来解决这个问题
2. 保证准确度，同时尽可能降低在离线训练和在线推断方面的开销
3. 在多种workload上进行了比较

读者观点

1. 设计有太多自定义的东西在，基本只能看个设计思想
2. 工程化的实现，细节比较多，复现比较困难
3. 数据库与人工智能的结合方向上，这里从缓存的角度来考虑，角度很好