Jiasen Li

—

Jiasen Li
Phone +86 13262776958
Emails
lijiasen0921@sjtu.edu.cn
lijiasen00921@gmail.com
(GMail is recommend)

Report of Lab2

1. Eviction

I use a victim cache for those which are evicted by dirty.
Because I am sure that they will be visited again.
BufferPool is Map: PageID->Page
I use a double mapping Map: PageID <-> TimeStamp
I implement high speed LRU replacement (O(logn))
I force the Victim Cache to be small and do Garbage Collection (Special Eviction) after each deletion and insertion. GC is only for Victim Cache.
I use discard inside eviction according to the hints in the code comment.

2. Searching

I think that a top-down B+ tree would run faster. Top-down B+ tree can be implemented just like the non-rotating Fan Haoqiang Treap.
Page Split(pid, field) returns the Split result of the pid sub-tree.
A top-down B+ tree would not save the parentID. A top-down B+ tree could get tmp parentID when it FINDLEAF. Split and Merge can also be done with the top-down method, which may save the time to update parentID.

However, the interface has ParentIDs and Tests ParentID update. As a result, I implement it normally and have 2 findLeafs.
Find Leaf through the Tree.

3. Inserting

Normal implementation. Split one by one upwards. 4 cases of split, root->internal internal->internal internal->leaf root->leaf.
If the "ParentID Check" is off, internal pages with many children updating parents would save a lot of time.
I add "markDirty" in getPage if Permissions.ReadWrite
I do Garbage Collection for Victim Cache After Inserting.
When coding, Sibling problems should be noticed.

4. Deleting and Merging

The interface has implemented the "handle……" and I only need to fill in merges.
The first version failed with Memory Problems. Tuples and RecordIDs should save memory because they are the major cost of Memory. I Changed Tuple and Record ID and then passed the tests.

5. Later

Later I add some codes for HeapFile similar to BTreeFile.

I think that the replacing strategy does make a change. Locally, I run 4 times faster with Victim Cache. Azure Pipeline runs 20 seconds approximately.