Shanghai Jiao Tong University
Lab2 Report: B+ Tree Index
Name: Yunxuan Xiao
Sid: 516030901268

Database 2019
April 30th

1 Design Decision

1.1 Page eviction for SimpleDB Buffer

Actually I have already implemented an LRU Cache in Lab1 for my own extensive test-case. So in Lab2 I simply changed some parts to fit for new functions like **flushPage()**, **discardPage()** and **flushAllPages()**.
For **InsertTuple()** and **DeleteTuple()** , we have to mark the pages that were dirtied by the operation by calling **markDirty()**, and insert the new versions of these pages that have been dirtied to my LRU Cache so that future requests see up-to-date pages.

1.2 Search Leaf Page in B+ Tree

This part is quite easy that we only have to recursively call **FindLeafPage()** method from top-to-bottom in the tree. In each InternalPage, we iterate through all the keys and find the first less-or-equal keys to the given key. Then we perform findleafpage on its left children page. When we reach a leaf page, the recursion stops.

1.3 Insert in B+ Tree

The problem in tuple insertion lays in page splitting. The procedure of page splitting can be formalized as follow:

1. prepare a new leaf/internal page
2. find the middle key which have to be "copied up"(leaf)/"pushed up"(internal)
3. find a valid parent page to insert middle key which contains empty slots (May leads to recursively parent page splitting)
4. insert the middle key into the valid parent page
5. transport a half tuples/entries on current full page to the new leaf/internal page.
6. update the parent pointer (and sibling pointers) for the former and new leaf pages.

1.4 Delete in B+ Tree

When deleting tuples/entries in B+ Tree, pages which is less than half full may have to steal elements from sibling pages or merge with others.

1.4.1 Steal from sibling

Denote the number of entries in current page as p_1, the number of entries in its neighbor page is p_2. To keep balance of the B+ tree, the total number of entry current page have to steal is $(p_2 - p_1)/2$, which leads to equal entries with its sibling.

1.4.2 Steal from sibling

For leaf page, we simply move $(p_2 - p_1)/2$ from sibling and change the parent keys as the leftmost/rightmost tuple key of right/left sibling.
For internal page we firstly pull down the parent key to current page, steal $(p_2 - p_1)/2$ from sibling, finally push up the leftmost/rightmost key in the right/left sibling.

1.5 Merge with sibling

Thanks to the pre-implemented method **deleteParentEntry**, it becomes a quite relaxing staff to merge two pages without the worry about recursively merging. The things left to do is as follow:

1. transport all tuples/entries from one page to another, the emptied page is about to be discarded.

2. reset the parent keys for all children pages, reset the sibling pointers.

3. Mark the deleted pages as empty page.

2 Missing and Incomplete Part

I notice in the descriptions of InsertTuple and DeleteTuple in BufferPool.java, it says *"Retrieve the specified page with the associated permissions. Will acquire a lock and may block if that lock is held by another transaction."*
But currently I did not implement any functions handling multi-threading and transaction locks. Although I passed all the unit test and system test, I think maybe later I still need to change my implementation to fit for transaction management.

3 Time, Difficulty and Confusion

I spend a whole day on this lab, from 10am in the morning to 3am in the next day. The confusion comes from these part:

1. **Deeply Nested API** It takes me a lot of time to figure out what's the function of these bunch of predefined APIs. Also it makes me confusing about the name of variables and methods. It often comes to me that I have a clear mind and decide to implement a function from scratch, but accidentally find that I used a wrong function with similar name or some useful function has already been written.

2. **Bug From previous Lab** The implementations in lab1 might cause some internal subtle errors since previously we did not consider some circumstance or requirements for lab2. I left some TODOs but unfortunately ignore them when debuging thus caused me some problem.