Parallel compaction and update

Solution Document:

Currently in carbon, compaction and updates are executed consecutively. And because of this, in case of large amount of data compaction and update operation has to wait for long amount of time to complete the update and compaction respectively. To solve this problem, we are planning to support compaction and update parallelly. In this document we have explained one of the solutions to support parallel compaction and update.

Current Implementation:

Compaction: In compaction we write multiple segments into a new segment based on the type of compaction (major, minor or custom).

For ex: If we have segments seg0, seg1, seg2, seg3 and we have run minor compaction with level 4,3. In this case a new segment seg0.1 will be created and data of all the segments will be written in seg0.1 and update the metadata information.

Update (Insert+delete): We perform the delete and insert operation for every update, and write the delete and insert delta file inside the segment on which update was performed and update the metadata information.

For ex: If there is update operation on seg1 then we will write the delete and insert delta file in the same segment seg1.

In case of parallel queries for both the operation this may lead to data inconsistency or incorrect result. So, to solve this problem currently we take compaction and update lock before starting of compaction or update, and we don't support parallel compaction and update. But this solution has the disadvantage of long waiting time to execute update if compaction is in progress or vice versa.

Now, we are planning to support parallel compaction and update and to do that, one of the methods is explained below:

Proposed solution:

Compaction: When the compaction is in progress then we will write the compacted segment at some temporary location and we will also maintain all the required metadata information like table status, segment status which help to find if any new update performed on any segment.

Update (Insert+delete): We are planning to change the location of delta insert file in update operation. At place of writing the delta insert file inside the same segment we are planning to write in a new segment. When update is in progress then we will write the delete delta file at some temporary location and we will also maintain the metadata information like table status, check points, segment info on which delete is getting performed, delete query plans which help to execute delete again if required.

For ex: If there is delete operation on seg1 then we will write the insert delta file in new segment (say seg4).

Based on query we may have three scenarios for execution:

1. Only compaction or update executing.
2. In case of parallel execution updates completes before compaction.
3. In case of parallel execution compaction completes before updates.

After complete of any operation either compaction or update we will take a lock and move the files written at temp location to final location and merge the table status and metadata info files.

<u>Scenario 1:</u>

a) <u>Only compaction is executing:</u>

If only compaction is executing then after writing the compacted segment at temp location say segment 0.1 in above example, we will first take a lock and move the seg0.1 at the place where all the segments are present and merge the tablestatus and others metadata info file with the file written at temp location.

b) <u>Only Update is executing:</u>

If only update is executing then after complete of update operation, we will take the lock and move the delete delta file inside the segment on which update was performed and merge the tablestatus and other metadata info.

<u>Scenario 2: In case of parallel execution updates completes before compaction</u>.

Consider the scenario where we have 4 segment -> seg0, seg1, seg2, seg3 and we are executing the compaction with level 4,3 and update operation on seg1 parallelly.

Now updates complete before compaction so, it will take a lock and move the delete delta file written at temp location inside the seg1 and merge the metadata and table status information. As we know that the delta insert file has been already written in new segment say seg4. After this update will be completed and lock will be released. We may trigger other update queries also in same way after complete of one update.

Let's consider after sometime compaction completes. So, it will acquire the lock and move the compacted segment seg0.1 to the location where all the segments are present. After that it will try to merge the metadata information and it can find that the update has been performed on seg1. So, the same delete operation will be executed on seg0.1 and move the delete delta file inside seg0.1 and merge the metadata info and release the lock.

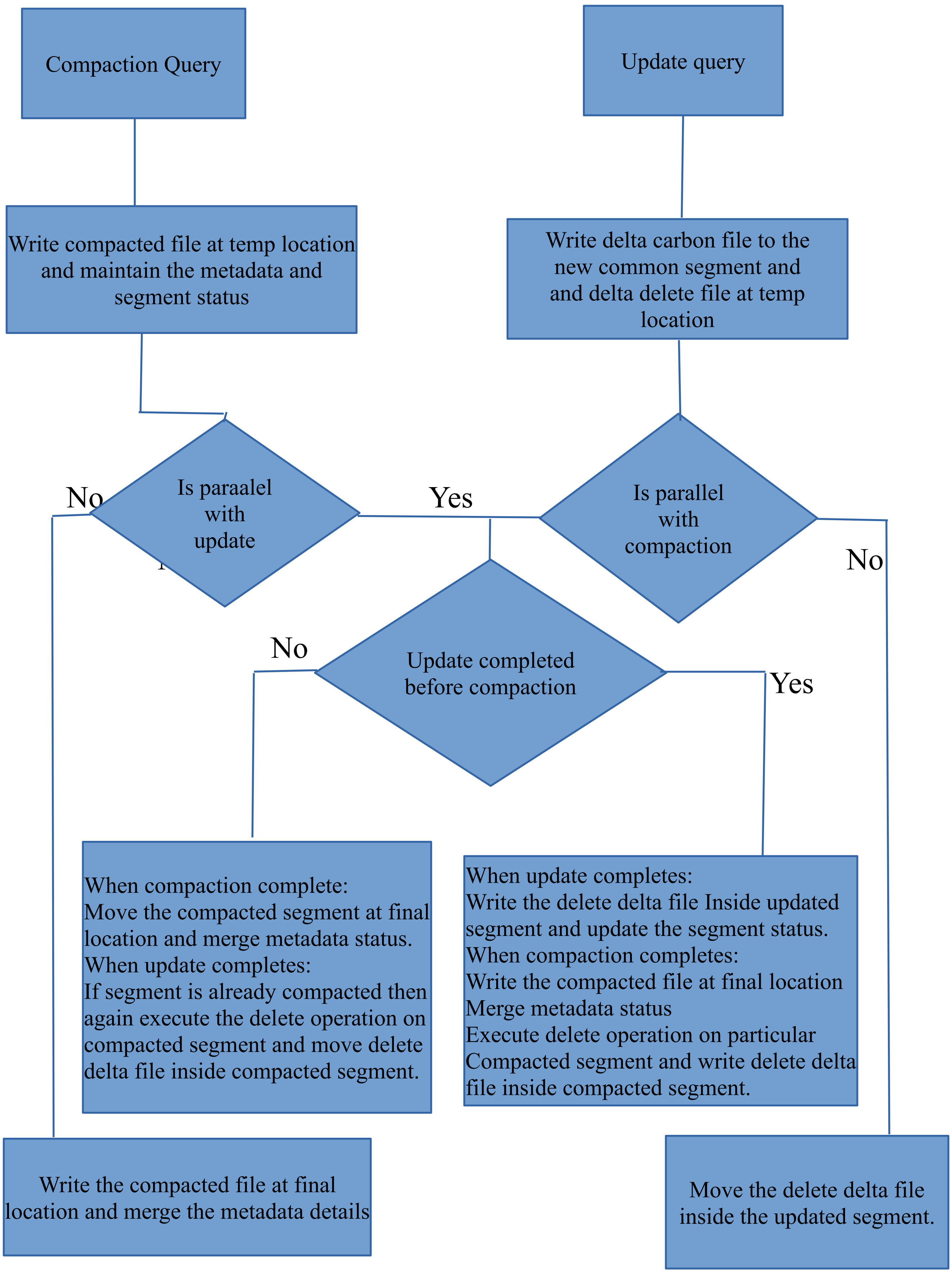
<u>Scenario 3: In case of parallel execution compaction completes before updates</u>.

Consider the same example as scenario 2 ->
where we have 4 segments: seg0, seg1, seg2, seg3 and we are executing the compaction with level 4,3 and update operation on seg1 parallelly.

Now compaction completes before update so it will acquire the lock and move the compacted segment seg0.1 to the location where all the segments are present and merge the metadata information and tablestatus file. Complete the compaction and release the lock.

Let's consider after sometime update completes which has written the delta insert file inside the new segment say seg4. Now it will acquire the lock and from the metadata info it can find that the seg1 is already compacted. In this case it will execute the same delete operation to the compacted segment seg0.1 and move the delete delta file inside seg0.1 and merge the metadata and table info and release the locks.

As we can notice that we may need to perform the delete operation twice in case of parallel execution of compaction and update. But we already know that delete operation is very fast and not cause much performance degradation.


Compaction Query
Update query
Write compacted file at temp location and maintain the metadata and segment status
Write delta carbon file to the new common segment and and delta delete file at temp location
No
Is paraalel with update
Yes
Is parallel with compaction
No
No
Update completed before compaction
Yes
When compaction complete: Move the compacted segment at final location and merge metadata status. When update completes: If segment is already compacted then again execute the delete operation on compacted segment and move delete delta file inside compacted segment.
When update completes: Write the delete delta file Inside updated segment and update the segment status. When compaction completes: Write the compacted file at final location Merge metadata status Execute delete operation on particular Compacted segment and write delete delta file inside compacted segment.
Write the compacted file at final location and merge the metadata details
Move the delete delta file inside the updated segment.