Problem

Currently the MOB compaction is implemented in HMaster. The compaction runs only in HMaster.

This introduces additional workloads and network I/O to HMaster during the compaction. And the localities of MOB files are lost after the compaction.

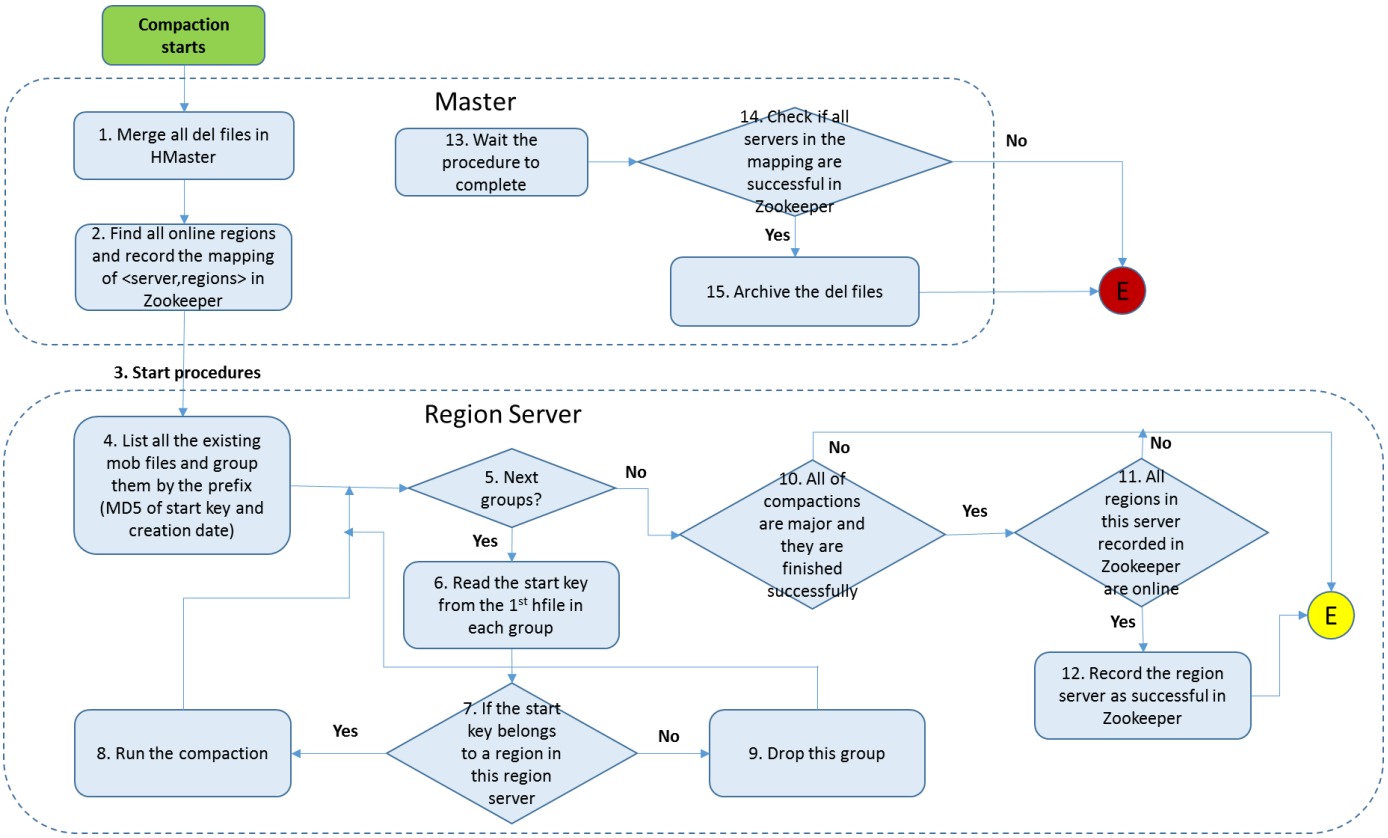
Solution

Like the compaction in HBase, the MOB compaction can be distributed to HRegionServer too.

The distributed MOB compaction can be implemented in HBase procedure.

1. The MOB compaction is triggered in HMaster by ScheduledChore.
2. HMaster starts a procedure to distribute the MOB compaction to region servers.

Details



1. When the compaction is called in HMaster, it selects all the existing del MOB files, and merge them into bigger files to make sure the number of del files is small.

2. HMaster finds all the alive regions and region servers, and record the mapping <regionServer, regions> in Zookeeper.
    a. HMaster needs to know if the del files can archived after the compaction. These del files can be archived only when all the regions are online and all sub-procedures are finished successfully. Thus HMaster records the mapping of <server, regions> to Zookeeper.
3. HMaster start a procedure for MOB compaction.
4. In each region server, it selects all the exiting MOB files and group them. The format of a file name is MD5(startKey)+"creation date"+UUID, MD5(startKey) and creation date are used for grouping, all the MOBs that have the same MD5(startKey) and creation date are in the same group.
5. For each group, we have to know the region where the grouped MOB files come from, region server has to read the metadata from the 1st MOB file in this group, we will discuss this later.
6. For each group, checks if the current group of MOB files belong to the region in the current region server.
    a. If true, run MOB compaction in this region server. If all MOB files in this group are selected during the compaction, this compaction is marked as major.
    b. If false, drop this group.
7. Wait until all groups are finished in this region server.
8. Check if all found regions in this region server that are recorded in Zookeeper before the procedure is started are still online, and all the groups of MOB files are compacted successfully in this region server and all of the compactions are major.
    a. If true, mark the compaction in this region server successful.
    b. If false, do nothing.
9. HMaster will wait until the procedure is finished.
10. At that time, HMaster checks the the Zookeeper to see if all the compactions in all region servers are marked as successful. If yes, it is a major compaction, HMaster directly archives all the del files.
11. Clean up the znodes in Zookeeper, and the mob compaction is finished.

How to know which region the MOB files come from?

As we know, the MOB files are never split even the regions split for multiple times. It means one MOB file might traverse many HBase regions.

In MOB compaction, we don't split MOB files either. We simply consider the MOB files belong to the region where they come from at the 1st time. So in step 6 we only pick up the first MOB file in that group and find out where it comes from.

Since the prefix of MOB file name is MD5 of a start key, it cannot be converted back to start key, we have to add a new item "MOB_REGION_STARTKEY" into metadata of MOB file so that we can know the start key of the region where this MOB file comes from by reading this metadata. For old MOB files that do not have such metadata, we can use the first key of each MOB file instead.

This demands a lot of reading to metadata of MOB files when compaction. Alternatively we can change the name format of the MOB file, we can use encoded name of start key instead of MD5 of start key, so

that we can get the start keys by only reading and converting the file names. But we cannot predict the length of the file name in this way since a start key of a region can be very long, and the MOB file name can be very long too, and consequently the reference cells in HBase take more storage spaces. It seems adding a new item to metadata is a more reasonable choice.