HDFS-15547 Disk-level tiering

Uber

Leon Gao
Ekanth Sethuramalingam

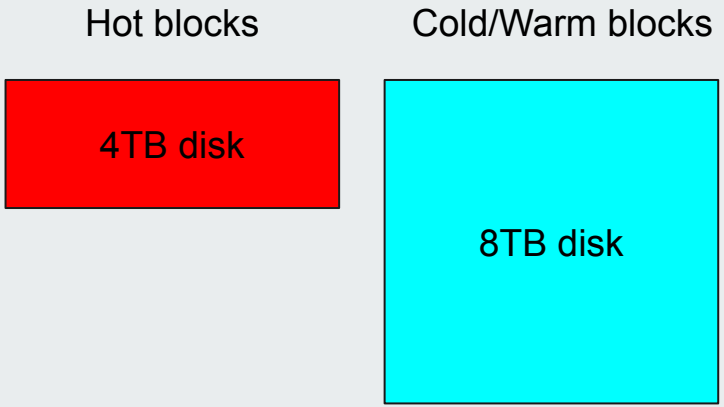
Motivation

- As HDD technology evolves in the past years, we have collected multiple types of HDD in our cluster:
  2TB disk -> 4TB -> 8TB

- We are looking forward to adopt 16TB disk in our production environment to achieve much better unit cost.

- Managing disk IO across different HDDs becomes a challenge.

Current setup



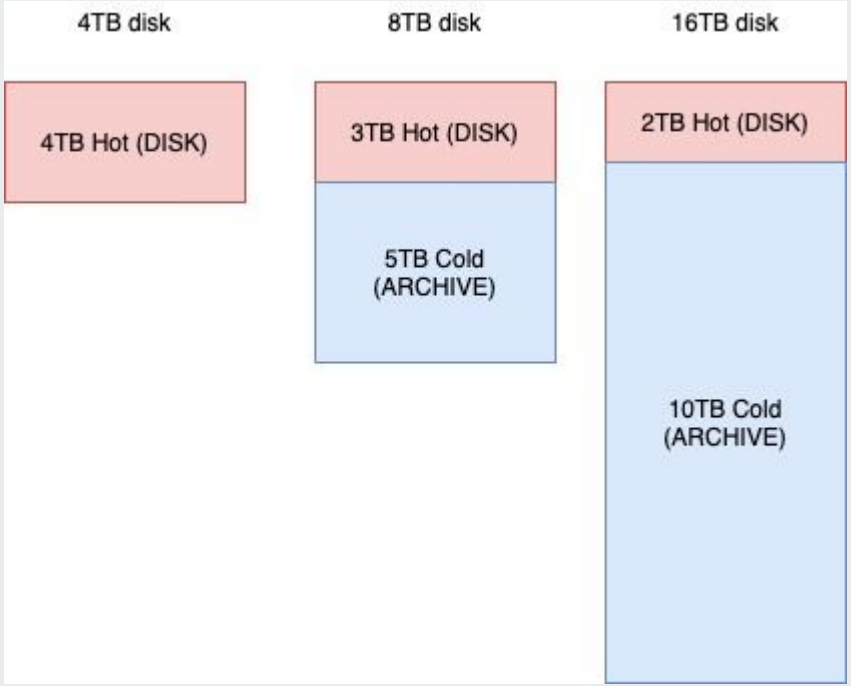
- HDFS data access is highly skewed
- Read traffic contributes to 80-90% of disk IO
- Cold disks IO is almost wasted, while some hot disks are very busy.
- Need a controlled way to mix cold/hot blocks on same disk

Hot disk VS Cold disk

One hour disk-util average

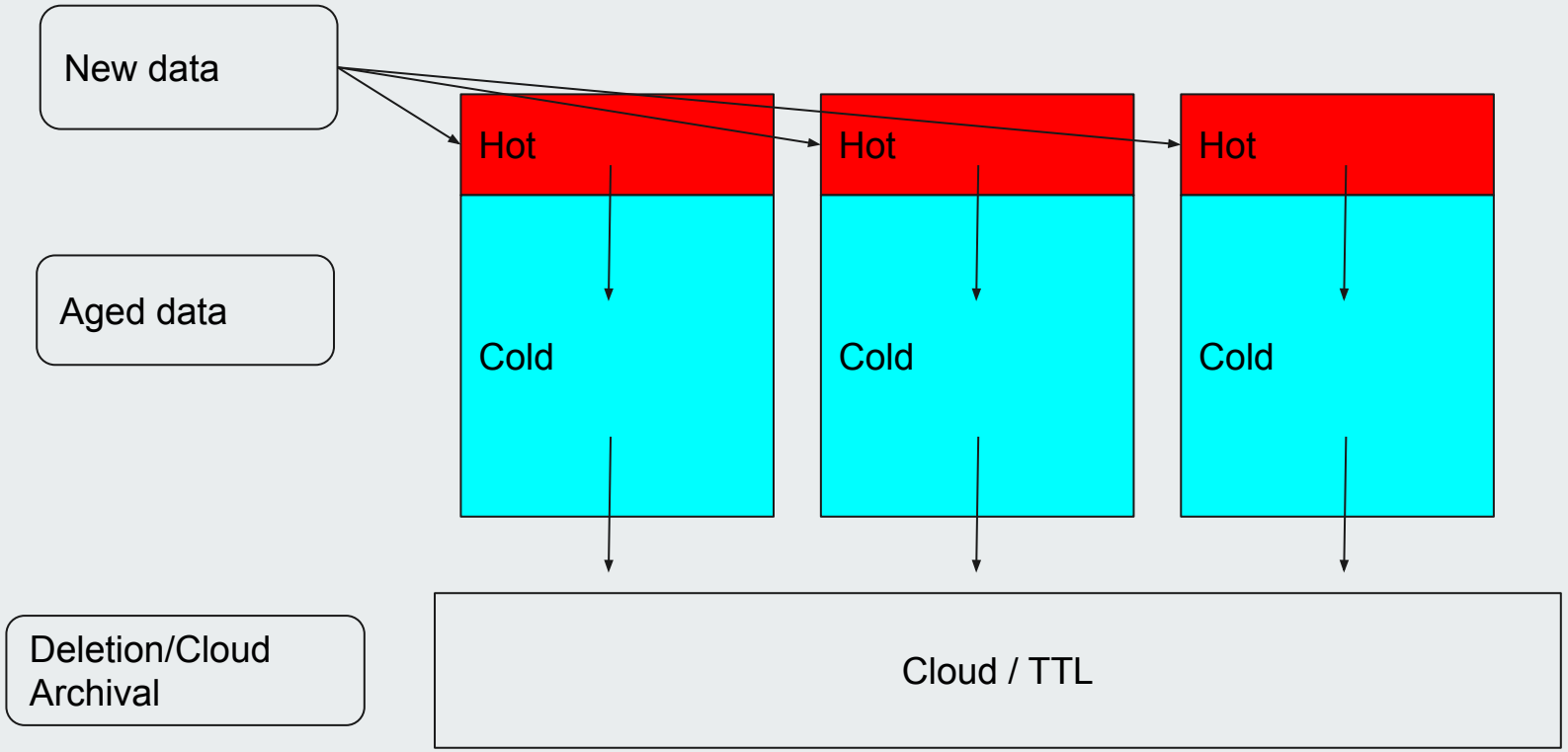
Cold disk (Yellow) IO is almost wasted, while hot disk is busy

High level idea



- Control disk IO by placing hot/cold blocks on the same disk
- Use up disk space with cold blocks and at the same time keep spindles busy for different SKUs
- The ratio of hot/warm is configured from datanode level. (DISK/ARCHIVE on same disk)

Data tiering flow on same disk without copying



Changes

- Able to configure DISK/ARCHIVE on same volume. Datanode should be able to control the capacity for each type and report the stats for each storage type correctly.

  For example, admins can create `/data01/dfs` and `/data01/dfs_archive` on the same mount `/data01`, and specify that 80% of the capacity belongs to ARCHIVE:

```
<property>
     <name>dfs.datanode.data.dir</name>
     <value>[DISK]/data01/dfs, [ARCHIVE]/data01/dfs_archive, …</value>
</property>

<property>
     <name>dfs.datanode.reserve-for-archive.percentage</name>
     <value>0.8</value>
</property>
```

Changes

- When mover tool move a block between DISK and ARCHIVE, prefer same mount and do rename instead of copy.

- (Nice to have) IO throttling on ARCHIVE portion to ensure the performance of DISK portion.

- (Nice to have) Live update ratio without restarting datanode