Hbase Performance and Reliability Enhancements

Agenda

- HBASE-451 Removes HTableDescriptor from HRegionInfo
- Load balancer enhancements
- HBASE-3777 Connection sharing
- HBASE-4015 TimeoutMonitor refactor to reduce contention
- HBASE-4377 Build .META. Table in offline mode
- HBASE-4213 instant schema update
- HBASE-2195 support for cyclic replication
- HBASE-4027 Off-heap cache
- Q & A

About myself

- Graduated from Tsinghua University
- Have been working on Hbase for over a year
- Promoted Hbase committer in June 2011
- Developer in Hadoop team at Ebay

Removing HTableDescriptor from HRegionInfo

- In 0.90 and prior, HRegionInfo refers to HTableDescriptor
  - Wastes a lot of heap in AssignmentManager
  - Prohibits high number of regions in large cluster

- In 0.92 and later, HRegionInfo only stores table name
  - Schema file, .tableinfo, is stored on hdfs under the table folder
  - HBASE-451 was logged in Feb 2008

Removing HTableDescriptor from HRegionInfo, cont'd

- New HRegion instances will load the HTableDescriptor from HDFS
- All schema operations result in appropriate changes to HTD on HDFS
- Migration from pre-0.92 clusters
  - Creates new .tableInfo files for all tables
  - Updates existing HRI for .META. by HRI without HTD
  - Set metamigrated in -ROOT- to record completion of migration
  - Continue with regular startup

Removing HTableDescriptor from HRegionInfo, cont'd

- Old HRI is in migration package and renamed to HRegionInfo090x
- New .tableInfo on HDFS follows the two step approach of .regionInfo
- HBASE-4032 HRegionInfo#getTableDesc backward compatibility
- HBASE-3970 HMaster crash/failure half way through meta migration
- HBASE-4301 META migration from 0.90 to trunk fails
- HBASE-4388 Second start after migration from 90 to trunk crashes

Load balancer enhancements

- In 0.90.x, daughter regions are placed on the same region server as the parent region
- HBASE-3586 randomly selected regions are offloaded from overloaded server(s)
- HBASE-3609 new and old regions from overloaded region servers would be assigned if new region server joins cluster
  - Otherwise, I find the new regions and put them on different underloaded servers
  - utilize the randomizer which shuffles the list of underloaded region servers

Load balancer enhancements, cont'd

- HBASE-3373 load balancing regions per table
- HBASE-3681 Check sloppiness of region load before balancing
- HBASE-3679 Use request histogram to help decide balancing action
- HBASE-4191 would use min cost maximum flow solver to utilize HRegion locality index
- http://zhihongyu.blogspot.com/2011/04/load-balancer-in-hbase-090.html

Connection Sharing

- In 0.90.x, new connection would be established for given Configuration instance
- This leads to too many connections on zookeeper
- HBASE-3777 Reference counting based connection sharing is implemented
- Sharing is determined by connection-specific properties, such as "hbase.zookeeper.quorum" – see HConnectionKey
- Htable.close() should be called if HTable instance is no longer in use
- TableOutputFormat. TableRecordWriter no longer calls HConnectionManager.deleteAllConnections(true)

Connection Sharing, cont'd

- HBASE-4508 backports HBASE-3777 to 0.90.5
- Rocketfuel.com has been running the backport since May, 2011
- HBASE-4087 stale connection cleanup in HBaseAdmin constructor
- http://zhihongyu.blogspot.com/2011/04/managing-connections-in-hbase-090-and.html

TimeoutMonitor refactor to reduce contention

- In 0.90.x, TimeoutMonitor used to be slow in responsiveness to status of region in zookeeper
- Initial redesign introduced new state RE_ALLOCATE which was later folded into OFFLINE state
- Gist is to compare the version in zookeeper with intended version
- Version for new unassigned node in zk is 0, expectedVersion passed is -1.
- HBASE-4203 speeds up Master restart if .META. was in OPENING state

TimeoutMonitor refactor to reduce contention

- Ramkrishna tested the implementation on a cluster with over 4000 regions
- When load balancer was reassigning regions, kill the destination region server and start it later
- The number of regions stayed constant
- When timeout happened, the recovery time was between 0.35 sec and 1.5 sec
- Hbck didn’t find any inconsistencies

Rebuild .META. Table in offline mode

- Hbck is extended to build .META. from .regioninfo
- HBASE-4506 allows hbck to be instantiated without connecting to active cluster
- missing regions on the file system and overlapping regions are printed
- User needs to move bad regions out
- hbase org.apache.hadoop.hbase.util.hbck.OfflineMetaRepair

Master-Master Replication

- Originating cluster Id is kept at ReplicationSink
- Replication source checks the HLog key to see if the cluster id equals the peer's cluster id
- Puts and Deletes store cluster Id in their attribute map
- HLogKey becomes versioned, utilizing the fact that encodedRegionName was written with Bytes.writeByteArray
- See javadoc in *HLogKey.readFields(DataInput in)*

Instant schema update

- HBASE-1730 Master issues close/open sequence
- HBASE-4213 avoids disabling the table or bulk assignment of regions
- Uses dedicated zk node for each region server
- Tolerates the drop or addition of region server(s) during alter
- Master failover doesn’t disrupt schema update
- Progress reporting and alter status

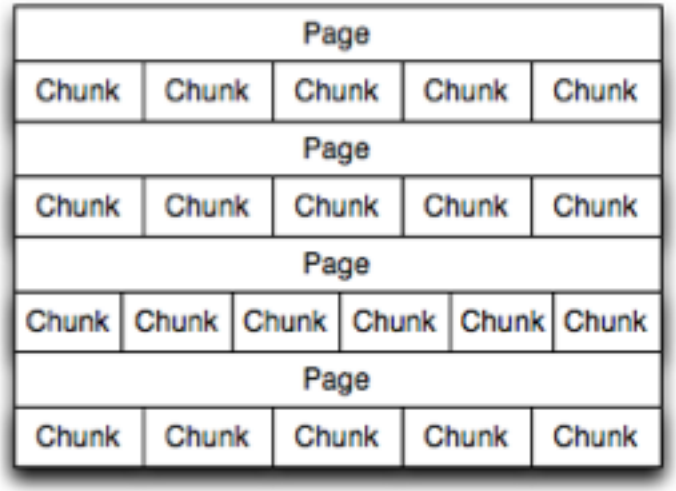
Instant schema update, cont'd

Start Time	Description	State	
Fri Nov 18 12:37:22 PST 2011	Checking alter schema request status for table = miweng_active	COMPLETE (since 5sec ago)	All region servers have successfully processed the schema changes for table = sea-lab-0,60020,1321647186776 sea-lab-4,60020,1321647187290 sea-lab-2,6 4,60020,1321647187290 sea-lab-2,60020,1321647187339 sea-lab-1,60020,13 2,60020,1321647187339 sea-lab-1,60020,1321647187258 sea-lab-3,60020,13 1,60020,1321647187258 sea-lab-3,60020,1321647188530 sea-lab-0,60020,13 3,60020,1321647188530 Total number of regions = 75 Processed regions = 75
Fri Nov 18 12:37:21 PST 2011	Creating schema change node for table = miweng_active	COMPLETE (since 5sec ago)	Created the ZK node for schema change. Current Alter Status = state= INPRO 0,60020,1321647186776 sea-lab-1,60020,1321647187258 sea-lab-2,60020,13
Fri Nov 18 12:37:21 PST 2011	Handling alter table request for table = miweng_active	COMPLETE (since 5sec ago)	Created ZK node for handling the alter table request for table = miweng_active

User Table	
miweng_active	{NAME => 'miweng_active', FAMILIES => [{NAME => 'CHANGE_SET', VERSIONS => '2147483647', COMPRESSION => 'LZO', MIN_VERSIONS {NAME => 'f1', MIN_VERSIONS => '0'}]}

Off Heap Cache

- SlabCache uses DirectByteBuffers
- Garbage collection pauses are greatly reduced
- We manage the eviction policy
- Its slab is allocated following memcached's model

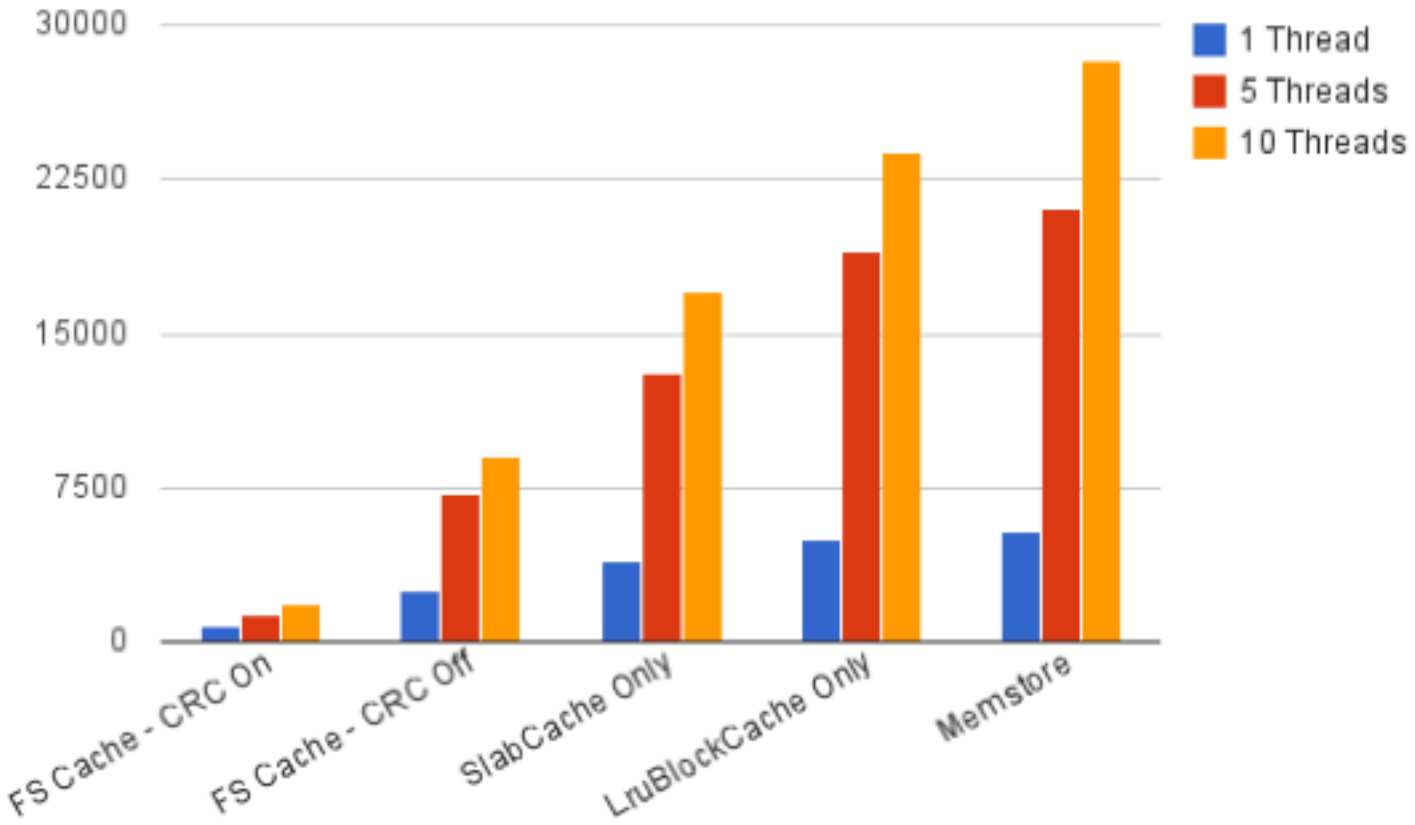


How SlabCache is integrated

- DoubleBlockCache - equivalent of L1 and L2 cache
- LruBlockCache = L1 Cache, on heap, and very fast
- SlabCache = L2 Cache, off heap, but still quite fast
- Write path: write to both caches
- Read path: check L1 cache, if we miss check L2 cache
- If hit in L2 Cache, copy the entry to L1 cache and return

Improvements

- Vertical axis is requests per second



Atomicity across multiple column families

- HBASE-2856 provides strong consistency across column families
- Stores memstoreTS in Hfile
- Took more than 3 months to develop, through multiple sub-tasks
- Backported to 0.92

Test Categorization

- Small tests are executed in a shared JVM - the maximum execution time for a test is 15 seconds and they do not use a cluster
- Medium tests are executed in a separate JVM - They're designed to run less than 50 seconds and can use a cluster
- Large tests are everything else. They are typically integration and regression tests
- See '**unit tests - pom.xml & surefire changed - categories are available'** from N Keywal
- See HBASE-4712

Modularizing HBase

- Currently there are two builds for 0.92 and 0.94, one for insecure Hbase, one for Hbase with security
- HBASE-4336 would modularize maven build
- Multiple pom.xml, one for each module: core, integration tests, server
- There would be separate jar files for different modules: core, server, security
- Work in progress

Q & A