HBASE-11331 [blockcache] lazy block decompression Report

Compare to current master blockcache

Report comparing master/trunk with *HBASE-11331 compressed blockcache* applied to current uncompressed blockcache.

1 Test Description

Ten clients in a single-context doing an all in-cache PerformanceEvaluation randomread against a single regionserver sitting on a 5-node HDFS. Keys are default regular 1k random content values FAST_DIFF prefix-encoded and LZ4 compressed followed by a part in-cache and then mostly missing the cache. We run first w/ unpatched master/trunk and then against a patched master/trunk.

Here's my little script that I run on a node to run the ten clients. We do a size that fits in-cache, then one that is part in-cache, and then another that is mostly out-of-cache. We do a little one minute runup, wait a minute, then run the test. If it does not finish inside 15minutes, we kill it.

```
#!/bin/sh
HOME=/home/stack
testtype=$1
date=`date -u +"%Y-%m-%dT%H:%M:%SZ"
echo testtype=$testtype $date` >> nohup.out
HBASE_HOME=$HOME/hbase-2.0.0-SNAPSHOT
warmuptime=120
runtime=900
for test in 10; do
```

```
  for i in 2.8 10 100; do
    sleep 60
    echo "`date` run $test $testtype warmup time=$warmuptime size=$i" >> nohup.out
    timeout $warmuptime nohup ${HBASE_HOME}/bin/hbase --config /home/stack/conf_hbase
org.apache.hadoop.hbase.PerformanceEvaluation --nomapred --valueRandom --size=$i
randomRead $test
    sleep 60
    echo "`date` run $test $testtype time=$runtime size=$i" >> nohup.out
    timeout $runtime nohup ${HBASE_HOME}/bin/hbase --config /home/stack/conf_hbase
org.apache.hadoop.hbase.PerformanceEvaluation --nomapred --valueRandom --size=$i
randomRead $test
  done
done
#mv $HOME/nohup.out $HOME/nohup.$testtype.$date
```

We are running block cache (CombinedBlockCache). Small heap of 8G. 5G of offheap.

```
<!--LRU Cache-->
<property>
<name>hfile.block.cache.size</name>
<value>0.5</value>
</property>

<!--Bucket cache-->
<property>
  <name>hbase.bucketcache.ioengine</name>
  <value>offheap</value>
</property>
<property>
  <name>hbase.bucketcache.size</name>
  <value>5120</value>
</property>
<property>
  <name>hbase.bucketcache.percentage.in.combinedcache</name>
  <value>0.8</value>
</property>
```

2 Finding

More GC. More CPU. Slower. Less throughput. But less i/o.

What would happen if a hot block was put back into the cache uncompressed so we did not decompress it over and over again? Or take it out of L2 and put it into L1 uncompressed?

3 Pictures

3.1 Chronology:

3.1.1 Clean trunk/master uncompressed

Thu Jul 3 21:37:46 PDT 2014 run 10 nopatch warmup time=120 size=2.8
Thu Jul 3 21:40:46 PDT 2014 run 10 nopatch time=900 size=2.8

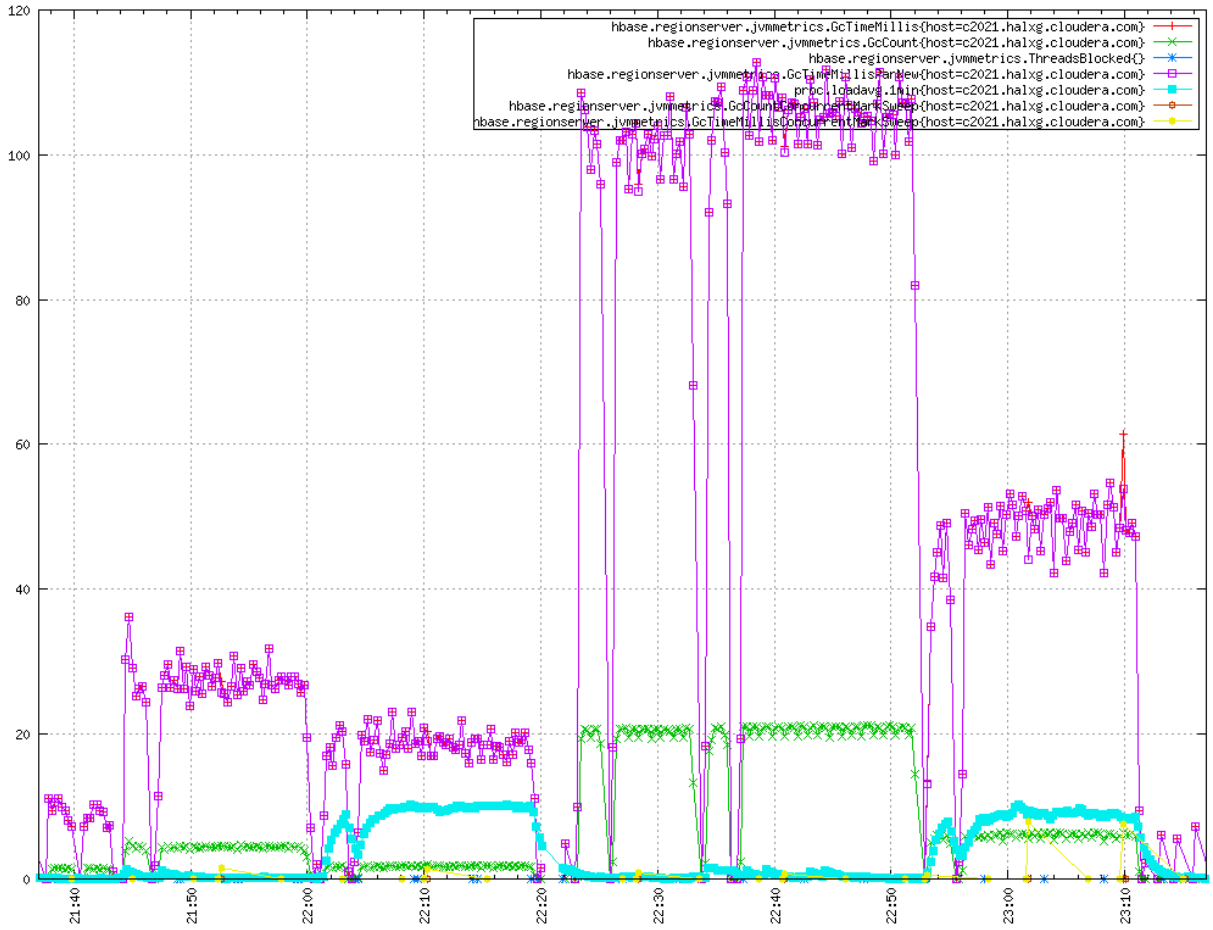
Thu Jul 3 21:44:07 PDT 2014 run 10 nopatch warmup time=120 size=10
Thu Jul 3 21:47:07 PDT 2014 run 10 nopatch time=900 size=10
Thu Jul 3 22:01:16 PDT 2014 run 10 nopatch warmup time=120 size=100
Thu Jul 3 22:04:16 PDT 2014 run 10 nopatch time=900 size=100

3.1.2 Patched trunk/master compressed

Thu Jul 3 22:23:06 PDT 2014 run 10 patched warmup time=120 size=2.8
Thu Jul 3 22:26:06 PDT 2014 run 10 patched time=900 size=2.8
Thu Jul 3 22:33:58 PDT 2014 run 10 patched warmup time=120 size=10
Thu Jul 3 22:36:58 PDT 2014 run 10 patched time=900 size=10
Thu Jul 3 22:52:59 PDT 2014 run 10 patched warmup time=120 size=100
Thu Jul 3 22:55:59 PDT 2014 run 10 patched time=900 size=100

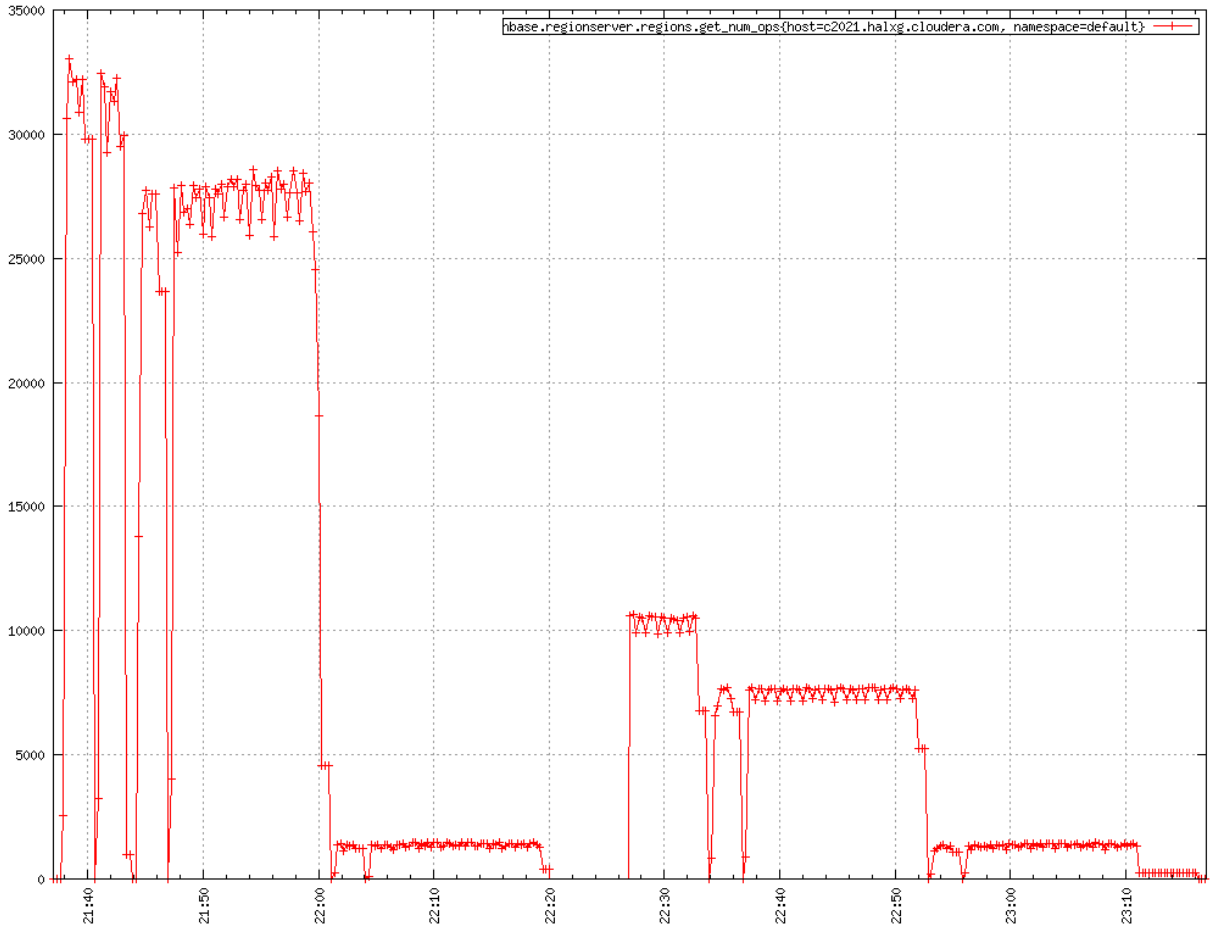
3.2 GC

Datablocks are in L2. Metablocks in L1 (CombinedBlockCache). They are compressed and encoded in L2 at least (and in L1? TODO). When reading all in-cache, lots fo GC decompressing compressed blocks over and over again. Way more when running with the HBASE-11331 patch (its run starts at around 22:23.



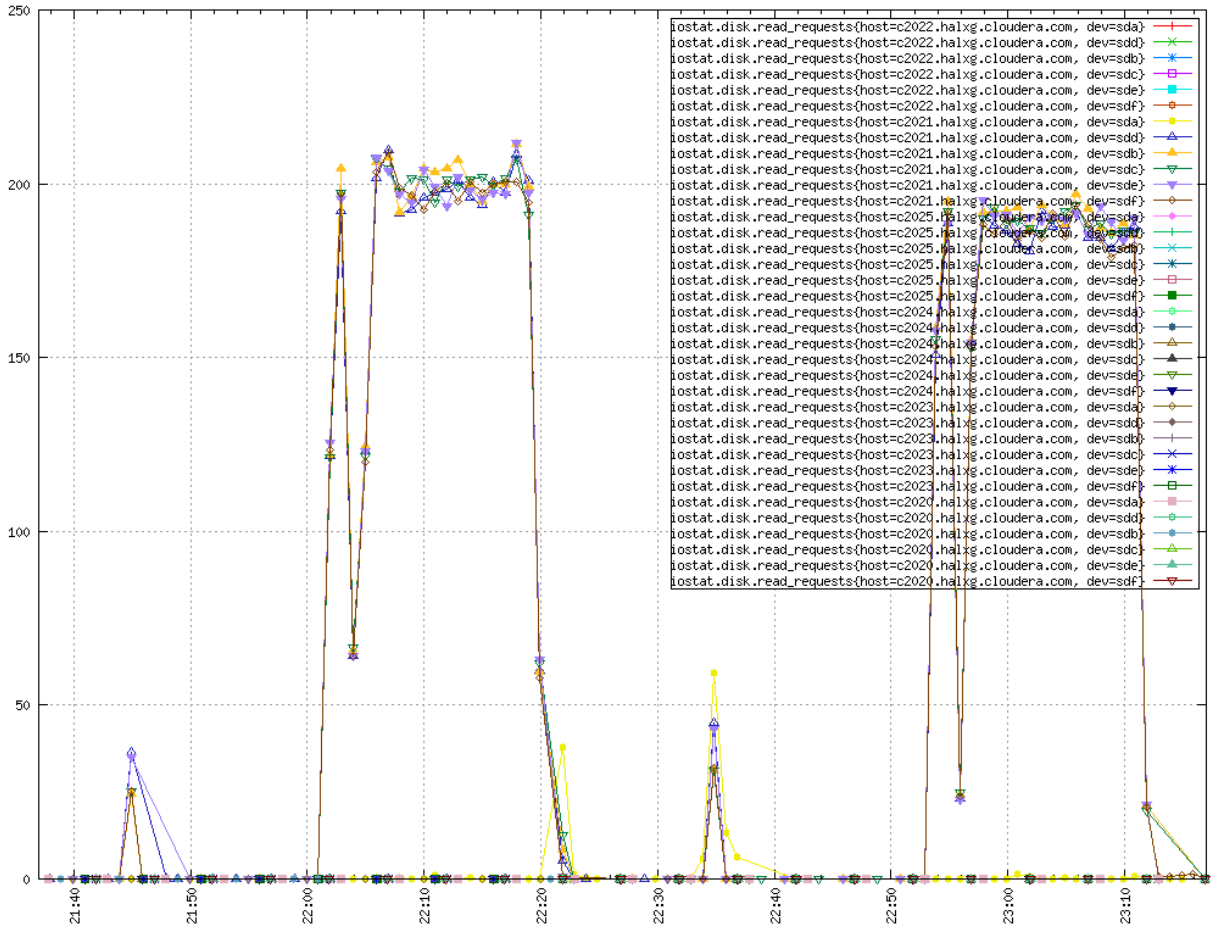
3.3 Gets per second

Many more gets-per-second when not compressed.



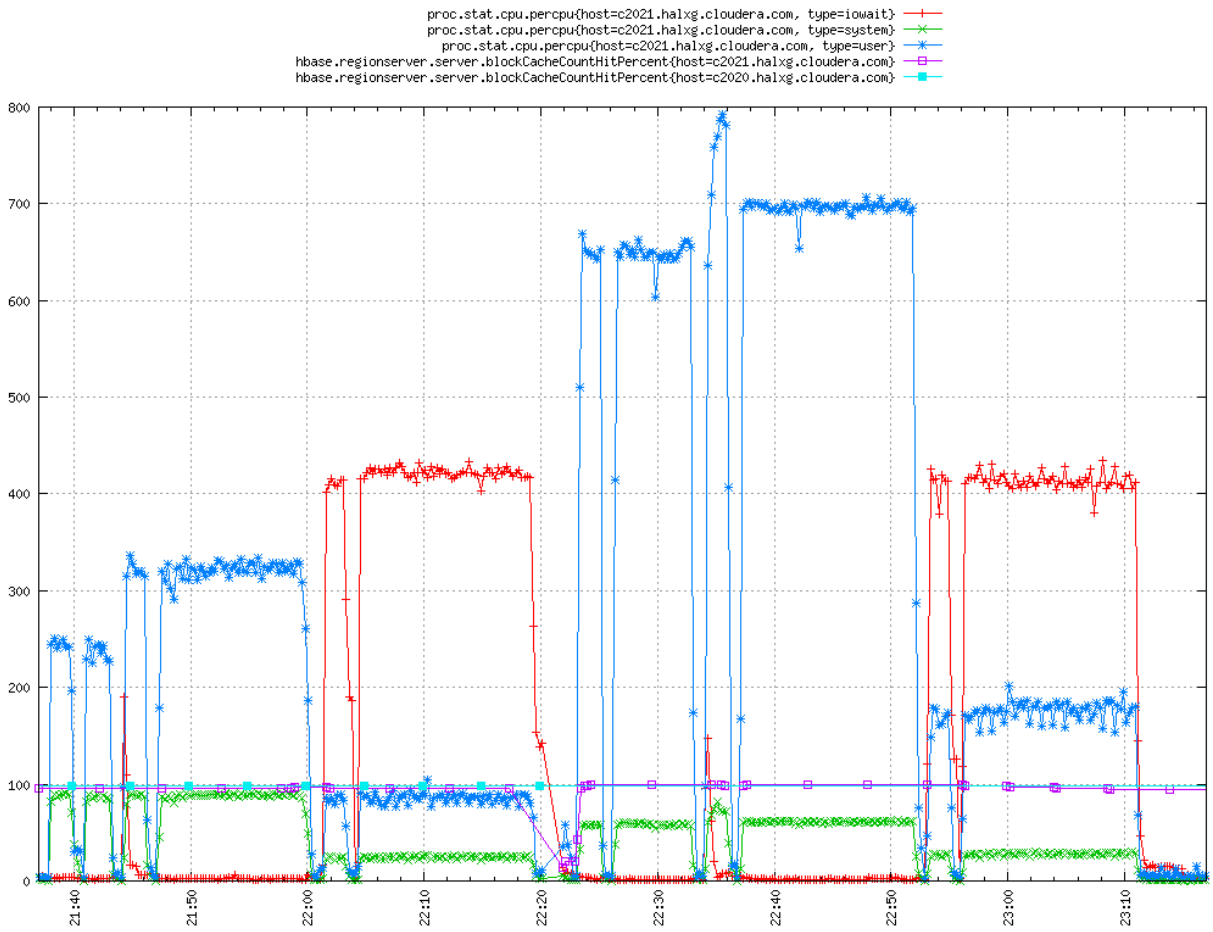
3.4 i/o

Less i/o when running with compression enabled.



3.5 CPU

More CPU when running compressed.

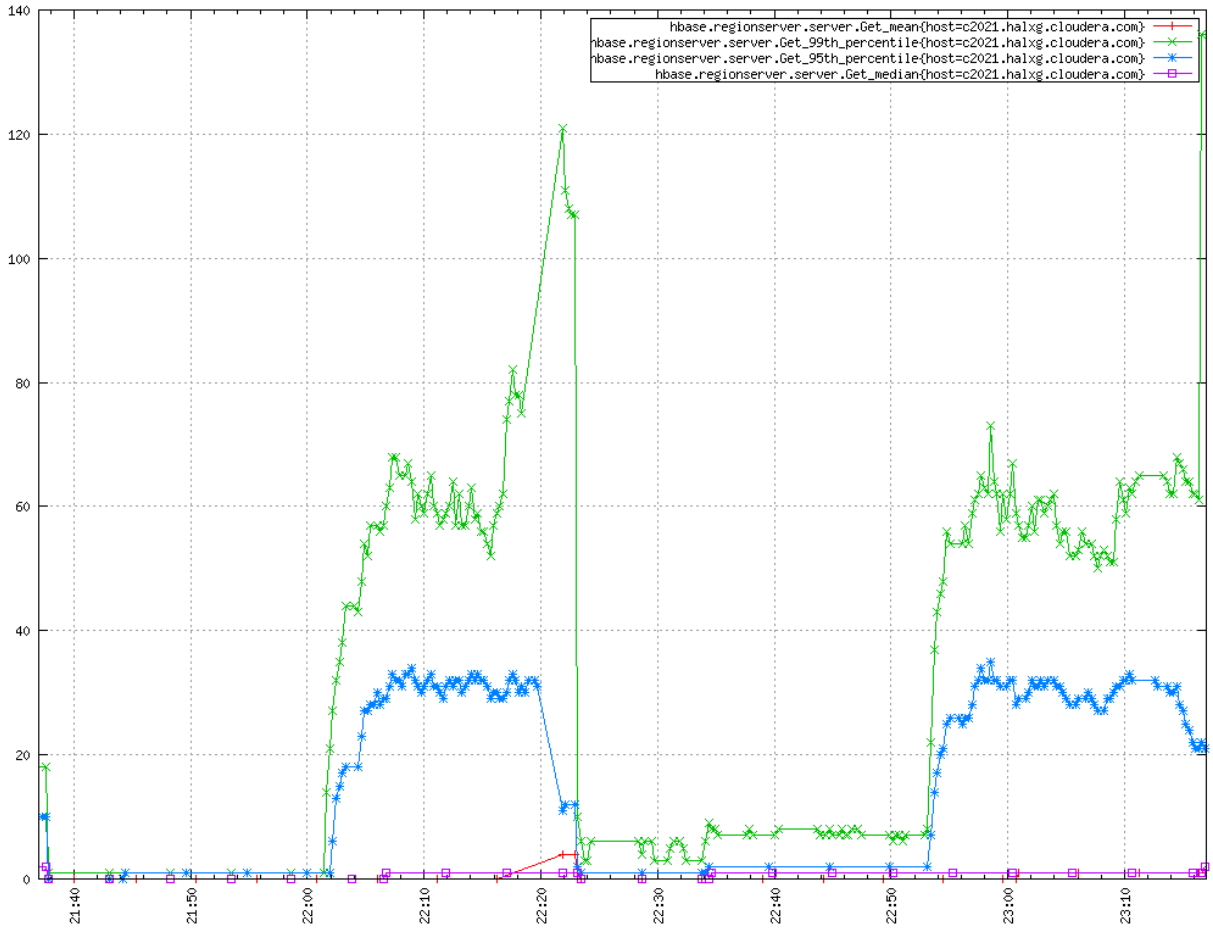


3.6 BC

This one is a bit odd. Slightly less evicted when compressed but we run evictions more often. Looks like we might have been missing cache when we were supposed to be in-cache when we were running unpatched/uncompressed.

3.7 Percentiles

About the same for both.


hbase.regionserver.server.Get_mean{host=c2021.halxg.cloudera.com}
hbase.regionserver.server.Get_99th_percentile{host=c2021.halxg.cloudera.com}
hbase.regionserver.server.Get_95th_percentile{host=c2021.halxg.cloudera.com}
hbase.regionserver.server.Get_median{host=c2021.halxg.cloudera.com}
140
120
100
80
60
40
20
0
21:40
21:50
22:00
22:10
22:20
22:30
22:40
22:50
23:00
23:10