HowToProfile

(this page has been created from an earlier document, while transferring the contents some of the commands listed here got truncated, they need to be fixed.)

How to profile pig on map reduce clusters

hadoop profile doc outlines the methodology for Java Map reduce jobs.
The following command line is a sample one for use with pig:
java -Dmapred.task.profile.maps=0-0 -Dmapred.tasks.profile.reduces=0-0 -Dmapred.task.profile=true -Dmapred.task.profile.params=<-agent.....> -cp <pig.jar pathname>:<dir containing of hadoop-site.xml> org.apache.pig.Main <pig script>

The <-agent..> is the relevant profiler specific option you would supply on the java command line

YourKit

YourKit is a commercial Java profiler. It has really good analysis features and more importantly much better performance (lesser impact on run time of your program). Yourkit has granted license for use by pig contributors. Contact another committer or pig pmc (private@pig.apache.org) for access to the licence key. You can download it from yourkit's website.

Set up

Get hold a Yourkit distribution (I have used yjp-7.0.7.zip), extract it and get hold of the license key. The key is required to use the UI which is the only way to examine the profile output.
Let us assume the Yourkit distribution has been extracted to BASEDIR. Assuming you are running on Linux and using version 7.0.7, the profiling agent is at: BASEDIR/yjp-7.0.7/bin/linux-x86-32/libyjpagent.so This is the profiling agent which can be provided with the java commandline. To use yourkit on a cluster, the above file needs to be copied to a readable location on each of the nodes of the cluster. Lets assume that this location is CLUSTER_BASEDIR.

Usage

See http://yourkit.com/docs/index.jsp for detailed docs.

CPU profiling

Here's a quick overview based on one developer's experience To use yourkit for a **standalone** java program - say pig on local file system, the commandline to use is:
java -agentpath:BASEDIR/yjp-7.0.7/bin/linux-x86-32/libyjpagent.so=dir=/tmp/yourkit_snapnshot,tracing,disablealloc,disablej2ee -cp <location of pig.jar> org.apache.pig.Main <pigscript>
In the above command line /tmp/yourkit_snapshot is the output directory into which yourkit outputs a ".snapshot" file. You can specify any directory to which you have write permissions. Yourkit seems to create the final dir in the path specification if it does not exist. The "tracing" option means that yourkit will trace the method calls to provide profile information (this gives accurate invocation counts since it is achieved by tracing every method call and is not based on sampling - which has the side effect that it is slower).

Pig Profiling "disablealloc" option means memory allocations are not traced. "disablej2ee" means j2ee specific profiling is disabled.
Using yourkit on a pig script running on a*cluster* in sampling mode: java -Dmapred.task.profile.maps=0-0 -Dmapred.task.profile.reduces=0-0 -Dmapred.task.profile=true -Dmapred.task.profile.params=-agentpath:CLUSTER_BASEDIR/libyjpagent.so=dir=/grid/0/tmp/yourkit_snapnshot,sampling,disablealloc,disablej2ee -cp <pig.jar pathname>:<dir containing of hadoop-site.xml> org.apache.pig.Main <pig script>

Using yourkit on a pig script running on a*cluster* in tracing mode:

- you need to disable the filter so that org.apache.* is also traced
- specify value for mapred.max.split.size smaller than block size, so that the map task has smaller input and finishes sooner.
- specify value for mapred.task.timeout so that it does not timeout

java -Dmapred.max.split.size=10000000 -Dmapred.task.timeout=60000000 -Dmapred.task.profile.maps=0-0 -Dmapred.task.profile.reduces=0-0 -Dmapred.task.profile=true -Dmapred.task.profile.params=-agentpath:CLUSTER_BASEDIR/libyjpagent.so=dir=/grid/0/tmp/yourkit_snapnshot,filters=/dev/null,tracing,disablealloc,disablej2ee -cp <pig.jar pathname>:<dir containing of hadoop-site.xml> org.apache.pig.Main <pig script>

With the above cmd, 0th mapper and reducer tasks are profiled and on the cluster machines running those tasks, a yourkit snapshot file is created at /grid/0/tmp/yourkit_snapnshot. This should be copied to the machine with the yourkit gui and loaded using the GUI to look at the profile informaiton. In the above cmd, "sampling" is used - to use tracing instead replace sampling with tracing in the above command.

Memory profiling

- Same steps as in CPU profiling with modified commandline to **NOT** disable memory allocation tracing:
  java -Dmapred.task.profile.maps=0-0 -Dmapred.task.profile.reduces=0-0 -Dmapred.task.profile=true -Dmapred.task.profile.params=-agentpath:CLUSTER_BASEDIR/libyjpagent.so=dir=/tmp/yourkit_snapnshot,sampling,disablej2ee -cp <pig.jar pathname>:<dir containing of hadoop-site.xml> org.apache.pig.Main <pig script>

GUI

- The GUI to view the profile output is present in: BASEDIR/yjp-7.0.7/bin/yjp.sh
- org.apache is not something most yourkit users are interested in exploring, so they are filtered out by default in the display. You need to click on Settings | Filters, and uncheck org.apache .

- On mac, the GUI does not work with java 1.6 . If you have java 1.6 as default, set export JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/1.5/Home/ , to use 1.5 instead .

HPROF

Hprof is the Sun profiler which comes with the JDK. It has many options for CPU and memory profiling. A general observation though is that the profiling using the "sampling" option doesn't seem to be accurate and that profiling using "tracing" (cpu=times or cpu=old) option is accurate but very very slow.

Usage

See http://java.sun.com/developer/technicalArticles/Programming/HPROF.html for all the gory details - skip to the next part of this section if you just want it to work but don't want the gory details.

CPU profiling

Some quick command lines:

CPU profiling using sampling:
java -agentlib:hprof=cpu=samples,interval=1,file=<outputfilename> -cp <pig.jar pathname> org.apache.pig.Main <pig script>
The above command line means that hprof will sample the running program at 1 ms interval times and provide profiling information. The observation has been that this output is not very accurate but sampling is fast and has very less impact on the running time of the pig script. The above command runs the script in one JVM locally. To profile ion a hadoop cluster see the topic above.

CPU profiling using tracing:
java -agentlib:hprof=cpu=times,file=<outputfilename> -cp <pig.jar pathname> org.apache.pig.Main <pig script>
The above command line means that hprof will inject byte codes and time each method call. So this is much much slower but very accurate.

Another option with tracing is to use the "old" option which is similar to the above but outputs in an older format.
java -agentlib:hprof=cpu=old,file=<outputfilename> -cp <pig.jar pathname> org.apache.pig.Main <pig script>

Using Hprof on pig script running on a cluster:

java -Dmapred.task.profile.maps=0-0 -Dmapred.tasks.profile.reduces=0-0 -Dmapred.task.profile=true -Dmapred.task.profile.params=-agentlib:hprof=cpu=times,file=<outputfilename> -cp <pig.jar pathname>:<dir containing of hadoop-site.xml> org.apache.pig.Main <pig script>

It is better for the <outputfilename> to be a location on /tmp or some writeable location - if it is relative, it is written to the tasks current working directory which is quite a long and not so easy to find pathname 🙂

Memory profiling

Memory profiling using sap memory analyzer

Step 1: Get heap dump:

1. In the pig commandline specify - -Dmapred.child.java.opts='-XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/tmp/pigtest.hprof -Xmx700M' which tells java to create a heap dump when OOM. Set -Xmx value in these params as desired, heap dump size will depend on this.

2. I am not sure but you may need also use keep.failed.task.files, otherwise dump file will be removed quickly. <I> I(thejas) could not prevent the files from being deleted even with that setting, so I used "-XX:HeapDumpPath=/dir/" also (in mapred-site.xml along with -XX:+HeapDumpOnOutOfMemoryError) to write to "dir" instead. Avoid using "/tmp" as the "dir".

3. In map-reduce UI, find out which machine the task is running on, go to that machine, in the directory $HADOOP_TMP/mapred/local/taskTracker/jobcache/jobidxxxx/attemptidxxxxx/work, you will find the dump file with the name java_pidxxxx.hprof . On STDOUT for the task in WEBUI, you can see the exact file location .

Step 2 (option 1) : use yourkit - it is able to deal with dumps that are larger than 1GB (it worked with a 1.4GB dump), but hung when it was trying to open a 2.1 GB dump on mac laptop with 4GB RAM. It seems yourkit can be used as a plugin from within eclipse as well (but i don't know if that will consume the limited number of licenses that yahoo has, even when you don't use yourkit). Open the .hprof snapshot file with yourkit to analyze the dump.

Step 2 (option 2) : Install memory analyzer in eclipse:

1. In eclipse, go to Help->Software update, add new update site: http://download.eclipse.org/technology/mat/0.8/update-site/

2. Follow the instruction to install memory analyzer. For detail, you can refer to http://www.eclipse.org/mat/downloads.php

3. Open perspective Memory Analysis, in the file menu, you will see a new entry "Open heap dump..."

4. Open the heap dump file we've got in step 1

5. There are multiple ways to view the memory, my favorite one is dominator tree, which gives you the largest memory consumer and percentage

6. The biggest problem is OOM in eclipse. Memory analyzer requires almost equal amount of memory to analyze the heap. If the heap dump is 1G, you need 1G memory to process that. If your installed memory is larger than the size of dump file, then you are good.