CloudMix User Manual

Table of Contents

1. Installation and Configuration of Software

1.1. Setting up Centos7.2

Please refer to: http://www.111cn.net/sys/centos/63645.html

1.2. Setting up JDK1.7

Please refer to: http://www.centoscn.com/image-text/install/2014/0827/3585.html

1.3. Setting up Python 2.7

Please refer to: http://www.cnblogs.com/dudu/p/4294238.html

1.4. Setting up gcc4.8.5

Please refer to: http://blog.csdn.net/majianfei1023/article/details/46811159

1.5. Setting up mysql5.5

Please refer to: http://jingyan.baidu.com/article/922554465e471a851648f4ed.html

1.6. Setting up perf

Please refer to: https://fukun.org/archives/03172543.html

1.7. Setting up Oprofile

Please refer to: http://xenoprof.sourceforge.net/

1.8. Setting up Apache Maven

Please refer to: http://maven.apache.org/download.cgi

1.9. Setting up Hadoop 2.7.3

Please refer to: http://hadoop.apache.org/releases.html

2. Software Deployment

Download the CloudMix software package from the BigDataBench website: http://prof.ict.ac.cn/BigDataBench/wp-content/uploads/2014/07/CloudMix.zip

2.1. Repository of SWBs

All the Reducible Workload Blocks (RWBs) in the fold:
/CloudMix/SWB_repository/corefunction/

These SWBs include 13 CPU RWBs(core1_new.s to core13_new.s) and 31 memory RWBs (core14_new.s-core44_new.s).

2.2. Profiling tools

The profiling tools are available in the fold:
CloudMix/ SWB_profiler/perfscripts/

There are four profiling scripts: cpu.py, cpi.py, mem.py, and mai.py, which are developed to test the workload behaviors of CPU usage, CPI, memory usage, and MAI, respectively.

```
$python mem.py mem log time
$python cpu.py cpu log time
$python cpi.py log time pid
$python mai.py log time pid
```

2.3. Workload traces

Downloading the 24-hour Google trace stored in MySQL database (1.8GB) from http://prof.ict.ac.cn/bdb_uploads/Multi-tenancy/Workload_Trace_24hours.tar.gz

(*Please contact us if you need the full version of workload trace stored in Impala (57GB)*)

2.4. Workload replication script generation

All the codes used to generate the replication scripts are in the fold:
/home/CloudMix/SWB_combiner /

2.5. Run job on YARN

If you want to run jobs on YARN, please install YARN Job Submitter.

Step 1:

Unpackage Jobsubmitter-src.tar.gz to /usr/local/, if you want to change the path, you need to modify "<jarPath>" in $JOBSUBMITTER_HOME/conf/example-job.xml

```
$ tar -zxvf Jobsubmitter-src.tar.gz
$ mv ./Jobsubmitter /usr/local/
```

Step 2: Add JOBSUBMITTER_HOME to environment.

```
$ export JOBSUBMITTER_HOME=/your-path/Jobsubmitter
```

Step 3: Enter the Jobsubmitter directory, make sure to "chmod +x" the build scripts and run it.

```
$ chmod +x build.sh
$ ./build.sh
```

3. Experimental Evaluation

3.1. Experimental Platform

The evaluation is conducted on the Huawei platform whose configuration details are listed in Table 1.

CPU type	Intel ®Xeon E5645
Number of processors	2
Cores per process	6
Frequency	2.4GHz
L1 DCache	6 x 32KB
L1 ICache	6 x 32KB
L2 Cache	6 x 256KB
L3 Cache	12MB
Memory size	32GB

Table 1. Configuration details of the experimental platform

3.2. Step 1: Trace selection

We demonstrate the effectiveness of CloudMix using a one-day workload trace from the Google datacenter. At this step, we selected one hour workloads from the trace.

Step 1.1: download the 24-hour Google trace, and load the data to the MySQL database

```
    $ mysql/ mysql –u root –p passwork Enter mysql
    mysql> create database google;
    mysql> use google;
    mysql> source …/google_trace.sql；
    mysql> source …/google_trace_data.sql；  //this may take a long time
```

Step 1.2: Workload trace selection

```
    insert into testdata801 select
jobid,starttime/1000000,endtime/1000000,machineid,cpuusage,memoryusage,cpi,mai
from task_usage where machineid=3349157760 and cpuusage>0.01 and cpuusage<1
and memoryusage>0.01 and memoryusage<1 and cpuusage!=0 and cpi!=0 and
memoryusage!=0 and mai!=0 and (endtime-starttime)<=300000000 and
endtime>=27896000000 and starttime<=31496000000;//One hour trace
```

3.3. Step 2: SWB profiling

The profiling tools can either be used to test the workload behaviors of individual SWBs, or test the workload behaviors of combined SWBs.

Step 2.1: Put the SWBs in the fold: /CloudMix/SWB_profiler /

```
    $./corefunctiongroup.sh
```

Step 2.2: Enter the fold: /CloudMix/SWB_profiler /perfscripts/

```
    $python mem.py mem log time
    $python cpi.py log time pid
    $python mai.py log time pid
```

Where log is the output of the profiling results, time is the time interval of profiling, pid is the process id of the SWB to be tested.

3.4. Step 3. Workload replication script generation

The generation of replication scripts and workloads have three steps:

Step 3.1: Optimal SWB combination.

Enter the /CloudMix/Examples/Original/ fold

```
    $./corefunctionmatch.sh ./testdata ./corefunctionorder/
```

Step 3.2: Generate the replication script.

Enter the /CloudMix/Examples/Original/ fold

```
    $./corefunctiongroup_min.sh  ./corefunctionorder  ./corefunctiongroup
```

```
$./loadreplay.sh ./testdata
```

Step 3.3: Generate the synthetic workloads according to the script.

Enter the /CloudMix/Examples/Original/ fold

```
$./corefunctiongroup.sh
```

3.5. Workload generation with duration reductions

This evaluation tested four reductions of evaluation durations: the durations are reduced by 2, 5, 10, and 100 times, respectively.

The detailed steps are as follows:

Step 1: Enter the /CloudMix/ Examples /

Step 2: Modify the iteration number of assembly language instructions in each SWB. In the four tests, the modified SWBs are stored in the folds /1_2Reduction/corefunction_backup0, /1_5Reduction/corefunction_backup00, /1_10Reduction/corefunction_backup000, and /1_100Reduction/corefunction_backup0000, respectively.

Step 3: Generate the replication script

```
// Reduce the duration by 2 times
$cd 1_2Reduction/
$./corefunctiongroup_min0.sh   ./corefunctionorder   ./corefunctiongroup0
$./loadreplay0.sh ./testdata

// Reduce the duration by 5 times
$cd 1_5Reduction/
$./corefunctiongroup_min00.sh   ./corefunctionorder   ./corefunctiongroup00
$./loadreplay00.sh ./testdata

// Reduce the duration by 10 times
$cd 1_10Reduction/
$./corefunctiongroup_min000.sh   ./corefunctionorder   ./corefunctiongroup000
$./loadreplay000.sh ./testdata

// Reduce the duration by 100 times
$cd 1_100Reduction/
$./corefunctiongroup_min0000.sh   ./corefunctionorder ./corefunctiongroup0000
$./loadreplay0000.sh ./testdata
```

4. Job Scheduling Optimization on YARN

Jobsubmitter is a YARN application which can run workloads on YARN as configuration. Workloads setting in XML format in the directory ./conf/ and executable workloads in ./tasks/.

Before all the steps below, make sure you have start up YARN and HDFS service with proper configurations.

```
$ HADOOP_HOME/sbin/start-dfs.sh
$ HADOOP_HOME/sbin/start-yarn.sh
```

4.1 Run CloudMix Workload

Step 1: Run CloudMix workload.

If you want to run CloudMix workload, you need to compile some assemble program. Enter $JOBSUBMITTER_HOME

```
$ chmod +x ./sh/*.sh
$./sh/workload-generation.sh
```

This command will generate executables in the directory ./tasks/assembly. The workloads' configuration files in ./conf.

Step 2:

① Run all 44 jobs in one queue, you can use YARN default configuration. Logs are put at ./logs. The detail run parameters please view the script ./sh/rundefault.sh

```
$ ./sh/rundefault.sh
```

② Run 44 jobs in two queue. Script below will configured your YARN to two queues which are named "A" and "B".

We select 12 settings for both Capacity Scheduler and Fair Scheduler, as shown in Table 2 and Table 3.

Num	Capacity	Maximum-capacity
1	A=20，B=80	A=40，B=80
2	A=20，B=80	A=60，B=80
3	A=20，B=80	A=80，B=80
4	A=30，B=70	A=40，B=70
5	A=30，B=70	A=60，B=70
6	A=30，B=70	A=80，B=70
7	A=40，B=60	A=40，B=60
8	A=40，B=60	A=60，B=60
9	A=40，B=60	A=80，B=60
10	A=50，B=50	A=50，B=50

11	A=50，B=50	A=60，B=50
12	A=50，B=50	A=80，B=50

Table 2. Configuration of Capacity Scheduler

Num	Weight	SchedulingPolicy
1	A=1, B=7	Fair
2	A=2, B=6	Fair
3	A=3, B=5	Fair
4	A=4, B=4	Fair
5	A=1, B=7	Fifo
6	A=2, B=6	Fifo
7	A=3, B=5	Fifo
8	A=4, B=4	Fifo
9	A=1, B=7	Drf
10	A=2, B=6	Drf
11	A=3, B=5	Drf
12	A=4, B=4	Drf

Table 3. Configuration of Fair Scheduler

Note: all the other YARN configuration parameters are put in ./schedulerxmls/.

The shell script “gen-best.sh” can run workloads at “queueA” and “queueB”.

```
$ ./sh/gen-best.sh [fair/capacity] [start-configuration] [end-configuration]
```

- Parameter 1: “capacity” or “fair”, declare what scheduler you want to use.
- Parameter 2: Index of scheduler setting file. The “index” is the number at the end of configure file’s name.
- Parameter 3: Index of scheduler setting file. The “index” is the number at the end of configure file’s name.

Capacity Scheduler

For example, if you want to run workloads with Capacity Scheduler’s configuration from “capacity-scheduler-1.xml” to “capacity-scheduler-12.xml”, then just run the command follows.

It will run the workloads with each configuration in sequence and generate task information in ./logs.

```
$ ./sh/gen-best.sh capacity 1 12
```

Fair Scheduler

If you want use “fair-scheduler-2.xml” in ./schedulerxmls/ with Fair Scheduler, just run:

```
$ ./sh/gen-best.sh fair 1 12
```

Running result.

All logs in ./logs, and LogAnalysis.jar will create a result file of log parsing which is named “capacity-res” or “fair-res” in Job Submitter’s root directory.

4.2 Run other workloads in addition to Google jobs

CloudMix’s workloads were selected from Google Trace. You can select other workloads information from trace and write the XML file in same format just like ./conf/example-job.xml. And you need to instead assemble program by other workloads(executables) you want to run.

For instance, script ./sh/runexampletask.sh run a bash as workload and you can set other types of workload, please view the script for detail.

```
$ ./sh/runexampletask.sh
```