Learning System to Classify Batch Jobs

Yutaka Kawai(1), Adil Hasan(2), Wataru Takase(1), and Takashi Sasaki(1)
(1)High Energy Accelerator Research Organization (KEK), Japan
(2)Kings’ College London, University of London, UK

Abstract
Many disciplines require large amounts of computing resources to process their data. Jobs are usually categorized according to their processing time which is a combination of factors (CPU, I/O, disk space etc.). For the user it is helpful to have a more refined categorization according to individual components such as CPU utilization, I/O usage, memory usage etc. which allows the user to group their job according to resource utilization.

In this poster we describe an approach to enable a grouping based on the individual components that makes efficient use of the computing resources from the perspective of the user. The approach makes use of the Torque job submission system as well as Python applications to achieve this.

Concept of the learning system
- Learning System Server (LSS) exports NFS
- Each compute node of Local Batch Scheduler (LBS) monitors cpu/memory utilization during job executions
- The monitored statistical data are stored in NFS
- LSS DB contains the historical data with using the statistical data
- LSS judges the job type based on the historical data in DB

Why we need to classify Jobs?
- We examine our data by job submissions with lots of computing and memory.
- We require our jobs should be submitted to suitable resources which have enough capacities for the jobs.
- However, it is difficult to know that our jobs mainly consume which resources are consumed before submitting jobs.

We tried to analyze resource usage data during several actual jobs (or small test jobs before running lots of jobs)

Next jobs can be classified as CPU consumption type or Memory one.

Monitoring Tool
“Dstat is a versatile replacement for vmstat, iostat, mpstat, netstat and ifstat.” (*)
Dstat:
- can be easily installed by “yum install dstat”
- simply outputs resource utilizations as csv file
- is Python based application (hope to use API?)
Dstat Options is “-cm”:
- -c : monitors CPU usage
- -m: monitors Memory usage

(*) ref: https://github.com/dagwieers/dstat

DB update
- PostgreSQL and Python are used for DB and language to access DB
- Several useful Python packages:
  - “psycopg2” to access PostgreSQL
  - “csv” to parse statistical data
  - “numpy” to get average values
- Each record contains 4 columns:
  - job_name : Job name
  - cpu : CPU utilization average
  - mem : Memory utilization average
  - job_id : identification value of the job

Statistical data collection
Prepare “prologue” and “epilogue” scripts to stat and kill a monitoring tool with specifying the log location. (This example is the case of using “Torque” as a batch scheduler)

prologue_script.sh
#!/bin/bash
job_id=$1
job_name=$4
datename=`date +%Y%m%d%H%M%S`
ps_no=$!
ps_no=`cat /tmp/dstat_ps`
ps_no=$!
echo "ps_no" > /tmp/dstat_ps
ps_no=`cat /tmp/dstat_ps`

epilogue_script.sh
#!/bin/bash
ps_no=`cat /tmp/dstat_ps`
echo "kill the process : " $ps_no
kill $ps_no

Judgment of job types
- Select CPU and Memory utilization data where “job_name” column is the same
- Get historical averages of CPU and Memory utilization
- Compare CPU vs Memory (which one is used more?)

Future Works
- Study for other resource consumption cases (IO, Network, etc.).
- How to make a judgment when resource usages are almost even.
- Other than “job name” can be considered as a selection key (script name, binary name, including some numbers or arguments, etc.).
- Automatically updates DB, judges job types, and select an ideal resource target whenever submitting jobs.

Acknowledgements
The authors would like to thank to Yoshiyuki Watase and Go Iwai at KEK for constructive discussions to improve slide materials.