Journal of Digital Contents Society Vol. 16 No. 5 Oct. 2015(pp. 691-697)
http://dx.doi.org/10.9728/dcs.2015.16.5.691

HPC 환경에서 사용자 로그 분석을 통한 작업 성공률 개선

윤준원*, 홍태영**, 공기식***, 박찬열****

요 약

슈퍼컴퓨터는 대량의 계산이 필요한 첨단 과학기술분야의 수치계산뿐만 아니라 산업분야의 신제품 설계 및 개발에도 다양하게 접목되어 사용되고 있다. KISTI의 슈퍼컴퓨터 4호기 Tachyon은 SUN Blade 서버기반으로 구성된 초병렬 컴퓨팅 시스템으로 3,200개의 컴퓨팅 노드와 인프라 노드들로 구분된다. 이 시스템은 현재 만여 명의 사용자와 170여개의 기관이 사용 중에 있으며, 수많은 작업들이 스케줄러를 통해 배치형태로 작업을 수행하고 있다. 또한 Tachyon에서는 작업 제출부터 종료까지 관련된 해당 스크립트와 수행 환경, 라이브러리, 작업수행로그 등을 저장하게 된다.

본 논문에서는 스케줄러로 사용되고 있는 Sun Grid Engine의 배치작업정보와 Tachyon 작업수행로그를 가지고 분석을 진행하였다. 특히, Tachyon에서 사용자가 수행했던 작업 결과 중 실패 작업을 구분하여 원인을 분석하고 그중 일부 작업에 대한 개선을 통해 정상 작업을 추출함으로써 시스템의 전체 성공률을 향상시킬 수 있다.

키워드 : 고성능컴퓨팅, 슈퍼컴퓨터, 스케줄러, 배치작업, 로그분석

Improving the Job Success Rate through Analysis of User Logs in HPC

JunWeon Yoon*, TaeYoung Hong**, Ki-Sik Kong***, ChanYeol Park****

Abstract

Supercomputers are used for many different areas including new product design of industries as well as state-of-the-art science and technology for large amount of computational needs. Tachyon is a 4th supercomputer built at KISTI that is a high-performance parallel computing system with 3,200 computing nodes and infrastructures. This system is currently about 10,000 users and over 170 organizations are used, the number of jobs they are performing work in batch type form through a scheduler. Also, this system logs lots of job scripts, execution environment, library, job status from the job submit to end.

In this paper, we analyzed batch jobs information from Sun Grid Engine, that use as a scheduler in Tachyon system, and job executed information in Tachyon System. In particular, we distinguished the fail jobs from the all tasks that users perform and we analyzed the cause of failure. Among them, we can extracted some of jobs that can be regarded as normal jobs through the improvement in those works logged as all of fail jobs.

Keywords : HPC, Supercomputer, Scheduler, Batch job, Log Analysis

1. 서론

KISTI 슈퍼컴퓨터 4호기 Tachyon은 SUN Blade 시스템을 기반으로 구성된 초병렬 컴퓨팅 시스템으로 대기, 환경, 물리, 천문, 열, 유체, 화학, 생명 분야 등의 다양한 계산과학 응용을 수행하고 있다. Tachyon은 3,200대의 컴퓨팅 노드와 인프라 노드로 구성되며 현재 만여 명의 사용자, 170여개 기관이 사용하고 있다[1].

※ Corresponding Author: ChanYeol Park
Received : September 18, 2015
Revised : October 19, 2015
Accepted : October 22, 2015
* Dept. of Supercomputing Center, KISTI
Tel: +82-42-869-0581, Fax: +82-42-869-0569
email: jwyoon@kisti,re.kr
** Dept. of Supercomputing Center, KISTI
*** Dept. of Multimedia, Namseoul University
**** Dept. of Supercomputing Center, KISTI

Tachyon에서는 클러스터의 배치(Batch) 작업 수행을 위한 스케줄러 소프트웨어로 Sun Grid Engine(이하, SGE)를 사용하고 있다[2][3]. 배치 작업을 제출하기 위해서는 사용자는 작업 스크립트 파일을 작성한 후 작업제출 명령(qsub)을 사용하게 되며, 이후 해당 큐(queue)에 대기과정을 거쳐 작업에 대한 계산자원이 확보되면 작업을 수행되게 된다[4].

본 논문에서는 Tachyon에서 수행한 사용자 작업 로그를 분석하고 실패한 작업에 대한 비율과 원인을 파악하였다. 이로써 사용자 또는 시스템 상의 오류가 있었는지를 분석할 수 있으며 실패 작업 중 정상으로 수행된 작업의 비율을 추출하여 전체의 작업 성공률의 비율을 높일 수 있었다.

2. 배 경

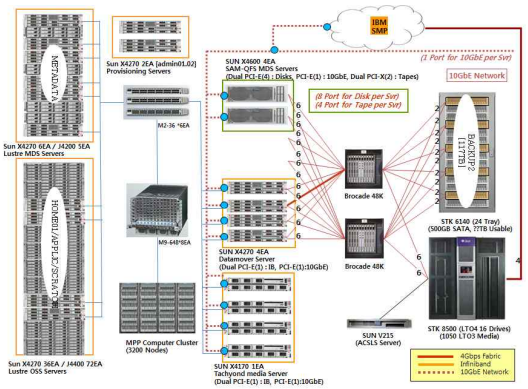
KISTI 슈퍼컴퓨터 4호기 Tachyon2은 SUN Blade 6275 기반으로 구성된 초병렬 컴퓨팅 시스템으로 이론최고성능(Rpeak) 300TFlops를 보이고 있으며, 3,200대의 컴퓨팅 노드와 인프라 노드들(Provisioning, Scheduler, Storage, Interconnector, Archiver, Cooling facilities 등)로 구분된다. (그림 1)은 Tachyon 시스템의 개요도를 나타내고 있다. Tachyon은 RedHat 5.3기반의 리눅스로 운영되고 있으며, 인터커넥션으로는 인피니밴드(Infiniband, 이하 IB) 네트워크의 2-layer, Fat-Tree 토폴로지로 구성이 되어 있다. IB는 계산노드와 병렬파일시스템의 I/O로 사용하고 있으며, 병렬 파일시스템으로 오픈소스 Lustre1.8이 설치되어 있다. Tachyon에서는 클러스터의 배치 작업 수행을 위한 스케줄러로 SGE을 사용하고 있다. 제출된 배치 작업은 스케줄링 정책에 따라 해당 큐(queue)에 대기과정을 거쳐 계산자원이 확보되면 작업을 수행하게 된다[5][6].

3. 배치작업관리

3.1 스케줄러(Sun Grid Engine)

스케줄러는 자원을 공유하기 위해 순서나 일정을 조정해주는 프로그램으로 SGE, Condor, Torque, PBS, LoadLeveler 등 다양한 솔루션들이 있다.

(그림 1) Tachyon 시스템 개요도



(Figure 1) Tachyon System Overview

Tachyon에서는 배치작업을 다루기 위해 SGE를 사용하고 있다. 스케줄러를 통해서 사용자는 작업이 어떤 곳에서 수행될지 신경 쓰지 않고 수많은 작업을 제출할 수 있다[7][8]. 배치 작업을 제출하기 위해서는 (그림 2)와 같이 작업 스크립트 파일을 작성한 후 SGE의 qsub 명령을 사용하게 되며, 이후 해당 큐(queue)에 대기 과정을 거쳐 계산자원이 확보되면 작업이 수행되게 된다. Tachyon에서는 배타적인 노드 할당 정책을 적용하여 한 노드에 한 사용자의 작업만이 실행될 수 있게 설정되어 있다.

(그림 2) 사용자 작업 스크립트

```
#!/bin/bash
#$ -V  //작업제출노드 쉘 환경변수를 컴퓨팅 노드 적용(default)
#$ -cwd //현재 디렉터리를 작업 디렉터리로 사용
#$ -N hybrid_job // 작업명
#$ -pe mpi_4cpu 8 //전체 MPI task 8개, 노드당 MPI task 4개
#$ -q normal     // 작업 수행 큐명
#$ -R yes        // Resource Reservation
##$ -wd /scratch/<user01>/hybrid # 작업 디렉터리를 설정
#$ -l h_rt=10:00:00  //작업 경과 시간 (hh:mm:ss)
                     // (wall clock time), 초과시 강제 작업 종료
#$ -l OMP_NUM_THREADS=2
                     // MPI 프로세스 당 OpenMP 쓰레드 수
#$ -M myEmailAddress  // 작업 관련 사용자 메일 주소
#$ -m e               // 작업 종료 시에 메일을 보냄
export OMP_NUM_THREADS=2

mpirun_rsh -hostfile $TMPDIR/machines -np
$NSLOTS ./hybrid.exe //사용자가 실행시킬 프로그램 파일
```

(Figure 2) User Job Script

3.2 SGE 작업 수행 로그

Tachyon에서는 작업의 시작과 종료시점에 각각 작업 정보를 저장하게 된다. 우선 작업이 제출되고 자원을 할당받아 시작이 되면 사용자의 작업 스크립트를 백업하고 작업이 완료되면 SGE의 로그를 추출, 재처리하여 <표 1>과 같은 형식으로 수행내역을 일별로 저장하게 된다.

여기에는 작업수행일자, 작업ID, 작업명, 작업제출(대기시간)과 시작 그리고 종료된 시간, 수행시간, CPU 사용개수, 종료코드, 실패코드, 쓰레드 개수 등과 같은 기본 작업 정보들이 저장된다. 수행 완료된 작업이 정상 또는 비정상으로 끝났는지를 확인하기 위해서는 결과 내역 중 16번째의 STATUS 필드가 D(Done) 또는 E(Error) 상태인지를 보고 확인할 수 있다.

본 논문에서는 Tachyon에서 수행한 사용자 작업 로그를 분석하고 실패한 작업에 대한 비율과 원인을 파악하였다. 이로써 실패 작업 중 정상으로 수행된 작업의 비율을 추출하여 전체의 작업 성공률의 비율을 높일 수 있다[9].

3.3 수행 로그 분석 절차

본 논문에서는 Tachyon 작업로그 중 일부인 2015년 1월부터 8월까지 수행했던 내역을 대상으로 분석을 다음과 같이 수행하였다.

<표 1> 작업 수행 로그 (작업완료 후)

1	2	3	4	5
DATE	JOBID	GID	UID	JOBNAME
20150821	20150821	na032	r000	jj_007_217
6	7	8	9	10
QNAME	SUBMIT (DATE)	START (DATE)	END (DATE)	WAIT(s)
ocean4special	20121026111256	20121028191152	20121029050924	201536
11	12	13	14	15
RUN(s)	CPUS	CPU USAGE	MEM USAGE	MAXV MEM
35852	80	2834281.74	68.41	1876.06
16	17	18	19	20
STATUS	E-CPU	E-RUN(s)	EXIT-CODE	FAILED
D	80	2868160	0(11)	0(11)
21	22			
OMP_NUM_THREADS	TASK_NUM			
1	0			

<Table 1> Job Execution Log(After the work is finished)

① 실패작업 추출

<표 1>의 형식으로 저장되어 있는 작업 로그들 중에 STATUS 필드가 "E"(Error)인 작업들을 우선 추출하였다. 작업이 정상적으로 종료되지 못한 이유로는 사용자의 작업 강제종료, 작업제출 스크립트의 오류, 작업수행시간제한(Wall-time limit) 초과, 시스템 상의 장애(하드웨어, 네트워크, 전원 등)등이 대표적이다[10].

② "E" 상태의 실패작업 분석

Tachyon에서는 작업 성공률은 정상적으로 종료된 즉, 상태가 "D"(Done)로 종료된 작업만을 대상으로 하고 있다. 하지만 실패작업 중에서도 작업수행시간이 초과되어 종료된 경우 실제 작업이 정상적으로 수행되었지만 SGE의 강제 종료 시그널에 의해 "E" 상태로 종료되게 된다. (그림 2)의 작업스크립트 중 h_rt는 해당 작업이 최대 얼마만큼 수행할 지를 제한하는 속성값(Wall-Clock Time)이다. Tachyon에서는 공용큐(normal)의 경우 범용의 목적으로 작업수행시간에 제한(48시간)을 설정하였다. 실제 사용자 어플리케이션을 살펴보면 대부분이 반복코드를 정상적으로 수행을 하다 제한시간에 걸려 종료되는 작업으로 실패 작업이라 볼 수 없다. SGE에서는<표 2>의 19번, 20번째 필드에 Exit code(종료코드)와 Failed code(실패코드)를 남기게 되는데 속성 값이 모두 "0"인 경우에는 작업이 성공적으로 수행되었음을 의미하며, 그 외에는 비정상적으로 작업이 완료된 경우로 원인에 따라 각각 다른 값을 저장하게 된다. <표 2>는 실패작업 중 작업수행시간이 초과되어 종료된 작업등 중의 일부로 "E" 상태로 작업이 종료되어 있다. 또 하나 살펴볼 것은 20번째 필드인 FAILED코드인데, SGE에서는 실패 코드 값에 따라 각각 의미를 부여하고 있다.

<표 2> 실패 작업 로그

DATE	QNAME	STATUS	E-CPU	E-RUN	EXIT-CODE	FAILED
20150715	normal	E	32	691200	0(1),137(3)	100(4)
20150715	normal	E	32	691264	0(1),129(2),137(1)	100(4)
20150715	normal	E	32	691232	0(1),137(3)	100(4)
20150715	normal	E	32	691232	0(1),137(3)	100(4)
20150715	normal	E	32	691232	0(1),129(1),137(2)	100(4)

<Table 2> Fail Job Logs

<표 3> SGE FAILED 코드 값
(출처:GE User Guide)

Code	Meaning
0	No failure /Job ran, exited normally
1	Failure before job (execd)
3	Before writing config
4	Before writing PID
5	On reading config file
6	Setting processor set
7,8	Before or in prolog
10	Failure in pestart
11	Failure before job (shepherd)
12,13	pe stop/PE stop procedure or failed
15	Failure epilog
19	No exit status
21	Failure in recognizing job
25	Rescheduling
26	Failure opening output(stderr/stdout)
27	no shell
28	no current working dir
29	AFS problem
30	Application error
36	Check daemon configuration
37	Qmaster enforced h_rt limit
100	Failure after job / Job ran, job killed by a signal.

<Table 3> SGE FAILED Code Value
(source: GE User Guide)

<표 2>의 작업들은 모드 FAILED 코드가 100임을 볼 수 있다. <표 3>은 관련 코드에 해당하는 값을 보여주고 있는데 100인 경우는 작업이 수행되었으나 리눅스 상에서 발생하는 시그널로 인해 작업이 중간에 종료 된 경우이다. 즉, SGE에서 작업제한시간 초과되어 리눅스 시그널을 이용해 작업 프로세스를 강제로 종료시킨 작업들이다.

③ "E"상태 중 작업수행시간제한 초과작업 추출

작업제한시간 초과로 종료된 작업들을 찾기 위해서는 위에서 언급하였듯이 백업된 사용자의 작업 스크립트와 종료시 남겨진 작업로그를 비교하여야 한다. SGE에서는 qsub을 통해 배치큐에 작업을 제출하게 되면 동시에 고유한 작업번호(<표 1>의 2번째 필드)를 부여받게 되는데 백업된 사용자의 작업 스크립트는 작업마다 주어지는 고유한 작업번호를 파일명으로 저장한다. 22번째 필드의 TASK_NUM 속성은 하나의 작업번호로 수행되는 TASK의 개수로 SGE에서는 array 작업을 통해 동일한 여러 개의 작업을 하나의 작업처럼 제출 할 수 있다. 본 실험에서는 TASK_NUM의 개수가 1개 이상인 즉, array작업을 통해 제출한 작업의 경우 제한시간을 초과하여 “E” 상태로 빠진 작업이 없음을 확인하였고, 대상작업 목록에서 제외하였다($22==0).

(그림 3) 작업수행시간제한 초과 작업 추출 스크립트

▪ 일별 작업 추출

```
joblogdir=/sge/job_logs/$Date.sge          // 일별로 로그저장
jobscrdir=/sge/job_scripts_backup/$JobID // 작업번호 파일명
// 실패작업 추출, array 작업 제외
joblog=`cat $joblogdir |   awk '{ if($22==0 && $16=="E")
print $2":"$11":"$13":"$14":"$16 }' | egrep "^[0-9]"`

for JLOG in $joblog  //한개의 작업단위로 반복하면서 수행
do
// 백업된 Job Script 파일 검색 후 h_rt(wall-time limit) 추출
  jobidfile=`lfs find $jobscrdir  -name "$jid"`
  h_rt=`cat $jobidfile | grep -v "^##" | grep h_rt | cut -d'='
-f2 | awk '{ print $1 }'`
  h_rt_sec=`cat $jobidfile | grep $h_rt | cut -d'=' -f2 | awk
-F':' '{ print sum=(expr ($1*3600)+($2*60)) }'`//초단위 변경
  // 시스템 종료 전후의 오차범위 ± 5초 값 계산
  hrt_p5=`echo $h_rt_sec | awk 'x=$1 { print sum=x+5 }'`
  hrt_m5=`echo $h_rt_sec | awk 'x=$1 { print sum=x-5 }'`
  // 해당 시간 범위 안에 끝나는 작업들 추출
  echo "$jid $hrt_m5 $hrt_p5 $h_rt_sec $runs $h_rt
$cpuusage $memusage $stat" | awk 'x=$2, y=$3 { if ($5>x
&& $5<y && $9=="E") print $1,$2,$3,$4,$5,$6,$7,$8,$9 }'
done >> errjob_hrt_$Date  // 해당 작업들 로그파일 생성

jobscount=`cat  $joblogdir  |  awk  '{  if($22==0)  print
$2":"$11":"$13":"$14":"$16 }' | grep "[0-9]" | wc -l`
// array job을 제외한 전체 대상 작업 개수
timelimitjobcount=`cat errjob_hrt_$Date|grep [0-9]| wc -l`
echo  "$Date:$timelimitjobcount/$jobscount"  >
errjob_hrt_count_$Date
//제한시간을 초과한 작업개수와 대상작업 개수를 카운트
```

▪ 월별 작업 추출

```
joblog=`cat ./Logs/errjob_hrt_count_$YearMonth*`
// 일별작업 로그(errjob_hrt_count_$Date)들을 반복해서 카운트
for JLOG in $joblog
do
  echo $JLOG
  job_date=`echo $JLOG | cut -d':' -f1`
  errfix=`echo $JLOG | cut -d'/' -f2 | cut -d'(' -f1`
  taskjob=`echo $JLOG | cut -d'/' -f3 | cut -d'(' -f1`

  sum_errfix=`echo "$sum_errfix $errfix" | awk '{ printf "%d", $1 + $2 }'`
  sum_taskjob=`echo "$sum_taskjob $taskjob" | awk '{ printf "%d", $1 + $2
}'`
done
// 전체 해당작업에 대해 결과값 합산
echo "Total Target Jobs(TASK_NUM>0) : " $sum_taskjob >>
result_$YearMonth
echo "Fixed Error Job Count : " $sum_errfix >> result_$YearMonth
```

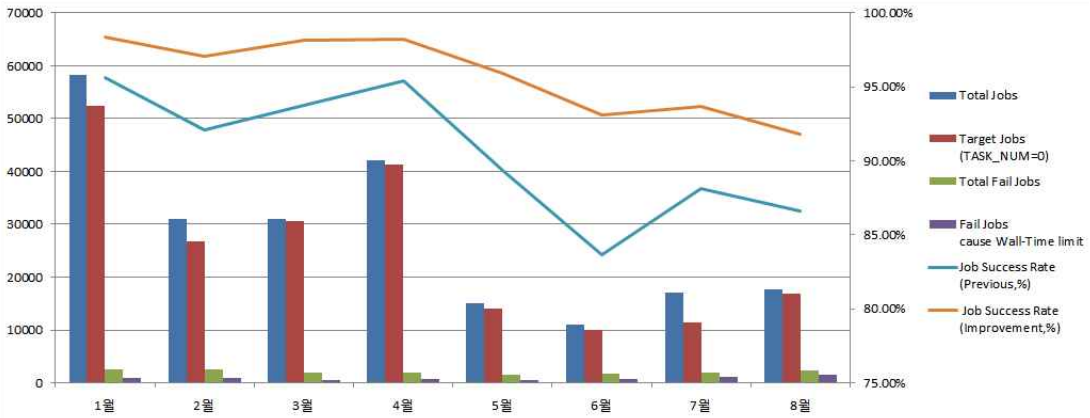
▪ 수행결과

```
[root@tachyon JobWallClockChk]# cat result_201504
** 2015.04 correction infomation of wall time limit error jobs **
Total Jobs :  42,071
Total Job Errors :  1,947
Total Target Jobs(TASK_NUM>0) :  41,271
Fixed Error Job Count :  740
-------------------------------------
Changing Ratio :  4.63 %  ->  1.76 %
```

(Figure 3) Extraction Script to Exceed Wall-Time limited jobs

(그림 4) 수행시간제한 초과작업 개선을 통한 향상률 실험결과 (2015.01~08)

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug
Total Jobs	58,248	31,013	31,063	42,071	14,956	11,086	16,988	17,731
Target Jobs (TASK_NUM=0)	52,385	26,770	30,572	41,271	13,963	9,984	11,496	16,814
Total Fail Jobs	2,555	2,443	1,942	1,947	1,589	1,813	2,019	2,371
Fail Jobs cause Wall-Time limit	939	919	566	740	607	765	1,077	1,457
Improvement Jobs	1,616	1,524	1,376	1,207	982	1,048	942	914
Job Success Rate (Previous)	95.61%	92.12%	93.75%	95.37%	89.38%	83.65%	88.12%	86.63%
Job Success Rate (Improvement)	98.39%	97.04%	98.18%	98.24%	95.94%	93.10%	93.66%	91.78%



(Figure 4) Experiment Results through Improving Exceed Wall-Time limited jobs(Jan~Aug,2015)

3.4 실험결과

본 논문에서는 2015년 1월부터 8월까지 Tachyon에서 수행된 작업로그를 대상으로 분석을 수행하였다. 작업로그 분석을 통해 오류의 원인을 분석하는 것은 시스템을 이해하는 중요한 척도 중에 하나이다. 가령, 시스템 환경이나 설정이 변경되었을 때 그에 대한 전체 시스템 성능이 어떠한 영향을 받는지 또한 사용자의 작업수행 로그를 통해 살펴볼 수 있다. 따라서 실패한 작업들을 구분하여 원인들을 파악하고 나아가 시스템 개선에 반영이 된다면 더욱 안정된 성능의 슈퍼컴퓨터 운영이 가능하다. 이를 위해 본 실험에서는 실패 작업들 중에 수행시간제한으로 종료된 작업들을 추출하였고 이를 성공률에 반영하였다. (그림 4)에서 보면 전체작업(Total Jobs)과 TASK_NUM의 개수가 0개인 성공률을 반영할 대상작업(Target Jobs)을 구분 하였다. 그리고 전체 실패작업(Total Fail Jobs)에서 작업초과시간제한으로 실패로 분류된 작업(Fail Jobs cause Wall-Time limit)을 구분하고 기존작업성공률(Job Success Rate- Previous)에 반영한 개선작업성공률(Job Success Rate- Improvement) 추출하였다. 실험결과를 보면 월 3%~5%정도의 성공률이 향상된 것을 살펴볼 수 있다.

슈퍼컴퓨터의 주요 목적 중에 하나는 대규모의 작업을 지원하는 것으로 KISTI에서는 거대규모의 응용과제를 선정하여 지원하고 있다. 이를 위해 일정기간 동안 별도의 전용노드를 지원하고 있으며 수행시간(Wall-Time)에 제한 없이 작업을 수행할 수 있다. 2015년 5월부터 거대규모과제, 즉 한 작업에 대량의 코어(1,000개~5,000개)를 사용하는 전용노드의 비율(전체의 약 40%이상)이 높아짐에 따라 전체 작업의 개수가 현저하게 줄어들었음을 확인할 수 있다.

4. 결 론

본 논문에서는 고성능 컴퓨팅 시스템인 슈퍼컴퓨터 Tachyon에서 나오는 실제 로그들을 구분하고 분석하였다. 그 중 실패한 작업들의 원인을 분석하였고, 시스템에 개선 반영하여 전체 성공률의 향상을 가져올 수 있었다. 실패한 작업들의 원인은 앞서 언급하였듯이 시스템 또는 사용자에 의해 발생하며 정확한 분석을 통해 원인을 파악하고 이를 반영함으로서 안정적인 시스템 성능을 얻을 수 있다.

향후, 사용자의 작업 중 자주 발생하는 실패 작업들에 대한 원인들을 재검토하고 분류하여 실패율을 낮추기 위한 방안과 사전 스크립트 검증 방법들에 대한 연구가 요구 되며 나아가, 시스템 로그(syslog)와 스케줄러의 작업 성공률을 비교하여 연관관계를 측정해보는 작업 또한 필요하다.

KISTI Tachyon은 2009년 말에 도입되어 2010년부터 서비스를 시작한 공공의 목적에 슈퍼컴퓨터로 시스템의 성능이 노후화 되어 국내 연구자의 자원 수요량을 충족시키지 못하고 있다. 빠른 시일 내에 새로운 시스템이 도입되어 국내 연구자의 지원이 다급한 실정이다.

References

[1] National Institute of Supercomputing and Networking, KISTI, http://www.nisn.re.kr

[2] F. Wang, S. Oral, G. Shipman, O. Drokin, T. Wang, and I. Huang, “Understanding lustre filesystem internals”, Oak Ridge National Lab, Technical Report ORNL/TM-2009/117, 2009

[3] G. Pfister, “An Introduction to the InfiniBand Architecture (http://www.infinibandta.org/)”, IEEE Press, 2001.

[4] G. Cawood, T. Seed, R. Abrol, T. Sloan, "TGO & JOSH: Grid Scheduling with Grid Engine & Globus", Proceedings of the UK e-Science All Hands Meetings, Nottingham, 2004.

[5] Templeton, D., “A Beginner’s Guide to Sun Grid Engine 6.2”, Whitepaper of Sun Microsystems, July 2009.

[6] Stillwell, M.; Vivien, F.; Casanova, H., "Dynamic Fractional Resource Scheduling versus Batch Scheduling," Parallel and Distributed Systems, IEEE Transactions, vol.23, no.3, pp.521-529, March 2012.

[7] C. Chaubal, "Scheduler Policies for Job Prioritization in the Sun N1 Grid Engine 6 System", Technical report, Sun BluePrints Online, Sun Microsystems, Inc., Santa Clara, CA, USA. http://www.sun.com/blueprints/1005/819-4325.pdf, 2005.

[8] J.H. Abawajy, “An efficient adaptive scheduling policy for high-performance computing”, Original Research Article Future Generation Computer Systems, Volume 25, Issue 3, pp.364-370, Mar 2009.

[9] J. W. Yoon, T. Y. Hong, C. Y. Park, H.C. Yu, “Analysis of Batch Job log to improve the success rate in HPC Environment”, International Conference on Convergence Technology, vol.2 No.1, pp.209-210, July,2013.

[10] El-Sayed, N., & Schroeder, B.., "Reading between the lines of failure logs: Understanding how HPC systems fail". In: Dependable Systems and Networks (DSN), 2013 43rd Annual IEEE/IFIP International Conference on, pp.1-12, June, 2013.

윤 준 원

2004년 : 고려대학교 대학원 컴퓨터학과(이학석사)
2011년 : 고려대학교 대학원 컴퓨터학과 박사 수료
2005년~현재 : KISTI 슈퍼컴퓨팅본부 선임연구원

관심분야 : 고성능컴퓨팅, 분산 컴퓨팅, 결함포용시스템, 슈퍼컴퓨팅, 스케줄러

홍 태 영

1999년 : 성균관대학교 물리학과 (이학사)
2002년 : 성균관대학교 대학원 물리학과(이학석사)
2005년~현재 : KISTI 슈퍼컴퓨팅 본부 선임연구원

관심분야 : 슈퍼컴퓨팅, 파일시스템, 인터커넥션 네트워크, 스케줄러

공 기 식

1999년 : 고려대학교 컴퓨터학과 (이학사)
2001년 : 고려대학교 컴퓨터학과 (이학석사)
2005년 : 고려대학교 컴퓨터학과 (이학박사)

2009년～현 재: 남서울대학교 멀티미디어학과 조교수

관심분야 : IPv6 이동성 관리, 광대역통합망, 미래인터넷, 분산 컴퓨팅

박 찬 열

1995년 : 고려대학교 대학원 컴퓨터학과(이학석사)
2000년 : 고려대학교 대학원 컴퓨터학과(이학박사)
20002년~현재 : KISTI 슈퍼컴퓨팅 본부 책임연구원

관심분야 : 그리드 컴퓨팅, 분산 컴퓨팅, 결함포용시스템, 슈퍼컴퓨팅