대규모 클러스터 시스템에서 배치작업 스케줄러를 활용한 성능 분석 데이터 수집 방법 연구

이재국[1], 권민우[1], 안도식[1], 홍태영[2]
[1]한국과학기술정보연구원 슈퍼컴퓨팅인프라센터 시스템팀
[2]한국과학기술정보연구원 슈퍼컴퓨팅인프라센터
{jklee, mwkwon, dsan, tyhong}@kisti.re.kr

A Study on Performance Analysis Data Collection Method Using Batch-job Scheduler on Large-Scale Cluster System

Jae-Kook Lee[1], Min-Woo Kwon[1], Do-Sik An[1], Taeyoung Hong[2]
[1]System Team, Div. of Supercomputing Infrastructure Center, KISTI
[2]Div. of Supercomputing Infrastructure Center, KISTI

요 약

사용자 응용 프로그램의 특징을 분석하고 효율적인 시스템 운영을 통하여 사용자 프로그램 최적화를 지원하기 위하여 소프트웨어 프로파일링을 수행한다. 특히 국가 슈퍼컴퓨터인 누리온과 같이 8,400대가 넘는 계산노드로 구성된 클러스터 시스템에서 응용 프로그램의 프로파일링 데이터를 사용자의 개입없이 수집하고 데이터를 분석하는 것에는 한계가 있다.

본 연구에서는 배치작업 스케줄러를 활용하여 사용자의 개입 없이 응용 프로그램의 프로파일링 데이터를 수집하기 위한 방법을 제안한다. 그리고 제안한 방법을 누리온에서 구현하고 사용자 응용 프로그램이 실행될 때 프로파일링 데이터가 수집되는 것을 확인한다.

1. 서론

사용자는 응용 프로그램의 성능을 분석하고 최적화하기 위하여 시스템의 CPU나 메모리 정보 등 프로프일링 데이터를 수집한다. 시스템 운영자 관점에서도 프로파일링 데이터는 시스템을 최적화하여 응용 프로그램이 원활히 수행될 수 있는 환경을 제공하는데에 유용한 정보이다. 그러나, 한국과학기술정보연구원(KISTI)에서 구축하여 운영중인 국가 슈퍼컴퓨터 5호기 누리온과 같이 8,437대(인텔 Knights Landing CPU 기반 8305대, 인텔 Skylake CPU 기반 132대) 계산노드로 구성된 초거대 클러스터 시스템[1]에서 사용자의 개입없이 프로파일링 데이터를 수집하는 것에는 한계가 있다.

본 논문에서는 클러스터 시스템에서 사용자 프로그램이 제출될 때 각 작업을 스케줄링하는 배치작업 스케줄러의 Prolog/Epilog 기능을 이용하여 사용자 개입없이 시스템 프로파일링 데이터를 수집하는 방법을 제안하고 클러스터 시스템인 누리온에서 PBS(Portable Batch System) 스케줄러를 활용하여 구현한 후 데이터가 수집되는 것을 확인한다.

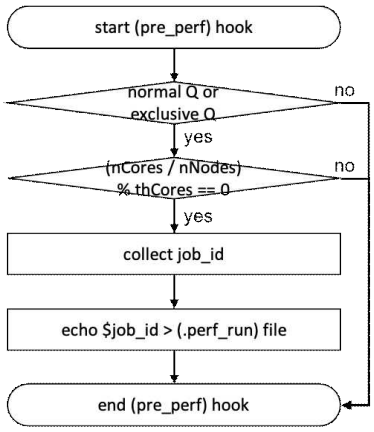
2. 관련연구

여러명의 사용자가 접속하여 사용하는 초고성능컴퓨팅(HPC) 환경에서는 배치작업 스케줄러를 통하여 사용자 응용 프로그램이 원활히 수행되도록 한다. 누리온에서도 PBS 스케줄러를 이용하여 사용자가 제출한 작업(프로그램)을 스케줄링한다. PBS 스케줄러는 사용자 작업이 실행되는 특정 시점에서 관리자가 정의한 작업을 수행할 수 있도록 후크라고 불리는 Prolog/Epilog 기능을 제공한다. 후크에서 관리자는 사용자 프로그램의 소스코드를 수정하지 않고 작업을 수락하거나 거부할 수 있으며, 수정할 수 있다[1-2].

리눅스에서는 기본으로 응용 프로그램 및 시스템의 특징을 모니터링을 위해 프로파일링 도구인 perf를 제공한다[3]. perf는 리눅스 커널에 포함된 시스템 영역에서부터 사용자의 프로그램 영역까지 한 번에 노드의 부하 및 캐시 미스, 버스 사이클 등 40여 개 CPU 및 메모리 관련 데이터 수집이 가능하다. 그러나, 계산노드의 모든 CPU 코어를 활용하여 응용 프로그램이 수행되는 경우 perf가 수행될 때 발생하는 인터럽트로 인하여 오버헤드가 발생하고 프로그램 수행시간에 지연이 발생한다[4].

3. 클러스터 시스템에서 프로파일링 데이터 수집 방법 설계

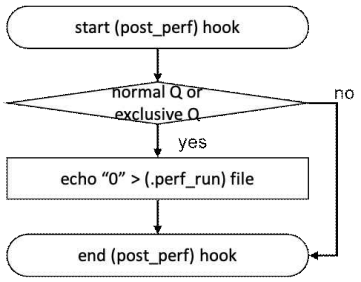
초고성능컴퓨팅 서비스 환경에서 사용자의 개입없이 응용 프로그램이 여러 개의 계산노드를 이용할 때 perf를 이용하여 프로파일링 데이터를 수집하기 위해 배치작업 스케줄러의 후크과 리눅스의 crontab을 활용한다. 사용자 개입없이 프로그램이 수행될 때 프로파일링 데이터를 수집하기 위한 후크는 사용자 응용 프로그램이 실행될 때 perf가 동작하게 하는 후크(pre_perf hook), 응용 프로그램이 끝나면 perf를 종료하기 위한 후크(post_perf hook)로 구성된다. 사용자 응용 프로그램이 모든 CPU 코어($_{max}$Core)를 사용하는 경우 오버헤드로 인하여 프로그램 수행시간에 지연이 발생하므로 임계값($_{th}$Core) 미만의 CPU 코어를 사용하는 프로그램이 수행될 때만 perf가 동작하도록 한다. 작업이 실행되면 지정한 큐에 제출된 응용 프로그램에 대해서만 동작하도록 큐를 확인한다. 지정한 큐에 제출된 작업이면 CPU 코어수를 체크하고 조건을 만족하면 작업 ID를 임의의 파일(.perf_run)에 기록하여 perf를 실행하는 스크립트에서 활용할 수 있도록 한다. (그림 1)는 pre_perf 후크 순서도이다.



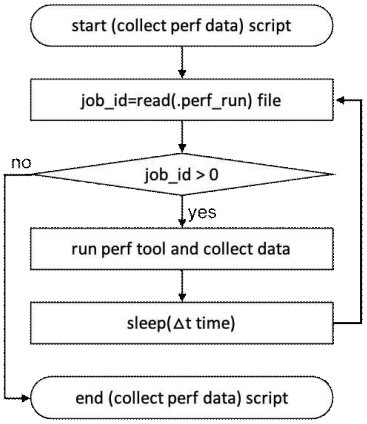
(그림 1) perf 실행을 위한 pre_perf 후크 순서도

post_perf 후크에서는 사용자 프로그램이 종료되면 perf도 함께 종료하기 위하여 작업 ID를 기록한 파일을 초기화한다. (그림 2)는 post_perf 후크 순서도이다.

여러 개의 계산노드에서 perf를 동시에 수행시키기 위해 본 논문에서는 리눅스의 crontab 기능을 이용하여 주기적으로 perf 데이터를 수집하는 스크립트가 동작하도록 한다. perf를 실행하는 스크립트는 작업 ID가 존재하는지 파일(.perf_run)을 확인한다. 작업 ID가 존재하면 perf를 수행하여 주기적($_{\triangle}$time)으로 프로파일링 데이터를 수집한다. perf 데이터 수집시 발생하는 오버헤드를 최소화하기 위하여 LLC loads, LLC-loads, LLC-stores, cache-misses, cache-references, page-faults, context-switches, cpu-clock, cpu-migrations, task-clock 등 중요한 10개 데이터만 수집하도록 한다. (그림 3)은 perf 데이터를 수집하는 스크립트 순서도이다.



(그림 2) perf 종료를 위한 post_perf 후크 순서도



(그림 3) perf를 이용한 데이터 수집 스크립트 순서도

4. 시험 및 데이터 수집

본 논문에서는 8,437대 계산노드로 구성된 누리온에서 PBS 스케줄러를 활용하여 제안한 방법을 구현하였다. 각 계산노드의 crontab에 perf를 수행하여 프로파일링 데이터를 수집하는 스크립트(collect perf data script)를 등록하였다. 스크립트에는 지정된 경로의 작업 ID에 해당하는 디렉토리($PERF_LOG_PATH/YYYY/MM/DD/Job_ID.pbs/)에 응용 프로그램이 수행되는 계산노드의 이름으로 perf 데이터가 수집되도록 하였다. (그림 5)는 2023년. 9월 20일에 4개의 계산노드(node2405, node3210, node3222, node3622)에서 수행된 13543631.pbs 작업에 대한 perf 데이터 수집 파일을 보여준다.

```
[10:20:35 root@bcm2 13543631.pbs]# pwd
/apps/pbs/perf_log/2023/09/20/13543631.pbs
[10:20:39 root@bcm2 13543631.pbs]# ls
node2405.perf  node3210.perf  node3222.perf  node3622.perf
```

(그림 4) 누리온 perf 데이터 수집 파일 경로 및 리스트 예제

(그림 6)은 13543631.pbs 사용자 작업이 수행될 때 node2405 계산노드에서 수집된 프로파일링 데이터 일부를 보여준다.

4. 결론 및 향후 연구 계획

본 연구에서는 사용자의 개입 없이 응용 프로그램 및 시스템의 성능 분석을 위해 필요한 프로파일링 데이터를 수집하기 위하여 배치작업 스케줄러의 후크 기능과 리눅스의 crontab을 활용하였따. 성능에 미치는 오버헤드를 최소화하기 위하여 본 논문에서는 임계값 미만의 CPU 코어를 이용하는 프로그램에 대해서만 데이터를 수집하도록 하였다. 그리고, 제안한 기법을 대규모 클러스터 시스템인 누리온에 적용하고 프로파일링 데이터가 지정된 경로에 수집되는 것을 확인하였다. 향후 수집된 데이터를 활용하여 응용 프로그램 및 시스템을 최적화 하기 위한 요소를 찾기 위하여 기계학습 및 AI를 활용한 빅데이터 분석 연구를 지속할 계획이다.

Acknowledgement

본 연구는 2023년도 한국과학기술정보연구원(KISTI) 기본사업 과제로 수행한 것입니다(K-23-L02-C01).

참고문헌

[1] 권민우, 윤준원, 홍태영, "PBS 작업 스케줄러 Hook를 이용한 슈퍼컴퓨터 5호기 계산노드 자동 점검 기능 구현," 한국정보처리학회 학술대회논문집, 대한민국, 2019, pp.101-102.

[2] PBS Professional 2021.1.2. Hooks Guide, PBS works, https://2021.help.altair.com/2021.1.2./PBS%20Professional/PBSHooks2021.1.2.pdf

[3] V.M. Weaver. "Linux perf_event Features and Overhead," The 2nd International Workshop on Performance Analysis of Workload Optimized Systems, Austin, Texas, 2013.

[4] 표지원, 방지우, 이재국, 홍태영, 엄현상, "Linux Perf를 이용한 슈퍼컴퓨터 CPU/메모리 관련 시스템 이벤트 수집 방안," 한국정보과학회 학술대회논문집, 대한민국, 2022, pp.48-50


```
# started on Wed Sep 20 23:58:03 2023


 Performance counter stats for 'system wide':

   511,651,360,055      LLC-loads                #  125.423 M/sec                    (39.93%)
    88,098,014,344      LLC-stores               #   21.596 M/sec                    (39.93%)
    23,968,335,011      cache-misses             #   11.418 % of all cache refs      (39.93%)
   209,908,726,266      cache-references         #   51.456 M/sec                    (39.94%)
           230,033      page-faults              #    0.056 K/sec
           126,781      context-switches         #    0.031 K/sec
     4,079,506.02 msec cpu-clock                #   67.973 CPUs utilized
             1,909      cpu-migrations           #    0.000 K/sec
 5,182,987,613,325      cycles                   #    1.271 GHz                      (39.95%)
     4,079,303.27 msec task-clock               #   67.970 CPUs utilized

     60.016257537 seconds time elapsed

# started on Wed Sep 20 23:59:03 2023


 Performance counter stats for 'system wide':

   515,089,691,072      LLC-loads                #  126.263 M/sec                    (40.01%)
    89,236,374,612      LLC-stores               #   21.874 M/sec                    (40.00%)
    24,168,109,324      cache-misses             #   11.810 % of all cache refs      (40.00%)
   204,643,515,603      cache-references         #   50.164 M/sec                    (40.00%)
           214,213      page-faults              #    0.053 K/sec
           127,740      context-switches         #    0.031 K/sec
     4,079,571.92 msec cpu-clock                #   67.976 CPUs utilized
             1,922      cpu-migrations           #    0.000 K/sec
 5,205,908,566,343      cycles                   #    1.276 GHz                      (40.01%)
     4,079,404.91 msec task-clock               #   67.973 CPUs utilized

     60.014908130 seconds time elapsed
```

(그림 5) 누리온 계산노드 perf 데이터 수집 예제