작업 배치 스케줄러와 컨테이너 오케스트레이션 툴을 활용한 이중 클러스터 서비스 환경 구현

권민우*, 이국화*, 안도식*, 홍태영*
*한국과학기술정보연구원(KISTI) 슈퍼컴퓨팅인프라센터
mwkwon81@kisti.re.kr

Implementation of dual cluster service environment using a job batch scheduler and a container orchestration tool

Min-Woo Kwon*, Gukhua Lee*, Do-Sik An*, Taeyoung Hong*
*Dept. of Supercomputing Infrastructure Center, KISTI

요 약

KISTI 슈퍼컴퓨팅인프라센터에서는 AI 연구자들을 위해 GPU기반의 클러스터 시스템인 뉴론을 구축하여 서비스하고 있다. 뉴론은 기본적으로 작업 배치 스케줄러인 SLURM을 통해 자원 분배 서비스를 제공하고 있다. 최근 컨테이너 이미지 기반의 클라우드 서비스에 대한 요구가 많아지면서 뉴론에서도 컨테이너 오케스트레이션 툴을 활용한 서비스인 웹 기반의 MyKSC를 제공하고 있다. 본 논문에서는 작업 배치 스케줄러와 컨테이너 오케스트레이션 툴을 활용한 이중 클러스터 서비스 환경을 구현하는 기법에 대해서 소개한다.

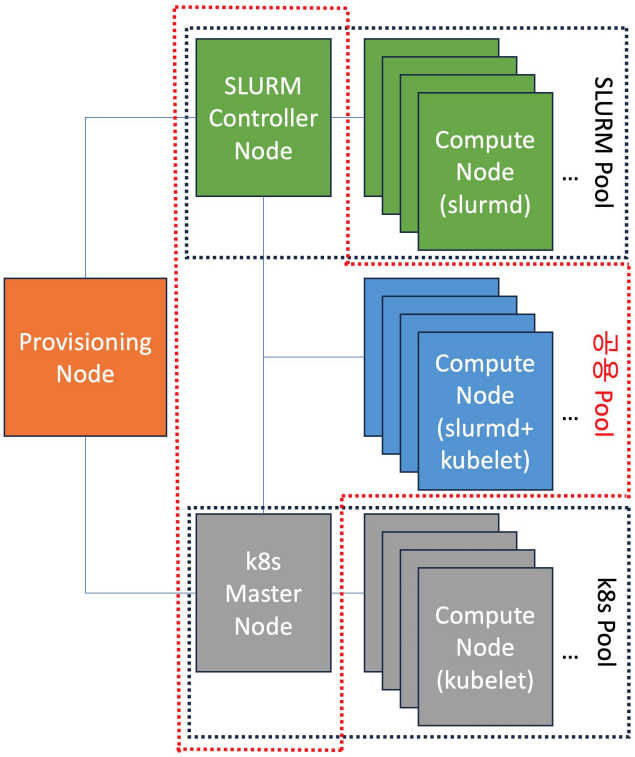
1. 서론

한국과학기술정보연구원(KISTI)의 슈퍼컴퓨팅인프라센터에서는 GPU를 이용한 성능 가속이 뛰어난 AI 관련 연구분야를 지원하기 위하여 GPU 기반의 클러스터 시스템인 뉴론을 구축하여 운영하고 있다[1]. 뉴론은 기본적으로 작업 배치 스케줄러인 SLURM을 이용하여 서비스 되며 사용자가 제출하는 작업 스크립트에 따라 계산 자원을 배분하여 서비스를 제공하고 있다[2]. 최근 AI 관련 연구가 활발하게 진행되면서 AI 관련 응용 소프트웨어의 변화가 급진적으로 이루어지고 있다. 이에 따라 기존의 HPC 클러스터 환경에서 사용자들이 AI 소프트웨어를 수행할 때 응용 소프트웨어의 버전 관리에 어려움을 겪고 있다. NVIDIA나 AMD와 같은 GPU 제조사들은 이런 연구자들의 어려움을 해소하기 위해 자체적으로 NVIDIA GPU Cloud(NGC)나 Infinity Hub와 같은 컨테이너 이미지 서비스를 제공하고 있다. 최근 이러한 컨테이너 이미지를 활용한 연구 수행이 활발해지면서 뉴론에서도 컨테이너 이미지 기반의 클라우드 서비스를 제공하기 위해 컨테이너 오케스트레이션 툴인 Kubernetes를 이용한 웹 기반의 MyKSC 서비스를 제공하고 있다[3]. 본 논문에서는 작업 배치 스케줄러인 SLURM과 컨테이너 오케스트레이션 툴인 Kubernetes를 이용한 이중 클러스터 서비스 환경을 구현하는 기법에 대해서 소개한다.

2. 이중 클러스터 서비스 환경을 위한 노드 구성

그림 1은 이중 클러스터 서비스 환경을 구현하기 위한 노드 구성을 보여준다. 뉴론과 같은 클러스터 시스템은 프로비저닝 노드를 통해 인프라와 계산노드의 OS 이미지가 관리되고 상태 모니터링, 로그 정보 수집 및 모든 노드 간의 공통의 계정 서비스를 제공하기 위한 LDAP 서비스 등이 운영된다. 그리고 HPC 작업 배치 스케줄러 서비스를 위한 SLURM Controller 노드와 이 노드와 통신하여 자원 배분을 담당하는 slurmd 데몬이 설치된 계산노드들이 SLURM Pool을 구성하게 된다. 또한 컨테이너 기반 클라우드 서비스를 제공하기 위한 Kubernetes(k8s) Master 노드와 이 노드와 통신하여 자원 배분을 담당하는 kubelet 데몬이 설치된 계산노드들이 k8s Pool을 구성하게 된다. 본 논문에서는 일부 계산노드에 slurmd와 kubelet 데몬을 동시에 설치하여 공용 Pool을 제공하고 경우에 따라 유동적으로 계산자

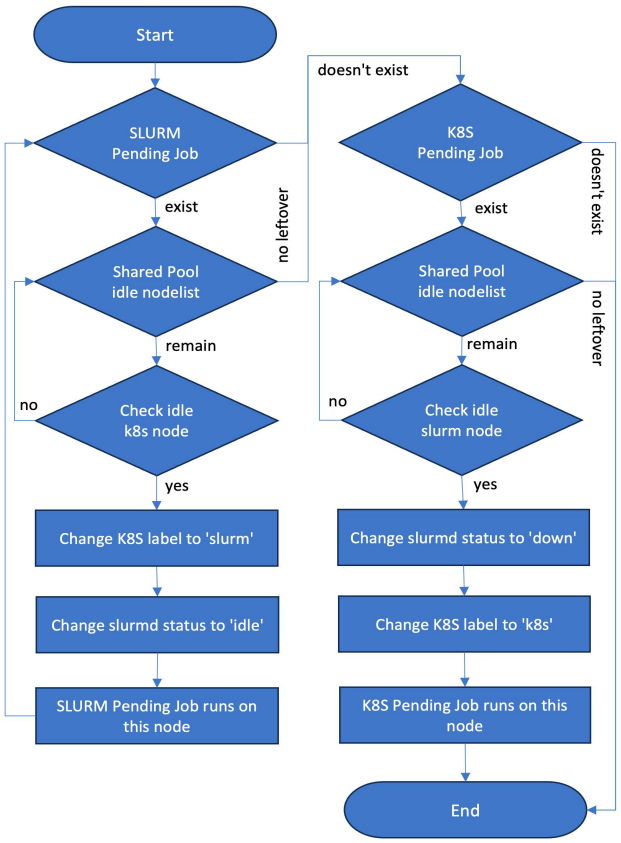
원을 배분하여 사용할 수 있는 기법을 소개한다.



(그림 1) 노드 구성도

3. 프로비저닝 노드를 활용한 공용 Pool 자원 관리

프로비저닝 노드에서 동작하는 Linux crontab에 아래 그림 2의 순서도에서 보여주는 공용 Pool 자동 자원 관리 쉘 스크립트를 등록하여 주기적으로 실행되도록 한다. 본 논문에서는 1분 단위로 스크립트가 수행되도록 하였다. 공용 Pool에 있는 계산노드에는 위에서 설명한 것과 같이 SLURM Controller(slurmctld)와 k8s Master에 의해 동시에 자원 관리가 가능하도록 slurmd와 kubelet 데몬을 동시에 설치한다. SLURM Pool에 있는 계산 자원이 전부 할당되어 사용자가 제출한 작업이 Pending(자원 할당 대기 중인 상태)된 경우, 공용 Pool의 계산노드 중 idle 상태인 노드를 찾는다. idle 상태의 노드가 k8s Master에 연동되어 있는 노드(k8s label이 'k8s'로 세팅되어 있는 노드)인 경우, slurmctld와 연동이 되도록 k8s label을 'slurm'으로 변경해주고 slumrd의 상태를 'idle'로 변경해준다. 그러면 SLURM 스케줄러에 Pending 상태로 대기 중인 작업이 해당 노드에 할당되게 된다. 반대로, 컨테이너 기반 클라우드 서비스에 Pending 작업이 있는 경우, 공용 Pool의 계산노드 중 idle 상태인 노드를 찾고 해당 노드가 slurmctld에 연동되어 있는 노드인 경우((k8s label이 'slurm'으로 세팅되어 있는 노드), k8s master와 연동이 되도록 slurmd의 상태를 'down'으로 변경하고 k8s label을 'k8s'로 변경한다. 그러면 컨테이너 기반 클라우드 서비스에 Pending 상태로 대기 중인 작업이 해당 노드에 할당되게 된다.



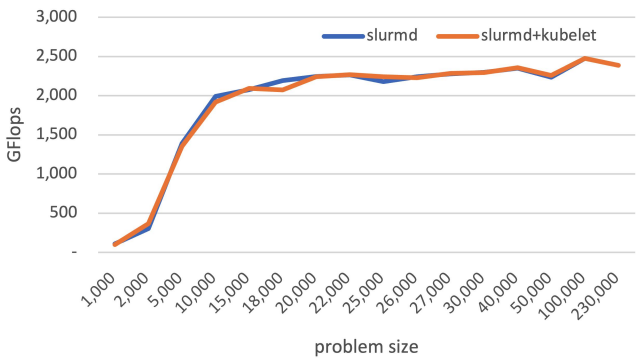
(그림 2) 공용 Pool 자동 자원 관리 기능 순서도

아래 표 1은 프로비저닝 노드에서 동작하는 공용 Pool 자동 관리 쉘 스크립트에서 계산노드의 연동 상태를 변경하기 위해 실행하는 커맨드를 보여준다[4,5].

<표 1> 계산노드의 연동상태를 변경하는 커맨드

State	Command
slurmctld 연동	kubectl label nodes (노드명) sched=slurm --overwrite
	scontrol update nodename=(노드명) state=idle
k8s master 연동	scontrol update nodename=(노드명) state=down reason=k8s_connect
	kubectl label nodes (노드명) sched=k8s --overwrite

그림 3은 slurmd와 kubelet 서비스 데몬을 함께 구동시켰을 때, 작업 성능의 저하가 발생하는지를 확인하기 위해 slurmd 서비스 데몬만 구동할 때와 slurmd와 kubelet 서비스 데몬을 함께 구동했을 때의 계산노드의 CPU Linpack Benchmark 성능을 보여준다. 성능 측정 결과 본 논문에서 제안하는 이중 클러스터 서비스를 위해 계산노드에서 slurmd와 kubelet을 함께 구동하여도 성능저하없이 작업 수행이 가능함을 확인할 수 있었다.



(그림 3) CPU Linpack Benchmark 수행 결과

4. 결론 및 향후 연구 방향

본 논문에서 제안하는 방식은 프로비저닝 노드에서 동작하는 Linux crontab 기능을 활용하여 단순한 쉘 스크립트로 구현이 가능하다는 장점을 가지고 있다. 반면에, 공용 Pool에 있는 계산노드에 SLURM 관련 패키지들과 Kubernetes 관련 패키지들이 동시에 설치되어야 하므로 시스템 소프트웨어 관리가 복잡해지는 단점을 가지고 있다. 향후에는 이러한 단점을 해결하기 위해 프로비저닝 노드의 기능을 고도화시킬 예정이다. 공용 Pool의 계산노드를 Diskless 방식으로 운영하면서 SLURM 스케줄러에 연동이 필요할 때는 프로비저닝 노드를 통해 SLURM 관련된 패키지들만 설치된 OS 이미지로 부팅을 시키고 Kubernetes와 연동이 필요할 때는 Kubernetes 관련 패키지들이 설치된 OS 이미지로 부팅을 시키는 방식으로 기능을 발전시킬 예정이다.

사 사

이 논문은 2024년도 한국과학기술정보연구원의 기본사업(과제명:국가 플래그십 초고성능컴퓨터 인프라 구축 및 서비스, 과제번호:K24L2M1C1)으로 수행된 연구입니다.

참고문헌

[1] 한국과학기술정보연구원 국가슈퍼컴퓨팅센터, 뉴론 소개, https://www.ksc.re.kr/byjw/neuron
[2] 뉴론 지침서, SLURM을 통한 작업 실행 https://docs-ksc.gitbook.io/neuron-user-guide/undefined/running-jobs-through-scheduler-slurm
[3] 한국과학기술정보연구원 슈퍼컴퓨터 웹 서비스 포털, MyKSC, https://my.ksc.re.kr/#/
[4] SLURM Workload Manager, scontrol https://slurm.schedmd.com/scontrol.html
[5] Kubernetes, Command line tool(kubectl), https://kubernetes.io/docs/reference/kubectl/