

쿠버네티스 멀티 클러스터 환경에서 생성 AI 추론 워크로드를 위한 Kueue 기반 효율적 Job 스케줄링

김철희, 김종원*
광주과학기술원

kchhaha3535@gm.gist.ac.kr, *jongwon@gist.ac.kr

Efficient Kueue-based Job Scheduling for Generative AI Inference Workloads in a Kubernetes Multi-cluster

Cheolhui Kim, JongWon Kim*
Gwangju Institute of Science and Technology

요 약

본 논문은 쿠버네티스 멀티 클러스터 환경에서 생성 AI 추론 워크로드를 보다 효율적으로 처리하기 위해, Kueue의 정적 ClusterQueue 지정 한계를 보완하는 동적 ClusterQueue 선택 전략을 제안한다. 제안한 전략은 (1) 각 ClusterQueue의 실시간 자원·대기열 지표를 제공하는 Quota API와 (2) Job 제출 단계에서 지표를 기반으로 'Light GPU Job'을 자동으로 CPU ClusterQueue로 fallback시키는 Job Submission Gateway로 구성된다. Stable Diffusion v1.5 모델을 활용해 GPU Job(steps=30)과 Light GPU Job(steps=5)을 Uniform(교차), Burst(집중) 두 시나리오로 실험한 결과, 제안된 전략은 정적 방식 대비 전체 Job 완료 시간을 각각 14.3%, 11.6% 단축하였다. 이는 GPU 대기열 혼잡을 줄이고 CPU 자원을 활용한 효과로, Kueue 기반 멀티 클러스터 스케줄링의 지연 최소화 가능성을 보여준다.

I. 서 론

최근 쿠버네티스의 배치(Job) 워크로드는 생성 AI 추론, 대규모 데이터 전처리, 과학 계산 등 다양한 연구·산업 환경에서 빠르게 활용 범위를 넓혀가고 있다.[1] 그러나 쿠버네티스의 기본 스케줄러는 Job 대기열 관리, 선점(preemption), 이질적 자원(GPU, CPU)의 혼합 배치와 같은 고급 스케줄링 기능을 제공하지 못한다. 이러한 한계를 보완하기 위해 Volcano, YuniKorn, Kueue와 같은 대기열 기반 스케줄러가 제안되어 왔다. 세 시스템은 공통적으로 Job을 대기열에 적재하고, 자원 상태, 우선순위, 사용자 정의 정책 등을 고려하여 스케줄링 결정을 내리지만, 구현 방식과 철학에는 차이가 있다.[1] 특히 Kueue는 Custom Resource Definition(CRD)과 컨트롤러만으로 구현된 쿠버네티스 네이티브 방식의 배치 스케줄링을 지향한다. [2] 핵심 개념인 ResourceFlavor는 동일 자원 타입의 세부 구성을, ClusterQueue(CQ)는 클러스터 범위의 자원 풀을, LocalQueue(LQ)는 네임스페이스별 하위 풀로써 다중 팀 환경에서 정책 분리를 용이하게 한다.

Kueue의 베타 기능인 MultiKueue는 하나의 Kueue Manager 클러스터와 여러 Kueue Worker 클러스터가 전역 CQ를 공유하도록 설계되어, 멀티 클러스터 환경에서도 일관된 Job 대기열 관리가 가능하다.[2] 그러나 Kueue는 Job 객체에 명시된 queue-name 라벨을 기준으로 제출 시점에 CQ가 고정되기 때문에, 실행 지연이나 클러스터 자원 불균형에 대한 동적 대응이 어렵다. 예를 들어, GPU CQ의 대기열이

과도하게 길어지더라도 해당 Job을 CPU CQ로 이동시킬 수 없어, 전체적인 스케줄링 효율성이 저하된다. 또한, Stable Diffusion v1.5와 같은 생성 AI 모델은 GPU가 아닌 CPU 환경에서도 실행 가능함에도 불구하고, Kueue의 구조적 제약으로 인해 이러한 유연한 자원 전환이 어렵다.[3]

본 연구는 Job 제출 시점에 CQ를 동적으로 선택할 수 있도록 하여, GPU 자원 상태에 따라 상대적으로 가벼운 GPU Job(이하 Light GPU Job)을 CPU CQ로 자동 fallback시키는 전략을 제안한다. 이를 통해 쿠버네티스 멀티 클러스터 환경에서 Kueue를 기반으로 한 클러스터 간 Job 스케줄링의 유연성과 지연 시간 최적화 방법을 실현하고, 생성 AI의 추론 작업이 보다 효율적으로 수행될 수 있도록 기여하는 것을 목표로 한다.

II. 쿠버네티스 멀티 클러스터 환경

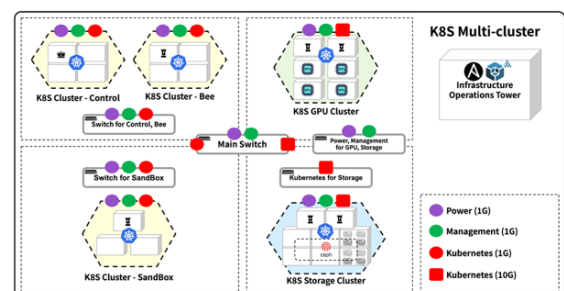


그림 1. 쿠버네티스 멀티 클러스터 환경 설계도

연구용 서버랙에는 그림 1 과 같이 총 다섯 개의 쿠버네티스 클러스터가 구축되어 있다. 모든 노드는 Power(1Gbps), Management(1Gbps), Kubernetes Plane(1/10Gbps) 세 NIC 으로 분리되어 원격 전원 제어, 구성 관리, 쿠버네티스 네트워크를 물리적으로 격리한다. 이번 실험에서는 Control Cluster 를 Kueue Manager 로, GPU Cluster 를 Kueue Worker 로 사용하며, 각각 CPU CQ 와 GPU CQ 를 포함한다. GPU Cluster 에는 Tesla T4 GPU 4 대가 탑재되어 있다. 이러한 환경은 CPU 와 GPU 자원이 네트워크, 하드웨어 차원에서 분리된 실험 조건을 제공해 동적 CQ 선택 전략의 효과를 측정하기에 적합하다.

III. 서비스 설계 및 구현

본 연구에서는 Stable Diffusion v1.5 모델을 사용하여, 추론에 사용되는 입력 변수인 steps 를 각각 30 과 5 로 하는 두 종류의 배치 추론 Job 을 대상으로 한다. 전자(이하 GPU Job)는 현실적으로 GPU 환경에서만 실행 가능하며, 후자(이하 Light GPU Job)는 추론 시간이 길어지지만 CPU 환경에서도 실행 가능하다.[3]

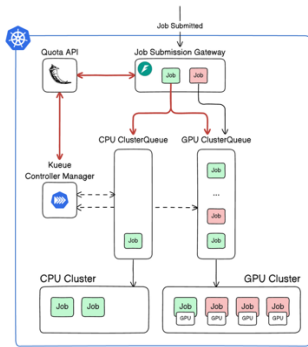


그림 2. 서비스 아키텍처

위 그림 2 는 전체 서비스 아키텍처를 시각화한 것이다. Quota API 는 Kueue Manager 로부터 각 CQ 의 상태를 조회한 값을 JSON 으로 반환한다.[4] Job Submission Gateway 는 입력으로 주어진 Job YAML 을 파싱해 Job 의 종류를 식별하고, Light GPU Job 의 경우 Quota API 로부터 각 CQ 의 실시간 자원 Quota 상태와 pendingWorkloads 수를 전달받아 동적 CQ 선택 알고리즘을 실행한다. 이 알고리즘은 GPU 자원의 점유 상태, GPU CQ 의 pendingWorkloads 값, CPU CQ 의 reservingWorkloads 값을 순차적으로 확인하고, 이 세 조건을 모두 만족할 경우 Light GPU Job 은 CPU CQ 를 선택하기 위해 입력받은 Job YAML 을 수정하여 Job 을 제출한다.[4] 그 외의 경우, 입력받은 Job YAML 을 수정없이 그대로 제출한다. 제출된 Job 은 CPU CQ 또는 GPU CQ 로 할당되고, Kueue 에 의해 승인되면 각각 CPU 클러스터, GPU 클러스터로 스케줄링된다.

IV. 검증 및 결과

동적 CQ 선택 알고리즘을 검증하기 위해 두 가지의 시나리오를 가정하였다. Uniform 시나리오에서는 1 초 간격으로 GPU Job 과 Light Job 을 교차하여 총 30 개를 입력하였다. Burst 시나리오는 GPU Job 15 개를 연속 제출한 직후 Light GPU Job 15 개를 연속 제출했다. 각 시나리오마다 동적 CQ 방식과 정적 CQ 방식을 적용해 각각 5 회 반복 측정했다. 처음 Job 을 제출한 시점부터, 모든 Job 이 실행을 마치기까지 걸리는 시간을 지표로 삼았다.

표 1. 실험 결과

시나리오	정적 CQ (s)	동적 CQ (s)	감소율 (%)
Uniform	201.6	172.8	14.3
Burst	235.5	208.2	11.6

Uniform 시나리오에서는 동적 CQ 선택 방식이 정적 CQ 방식에 비해 14.3%의 낮은 실행 시간을 보였으며, Burst 시나리오에서는 이보다 소폭 적은 11.6%의 낮은 실행 시간을 보였다. 이는 동적 CQ 선택 전략이 일부 Light GPU Job 을 CPU 환경에서 실행되도록 스케줄링하여, GPU 자원에 대한 대기열 밀집을 완화했기 때문으로 해석된다. 이로부터 동적 CQ 선택 전략을 통해 전체 생성 AI 추론 Job 의 실행 시간을 단축했음을 확인할 수 있다.

V. 결론 및 향후 연구

본 논문은 Kueue 환경에서 Job 제출 시점에 GPU 자원 및 Job 대기열 상태에 따라 Light GPU Job 을 CPU ClusterQueue 로 자동 fallback 시키는 동적 ClusterQueue 선택 전략을 제안하였다. 이 전략은 Kueue 의 구조적 한계인 정적 ClusterQueue 지정 문제를 보완하며, 쿠버네티스 멀티 클러스터 환경에서 클러스터 간 Job 스케줄링의 유연성과 추론 지연 시간 최적화 가능성을 실험적으로 입증하였다.

향후에는 Kueue 자체에서 Admission 단계에서 ClusterQueue 를 동적으로 결정할 수 있는 기능이 공식적으로 지원된다면, 외부 Gateway 없이도 더 단순한 구조로 유사한 효과를 기대할 수 있다. 또한, 추후 연구에는 H100, A100, L40 등 다양한 성능의 GPU 풀을 계층적으로 구성하여, 상위 사양 GPU 가 부족할 때 자동으로 하위 성능 GPU 로 우회하는 방식의 동적 스케줄링 전략을 탐색해볼 수 있을 것이다.

ACKNOWLEDGMENT

본 논문은 2019 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2019-0-01842, 인공지능대학원지원(광주과학기술원))

참 고 문 헌

- [1] Wei Huang, Shiming Zhang, "A Comparative Analysis of Kueue, Volcano, and YuniKorn", Cloud Native Computing Foundation, YouTube, <https://www.youtube.com/watch?v=njT5r3JjIaA>, 2025.
- [2] Kubernetes SIGs, Kueue, <https://kueue.sigs.k8s.io/docs>, 2025.
- [3] Jingxu Ng, Chao Lv, Pengcheng Zhao, Wenjie Niu, Jiachong Lin, Minjie Pan, Yuhao Liang, Yuanming Wang, "Open-Source Acceleration of Stable-Diffusion.cpp Deployable on All Devices", arXiv:2412.05781, <https://arxiv.org/abs/2412.05781>, 2024.
- [4] Job Submission Gateway Implementation, GitHub Repository, <https://github.com/Kimcheolhui/kueue-scheduling>, 2025.