Accelerator-Aware Kubernetes Scheduler for DNN Tasks on Edge Computing Environment

Jungae Park
Kookmin University
Seoul, Korea
barkjungae@kookmin.ac.kr

Unho Choi
Kookmin University
Seoul, Korea
yms04089@kookmin.ac.kr

Seungwoo Kum
Korea Electronics Technology Institute
Seoul, Korea
swkum@keti.re.kr

Jaewon Moon
Korea Electronics Technology Institute
Seoul, Korea
jwmoon@keti.re.kr

Kyungyong Lee
Kookmin University
Seoul, Korea
leeky@kookmin.ac.kr

ABSTRACT

The compute capability of edge devices is expanding owing to the wide adoption of edge computing for various application scenarios and specialized hardware explicitly developed for an edge environment. A container orchestration platform, Kubernetes is widely used to maintain edge computing resources efficiently, but it suffers from a limited scheduling capacity. We present a design and implementation of an accelerator information extraction module to improve the scheduling capability of a standard Kubernetes implementation by providing rich hardware information. Furthermore, we present a plausible advancement of the Kubernetes scheduler by considering detailed workload characteristics and attached specialized accelerator hardware information.

CCS CONCEPTS

• **Computer systems organization → Distributed architectures**.

KEYWORDS

edge computing, accelerator, scheduler, Kubernetes

ACM Reference Format:
Jungae Park, Unho Choi, Seungwoo Kum, Jaewon Moon, and Kyungyong Lee. 2021. Accelerator-Aware Kubernetes Scheduler for DNN Tasks on Edge Computing Environment. In *The Sixth ACM/IEEE Symposium on Edge Computing (SEC '21), December 14–17, 2021, San Jose, CA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3453142.3491411

1 BACKGROUND AND MOTIVATION

Edge computing is a distributed computing system that brings dataset computation to a location closer to the data source. Edge computing improves response time and saves bandwidth by avoiding the transfer of datasets to a remote cloud environment. Compared to ordinary public cloud computing, where a service vendor is responsible for maintaining compute resources, an edge computing environment is decentralized, and edge resources can be unstable compared to centralized cloud computing resources. Thus, it can be challenging to maintain edge computing resources and schedule jobs reliably. Kubernetes [3] is a container orchestration platform that provides a highly available container execution environment while providing fault-tolerance and auto-scaling on top of Linux container (LXC) technology [5]. Because Kubernetes can maintain resources and containers in a fault-tolerant way, managing relatively unstable edge computing resources is well-suited.

Edge computing allows running compute-intensive tasks near data sources, and it can meet geographical regulations and lessen security concerns. Savi et. al. [7] proposed a blockchain-based edge computing resource brokerage platform that extends Kubernetes. The proposed system schedules compute-intensive Deep Neural Network (DNN) tasks to a pool of remotely located edge resources to meet a geographical constraint. Using Kubernetes as an underlying container orchestration platform reduces resource management overhead while providing limited scheduling capabilities. In the vanilla Kubernetes distribution, the default *kube-scheduler* relies on user-defined job requirements and simple worker node hardware characteristics, and it becomes users' responsibility to provide rich hardware information and build an optimal scheduler on an edge environment for diverse tasks.

Reza et. al. [4] thoroughly evaluated the inference performance of three different DNN models (MobileNetV1, MobileNetV2 [6], and InceptionV3 [8]) on edge devices with diverse accelerators, such as NVIDIA Jetson TX2, NVIDIA Jetson Nano, and Google Edge Tensor Processing Unit (TPU). The researchers discovered that no single edge-accelerator device could provide optimal performance for all models. Google Edge TPU performs best for smaller models (MobileNetV1 and MobileNetV2), while NVIDIA Jetson Nano performs best for larger models (InceptionV3). Furthermore, the authors discovered that the host to accelerator device interface could significantly impact the inference performance especially for Google Edge TPU. Other than the DNN inference, the federated learning [1], which trains a DNN model using various edge devices with the accelerator, can show performance diversity.

It is crucial to reference complex relationships between workload and hardware characteristics to enhance scheduling quality, but it is barely supported in the current Kubernetes distribution. To improve the scheduling capability of Kubernetes, we present an accelerator discovery extension module for Kubernetes on edge computing environment with its implementation detail. Using the accelerator

SEC '21, December 14–17, 2021, San Jose, CA, USA

ACM ISBN 978-1-4503-8390-5/21/12.
https://doi.org/10.1145/3453142.3491411

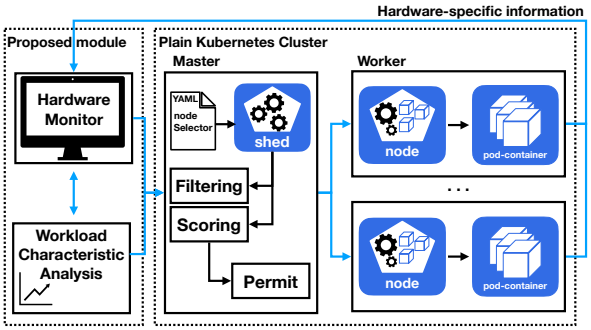

Figure 1: Kubernetes scheduler and the proposed extension module to enhance scheduling capacity

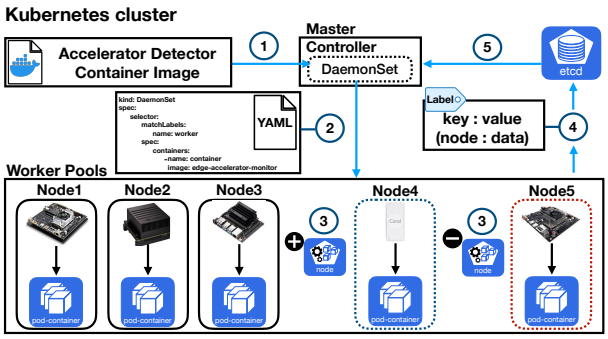

Figure 2: The implementation of accelerator hardware information extraction module

hardware information, we envision the possibility of enhancing the Kubernetes scheduler by referencing workload characteristics and accelerator detail while making a scheduling decision.

2 SYSTEM DESIGN AND IMPLEMENTATION

Using a rich set of information regarding workload and hardware characteristics can enhance the capability of the Kubernetes scheduler. Figure 1 presents an architecture of plain Kubernetes scheduler with a proposed extension module. The vanilla Kubernetes scheduler, which is shown in the right part of Figure 1, relies on limited resource information, such as available CPU cores and memory, and pod-to-node matchmaking happens based on label matching requirement manually specified by end-users and the internal ranking mechanism provided by Kubernetes. To enhance the scheduler capability, we propose to add a hardware monitor module that collects detailed and timely worker nodes' hardware information, including edge-accelerator devices. We propose adding a workload character analysis module that analyzes users' submitted tasks and extracts core components to execute a task to reflect the workload detail. The workload analyzer needs to consider various workload scenarios, such as DNN inference or training with different models and datasets and should be able to model characteristics of workloads on arbitrary accelerator hardware. The detailed hardware information and workload analysis results become input to a Kubernetes scheduler to enhance scheduling quality.

The automatic accelerator hardware and information extraction module for Kubernetes have been completed, and the workload analysis module is currently being developed. Figure 2 presents the implementation detail of accelerator information extraction module. The module aims to provide details of accelerator hardware attached to an edge device to help the Kubernetes scheduler make an informed decision. The prototype implementation is built with five representative edge-accelerator devices, NVIDIA Jetson TX1, TX2, Nano, Xavier, and Google Edge TPU hosted by Raspberry PI. We built a container image that can extract detailed accelerator information, and we made the image publicly available[1][2]. To detect NVIDIA Jetson devices, we reference */proc* file system of a host machine. To detect the Google Edge TPU device, we use the *lsusb* system command.

A worker node should start a pod that runs the accelerator detector container and report hardware detail to a master node. The update period is determined based on the characteristics of reported metrics, which are either static or dynamic. To automatically start a pod with a designated container when a node joins a cluster, we use Kubernetes *DaemonSet* feature. After a master node receives worker nodes' accelerator information, it records the received information using Kubernetes*Label* so that it can be easily integrated with the original Kubernetes scheduler.

3 DISCUSSION

A resource scheduler can make an informed decision using detailed accelerator information. It becomes advantageous for DNN workloads that exhibit varying performance for accelerator devices, [4]. In a market edge federation system [7], providing hardware capacity is crucial for buyers and sellers to interact appropriately.

To support a workload-aware edge computing scheduler using Kubernetes, the performance prediction of diverse DNN workloads on edge devices is crucial. Paleo [2] provided an algorithm to predict various DNN model training times on different GPU devices, and Mark [9] proposed a cost-efficient DNN inference system using multiple cloud services. However, most relevant prior work did not consider using edge computing resources with accelerators. Further research should be conducted to characterize the performance of various DNN workloads executed on edge devices.

The current version of the accelerator-detector module requires a manual implementation to extract characteristics of distinct hardware devices. This may limit the system's applicability for a newly released accelerator device, and generalizing diverse accelerator characteristics is required to improve the system's practicability.

ACKNOWLEDGMENTS

This work is supported by the Institute of Information Communications Technology Planning Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2021-0-01578) and the National Research Foundation of Korea (NRF) (Nos. NRF-2020R1A2C1102544 and NRF-2015R1A5A7037615)

[1]https://hub.docker.com/repository/docker/kmubigdata/edge-accelerator-monitor
[2]https://github.com/kmu-bigdata/edge-accelerator-monitor

REFERENCES

[1] Kallista A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards Federated Learning at Scale: System Design. *CoRR* abs/1902.01046 (2019). arXiv:1902.01046 http://arxiv.org/abs/1902.01046

[2] Hang Qi, Evan R. Sparks, and Ameet Talwalkar. 2017. Paleo: A Performance Model for Deep Neural Networks. In *Proceedings of the International Conference on Learning Representations*.

[3] David K. Rensin. 2015. *Kubernetes - Scheduling the Future at Cloud Scale*. 1005 Gravenstein Highway North Sebastopol, CA 95472. All pages. http://www.oreilly.com/webops-perf/free/kubernetes.csp

[4] Sheikh Rufsan Reza, Yuzhong Yan, Xishuang Dong, and Lijun Qian. 2021. Inference Performance Comparison of Convolutional Neural Networks on Edge Devices. In *Science and Technologies for Smart Cities*, Sara Paiva, Sérgio Ivan Lopes, Rafik Zitouni, Nishu Gupta, Sérgio F. Lopes, and Takuro Yonezawa (Eds.). Springer International Publishing, Cham, 323–335.

[5] Rami Rosen. 2014. Linux containers and the future cloud. *Linux J* 240, 4 (2014), 86–95.

[6] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

[7] Marco Savi, Daniele Santoro, Katarzyna Di Meo, Daniele Pizzolli, Miguel Pincheira, Raffaele Giaffreda, Silvio Cretti, Seung-woo Kum, and Domenico Siracusa. 2020. A Blockchain-based Brokerage Platform for Fog Computing Resource Federation. In *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. 147–149. https://doi.org/10.1109/ICIN48450.2020.9059337

[8] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the Inception Architecture for Computer Vision. *CoRR* abs/1512.00567 (2015). arXiv:1512.00567 http://arxiv.org/abs/1512.00567

[9] Chengliang Zhang, Minchen Yu, Wei Wang, and Feng Yan. 2019. MArk: Exploiting Cloud Services for Cost-Effective, SLO-Aware Machine Learning Inference Serving. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. USENIX Association, Renton, WA, 1049–1062. https://www.usenix.org/conference/atc19/presentation/zhang-chengliang