YOYOWORKS

XPU Technical White Paper

Version：2.0.21

Date：2021-09-12

Table of Contents

©2021 YOYOWORKS www.yoyoworks.com

©2021 YOYOWORKS www.yoyoworks.com

1 .Introduction

1.1 AI Industry Status

AI (Artificial Intelligence), 5G, big data centers, and industrial Internet together constitute the core breakthrough areas of the new infrastructure in the world. AI will penetrate into various industries and become more and more important.

Deloitte's 'Global Artificial Intelligence Development White Paper' released in the first half of 2020 predicts that worldwide artificial intelligence market will be more than 6 trillion US dollars in 2025; And artificial intelligence industry in China will grow to 160 billion RMB by 2020, driving the related industries to one trillion RMB.

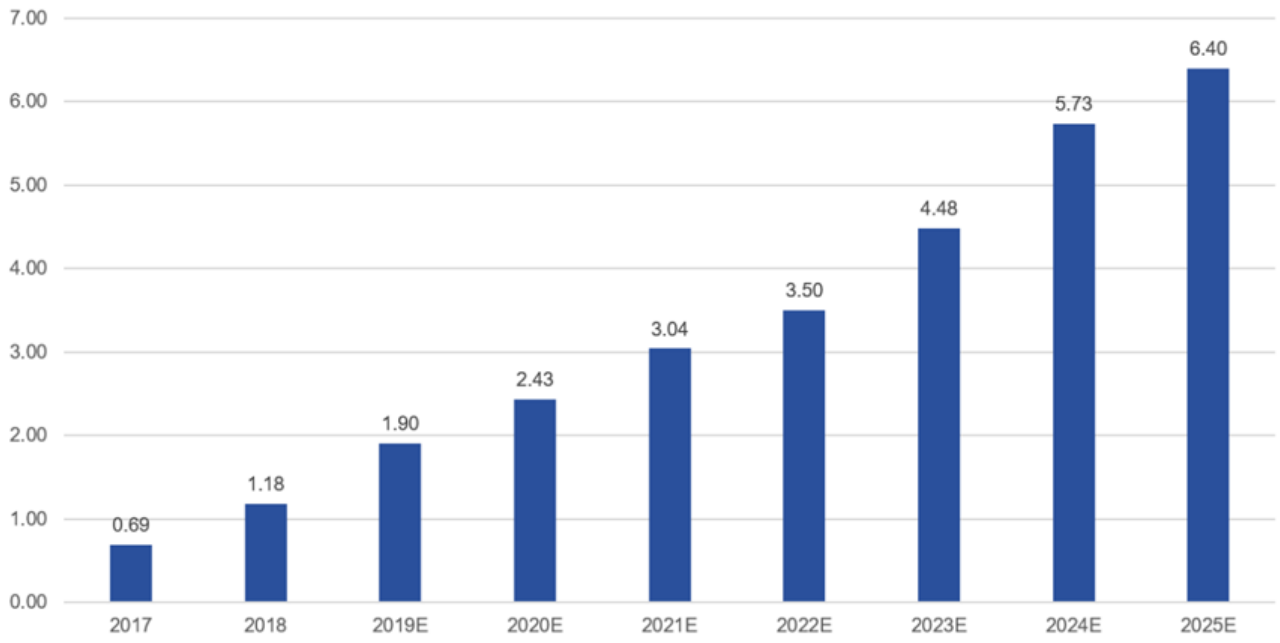

Figure 1AI market in every year (US dollars)

©2021 YOYOWORKS www.yoyoworks.com

1.2 GPU Virtualization Status

AI accelerators, including GPUs and FPGAs, just grow to be a fundamental AI market segment. GPU usually can be used by AI applications exclusively, and it results in high AI computing costs. Currently NVIDIA GPU is the dominant accelerator for AI computing, so the following introduction focuses on current state of technology based on NVIDIA GPU.

NVIDIA GRID Solutions

NVIDIA GRID technology is currently based on virtual machine platforms (VMware, Xen, KVM). Also, NVIDIA charges expensive software license fee for GRID product.

The other solution from NVIDIA is MPS technology. Through MPS server and MPS client, multiple GPU tasks can share the GPU. For container platforms, this way of sharing GPUs is a viable option. However, this command proxy technology has a drawback. If the MPS Server hangs for any issue caused by one MPS client, it will block all MPS clients under the same MPS server. So this solution is hardly used in the production environment.

Third-Party Solutions

a. Implementation in the NVIDIACUDA Library API layer, which uses hijacking calls to the CUDA API to achieve GPU virtualization. The disadvantage of this type of solution is that application compatibility depends on the releases from the native manufacture (NVIDIA), each API version needs to be aligned, and AI applications need to be recompiled. Thus the maintenance cost is high.

©2021 YOYOWORKS www.yoyoworks.com

b. Implementation in the Linux kernel. With zero-intrusion design, the implementation is not coupled with one CUDA version, and the application is not aware that it is running on a virtual GPU.

©2021 YOYOWORKS www.yoyoworks.com

2 . XPU Overview

XPU is a container GPU virtualization product introduced by YOYOWORKS LLC ("YOYO"). XPU adopts the kernel layer design described above. The core is to split the GPU at the kernel layer into many resource shares, and then simulate them into XPU devices (vGPU) for containers. XPU framework unbinds AI applications with physical GPUs, and binds them to virtual GPUs. With isolating fault, video memory, and computing in the scope of virtual GPU, XPU achieve business applications concurrency and better GPU hardware utilization.

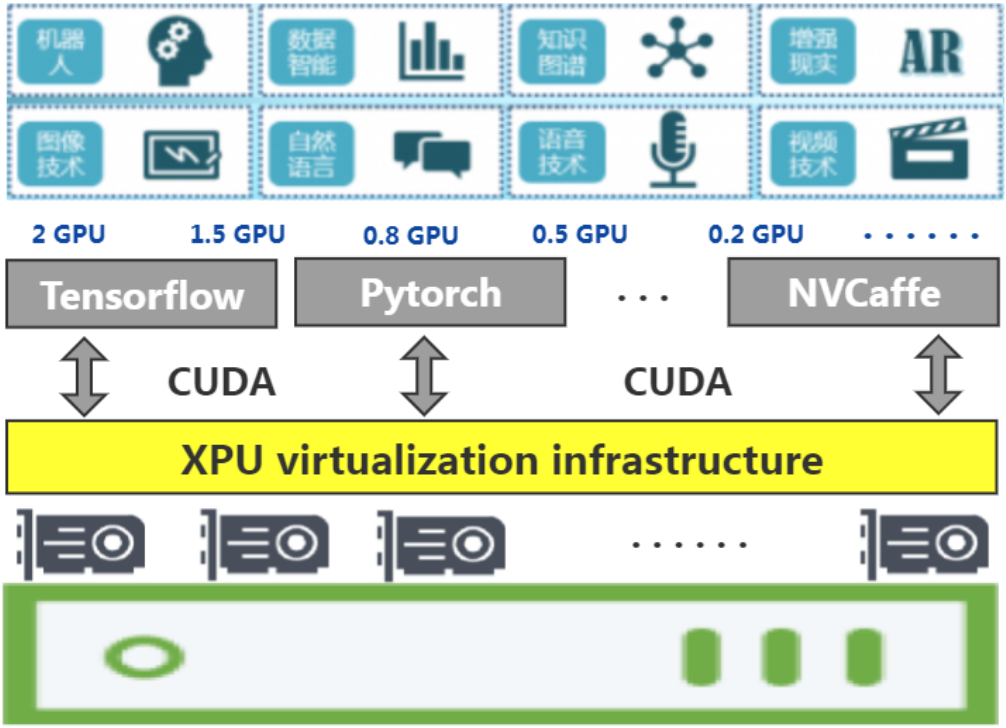

Figure 2XPUArchitecture

©2021 YOYOWORKS www.yoyoworks.com

3 . XPU Advantages

XPU adopts a zero-intrusion architecture design, provides virtual GPU devices for containers through kernel module, services and container runtime, isolates video memory and computing. It is well tuned for AI workload.

- **High Performance**
  - The GPU virtualization kernel layer engine achieves very good performance. Enabling XPU virtualization, usually the performance degradation is less than 5%.
  - Supports slicing the GPU into fine-granularity virtual GPUs, allowing multiple AI loads to run in parallel and improving physical GPU utilization.
  - Improving the comprehensive utilization rate of GPU by up to 3-10 times.
- **Security**
  - Achieving the computing isolation, video memory isolation and fault isolation among containers which share the same group of GPUs.
- **Compatibility**
  - Software compatibility: XPU guarantees application backward and forward compatibility. Legacy AI applications for example, container images such as NGC, can run on the XPU based platform without porting overhead or performance loss.
  - Hardware compatibility: XPU supports all series of NVIDIA Pascal and later GPUs (Telsa, Quadro/RTX, GeForce); it supports bare metal and virtual machine environments, and supports physical GPUs (bare metal or passthru GPU) and vGPUs.

©2021 YOYOWORKS www.yoyoworks.com

- Architecture compatibility: Thanks to the universal interface XPU driver module layer, XPU perfectly supports the framework and applications of CUDA8.0 and later versions.
- Ecology compatibility: compatible with industry standard Kubernetes and NVIDIA Docker products and solutions.

©2021 YOYOWORKS www.yoyoworks.com

4 . XPU Architecture

4.1 XPU Logical Architecture

The XPU has two components: the host-side component and the container-side component. The host-side component includes: XPU toolkit, XPU service & XPU (kernel) driver module. XPU toolkit and service take charge of virtual GPU management, and XPU driver module virtualizes one physical GPU to many vGPUs which can be assigned and used by containers. It is the host-side component that enforces fault isolation, compute isolation & memory isolation among containers. The container-side component includes: XPU container runtime. XPU container runtime helps the container recognize the virtualized GPUs just as physical GPUs, so the AI application inside the container can use the virtual GPUs even without knowing those are virtual GPUs. The diagram is as follows:

©2021 YOYOWORKS www.yoyoworks.com

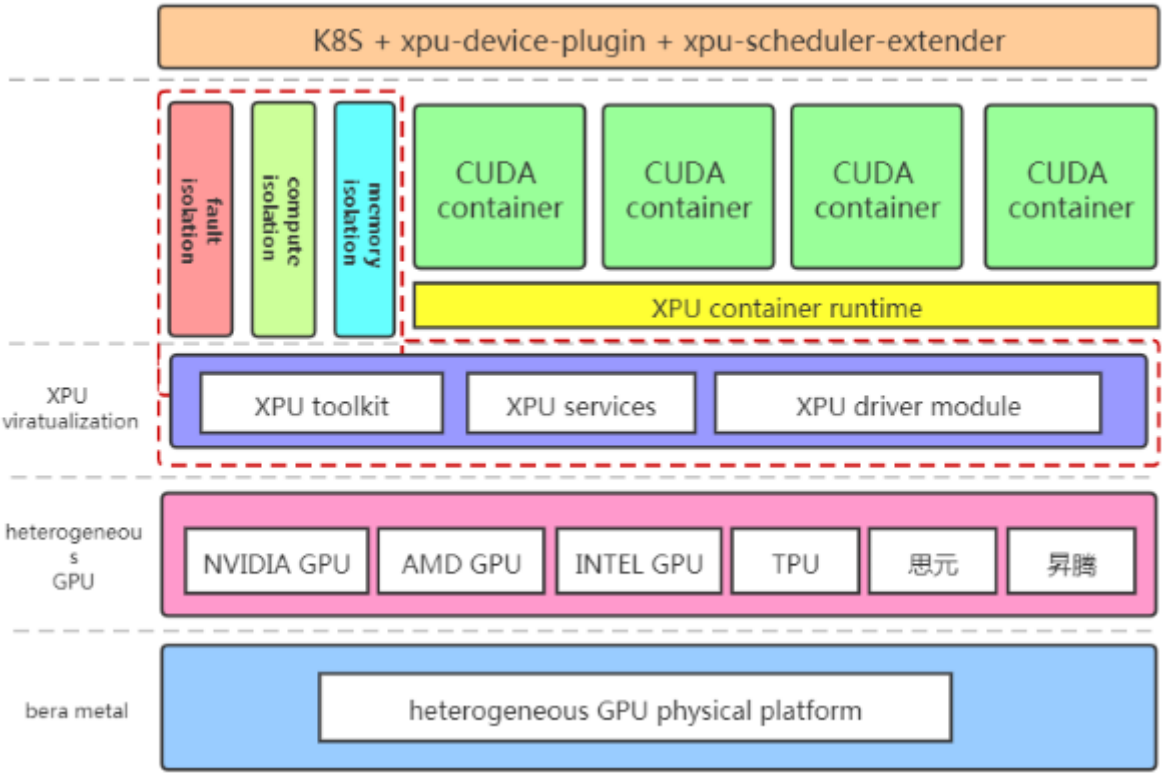

Figure 3 XPU Logical Architecture

©2021 YOYOWORKS www.yoyoworks.com

4.2 XPU Implementation in NVIDIA GPU

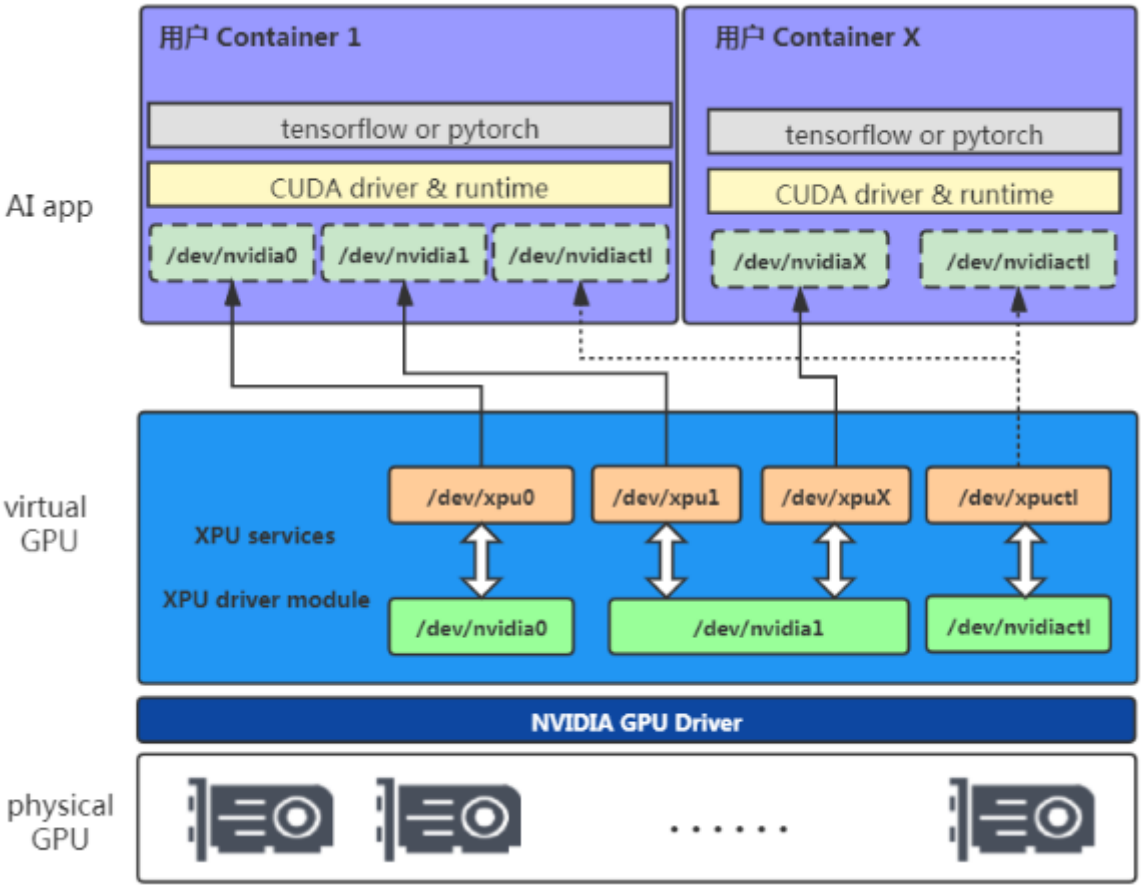

Figure 4 XPU Implementation in NVIDIA GPU

As shown in the figure above, XPU virtualizes the NVIDIA GPU into different virtual GPUs (XPU) devices and mapping them to the container, and the AI application containers can only access its own devices. It is the XPU virtualization layer that guarantees the isolation of video memory, computing, and faults among containers.

4.3 XPU Components

XPU Driver Module

XPU kernel driver layer is the core module that implements the virtual GPU. The physical GPU is virtualized, and unified XPU devices is simulated for use by containers. At the same time, the

©2021 YOYOWORKS www.yoyoworks.com

module discovers and manages physical GPU resources, abstracts the physical GPU into XPU, and enforces computing, video memory, and fault isolation by control the GPU access from applications.

XPU Service

Provides the XPU daemon and a few management utilities to control and manage XPU devices, including but not limited to: creating, deleting, monitoring the use of XPU, etc.

XPU Toolkit

Based on XPU services, a set of CLI commands are implemented to acquire, release, and dynamically adjust the XPU resources.

XPU Container Runtime

A set of libraries and utilities for managing XPU by Docker and Kubernetes. This component is a set of container runtimes used to creating, deleting and monitoring XPU devices. Users can acquire XPU devices of different size for containers by setting Docker environment variables. Also, this component is the basic component for Kubernetes to managing XPU devices.

©2021 YOYOWORKS www.yoyoworks.com

5 . XPU Deployment

XPU supports deployment on bare metal servers and on virtual machines.XPU supports mainstream Linux distributions such as RHEL, CentOS and Ubuntu Server.

5.1 XPU with Docker

XPU adopts a zero-intrusion deployment method and supports native containers to run AI applications, which greatly simplifies the operation and maintenance of algorithm engineers and management of AI infrastructure.

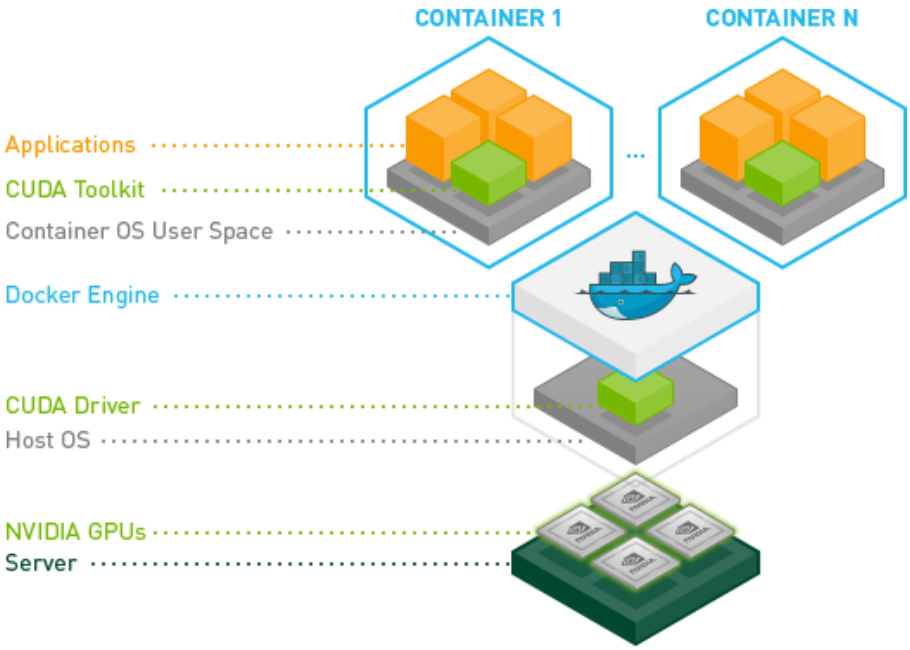

Figure 5NVIDIA Docker

Users can acquire XPU devices of different size for the container via setting Docker environment variables.

©2021 YOYOWORKS www.yoyoworks.com

5.2 XPU with Kubernetes

XPU Uses K8S device plugin and extended resources mechanism to integrate XPU into K8S cluster. The K8S device plugin mechanism is shown in the figure below:

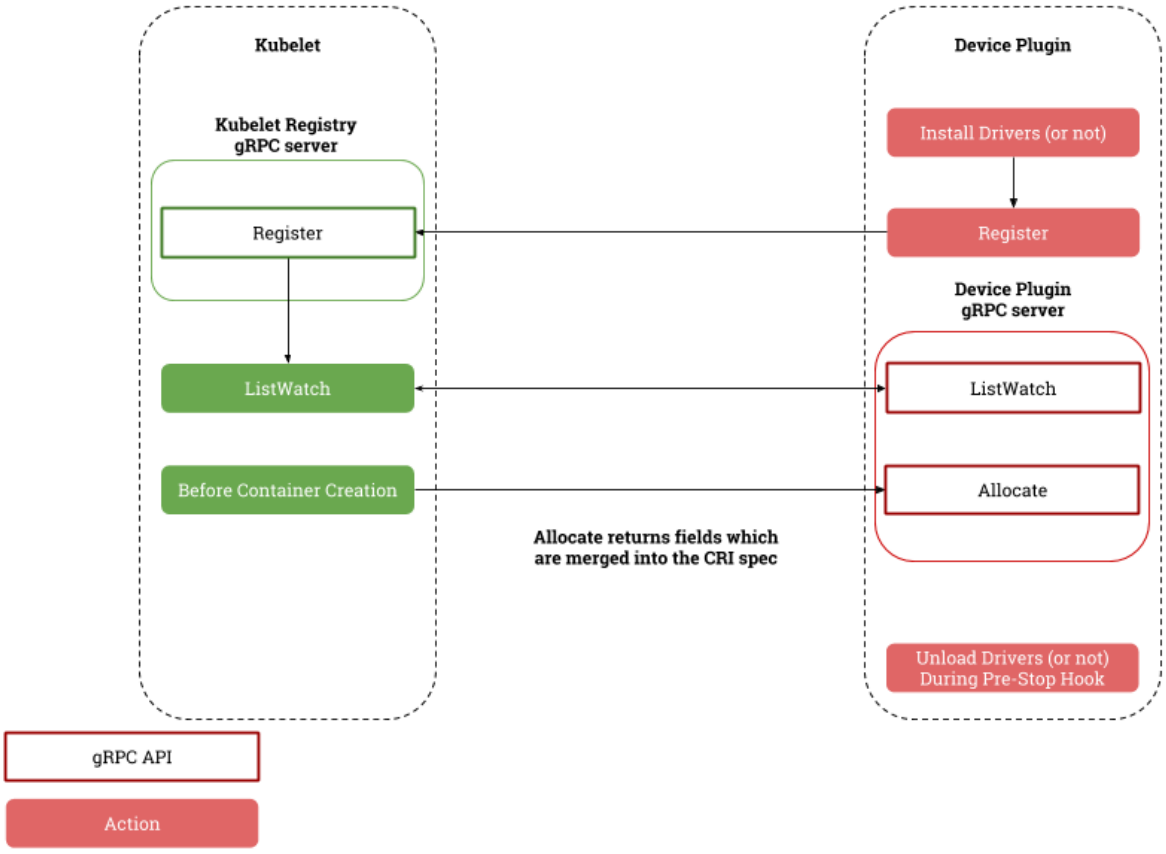

Figure 6K8S Device Plugin

Two plug-ins are used for Kubernetes to control and manage XPU. System administrators only need to complete the centralized configuration and scheduling management of GPU resources in K8S. In addition, system administrators are allowed to dispatch all data center resources through a single interface to realize SDDC (Software Defined Data Center).

The two plug-ins provided by XPU for Kubernetes:

- XPU Kubernetes Device Plugin

- Collecting XPU resource pool information by communicating with XPU Driver Module and XPU Service.
- Registering the resource named *yoyoworks.com/xpu-shares* with Kubernetes through the Device Plugin standard defined by Kubernetes.

- XPU Kubernetes Scheduler Extender
  - Providing loosely coupled scheduling extension functions based on HTTP API.
  - Registering the resource keyword named *yoyoworks.com/xpu-shares* with Kubernetes through the configuration file, and making it point to the HTTP service address of XPU Kubernetes Scheduler Extender.

©2021 YOYOWORKS www.yoyoworks.com

6 . Practice Scenarios

XPU supports users to dynamically allocate and release required GPU resources during the life cycle of the container. Thus XPU achieves true dynamic scaling of GPU resources, which greatly improves the flexibility of GPU resource scheduling. XPU supports the allocation of GPU resources on-demand, maximizing the use of computing resources. According to the training or inference model size, users can dynamically adjust the size of the computing according to the needs of the AI model. According to user needs, the allocation method can be designed into the following two types:

Distributed Scheduling

With this method, users can make full use of resources when they are idle by scheduling the computing tasks on multiple cards as much as possible. It is mainly used for the scenario that a variety of tasks are started at different times. When some tasks are running, the other tasks are idle. This scheduling method can decrease task completion time as much as possible.

Centralized Scheduling

With this method, users can consolidate their tasks on least GPUs as much as possible. In this way, maximum tasks can be scheduled on the GPU pool, and we can achieve best batch throughput.

©2021 YOYOWORKS www.yoyoworks.com

7 . Performance Test

7.1 Test Environment

Hardware Configuration

CPU: Intel Xeon Gold 6132

Memory:128GB

GPU: NVIDIA Tesla P40

Software Configuration

NVIDIA GPU Driver 465.31

NVIDIA CUDA 8.0

Test Cases

Tensorflow，AlexNet benchmark，MNIST，CIFAR10

7.2 Test Results

Native GPU: the test case is run on a Docker container that has not been virtualized, that is, a physical GPU is assigned to the container.

XPU: the test case is run on a virtualized Docker container, that is, XPU is assigned to the container.

XPU=0.5GPU: an XPU uses 50% of the computing resource and memory of the physical GPU.

©2021 YOYOWORKS www.yoyoworks.com

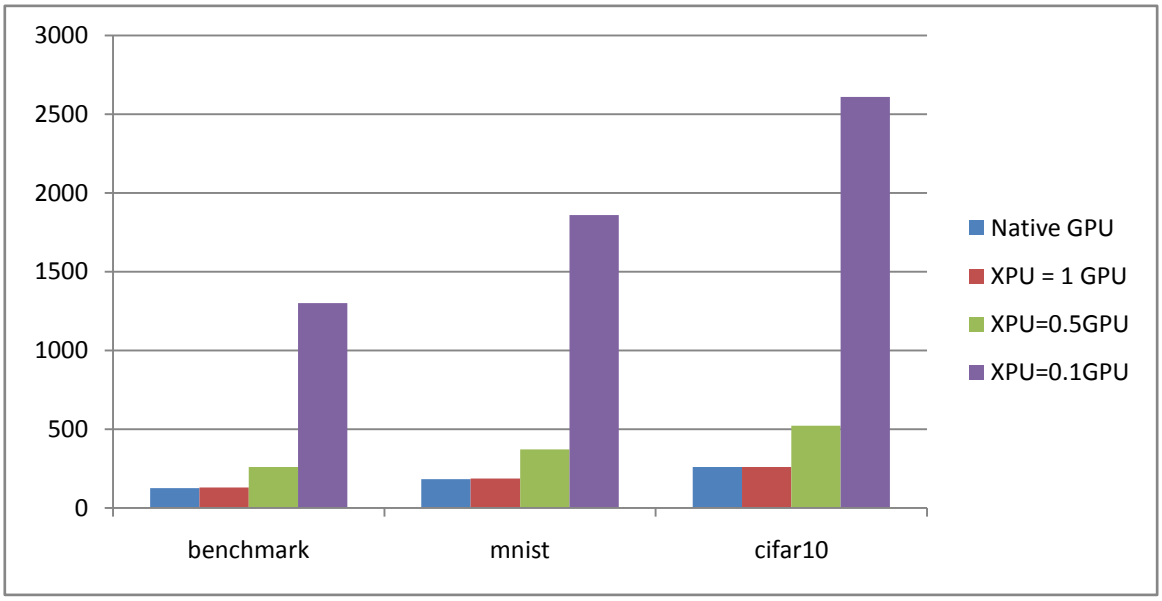

Figure 7XPU Performance test results

The test is performed on different data sets, and the time spent training is based on XPUs of different size.

7.3 Conclusion

a. When XPU=1GPU, the running times are approximately equal, indicating that the performance loss of the XPU virtualization layer to the physical GPU is close to zero;

b. When XPU=0.5GPU or XPU=0.1GPU, the calculation time is approximately 1 and 10 times the calculation time of the physical GPU. XPU effectively splits and isolates the physical GPU computing resources; the sum of the latter XPU is approximately equal to 1 GPU computing, indicating that XPU virtualization has minimal loss of GPU performance.

©2021 YOYOWORKS www.yoyoworks.com

8 . Compatibilities

NVIDIA GPU

All Pascal and later GPUs, including all series of Tesla/RTX/Quadro/GeForce GPUs:

-Support physical NVIDIA GPUs in bare metal environment

-Supports passthru NVIDIA GPUs in virtual machines environment

-Support NVIDIA GRID vGPU in virtual machines environment

NVIDIAGPU：

Tesla	A100/A10/A16/A30/A40 T4 V100 P100/P40/P6/P4
RTX	A6000/A5000/A4000
Quadro	RTX8000/RTX6000/RTX5000/RTX4000 P6000/P5000/P4000
GeForce	All 30XX, for example3090/3080Ti All 20XX, for example 2080/2080Ti All 10XX, for example 1080

Note:

-XPU is supposed to be compatible with all Pascal and later GPUs, but the above are well supported.

OS

CentOS 7.x/8.x，RHEL 7.x/8.x (64 bit)

Ubuntu Server 18.04/20.04 LTS (64 bit)

NVIDIA CUDA

Fully supports CUDA 8.x, CUDA 9.x, CUDA 10.x, CUDA 11.x containers and CUDA applications

Note:

©2021 YOYOWORKS www.yoyoworks.com

-Only supports 64-bit CUDA containers and CUDA applications

Cloud

Docker:19.03 and later versions

Kubernetes: 1.18 and later versions

©2021 YOYOWORKS www.yoyoworks.com