The differences between CUDA version PyTorch (Traditional PyTorch) and ROCm PyTorch:

1. Kernel Function Language: The kernel functions in traditional PyTorch are typically written in CUDA, while the kernel functions in ROCm PyTorch are written in HIP. This means that the implementation of core parallel computing operations differs at the source code level.
2. Compilation and Build System: Traditional PyTorch utilizes the CUDA compilation toolchain and build system, whereas ROCm PyTorch uses the ROCm compilation toolchain and build system. This results in different tools and configurations required during the build and installation processes for the two PyTorch versions.
3. Underlying Library Calls: Traditional PyTorch relies on CUDA-related underlying libraries such as cuDNN and NCCL, while ROCm PyTorch uses ROCm-related libraries like MIOpen and HIP. These libraries provide tight integration with the hardware and software stack to achieve high-performance computing on different platforms.
4. Platform-Specific Code: ROCm PyTorch may contain platform-specific code to handle specific functionalities and features of AMD GPUs. This code is typically aimed at optimizing and supporting specific functionalities on AMD GPUs.

https://www.youtube.com/watch?v=mPrfh7MNV\_0

The relation between CUDA version PyTorch (Traditional PyTorch) and ROCm PyTorch:

The ROCm version of PyTorch is not completely independent of CUDA. Although ROCm PyTorch is an optimized version for running on AMD GPUs, it still has dependencies and associations with CUDA.

In ROCm PyTorch, certain core functionalities and modules still rely on CUDA. This is because some PyTorch operations and underlying libraries, such as cuDNN and NCCL, currently provide the best performance and functionality on CUDA. Therefore, when running ROCm PyTorch on AMD GPUs, there is still a need for some level of CUDA support.

ROCm PyTorch uses the HIP language to adapt parallel computing for AMD GPUs, allowing for the conversion of some PyTorch code originally written in CUDA to HIP. This is done to enable compilation and execution on AMD GPUs. However, for operations that still depend on CUDA, ROCm PyTorch utilizes compatible CUDA paths to maintain functionality and optimize performance.

In summary, ROCm PyTorch has dependencies on CUDA, despite utilizing the HIP language and the ROCm software stack for AMD GPUs. This is because certain core functionalities and underlying libraries still require CUDA for optimal performance and functionality.

See

system version : lsb\_release -a

rocm environment : rocminfo

Preconditions :

The minimum requirements of the single node server are that it should have X86-64 CPU or CPUs along with GPU(s) of AMD instinct MI100 and GPU of Radeon instinct MI50 (S). The operating system used for this should be Centos 8.3 or higher OR Ubuntu 18.04 or higher version. The driver for ROCm should be compatible with the 4.2 version, and the Docker Engine Singularity container is used at runtime.



 