Group 64: Enabling GPU Support on NumS

Kunal Agarwal, Parth Baokar, Brian Park

Problem

- GPU allows for faster computation compared to CPU.
- CUDA C/C++ is not easy to program.
- NumS usage was originally intended for cloud computing, not HPC systems.

Approaches

- Single GPU with CuPy
- Multi-GPU with CuPy and NCCL
  - Able to utilize NVLink connections for GPU-GPU memory transfers.
  - Using NCCL is not fault tolerant and may lead to deadlocks (just like MPI).
- Multi-GPU with CuPy and Ray
  - Ray incurs a lot of overhead due to Object Store being part of main memory.
  - It is not aware of NVLink connections.

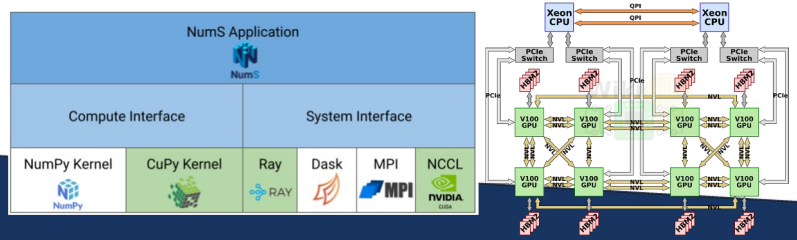
Benefits

- CuPy uses optimized packages from CUDA Toolkit (cuBLAS, NCCL, etc).
- Ease of use and debugging with Python compared to CUDA C/C++.
- One can easily change NumS backend from CPU to GPU with a single line of code.
- Uses existing NumS kernels and algorithms that are communication optimal.

Challenges

- CuPy kernels execute fast, so overhead of function dispatch becomes noticeable.
- Accessing low-level CUDA API through Python is not elegant and has limited support.
- GPU memory is limited, thus hits OOM earlier (32GB for each V100).

Design



Preliminary Benchmarks

- Benchmarks done on Bridges-2 with 8 NVIDIA V100s connected with NVLink using Multi-GPU with NCCL backend.