Cloud Platform Optimization for HPC

Aman Verma(✉)

Microsoft, Redmond, WA 98052, USA
Verma.Aman@microsoft.com

Abstract. The special requirements of HPC have typically been tacked onto existing cloud infrastructure and practices. As a result, most cloud offerings aren't completely optimized for HPC, or aren't yet feature-complete as far as traditional supercomputing experience is concerned. This work addresses the progress made in (1) optimizing the performance of HPC workloads in a cloud environment, and (2) evolving the usability of cloud HPC environments. Specifically, this work discusses efforts made to minimize and eliminate the impact of virtualization on HPC workloads on cloud infrastructure and move towards a more familiar supercomputing experience. Initial experience with "cloud-native" HPC is also discussed. In many aspects, this work is inspired by and impactful for many HPC workloads in many disciplines including earth sciences and manufacturing.

Keywords: HPC · Cloud · Performance · Scalability

1 Introduction

The advent of cloud computing offers the promise of virtually unlimited resources, elasticity in scale, available on demand, with the appeal of access to the latest advances in technology in both hardware and software. The availability, flexibility and elasticity of cloud computing makes it appealing to a wide variety of workloads, including those in science and engineering. Many of the problems in the domain of scientific computing generally fall in at least one of the following 2 classes: (1) simulation of modeled representation of the physical world (computational physics, chemistry, mechanics, etc.), and (2) analysis of large amount of data (astrophysics, genomics, etc.). Both classes of problems, but more so the first have special demands on the computing infrastructure compared to other non-scientific computing workloads. Hence, many of these computing-intensive scientific computing workloads resort to "high performance computing (HPC)" primarily to minimize the time to solution or "time-to-science".

Cloud infrastructure has had to adapt to meet the "high performance" requirements of such workloads but mostly as an afterthought. This is primarily due to the fact that the vast majority of the public cloud as we know it had been built for, and has evolved out of the demand for consumer applications such as hosting websites, databases, object storage, content delivery, gaming, etc. Virtualization is one of the key technologies that has enabled the simulation and pooling of multiple dedicated resources from limited physical hardware. Hence virtualization is a common technology adopted

D. K. Panda (Ed.): SCFA 2020, LNCS 12082, pp. 55–64, 2020.
https://doi.org/10.1007/978-3-030-48842-0_4

by cloud providers to reduce hardware, save energy and offer more customized resource options to customers at attractive price points while leveraging the economies of scale.

Given all its advantages though, virtualization does not come free and incurs resource overhead. Ultimately, this overhead must be accounted for within the same pool of physical hardware, leaving slightly reduced resources for the customer workload, or worse, noise or interruption adversely impacting the customer workload. This is in stark contrast to the bare-metal environments the HPC community has been used to, over decades, by dint of practice and familiarity. This work addresses the progress made in (1) optimizing the impact of virtualization on HPC workloads in a cloud environment, and (2) evolving the usability of cloud HPC environments. Specifically, this work discusses efforts made to minimize and eliminate the impact of virtualization on HPC workloads on cloud infrastructure and move towards a more familiar supercomputing experience. The impact of such work is felt resoundingly in all industries that (1) traditionally have always been at the forefront of advancing HPC capabilities, and (2) are currently undergoing an all-encompassing digital transformation leveraging cloud computing.

2 Existing Gaps

The special requirements of HPC have typically been tacked onto existing cloud infrastructure and practices. As a result, most cloud offerings aren't completely optimized for HPC, or aren't yet feature-complete as far as traditional supercomputing experience is concerned. The key gaps are in the areas of:

(1) minimizing and eliminating impact of the virtualization layer (hypervisor),
(2) bare-metal-like representation of the hardware to the workload, and
(3) the HPC software ecosystem.

While the third item concerns more with the readiness and usability of the environment, the first two items directly impact the performance of HPC workloads.

The first two "performance" related items can be addressed by truly bare-metal instances which many cloud providers offer, and which come with a different set of considerations. Another common theme among cloud providers is that they offer instances with specific features exposed natively, in as bare-metal a state as possible, through the virtualization layer. The implementations differ and so do the feature-set and underlying performance and ease of usage. The "usability" is commonly addressed through one of two ways: (1) ready-to-use operating system images, preloaded with the right drivers, packages and applications to use the features out of the box, and (2) scripts or instructions to enable and use features. Solution approaches to address the existing gaps as listed above are described in greater detail as follows.

3 Methods

The optimization performed on the cloud computing platform are described as follows.

3.1 Eliminate "Jitter"

One of the biggest concerns of running an HPC (or any) workload on shared resources such as on the cloud is that of reduced performance due to a "noisy neighbor". At least on the aspect of sharing compute resources, this can be rather trivially addressed by hosting only 1 customer virtual machine (VM) per compute node. While the economics of this depends on the specifications of the compute node and the customer workload, this arrangement makes complete sense for HPC workloads. Compute resource intensive workloads, such as in HPC and AI, should first scale up (on the same compute node) before scaling out (to multiple compute nodes). Hence providing a full compute node per customer VM eliminates the "noisy neighbor" issue.

The issue of minimizing and eliminating the impact of the hypervisor can be addressed separately for compute and networking. The compute resources allocated to the hypervisor can be separate from the compute resources allocated to the customer VM or workload. On Azure, where the hypervisor is essentially a very stripped down version of Windows Server, this is accomplished using Minroot [1] and CpuGroups [2]. Minroot is used to constrain and isolate the compute resources (host virtual processors) allocated to the hypervisor. CpuGroups is used to group, constrain and isolate the compute resources (host virtual processors) allocated to the customer VM(s). As Fig. 1 illustrates, in the case of HPC on Azure HB-series, there is 1 customer VM per node and the hypervisor resources (physical cores 0–3) are separate from the VM's resources (physical cores 4–63). The VM on this 64-core node sees 60 cores divided across 15 NUMA nodes, isolated from any interference or "jitter" from the hypervisor.

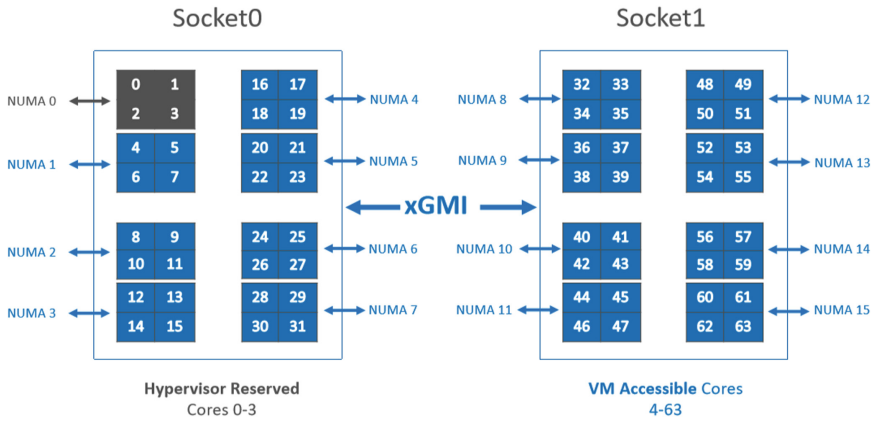

Fig. 1. Separation and isolation of the hypervisor from the VM to eliminate 'jitter'.

Performance jitter due to noisy neighbors in networking is a different topic but is an integral one when eliminating jitter holistically from a system. Such "networking jitter"

can be trivially eliminated in a single node case when there is no inter-node communication over a network. However this trivial case is not interesting since the compute hypervisor jitter really manifests and becomes important when involving inter-node communication at a large enough scale. On Azure, the network jitter is attempted to be mitigated with the use of a balanced, non-blocking, fat-tree cluster and Adaptive Routing (AR) on the InfiniBand fabric [10]. With destination-based routing, AR enables the source node to select alternate paths to the same destination, allowing congested paths in the network to be avoided. This mitigation of networking jitter is demonstrated in Fig. 2 where enabling AR improves application (Star-CCM+) performance, particularly at scale on Azure HB-series with improvement up to 17% higher at 96 nodes when compared to AR disabled.

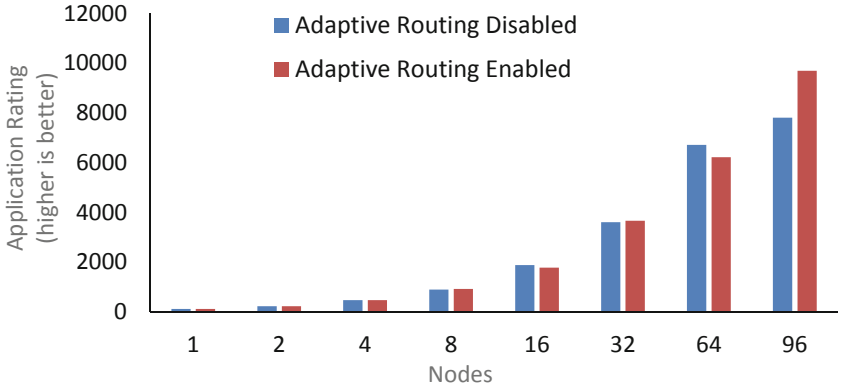

Fig. 2. Adaptive routing in the InfiniBand fabric mitigates "networking jitter".

3.2 Resource Virtualization

The impact of the virtualization on the networking is overcome through Single Root Input/Output Virtualization (SR-IOV). This technology allows device virtualization without using device emulation by enabling the PCIe resources to expose virtual functions (VFs) to virtual components (such as network adapter). This allows the hypervisor to map VFs to VM(s), which can achieve native device performance without using passthrough [3]. For HPC on Azure (e.g. HC-series), this is used for the InfiniBand network. This allows HPC and AI workloads to take advantage of all Message Passing Interface (MPI) implementations (and other frameworks based on MPI such as Horovod) and Remote Direct Memory Access (RDMA) verbs natively. SR-IOV for InfiniBand allows (1) customers to bring over a familiar (and any) HPC stack to the cloud, and (2) expose advanced networking features for optimized performance (HCOLL, SHARP, etc.). Figures 3 and 4 demonstrate the native performance of MPI point-to-point benchmarks – latency and bandwidth. This data is from running the OSU microbenchmarks with 3 MPI implementations: HPC-X, IntelMPI 2018, MVAPICH2 on Azure HC-series and HB-series for the latency and bandwidth tests respectively.

Complementary to the work above, performance with SR-IOV for InfiniBand had been shown to be comparable to that of bare-metal InfiniBand [4]. For compute

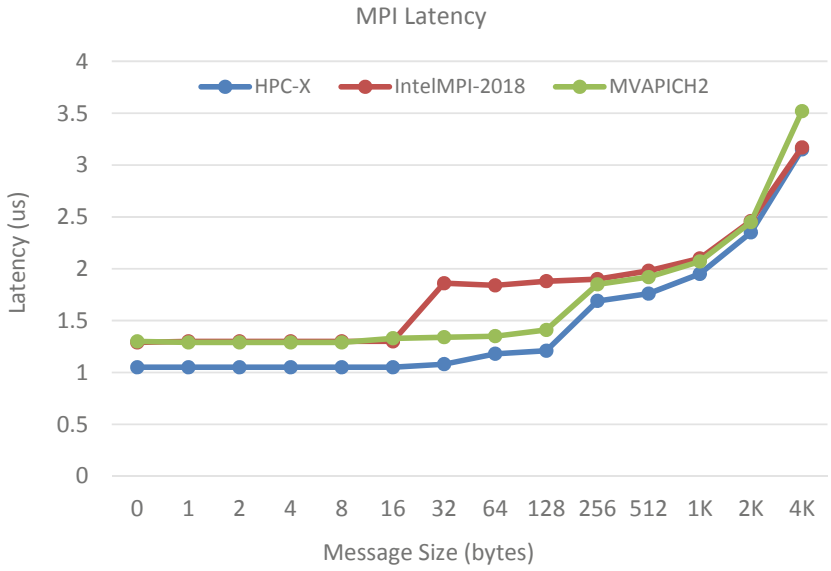

Fig. 3. Massive improvement in MPI latency due to platform updates, close to bare-metal performance expectations.

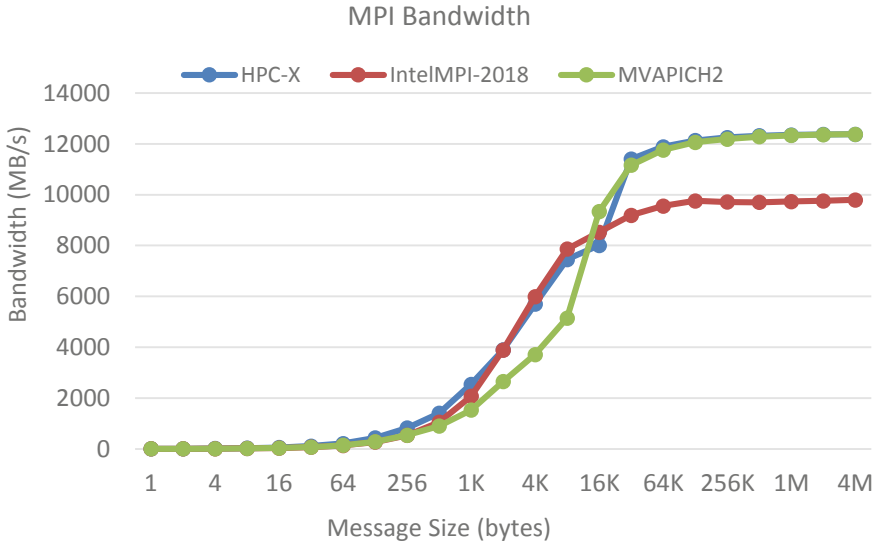

Fig. 4. MPI bandwidth near line rate (InfiniBand EDR 100 Gbps).

resources, there are other challenges with respect to mapping the physical CPU topology as-is to the virtual representation. This is especially important for chiplet-like designs with multiple NUMA nodes (or groups of L3-cache). On Azure HB-series and HC-series, the absence such mapping in a deterministic and inconsistent manner has

resulted in differing performance of different realizations of the same experiment. Figure 5 is a fictional representation of the "scrambling" of the NUMA node numbering in the VM for a 2 socket, 16 NUMA node architecture. Applications may experience the impact of this in the way of reduced performance when, for instance:

- the data it was sharing with a NUMA node it was assuming to a neighbor is actually across an inter-socket link, or
- a process pinned to a core in a NUMA node assuming the NIC is affinitized nearby, attempts to broadcast message elsewhere only to realize that the bandwidth is reduced on account of the physical NIC being on a far NUMA node.

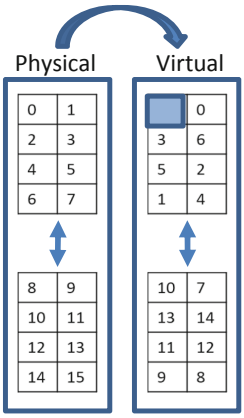

Fig. 5. Representation of the challenges of inconsistent NUMA node mapping.

Having an accurate view of where the workload processes (within the VM) are running on the physical node is important to plan out proper process placement and pinning, and eke out optimal performance. This issue is addressed in later version of the hypervisor which enables deterministic and consistent mapping of the NUMA nodes from the physical topology to the virtual presentation (pNUMA->vNUMA mapping). Note that corresponding pCore->vCore mapping at a granular core level is still ongoing work.

3.3 Software Ecosystem

The above work has been focused on the "performance" aspects of the platform; the "usability" side of the platform is equally important. Users of supercomputing facilities are accustomed to seeing pre-configured scientific computing and HPC packages available as ready-to-use, loadable modules. This greatly reduces the barrier to entry for new, first time users for such platforms, maintain control over the proliferation of custom environments, as well as provide some guarantees on function and performance of the various applications and packages. Spack is gaining popularity among system

administrators of such facilities as a flexible package manager to support multiple versions, configurations, compilers and platforms.

An optimized, performant, and easy to use HPC software ecosystem allows customers to get native and designed-for performance right away. To this end, the following are made available on Azure:

(1) optimized VM OS images based on CentOS 7.6 and 7.7, with popular MPI implementations and scientific computing libraries [5],
(2) an optimized MPI implementation (MVAPICH2-Azure),
(3) native support for Open Container Initiative (OCI) format container images [6], including Singularity.sif image files
(4) recipes for scientific computing containers [7, 8], and
(5) Spack repo with integrated buildcache on Azure Blob (object storage) [11].

4 Results

The composite result of the progress made in (1) optimizing the performance of HPC workloads in a cloud environment, and (2) evolving the usability of cloud HPC environments is illustrated in Figs. 6, 7, 8 and 9. Figure 6 shows the performance of an open source reservoir simulator OPM Flow [9]. Expanding beyond the scope of traditional HPC applications, Fig. 7 shows the advantages offered by RDMA over InfiniBand for "big data" workflows leveraging SparkRDMA [12]. Figure 9 shows an example of running Horovod, a popular distributed AI training framework for TensorFlow; the efficiency of RDMA over InfiniBand outpacing that of IPoIB even at the relatively small scales. Both these experiments (Figs. 7 and 8) are performed on the Azure HC-series, with the Horovod experiment using Intel MPI. Finally Fig. 9 shows the scaling of a popular CFD application (Star-CCM+) up to 37000 cores on Azure HB_v2-series. This has now been extended to 57,000 cores which is a record run for HPC in the cloud.

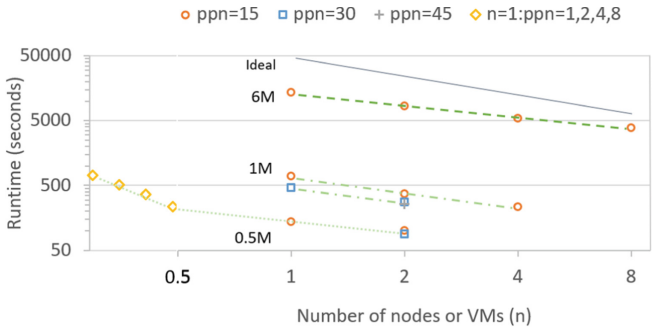

Fig. 6. Comparison of the runtime for the OPM reservoir simulator on HB with 3 different cases (0.5 M, 1 M and 6 M) for a combination of nodes and processes per node.

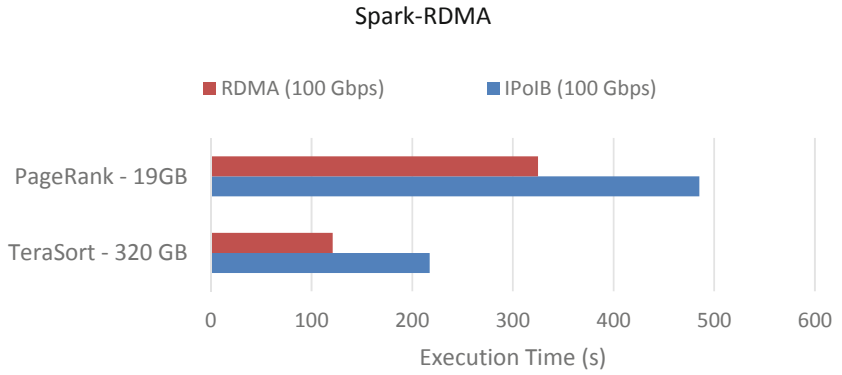

Fig. 7. Advantage of RDMA over InfiniBand for "big data" workloads using SparkRDMA.

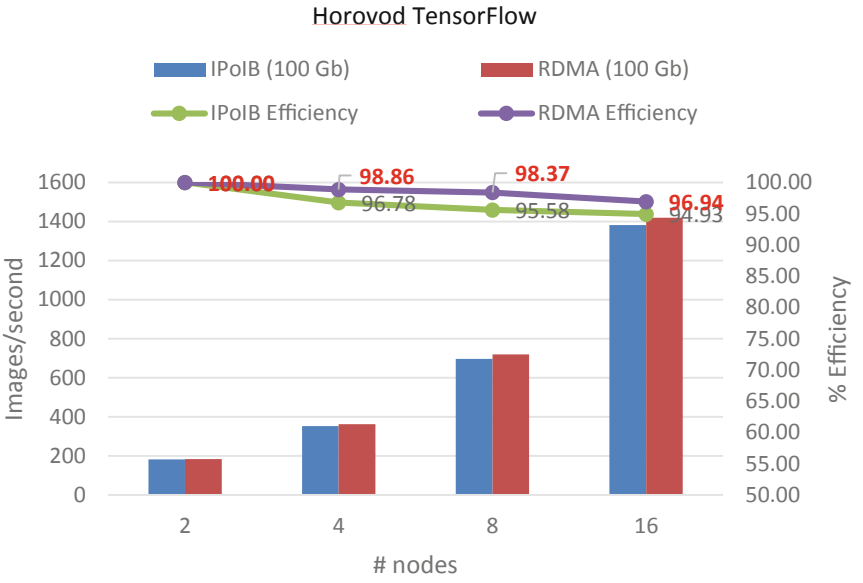

Fig. 8. Running Horovod, a distributed AI training framework for TensorFlow on an HPC platform.

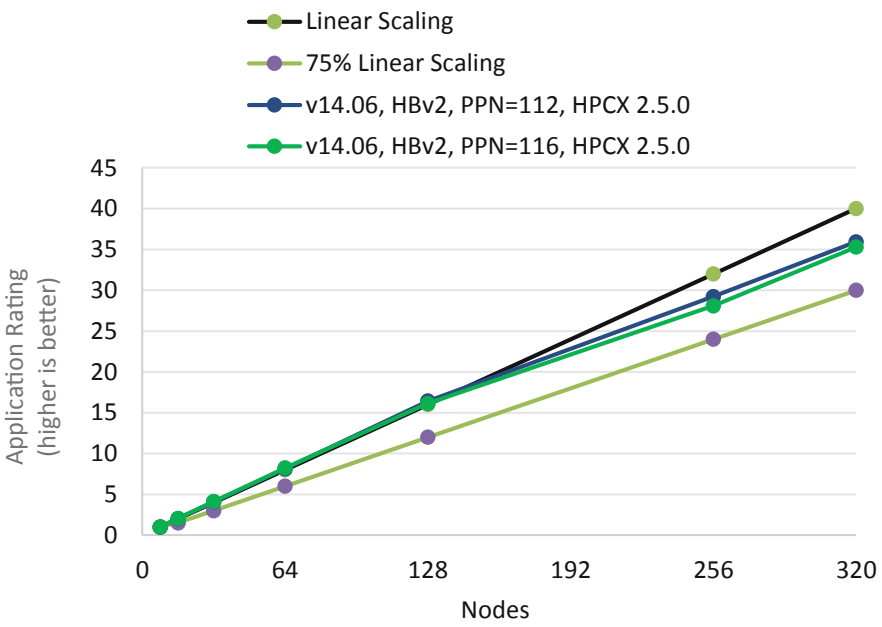

Fig. 9. Scaling of the CFD simulator Star-CCM+ up to 37000 cores.

5 Future Work

From an HPC platform perspective, the above work described the optimizations made for compute and clustered networking to enable not just the traditional HPC workloads, but also the emerging "big data" and AI workloads. A key piece of the puzzle to achieve complete parity with on-prem infrastructure, ecosystem and experience is HPC storage. While there is significant momentum in the convergence of HPC compute and networking to support the traditional HPC and AI workloads, the storage space appears to be evolving disjoint requirements and preferences. There may be a "divergence of HPC and AI" as far as storage is concerned, but this is evolving. There is ongoing work with "cloud-native" HPC which concerns "cloud-native" orchestration of resources, monitoring, logging, and interaction with distributed data stores.

References

1. https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/manage/manage-hyper-v-minroot-2016
2. https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/manage/manage-hyper-v-cpugroups
3. Efficient High-Performance Computing with Infiniband Hardware Virtualization. http://dataslys.cs.iit.edu/reports/2014_IIT_virtualization-fermicloud.pdf

4. SR-IOV Support for Virtualization on InfiniBand Clusters: Early Experience. http://mvapich.cse.ohio-state.edu:8080/static/media/publications/abstract/sriov-ccgrid13.pdf
5. https://techcommunity.microsoft.com/t5/Azure-Compute/Azure-CentOS-7-6-7-7-HPC-Images/ba-p/977094
6. https://techcommunity.microsoft.com/t5/Azure-Compute/Singularity-on-Azure-Containers-for-HPC/ba-p/464174
7. https://docs.nvidia.com/ngc/ngc-azure-setup-guide/index.html
8. https://github.com/vermagit/hpc-containers/tree/master/singularity/recipes
9. https://techcommunity.microsoft.com/t5/Azure-Compute/Reservoir-Simulation-on-Azure-HPC-for-Oil-amp-Gas/ba-p/791986
10. https://community.mellanox.com/s/article/How-To-Configure-Adaptive-Routing-and-SHIELD
11. https://github.com/Azure/azurehpc/tree/master/apps/spack
12. https://github.com/Mellanox/SparkRDMA