Performance Analysis of mdx II: A Next-Generation Cloud Platform for Cross-Disciplinary Data Science Research

Keichi Takahashi[a], Tomonori Hayami[b], Yu Mukaizono, Yuki Teramae and Susumu Date[c]
D3 Center, University of Osaka, Osaka, Japan

Keywords: Cloud Computing, Data Science, Performance Evaluation, OpenStack.

Abstract: mdx II is an Infrastructure-as-a-Service (IaaS) cloud platform designed to accelerate data science research and foster cross-disciplinary collaborations among universities and research institutions in Japan. Unlike traditional high-performance computing systems, mdx II leverages OpenStack to provide customizable and isolated computing environments consisting of virtual machines, virtual networks, and advanced storage. This paper presents a comprehensive performance evaluation of mdx II, including a comparison to Amazon Web Services (AWS). We evaluated the performance of a 16-vCPU VM from multiple aspects including floating-point computing performance, memory throughput, network throughput, file system and object storage performance, and real-world application performance. Compared to an AWS 16-vCPU instance, the results indicated that mdx II outperforms AWS in many aspects and demonstrated that mdx II holds significant promise for high-performance data analytics (HPDA) workloads. We also evaluated the virtualization overhead using a 224-vCPU VM occupying an entire host. The results suggested that the virtualization overhead is minimal for compute-intensive benchmarks, while memory-intensive benchmarks experienced larger overheads. These findings are expected to help users of mdx II to obtain high performance for their data science workloads and offer insights to the designers of future data-centric cloud platforms.

1 INTRODUCTION

The rapid advancements in data collection and data analysis capabilities have led to the widespread adoption of data-driven approaches in scientific research. There is thus an increasing demand within the academic community for computational infrastructures that facilitate the aggregation, storage, and analysis of large-scale data. However, the functionalities and performance requirements of such infrastructures vary significantly across different academic fields and projects, making it impractical to develop a singular infrastructure tailored to a specific field or project.

To address this situation, research institutions in Japan envisioned a concept of a cloud platform known as *mdx*. mdx is jointly procured and operated by nine national universities and two research institutes and provides service to educational and research institutions and private companies across Japan. The first implementation of the mdx concept, named mdx I (Suzumura et al., 2022), was installed at the University of Tokyo and began offering services to users in September 2021. To enable continued operation during system maintenance or replacement, and to strengthen disaster resistance and fault tolerance, the second-generation mdx named mdx II was installed at the University of Osaka (Figure 1) and started its service in November 2024.

mdx II is an Infrastructure-as-a-Service (IaaS) cloud platform for data science research. Unlike traditional High-Performance Computing (HPC) systems that use batch job schedulers and bare metal servers, mdx II is based on the OpenStack cloud computing platform. This allows users to create isolated, tailor-made computing environments consisting of virtual machines, virtual storage, and virtual networks, to meet their diverse compute, storage, and network requirements. mdx II also provides a variety of storage types and access methods to support data ingestion block storage, parallel file system, and object storage which can be accessed through Lustre, S3 API, and web interface.

Although at the hardware level mdx II resembles traditional supercomputers, its software stack and use

[a] https://orcid.org/0000-0002-1607-5694
[b] https://orcid.org/0009-0003-8165-2253
[c] https://orcid.org/0000-0001-7159-289X

Takahashi, K., Hayami, T., Mukaizono, Y., Teramae, Y. and Date, S.
Performance Analysis of mdx II: A Next-Generation Cloud Platform for Cross-Disciplinary Data Science Research.
DOI: 10.5220/0013485100003950

In *Proceedings of the 15th International Conference on Cloud Computing and Services Science (CLOSER 2025)*, pages 92-100
ISBN: 978-989-758-747-4; ISSN: 2184-5042

Figure 1: Server racks installed with servers, storage, and network devices constituting mdx II.

cases are significantly different from supercomputers. In this paper, we thus carry out a comprehensive performance evaluation of mdx II to provide current and potential users with excepted performance characteristics of the system and suggestions to optimize the performance of High-Performance Data Analytics (HPDA) workloads on mdx II. Based on the performance evaluation and analysis, we also aim to provide feedback on the design and configuration of the mdx II system to its operators, as well as designers of future data-centric cloud platforms.

The rest of this paper is structured as follows. Section 2 briefly introduces the overall architecture of the mdx II system, and compares it with other academic cloud systems. Section 3 carries out a comprehensive performance evaluation and analysis of the mdx II system. Section 4 concludes this paper and discusses future work.

2 BACKGROUND

2.1 Overview of mdx II

Figure 2 shows an overview of the mdx II system. mdx II comprises 60 compute nodes each equipped with two Intel Xeon Platinum 8480+ (Sapphire Rapids) processors and 512 GiB of DDR5-4800 SDRAM. The peak floating-point computing performance of a single compute node is 7168 GFLOP/s at a base frequency of 2.0 GHz, and the peak memory bandwidth is 614 GB/s. Out of the 60 nodes, 54 nodes are managed by the Red Hat OpenStack Platform (RHOSP)[1] and 6 nodes are managed by VMware vSphere. The vSphere-managed nodes are named interoperability nodes, and are designed to allow users to migrate VMs deployed on mdx I, which also uses vSphere. In this work, we evaluate the RHOSP-managed nodes only.

Two storage systems are provided to compute nodes. One is an all-NVMe Lustre parallel file system (DDN EXAScaler) with a capacity of 553 TB, and the other is an S3-compatible object storage (Cloudian HyperStore) with a capacity of 432 TB.

The Lustre file system can be directly mounted by multiple VMs and used to read and write data from multiple VMs. The Lustre file system is also used to store VM disk images. Specifically, the Lustre file system is exported as a Network File System (NFS) volume, which is then accessed by RHOPS's block storage service to create and manage volumes. In addition, the Lustre file system is also accessible via S3 protocol through the S3 Data Services (S3DS), which is a Lustre-S3 gateway offered by DDN, and a web interface based on Nextcloud[2]. Physically, the Lustre file system is composed of a single DDN ES400NVX2 appliance equipped with 24 NVMe SSDs each with a 30 TB capacity. The appliance hosts two Lustre Metadata Servers (MDS) and four Object Storage Servers (OSSs) as VMs to provide a Lustre file system.

The S3-compatible object storage is deployed as a part of an existing supercomputer (SQUID) (Date et al., 2023) installed at the University of Osaka, and is connected to mdx II via 10 Gbps Ethernet. The object storage is composed of six Cloudian HyperStore HSA-1610 appliances that form a cluster. The HyperStore cluster is configured with a 4+2 erasure coding scheme, where each object is encoded into four data fragments and two parity fragments, which are stored on different nodes.

All compute nodes, servers, and Lustre storage are interconnected with a 200 Gbps Ethernet network. The overlay network is realized using Generic Network Virtualization Encapsulation (GENEVE) (Gross et al., 2020). PCI Passthrough or Single Root I/O Virtualization (SR-IOV) (Lockwood et al., 2014) are not utilized in mdx II because VMs with these technologies cannot be migrated to other private or public clouds, and one future goal of mdx II is to allow seamless migration of VMs between mdx II and public clouds. In terms of external connection, the system is connected to a Japanese research network (Science Information NETwork, SINET) and the internet through the University of Osaka's campus network.

[1] https://access.redhat.com/products/red-hat-openstack-platform

[2] https://nextcloud.com/

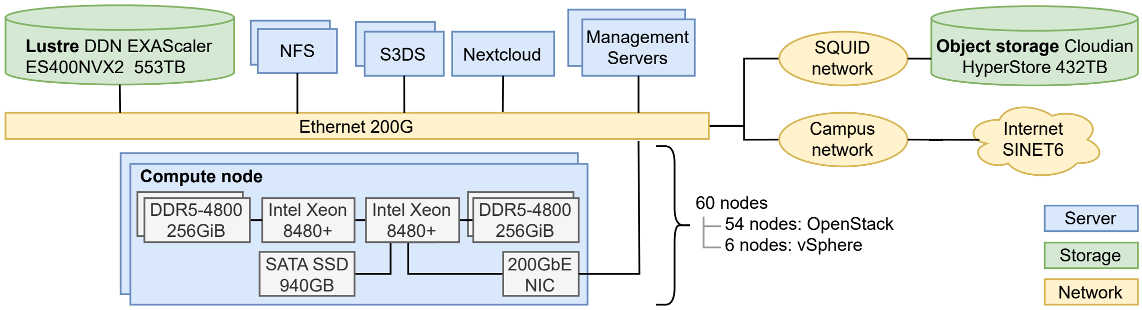


Figure 2: Overall architecture of mdx II.

2.2 Related Work

Jetstream (Stewart et al., 2015) is the first production cloud system within the NSF-funded Extreme Science and Engineering Discovery Environment (XSEDE) (Towns et al., 2014) ecosystem, designed to support interactive computing for researchers who do not fit traditional HPC models. Jetstream is an OpenStack-based cloud system that allows users to provision VMs, and supports authentication and data movement via Globus. Building on these foundations, Jetstream 2 (Hancock et al., 2021) was introduced as an evolution of the original Jetstream system, featuring heterogeneous hardware including GPUs, software-defined storage, and container orchestrations. The primary system of Jetstream 2 is installed at Indiana University[3], and is composed of 384 compute nodes, 32 large memory nodes, and 90 GPU nodes. The system is based on AMD EPYC 7713 (Milan) CPUs and NVIDIA A100 40 GB GPUs.

mdx I (Suzumura et al., 2022) is the predecessor of mdx II, which is installed at the University of Tokyo. mdx I is composed of 368 CPU nodes and 40 GPU nodes, equipped with Intel Intel Xeon Platinum 8368 (IceLake-SP) CPUs and NVIDIA A100 40 GB GPUs. Regarding storage, mdx I provides an SSD-based Lustre file system, an HDD-based Lustre file system, and an S3-compliant object storage. mdx I emphasizes strong network isolation between tenants and high communication performance. In particular, mdx I uses Virtual eXtensible LAN (VXLAN) to isolate tenants, SR-IOV to connect the host NICs to VMs while bypassing the hypervisor, and RDMA over Converged Ethernet (RoCE) to enable low-latency, high-throughput communication between VMs, or VMs and storage.

While mdx I, mdx II and Jetstream share similar goals, an in-depth performance analysis of these systems has not been published to the best of our knowledge. Apart from some performance measurement results obtained as a part of the system acceptance test (Stewart et al., 2016), no comprehensive performance evaluation has been conducted, and no real-world benchmark results or comparisons with public clouds have been published so far. Therefore, this paper evaluates and analyzes the performance of mdx II, a state-of-the-art academic cloud, and offers insights such as performance characteristics and potential bottlenecks to aid future academic cloud design.

[3]https://docs.jetstream-cloud.org/overview/config/

3 PERFORMANCE ANALYSIS

3.1 Evaluation Method

As with public clouds, mdx II allows users to flexibly choose the number of vCPUs for a VM, currently ranging from 1 to 224 vCPUs. However, it is infeasible to evaluate all VM configurations. Thus, in the first half of the evaluation, we focus on a 16-vCPU VM, since the minimum purchasable vCPU quota is currently 16, and it is expected that many users will launch VMs of this size. In the second half of the evaluation, we focus on a 224-vCPU VM, since it occupies a full compute node and thus allows us to directly compare its performance with a bare metal server that has the same hardware configuration.

3.2 16-vCPU VM

We use the mdx II `vc16m32g` instance type, which is equipped with 16 vCPUs and 32 GiB of memory, for the evaluation. As a baseline, we use Amazon Web Services (AWS), a widely known public cloud service. Specifically, we use the `c7i.4xlarge` instance type, which uses CPUs of the same generation (Sapphire Rapids) as mdx II and is equipped with 16 vCPUs and 32 GiB of memory, exactly matching the `vc16m32g` instance type.

Table 1: Compute and memory performance of a 16-vCPU mdx II VM and AWS instance.

	Compute	Memory
mdx II	1344 GFLOPS	164 GB/s
AWS	656 GFLOPS	97 GB/s

3.2.1 Computing Performance

We use the Intel-optimized LINPACK Benchmark included in the Intel oneAPI HPC Toolkit 2025.0.1 to measure the floating-point computing performance and BabelStream[4] 5.0 to measure the memory throughput. BabelStream is compiled with the Intel oneAPI DPC++/C++ Compiler using the compiler flags `-O3 -march=native -qopt-zmm-usage=high -qopt-streaming-stores=always`.

Table 1 compares the LINPACK performance and memory throughput between mdx II and AWS. The LINPACK performance of mdx II reaches 1.34 TFLOPS, and is twice that of AWS. This is likely because mdx II does not use vCPU pinning and thus vCPUs are executed on different physical cores. Contrastingly, AWS pins vCPUs to logical cores to minimize interference with other instances. Thus, two vCPUs share a single physical core on AWS and results in half the performance of mdx II. The mdx II instance achieved 1.7× higher memory throughput, which could also attribute to a weaker resource isolation.

3.2.2 Network Performance

To assess the network performance ofa VM, we use iPerf 3.18[5] and measure the TCP throughput between VMs running on either the same compute node or two different compute nodes. Since a single TCP stream cannot saturate the 200 Gbps physical link bandwidth, we generate multiple parallel TCP streams. We also enable the zero-copy option (`-Z`) in iPerf, which uses the `sendfile()` system call instead of the `write()` system call to reduce the CPU load and improve the throughput.

Figure 3 shows the throughput between two VMs using a varying number of TCP streams. The single-stream throughput between two VMs running on the same node is 31.6 Gbps, while the throughput between VMs running on different nodes is 12.9 Gbps. As we increase the number of parallel TCP streams, both the intra- and inter-node throughput does not improve significantly. This is because by default, the virtio-net/vhost-net (Bugnion et al., 2017) paravirtualized NIC uses only a single queue to communicate between the guest and host kernels, and thus all packet transmissions are serialized. Therefore, the total throughput does not improve with the number of parallel TCP streams.

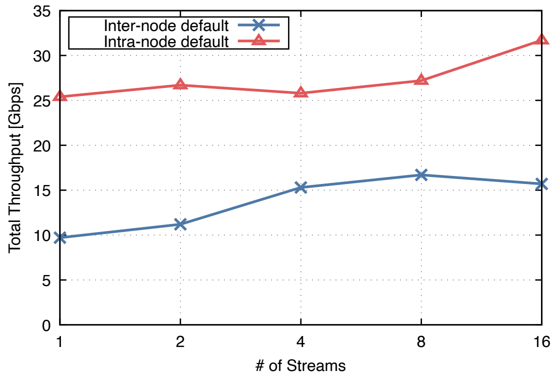


Figure 3: Total TCP throughput between two VMs.

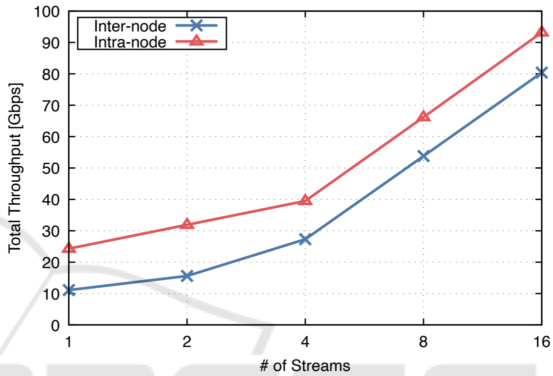


Figure 4: Total TCP throughput between two VMs with virtio multiqueue enabled.

One optimization to address this problem is to set up multiple queues between the guest and host kernels. This feature is called *virtio multiqueue*, and can be enabled by setting the `hw:vif_multiqueue_enabled` flavor extra spec in OpenStack. Figure 4 shows the throughput with virtio multiqueue enabled. Evidently, the throughput increases with the number of TCP streams when multiqueue is enabled. With 16 streams, the inter-node throughput reaches 80.4 Gbps, and the intra-node throughput reaches 93.2 Gbps, demonstrating a significant benefit of virtio multiqueue. Nonetheless, the inter-node throughput is still lower than the 200 Gbps link bandwidth of the host. We therefore investigate whether a higher total throughput can be achieved when multiple VMs simultaneously generate multiple TCP streams.

Figure 5 shows the total TCP throughput between multiple pairs of VMs running either on the same compute node or two different nodes. The number

[4] https://github.com/UoB-HPC/BabelStream
[5] https://github.com/esnet/iperf

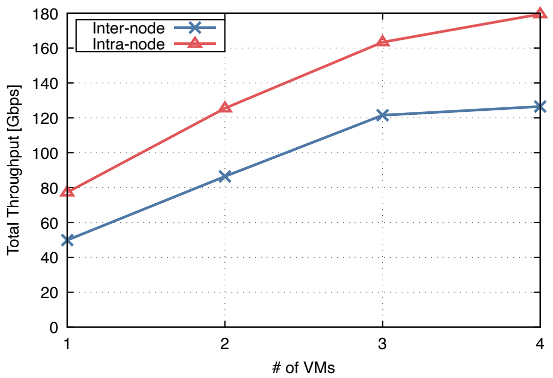


Figure 5: Total TCP throughput between multiple VM pairs (16 TCP streams per VM).

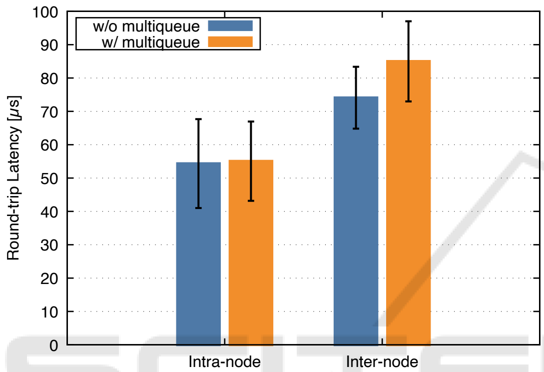


Figure 6: TCP latency between two VMs (error bars indicate standard deviation).

of TCP streams generated by a single VM is 16. The result shows that the total inter-node throughput increases linearly up to three VMs, and saturates at 126 Gbps. We believe the performance gap between the achieved throughput and the 200 Gbps host NIC bandwidth is due to the various overheads of virtual networking.

To investigate whether virtio multiqueue has any impact on the network latency, we use netperf[6] 2.7.0. The benchmark is launched with the `TCP_RR` test type, which repeatedly exchanges a request and a response between the client and server over TCP to measure the round-trip network latency. Figure 6 shows the measured latency. The mean round-trip latency between VMs running on the same node is 55μs both with and without virtio multiqueue, indicating that no measurable overhead is imposed in terms of latency. The mean round-trip latency between two VMs running on different nodes is 74μs without multiqueue and 85μs with multiqueue. Considering that virtio offers a multi-fold improvement in throughput, we believe this 15% increase in latency is a reasonable trade-off.

In summary, mdx II delivers up to 80 Gbps network throughput to VMs, outperforming most public cloud instances. This performance, however, is only achievable when virtio multiqueue is enabled. We are thus recommending the operators of mdx II to enable virtio multiqueue by default.

[6]https://github.com/HewlettPackard/netperf

Table 2: Comparison of sequential I/O performance (1 MB).

	Read	Write
mdx II (Block)	4.21 GB/s	1.75 GB/s
mdx II (Lustre)	9.82 GB/s	7.74 GB/s
AWS (Block)	1.05 GB/s	1.05 GB/s

3.2.3 File System Performance

Since data science workloads are often I/O-bound (Philip Chen and Zhang, 2014), the file system performance of a cloud platform becomes crucial. Here, we compare the throughput and IOPS of mdx II block storage and Lustre storage, and AWS block storage. We use the Flexible I/O tester (fio)[7] 3.38 to measure the file access performance. To saturate the I/O stack, we use the libaio (`--ioengine=libaio`) asynchronous I/O backend and a sufficiently large (256) number of in-flight I/O requests (`--iodepth`). To exclude the effect of the page cache, the `O_DIRECT` flag is set (`--direct=1`) to bypass the page cache. On AWS, we use the General Purpose SSD (gp3) volume type. Since its default I/O performance is limited (only 3 KIOPS and 125 MiB/s throughput), we provision the I/O performance to its maximum (16 KIOPS and 1 GiB/s throughput).

Table 2 summarizes the sequential I/O performance. The block storage of mdx II offers 4.21 GB/s in read throughput and 1.75 GB/s in write throughput. The Lustre storage of mdx II offers 9.82 GB/s write throughput and 7.74 GB/s read throughput, surpassing the performance of the block storage. The OpenStack block storage with the NFS backend works by mounting an NFS volume on the host, and exposing the image file stored on the NFS volume to the guest via a virtio-blk (Russell, 2008; Bugnion et al., 2017) paravirtualized block device. In the case of mdx II, the Lustre-NFS gateway exposes the Lustre volume as an NFS volume. In contrast, Lustre distributes file accesses to multiple Object Storage Service (OSS) servers and thus provides higher aggregate performance. The read throughput of Lustre is close to the network throughput measured by iPerf (10.05 GB/s), suggesting its is bottlenecked by the guest network performance. The AWS instance exactly delivers the provisioned 1 GiB/s performance.

Table 3 summarizes the random I/O perfor-

[7]https://github.com/axboe/fio

Table 3: Comparison of random I/O performance (4 KB).

	Read	Write
mdx II (Block)	61 KIOPS	21 KIOPS
mdx II (Lustre)	416 KIOPS	164 KIOPS
AWS (Block)	16 KIOPS	16 KIOPS

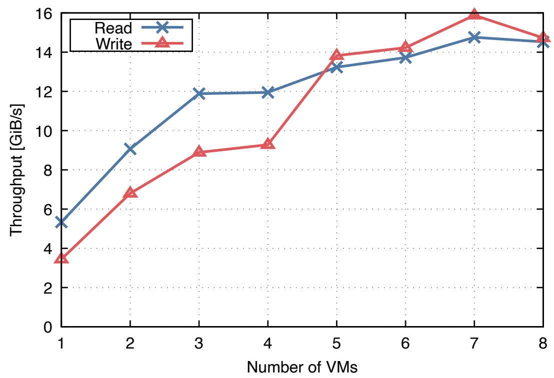


Figure 7: Total Lustre throughput when accessed from multiple VMs.

mance. Again, AWS exactly delivers the provisioned 16 KIOPS. The mdx II block storage achieves 61 KIOPS in read and 21 KIOPS in write access, exceeding the AWS block storage. The mdx II Lustre storage delivers even higher performance of 416 KIOPS in read and 164 KIOPS in write access.

To investigate the performance of the Lustre storage, we run the IOR[8] parallel I/O benchmark on multiple VMs running on different compute nodes, and measure the sequential read and write performance.

Figure 7 shows the total read and write throughput. The read throughput measured by IOR was 5.33 GB/s, and the read throughput was 3.44 GB/s when the number of VMs was one. The reason the throughput is lower than the throughput measured by fio is that IOR does not support asynchronous I/O, and thus only one I/O operation can be in-flight. The total read and write throughput gradually increases with the number of VMs, and saturates at approximately 15 GB/s.

In summary, the block storage of mdx II outperforms that of AWS, especially in read performance. Furthermore, the Lustre storage offers considerably higher throughput and IOPS than the block storage. It should be noted that this evaluation was conducted immediately after the launch of the mdx II service when system utilization was still low. Therefore, the I/O performance might become lower due to contention and interference when the system is highly utilized.

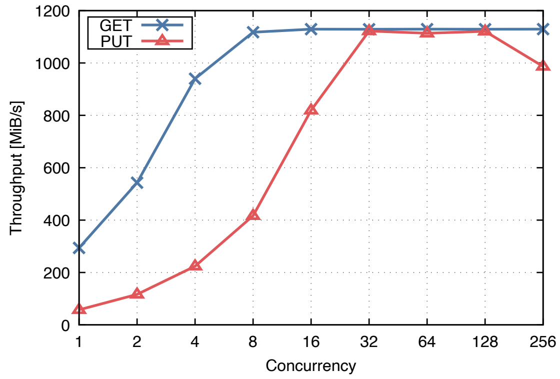


Figure 8: Cloudian HyperStore throughput.

3.2.4 Object Storage Performance

AWS S3-compatible object storages are nowadays widely used as data lakes, and thus many data science libraries and frameworks support directly loading data from S3-compatible object storage. To evaluate the performance of the object storages available in mdx II, we use warp[9] 1.0.8, which is a benchmark tool for S3-compatible storage. We configure warp to either upload or download 2500 objects each of which is 10 MiB in size. We vary the number of concurrent operations and measure the total throughput of GET or PUT operations of the object storage.

Figure 8 shows the throughput of Cloudian HyperStore. The single-client PUT throughput is 293 MiB/s, and the GET throughput is 57 MiB/s. The throughput increases with the number of clients, and saturates at approximately 1120 MiB/s. This is because the HpyerStore object storage exists on an external supercomputer (SQUID), and the link bandwidth between SQUID and mdx II is 10 Gbps.

Figure 9 shows the throughput of DDN S3DS. The single-client PUT throughput is 898 MiB/s and the GET throughput 227 MiB/s, indicating 3–4× higher performance than HyperStore. The GET performance improves considerably with the number of clients and reaches 9444 MiB/s with 128 clients. This throughput is close to the maximum effective network throughput, and indicates the guest network performance is the bottleneck. The PUT throughput saturates at 1435 MiB/s with 16 clients and does not improve further with more clients. The reasons for the observed poor performance and scalability of PUT operations compared to GET operations remain unclear. We plan to further investigate the PUT performance in our future work.

To evaluate the performance of S3DS, we run warp on multiple VMs each running on different hosts and measure the performance of S3DS when accessed

[8]https://github.com/hpc/ior

[9]https://github.com/minio/warp

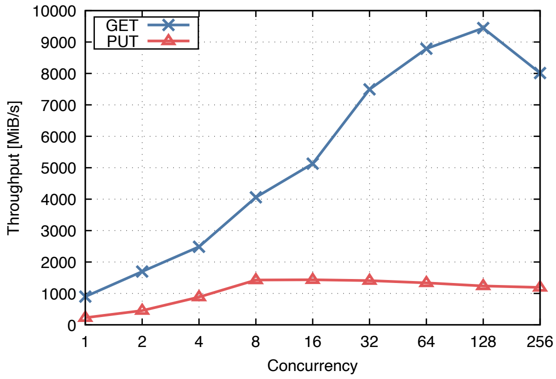


Figure 9: DDN S3DS throughput.

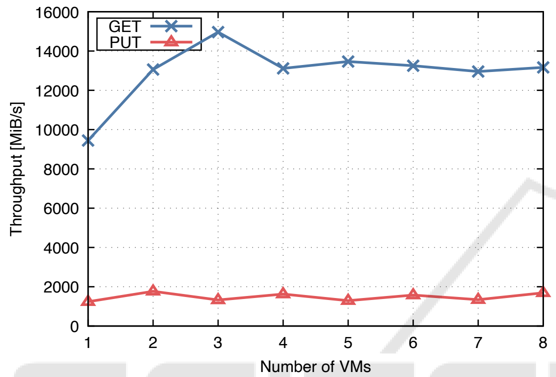


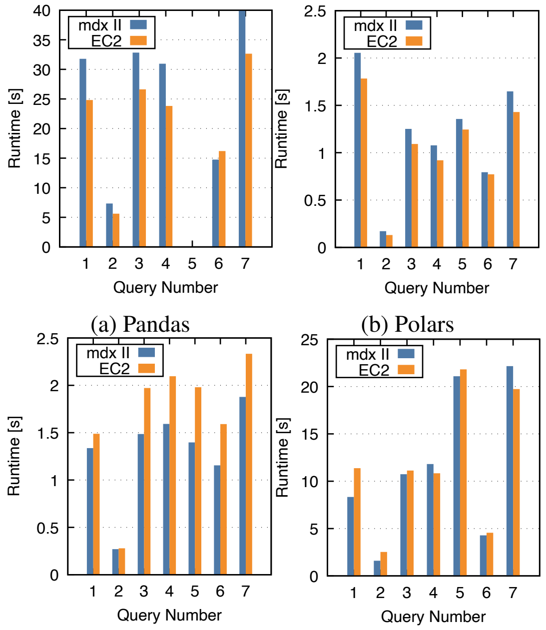
Figure 10: DDN S3DS throughput when accessed from multiple VMs.

from multiple clients in parallel. Figure 10 plots the total throughput with respect to the number of clients (VMs). The peak GET performance is achieved with three clients, and the throughput is 12.75 GiB/s. Since this is lower than the Lustre throughput, the S3DS server is the bottleneck. The PUT performance does not scale with the number of clients, suggesting that the bottleneck is not on the client side.

In summary, Cloudian HyperStore is limited in performance due to the narrow bandwidth between it and the mdx II compute nodes. Thus, it should be avoided if the workload requires high S3 access performance. In such cases, the data should be staged to DDN S3DS.

3.2.5 Data Science Application Performance

Finally, we assess the real-world performance of mdx II in data science workloads. Since Python its ecosystem remains to be the de facto standard in data science (Castro et al., 2023; Raschka et al., 2020), we compare the performance of Python-based data science workloads on mdx II and AWS. Specifically, we use the Polars Decision Support (PDS) benchmarks[10],



(a) Pandas (b) Polars

(c) DuckDB (d) Dask

Figure 11: Polars Decision Support (PDS) benchmarks results.

which is an implementation of the well-known TPC-H[11] benchmark (Dreseler et al., 2020) for measuring online analytical processing performance, in different Python libraries including Pandas, Polars, DuckDB, and Dask. The PDS benchmark allows for running queries on different dataset sizes. Here, we configure the benchmark to run on a dataset of approximately 10 GB in size.

Figure 11 compares the runtime for completing each query included in the benchmark suite on mdx II and AWS. The figure shows that there is generally only a small difference in the execution time of each query between mdx II and AWS, but Pandas performs slightly better on AWS and DuckDB performs better on mdx II. However, it should be noted that these performance differences between mdx II and AWS for each library are much smaller than the performance differences between different libraries. Thus, the choice of the appropriate library for each workload (in this case, Polars or DuckDB) is critical for maximizing the query performance.

3.3 224-vCPU VM

Since virtualization inherently imposes a performance overhead, we aim to quantify this overhead on the mdx II system in this evaluation. We create a VM occupying a full compute node (`vc224m498g` instance type) and compare its performance to a bare metal

[10] https://github.com/pola-rs/polars-benchmark

[11] https://www.tpc.org/tpch/

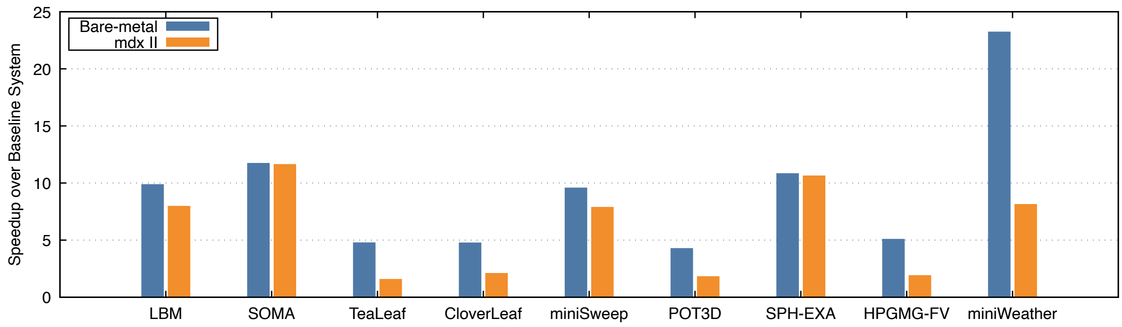


Figure 12: SPEChpc 2021 tiny size results on a bare metal server and mdx II.

Table 4: Compute and memory performance of a 224-vCPU mdx II VM and a bare metal server

	Compute	Memory
mdx II	4965 GFLOPS	383 GB/s
Bare metal	4819 GFLOPS	490 GB/s

server with a similar hardware configuration as an mdx II compute node.

3.3.1 Computing Performance

We first compare the computing performance and memory throughput. Here, the baseline is a supercomputer named *Laurel* installed at Kyoto University. This system is equipped with two Intel Xeon Platinum 8480+ CPUs and 512 GiB of DDR5-4800 SDRAM, matching mdx II. Of course, other components such as storage and interconnect differ from mdx II, but we believe their impact in these node-level benchmarks is negligible. The performance is obtained from a previous work (Fukazawa and Takahashi, 2024).

Table 4 summarizes the computing and memory performance. The computing and memory performance is measured in the same way as the 16 vCPU case described in Section 3.2.1. The result shows that the virtualization overhead imposed to the floating-point computing performance is minimal. However, the overhead imposed on the memory throughput is clear. The mdx II VM delivers 383 GB/s memory throughput, which is 22% lower than the bare metal server. One reason behind this large performance degradation is the lack of vCPU pinning in mdx II. Because vCPUs are not pinned to host cores, vCPUs can freely move between the two CPU sockets. This results in a large cross-socket traffic volume in memory-intensive applications, and degrades the effective memory throughput.

3.3.2 Real-World Application Performance

To evaluate the impact of virtualization overhead in real-world applications, we use the SPEChpc 2021 (Li et al., 2022) benchmark suite. Various organizations have published the SPEChpc scores of their systems on the SPEChpc official website[12]. In particular, Intel has published measurement results on a server equipped with two Intel Xeon Platinum 8480+ CPUs and 512 GiB of DDR5-4800 SDRAM, which matches mdx II.

Figure 12 compares the performance of the SPEChpc *tiny* suite on the two systems. The vertical axis shows represents performance of each benchmark, defined as the speedup over a baseline system (the *Taurus* system at TU Dresden). The plot shows that the virtualization overhead is relatively small for the LBM, SOMA, miniSweep and SPH-EXA. On the other hand, the overhead is large for TeaLeaf, CLoverLeaf, POT3D, HPGMG-FV and miniWeather. TeaLeaf and miniWeather can only achieve one-third of the bare metal performance on mdx II. These benchmarks that experience a large performance degradation on mdx II are generally memory-bound, and thus suffer from the lower memory throughput on mdx II.

4 CONCLUSIONS AND FUTURE WORK

Our performance evaluation of the mdx II cloud platform demonstrated its superiority over AWS in various metrics, including floating-point computing, memory throughput, and storage I/O performance. This positions mdx II as an IaaS platform ideal for HPDA workloads, particularly due to its advanced storage options like Lustre. While the virtualization

[12]https://www.spec.org/hpc2021/results/hpc2021tiny.html

overhead is notable in memory-intensive tasks, its minimal impact on compute-intensive benchmarks indicates mdx II capability to effectively support diverse HPC and HPDA applications.

Future efforts will focus on analyzing and optimizing the performance of various large-scale real-world data science workloads on mdx II, solidifying the role of mdx II in advancing data science research and facilitating cross-disciplinary collaborations. Another direction is to explore the energy efficiency and cost-effectiveness of mdx II in comparison to other academic and public clouds.

ACKNOWLEDGMENTS

This work was partially supported by JST ACT-X Grant Number JPMJAX24M6, as well as JSPS KAKENHI Grant Numbers JP20K19808 and JP23K16890. The mdx II system was used to carry out experiments.

REFERENCES

Bugnion, E., Nieh, J., and Tsafrir, D. (2017). *Hardware and Software Support for Virtualization*, volume 12. Morgan & Claypool.

Castro, O., Bruneau, P., Sottet, J. S., and Torregrossa, D. (2023). Landscape of High-Performance Python to Develop Data Science and Machine Learning Applications. *ACM Computing Surveys*, 56(3).

Date, S., Kido, Y., Katsuura, Y., Teramae, Y., and Kigoshi, S. (2023). Supercomputer for Quest to Unsolved Interdisciplinary Datascience (SQUID) and its Five Challenges. In *Sustained Simulation Performance 2021*, pages 1–19.

Dreseler, M., Boissier, M., Rabl, T., and Uflacker, M. (2020). Quantifying TPC-H choke points and their optimizations. In *Proceedings of the VLDB Endowment*, volume 13, pages 1206–1220.

Fukazawa, K. and Takahashi, R. (2024). Performance Evaluation of the Fourth-Generation Xeon with Different Memory Characteristics. *ACM International Conference Proceeding Series*, (May):55–62.

Gross, J., Ganga, I., and Sridhar, T. (2020). Geneve: Generic Network Virtualization Encapsulation. Technical report.

Hancock, D. Y., Fischer, J., Lowe, J. M., Snapp-Childs, W., Pierce, M., Marru, S., Coulter, J. E., Vaughn, M., Beck, B., Merchant, N., Skidmore, E., and Jacobs, G. (2021). Jetstream2: Accelerating cloud computing via Jetstream. In *Practice and Experience in Advanced Research Computing*, number Ci, pages 1–8, New York, NY, USA. ACM.

Li, J., Bobyr, A., Boehm, S., Brantley, W., Brunst, H., Cavelan, A., Chandrasekaran, S., Cheng, J., Ciorba, F. M., Colgrove, M., Curtis, T., Daley, C., Ferrato, M., De Souza, M. G., Hagerty, N., Henschel, R., Juckeland, G., Kelling, J., Li, K., Lieberman, R., McMahon, K., Melnichenko, E., Neggaz, M. A., Ono, H., Ponder, C., Raddatz, D., Schueller, S., Searles, R., Vasilev, F., Vergara, V. M., Wang, B., Wesarg, B., Wienke, S., and Zavala, M. (2022). SPEChpc 2021 Benchmark Suites for Modern HPC Systems. In *ICPE 2022 - Companion of the 2022 ACM/SPEC International Conference on Performance Engineering*, pages 15–16.

Lockwood, G. K., Tatineni, M., and Wagner, R. (2014). SR-IOV: Performance Benefits for Virtualized Interconnects. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment*, pages 1–7, New York, NY, USA. ACM.

Philip Chen, C. L. and Zhang, C. Y. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Information Sciences*, 275:314–347.

Raschka, S., Patterson, J., and Nolet, C. (2020). Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information (Switzerland)*, 11(4).

Russell, R. (2008). Virtio: Towards a de-facto standard for virtual I/O devices. *Operating Systems Review (ACM)*, 42(5):95–103.

Stewart, C. A., Fischer, J., Merchant, N., Stanzione, D. C., Hancock, D. Y., Cockerill, T., Miller, T., Taylor, J., Vaughn, M., Liming, L., Lowe, J. M., and Skidmore, E. (2016). Jetstream - Performance, early experiences, and early results. *ACM International Conference Proceeding Series*, 17-21-July-2016.

Stewart, C. A., Hancock, D., Stanzioneb, D., Turnerd, G., Cockerill, T. M., Merchant, N., Taylor, J., Vaughn, M., Foster, I., Skidmore, E., Tuecke, S., and Gaffney, N. I. (2015). Jetstream: A self-provisioned, scalable science and engineering cloud environment. *ACM International Conference Proceeding Series*, 2015-July.

Suzumura, T., Sugiki, A., Takizawa, H., Imakura, A., Nakamura, H., Taura, K., Kudoh, T., Hanawa, T., Sekiya, Y., Kobayashi, H., Kuga, Y., Nakamura, R., Jiang, R., Kawase, J., Hanai, M., Miyazaki, H., Ishizaki, T., Shimotoku, D., Miyamoto, D., Aida, K., Takefusa, A., Kurimoto, T., Sasayama, K., Kitagawa, N., Fujiwara, I., Tanimura, Y., Aoki, T., Endo, T., Ohshima, S., Fukazawa, K., Date, S., and Uchibayashi, T. (2022). mdx: A Cloud Platform for Supporting Data Science and Cross-Disciplinary Research Collaborations. In *Proceedings of the 2022 IEEE International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing, International Conference on Cy*, pages 1–7. IEEE.

Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G. D., Roskies, R., Scott, J. R., and Wilkens-Diehr, N. (2014). XSEDE: Accelerating scientific discovery. *Computing in Science and Engineering*, 16(5):62–74.