Exploring the Viability of Composable Architectures to Overcome Memory Limitations in High Performance Computing Workflows

Wesley A. Brashear*
HPRC, Texas A&M University,
College Station, TX
wbrashear@tamu.edu

Varshani P. Reddy
HPRC, Texas A&M University,
College Station, TX
varshanipreddy@tamu.edu

Steven K. Baum
HPRC, Texas A&M University,
College Station, TX
baum@tamu.edu

Dhruva K. Chakravorty
HPRC, Texas A&M University,
College Station, TX
chakravorty@tamu.edu

Francis M. Dang
HPRC, Texas A&M University,
College Station, TX
francis@tamu.edu

Lisa M. Perez
HPRC, Texas A&M University,
College Station, TX
perez@tamu.edu

Honggao Liu
HPRC, Texas A&M University,
College Station, TX
honggao@tamu.edu

ABSTRACT

High Performance Computing (HPC) workflows across disciplines often require large amounts of memory which can result in bottlenecks when system memory is exceeded. Technologies that bridge the latency gap between traditional Hard Disk Drive (HDD) and Solid State Drive (SSD) SATA/SAS storage and volatile DRAM offer a way to extend available memory on HPC systems at a fraction of the cost of traditional DRAM. We developed synthetic benchmarks to test the performance of various configurations that leverage NVMe (non-volatile memory express) SSDs and Lustre storage over a Liqid composable infrastructure. Configurations included mounting the NVMe SSDs as swap connected via PCIe (Peripheral Component Interconnect express) Gen4 fabric over a software-defined composable infrastructure and having the same NVMe SSDs and the Lustre space being managed by MemVerge Memory Machine software. The nodes with NVMe SSD swap and MemVerge-managed NVMe SSDs performed similarly and completed synthetic benchmark runs with little to no increase in runtime compared to nodes configured with traditional DRAM alone. This is surprising given the latency differences between DRAM and NVMe SSDs and the results bode well for the adoption of composable architecture. The approaches described within this paper offer HPC resource providers a cost-effective way to increase memory bandwidth while sacrificing very little performance.

CCS CONCEPTS

• **Hardware** → Communication hardware, interfaces and storage; Emerging technologies.

*High Performance Research Computing

This work is licensed under a Creative Commons Attribution International 4.0 License.

PEARC '24, July 21–25, 2024, Providence, RI, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0419-2/24/07
https://doi.org/10.1145/3626203.3670620

ACM Reference Format:
Wesley A. Brashear, Varshani P. Reddy, Steven K. Baum, Dhruva K. Chakravorty, Francis M. Dang, Lisa M. Perez, and Honggao Liu. 2024. Exploring the Viability of Composable Architectures to Overcome Memory Limitations in High Performance Computing Workflows. In *Practice and Experience in Advanced Research Computing (PEARC '24), July 21–25, 2024, Providence, RI, USA*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3626203.3670620

1 INTRODUCTION

Memory capacity and I/O performance are frequent bottlenecks in many modern High Performance Computing (HPC) applications and workflows across a variety of disciplines [1]. In traditional HPC applications, such as quantum chemistry, bioinformatics, and fluid dynamics, even small problems frequently require large amounts of dedicated memory. This issue is exacerbated by the amount of memory being consumed by the recent explosion in Artificial Intelligence and Machine Learning (AI/ML) workloads. Indeed, the growth in speed at which HPC hardware can conduct floating-point operations has outpaced memory bandwidth gains by approximately 2x every two years over the past 20 years [2]. Overcoming these bottlenecks is a key challenge as we embrace exascale computing and address growing demands for HPC resources [3].

The need for increased memory has been exacerbated by current HPC workflows, but it is not a novel limiting factor in the design of scientific applications and workflows. As such, several methods have been employed at both the software and hardware levels that help mitigate DRAM memory limitations. Some programs have been written to utilize large I/O operations to disk to reduce RAM usage and ensure the program completes successfully [4–6]. Some workflows incorporate libraries or models that allow the use of disaggregated logically addressable memory space [e.g. PGAS models [7]). Increasing swap space (the portion of the storage memory that is dedicated as a substitute for DRAM memory) can also help alleviate memory limitation. However, some of these methods require substantial I/O that is typically orders of magnitude slower than traditional RAM.

Table 1: Node configurations used to benchmark performance of MemVerge-managed composable memory and increased swap space through PCIe connected Intel Optane SSDs and a Lustre parallel distributed file system.

Configuration Name	Total DRAM	Extended Memory	MemVerge-managed DRAM
Standard Node	512 GB	16 GB Swap	NA
Increased Swap	512 GB	1.4 TB Intel Optane SSDs as Swap	NA
Reduced DRAM	256 GB	1.4 TB Intel Optane SSDs as Swap	NA
MemVerge 400 GB DRAM Tier	512 GB	2.7 TB Intel Optane as MemVerge Disk Tier	400 GB
MemVerge 250 GB DRAM Tier	512 GB	1.3 TB Intel Optane as MemVerge Disk Tier	250 GB
MemVerge Over Lustre	512 GB	1.3 TB Lustre space as MemVerge Disk Tier	250 GB

Recent advances in HPC hardware offer promising solutions to these bottlenecks that reduce latency and bridge the gap between traditional DRAM and Hard Disk Drive (HDD)/Solid State Drive (SSD) storage devices [8]. Technologies such as persistent memory (PMEM) in server DIMM slots or NVMe (non-volatile memory express) SSDs can be connected via PCIe (Peripheral Component Interconnect Express) over hardware composable infrastructures which allows faster access to data and reduced runtimes [9]. Given these advances, we sought to test the performance of various node configurations using non-volatile storage to extend memory capacity. The approaches discussed herein can help alleviate the need for large memory nodes and provide alternative approaches for completing scientific workflows that require large amounts of memory.

2 METHODS

2.1 ACES Composable Testbed

ACES (Accelerating Computing for Emerging Sciences) is a National Science Foundation-funded testbed system housed at Texas A&M's High Performance Research Computing [10]. This system has a number of different accelerators (e.g. Intel Max 1100 GPUs, NVIDIA H100 GPUs) that, along with Intel Optane SSDs, can be composed through the Liqid composable infrastructure over PCIe (Peripheral Component Interconnect Express) Gen4 and Gen5 fabrics. There are 48 Intel Optane SSDs, comprising an additional ~18TB of composable memory that can be managed through the MemVerge MemoryMachine software [11] or configured as PCIe-connected swap. In previous works, we have investigated the effectiveness of composing accelerators to match the needs of composable workloads. Here, we extend this approach toward managing memory-requiring workloads [12-14]. We used multiple node configurations to test the efficacy of increased swap space (using composed Intel Optane SSDs as a swap device), MemVerge-managed composable memory (through both composed Intel Optane SSDs and reserved space on the 2.3 PB DDN Lustre parallel distributed file system connected through an NDR InfiniBand Network) in relation to standard node architecture (Table 1). Runs on MemVerge nodes were managed with the MemoryMachine software, which allows the user to configure the amount of DRAM utilized by the program (DRAM Tier Limit), before non-DRAM resources are utilized. All nodes were equipped with 2 48-core Intel Xeon 8468 Sapphire Rapids CPUs and each run used all 96-cores available on each node.

2.2 Synthetic Benchmarks with Dense Matrix Multiplication

To test the efficiency of using Intel Optane SSDs as composed memory, we developed a synthetic benchmark that would utilize large amounts of memory over a relatively short runtime: an R script that conducts parallelized matrix multiplication with double-precision floating point values [15]. This allowed us to test how utilizing composed memory might affect applications/workflows utilizing dense linear algebra, a common component of HPC algorithms [16]. The R script reports memory utilization and runtime to compare against metrics we collected through mvmcli and Memory Viewer (proprietary software from MemVerge), Linux free command, and /proc/meminfo. These benchmarks were first run on standard compute nodes (no composed memory) while running Memory Viewer, an application from MemVerge that can be used to profile memory usage (i.e. hot vs cold memory), to inform parameters for running jobs with MemoryMachine and on nodes with Intel Optane SSDs configured as swap devices. We completed at least three runs for each node configuration/matrix size and reported the average performance for each. Upper DRAM tier limits and hugepage numbers (hugepage size of 2 MB) for MemVerge-managed nodes were set according to recommendations from MemVerge: an upper limit of 400 GB MemVerge-managed memory for 512 GB DRAM nodes and hugepage numbers equal to the GB DRAM tier limit multiplied by 500 (the number of huge pages required for each GB of managed DRAM).

3 RESULTS

We ran the R script for matrix multiplication with two independent matrices (125k x 125k dimensions) with random numbers in a normal distribution with a mean value of 1000 and a standard deviation of 100 to profile memory usage using the proprietary Memory Viewer software from MemVerge. While this program runs as a graphical user interface (GUI) and produces a graph of memory usage, the raw data logged by this process was replotted for clarity (Figure 1). The script uses a max of ~350 GB of DRAM, but for a majority of the run the amount of hot RAM is < 125 GB.

The average runtimes, average maximum DRAM, and average extended memory for each node configuration across multiple matrix dimensions are shown in Table 2. The nodes that utilized Intel Optane SSDs configured as additional swap space outperformed the MemVerge-managed nodes using both the Intel Optane SSDs and the Lustre file system except for the calculations with 150,000 x 150,000 matrices. In these runs, the MemVerge-managed node with

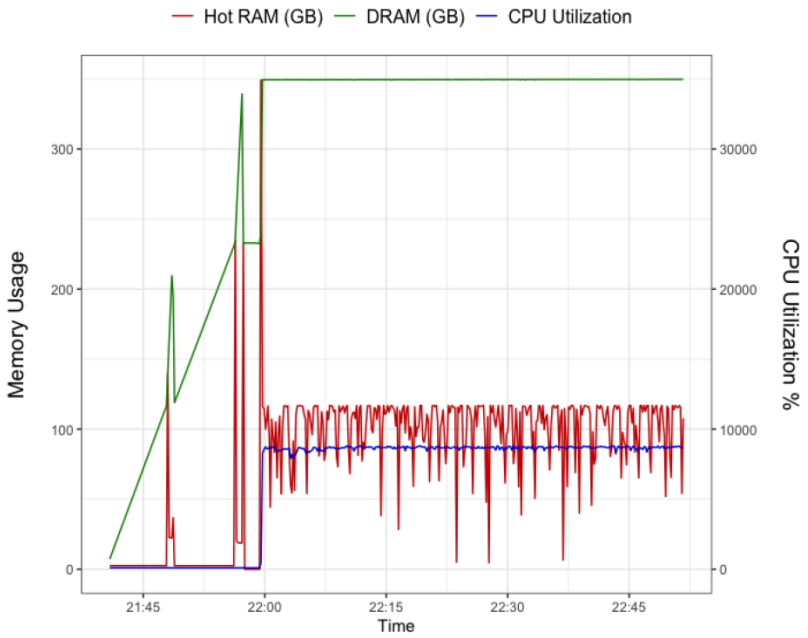

Figure 1: Memory profile of R script conducting matrix multiplication with two matrices of 125000 x 125000 dimension. The stats from this run were generated using the Memory Viewer software from MemVerge and depict the total amount of DRAM used for the run, Hot RAM (memory which is being frequently accessed), and the CPU utilization of the run.

Intel Optane SSDs outperformed the node with reduced DRAM (256 GB) but were still slightly slower than the node with 512 GB of DRAM and Intel Optane SSDs configured as swap.

4 DISCUSSION

The DRAM tier limits used in this study were selected by starting with a small portion of the job (~10%) using extended memory and then slowly increasing the threshold until the minimum DRAM tier needed for desired performance was reached [17]. We found that, using the Memory Viewer software from MemVerge, this tier should ideally be set slightly higher than the amount of hot memory typically utilized across the length of the application's runtime. This limits the amount of time the job spends accessing memory stored in non-volatile storage to help mitigate the drop in performance that comes from using storage with higher latencies. In practice, this also limits the type of programs/applications that might benefit from using this technology. It may therefore be beneficial for HPC resource providers and facilitators to identify the best candidate applications within their areas of expertise that would benefit from utilizing non-volatile memory and instruct users on how best to use these resources. A centralized repository for these identified applications would benefit the HPC community and utilization of existing profiling applications could help in this endeavor [18].

The similarities in performance between the MemVerge-managed extended memory nodes and those where swap was configured as PCIe-connected Intel Optane SSDs exhibits the lack of need for third-party interfaces for managing extended memory for applications using dense linear algebra. The Memory Viewer software allowed us to easily set a file on our Lustre file system to act as extended memory whereas this process is much more difficult without it. Although it did show a dramatic decrease in speed, the job was able to complete with over half the full amount of memory being written to this space. It would be useful for HPC centers to identify specific workflows that will benefit. Otherwise, future development is needed to reduce barriers to entry for general use.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation grants Fostering Accelerated Scientific Transformations, Education, and Research (FASTER, NSF award number 2019129) and Accelerating Computing for Emerging Sciences (ACES, NSF award number 2112356).

REFERENCES

[1] Michèle Weiland, Holger Brunst, Tiago Quintino, Nick Johnson, Olivier Iffrig, Simon Smart, Christian Herold, Antonino Bonanni, Adrian Jackson, Mark Parsons. 2019. An Early Evaluation of Intel's Optane DC Persistent Memory Module and its Impact on High-Performance Scientific Applications. In *SC '19: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 19 pages. https://doi.org/10.1145/3295500.3356159

[2] Amir Gholami, Zhewei Yao, Sehoon Kim, Coleman Hooper, Michael W. Mahoney, Kurt Keutzer. 2024. AI and Memory Wall. In *EEE Micro*, doi: 10.1109/MM.2024.3373763

[3] Stijn Heldens, Pieter Hijma, Bev Van Werkhoven, Jason Maassen, Adam S. Z. Belloum, Rob V. Van Nieuwpoort. 2020. The Landscape of Exascale Research: A Data-Driven Literature Analysis. ACM Computing Surveys. https://dl.acm.org/doi/10.1145/3372390

[4] Petr Danecek, James K Bonfield, Jennifer Liddle, John Marshall, Valeriu Ohan, Martin O Pollard, Andrew Whitwham, Thomas Keane, Shane A McCarthy, Robert

Table 2: Runtime statistics for matrix multiplication in R across multiple node configurations with different matrix sizes. Extended Memory refers to either swap or MemVerge managed memory devices depending on node configurations.

Matrix Sizes			
	100k x 100k		
Node Configuration	Runtime (minutes)	Max DRAM	Extended Memory
Standard Node	19.5	228.6	5.7
Large Swap	18.1	229.0	11.5
Reduced DRAM	17.3	229.0	0.0
MemVerge 250 GB DRAM Tier	23.4	224.0	0.1
MemVerge 400 GB DRAM Tier	23.2	224.0	0.0
MemVerge over Lustre	21.4	224.0	0.1
	125k x 125k		
	Runtime (minutes)	Max DRAM	Extended Memory
Standard Node	32.9	354.3	11.6
Large Swap	31.2	355.5	0.3
Reduced DRAM	33.8	250.0	223.3
MemVerge 250 GB DRAM Tier	37.5	250.0	99.9
MemVerge 400 GB DRAM Tier	37.9	349.9	0.0
MemVerge over Lustre	50.8	250.0	99.9
	150k x 150k		
	Runtime (minutes)	Max DRAM	Extended Memory
Standard Node	NA	NA	NA
Large Swap	57.0	502.0	93.0
Reduced DRAM	68.3	250.0	419.5
MemVerge 250 GB DRAM Tier	64.2	250.0	253.6
MemVerge 400 GB DRAM Tier	61.4	390.6	112.9
MemVerge over Lustre	864.1	250.0	253.6

M Davies, Heng Li. 2021. Twelve years of SAMtools and BCFtools *GigaScience*. https://doi.org/10.1093/gigascience/giab008

[5] Brandi L. Cantarel, Ian Korf, Sofia M.C. Robb, Genis Parra, Eric Ross, Barry Moore, Carson Holt, Alejandro Sanchez Alvarado, Mark Yandell. 2008. MAKER: An easy-to-use annotation pipeline designed for emerging model organism genomes. *Genome Research*. doi: 10.1101/gr.6743907

[6] Hans-Joachim Werner, Peter J. Knowles, Frederick R. Manby, Joshua A. Black, Klaus Doll, Andreas Heßelmann, Daniel Kats, Andreas Köhn, Tatiana Korona David A. Kreplin, Qianli Ma, Thomas F. Miller, III, Alexander Mitrushchenkov, Kirk A. Peterson, Iakov Polyak, Guntram Rauhut, Marat Sibaev. 2020. The Molpro quantum chemistry package. *The Journal of Chemical Physics* https://doi.org/10.1063/5.0005081

[7] Rob F. Van Der Wijngaart, Srinivas Sridharan, Abdullah Kayi, Gabriele Jost, Jeff R. Hammond, Timothy G. Mattson, Jacob E. Nelson. 2015. Using the Parallel Research Kernels to Study PGAS Models. *in the 9th International Conference on Partitioned Global Address Space Programming Models, Washington, DC, USA. pp. 76-81,* doi: 10.1109/PGAS.2015.24.

[8] Sungjoon Koh, Junhyeok Jang, Changrim Lee, Miryeong Kwon, Jie Zhang, Myoungsoo Jung. 2019. Faster than Flash: An In-Depth Study of System Challenges for Emerging Ultra-Low Latency SSDs. *2019 IEEE International Symposium on Workload Characterization (IISWC)*, Orlando, FL, USA, 2019, doi: 10.1109/IISWC47752.2019.9042009

[9] Dina Fakhry, Mohamed Abdelsalam, M. Watheq El-Kharashi, Mona Safar. 2023. A review on computational storage devices and near memory computing for high performance applications. *Memories - Materials, Devices, Circuits and Systems*. https://doi.org/10.1016/j.memori.2023.100051

[10] ACES (Accelerating Computing for Emerging Sciences). Retrieved February 28, 2024, from https://hprc.tamu.edu/aces/

[11] MemVerge Memory Machine. Retrieved April 26, 2024 from https://docs.memverge.com/mvmm/2.3/userguide/index.html

[12] Zhenhua He, Aditi Saluja, Richard E. Lawrence, Dhruva K. Chakravorty, Francis Dang, Lisa M. Perez, and Honggao Liu. 2023. Performance of Distributed Deep Learning Workloads on a Composable Cyberinfrastructure. In Practice and Experience in Advanced Research Computing (PEARC '23), Portland, OR, USA. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3569951.3593601

[13] Abhinand S. Nasari, Lujun Zhai, Zhenhua He, Hieu T. Le, Suxia Cui, Jian Tao, Dhruva K. Chakravorty, and Honggao Liu. 2023. Porting AI/ML Models to Intelligence Processing Units (IPUs). In Practice and Experience in Advanced Research Computing (PEARC '23), Portland, OR, USA. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3569951.3603632

[14] Hieu T. Le, Zhenhua He, Mai Le, Dhruva K. Chakravorty, Akhil Chilumuru, Yan Yao, Jiefu Chen 2024. Performance Benchmarking and Lessons Learned from Porting AI/ML Workloads to Intelligence Processing Units. July 21 - 25, 2024, PEARC Conference Series, Providence, Rhode Island, USA, 12 pages. (Accepted)

[15] Composable Memory https://github.com/wabrashear/ComposableMemory

[16] Krste Asanovic, Ras Bodik, Bryan C. Catanzaro, Joseph J. Gebis, Parry Husbands, Kurt Keutzer, David A. Patterson, William L. Plishker, John Shalf, Samuel W. Williams, Katherine A. Yelick. 2006. The Landscape of Parallel Computing Research: A View from Berkeley. Technical Report No. UCB/EECS-2006-183 http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html

[17] Setting Volatile Tier Memory. Retrieved April 26, 2024 from https://docs.memverge.com/mvmm/2.3/userguide/oxy_ex-3/topics/t_mvmm_setting_dram_cache.html

[18] Jacob Wahlgren, Maya Gokhale, Ivy B. Peng. 2022. Evaluating Emerging CXL-enabled Memory Pooling for HPC Systems. in IEEE/ACM Workshop on Memory Centric High Performance Computing (MCHPC), Dallas, TX, USA. doi: 10.1109/MCHPC56545.2022.00007