Defining Performance of Scientific Application Workloads on the AMD Milan Platform

Tsai-Wei Wu
tsaiwei@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Stephen Lien Harrell
sharrell@tacc.utexas.edu
Texas Advanced Computing Center
The University of Texas
Austin, Texas, USA

Geoffrey Lentner
glentner@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Alex Younts
ay@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Sam Weekly
sweekly@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Zoey Mertes
zmertes@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Amiya Maji
amaji@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Preston Smith
psmith@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

Xiao Zhu
zhu472@purdue.edu
Research Computing
Purdue University
West Lafayette, Indiana, USA

ABSTRACT

Understanding the capabilities of new architectures is key to informing system purchases and good long-term ROI (Return of Investment) for cluster installations. The newest AMD architecture, Milan, has become available first on Microsoft Azure and we use this early access to measure the performance of this 3rd Generation AMD EPYC processor. In this paper single node performance is gathered for seven popular scientific applications and benchmark test-suites. Quantitative comparisons are carried out between two independent platforms, Milan and its architectural predecessor Rome, for performance evaluations. Our results have shown that Milan architecture have improved performance and met our projections.

CCS CONCEPTS

• **Computer systems organization** → Multicore architectures; • **Computing methodologies** → Massively parallel and high-performance simulations; • **General and reference** → **Performance**.

KEYWORDS

performance, benchmark, architecture, scientific application, HPC

ACM Reference Format:
Tsai-Wei Wu, Stephen Lien Harrell, Geoffrey Lentner, Alex Younts, Sam Weekly, Zoey Mertes, Amiya Maji, Preston Smith, and Xiao Zhu. 2021. Defining Performance of Scientific Application Workloads on the AMD Milan Platform. In *Practice and Experience in Advanced Research Computing (PEARC '21), July 18–22, 2021, Boston, MA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3437359.3465596

PEARC '21, July 18–22, 2021, Boston, MA, USA
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8292-2/21/07.
https://doi.org/10.1145/3437359.3465596

1 INTRODUCTION

The performance of HPC architectures is a key indicator for many academic research computing organizations. For instance, when purchasing systems in heterogeneous environments [5], and when calculating ROI and TCO (Total Cost of Ownership). Additionally, measured architecture performance is crucial to set expectations of the performance of a machine and in-turn provide a baseline to understand and identify soft-failures within an HPC cluster. Lastly, historical performance data can inform interactions with vendors and deepen understanding of vendor road maps.

At Purdue University, the community cluster program has supported scientists from every corner of our campus since 2004, building upon decades of experience at Purdue in scientific computing [10]. This program makes more computing power available for Purdue researchers than faculty and campus units could individually afford, and provides this crucial service at the lowest cost to the institution. The Bell cluster, featuring 2nd Generation AMD EPYC "Rome" processors, is the most recent example of success.

With this experience in mind, Purdue proposed and was awarded to build and operate the "Anvil" system [19] by the National Science Foundation (NSF) within the NSF's XSEDE program [20]. Equipped with the latest 3rd Generation AMD EPYC processors, the CPU-based partition of Anvil will provide 1,000 nodes of HPC capacity. Anvil was designed and proposed before any Milan EPYC processors existed, and the system's anticipated performance was based on projections using 2nd Generation AMD EPYC processors. The Anvil design team projected a 1.2x speedup when moving to the 3rd Generation EPYC. The benchmark and comparison below is made possible with the recent availability of HBv3 instances on Microsoft

Azure, based on the same family of 64-core AMD processors that will underpin Anvil.

In this work, we study the performance characteristics of seven key applications from the different scientific domains (e.g. computational fluid dynamics, molecular dynamics simulation and weather forecasting) which we expect to support on the Anvil system. We compare these single-node application benchmarks with the projected speedup and call out some specific performance characteristics of the platform.

2 ARCHITECTURE BACKGROUND

AMD's 3rd Generation EPYC architecture, code named "Milan," is the server product line from AMD based on their "Zen3" processor core. A "Zen3"-based processor is composed of one or more core complexes. These complexes have 8 Zen 3 cores. The highest performance 3rd Generation processor has 8 core complex and can come into dual processor servers for a total of 128 cores [8]. The 1st generation EPYC processor introduced the concept of multiple silicon dies per package to AMD's processors which was an evolution following trends in the silicon industry. This original design connected its core complexes together using point to point links. Core counts were doubled from competitors for more total throughput but connecting the memory directly to core complexes produced a platform with a challenging memory layout [3]. The 2nd Generation EPYC Processors kept the same fundamentals but improved internal processor core efficiencies and solved some memory performance problems by isolating all external I/O to one task specific die on the processor [13]. 3rd Generation AMD EPYC processors bring most performance by optimizing each core for more instructions per second and shakes up the memory layout to decrease complexity and hopefully latency [14]. These evolutionary changes inside the processor socket and continued into system integration should provide more throughput and more performance [8].

Integrating the 2nd and 3rd Generation AMD EPYC Processors into a complete compute cluster for "Bell" and "Anvil" have brought evolutionary changes as well. For comparison, Intel's 1st and 2nd generation Scalable processor consumed a lot less power per socket, approaching 115 watts per processor socket [9]. The processor used in "Anvil" will consume 280 watts of power per socket [4] which is increased from "Bell"'s 225 watts. The increase in efficiencies and performance have come at an increase in power. When considering these processors it became clear that we needed to evolve our data center to keep up with hotter processors. This meant bringing liquid water direct to the processor with CoolIT's direct to chip cooling solution in addition to the back of rack water cooled radiator doors. By taking water directly into "Bell" nodes we brought density to 40 nodes per server rack. The "Anvil" system continued the same tradition by increasing power density with 480v power directly to racks and using active cooling radiators with fans units. These changes allowed us to put 60 nodes per rack and support the increased power requirements to keep the processors cool. This evolution has allowed us to keep pace and provide a solution we hope provides comparable performance to systems run by much larger organizations with modern data centers.

3 EXPERIMENTAL SETUP

3.1 Hardware

All experiments were carried out on two separate platforms, the Bell cluster and the HBv3 instance at Microsoft Azure, only available for early-user testing at the time of writing.

3.1.1 The Bell Cluster at Purdue [2] was built through a partnership with Dell and AMD in late 2020. Bell consists of Dell compute nodes with two 64-core AMD EPYC 7662 "Rome" processors (128 cores per node) and 256 GB of DDR4 memory. All nodes have 100 Gbps HDR Infiniband interconnect. Simultaneous Multithreading (SMT) is disabled on Bell. We partition the CPU into *four NUMA domains per socket* (NPS=4). This NUMA configuration has been suggested by vendors to provides the a good combination of memory bandwidth and memory latency for a HPC/HTC workload. Considering that most applications will fully subscribe all the cores on each node, the InfiniBand device on Bell is selected as the *Preferred IO Device* to achieve full message rate when all CPU cores are active. This is crucial for HPC applications and especially for multi-node scalability.

3.1.2 The Microsoft Azure HBv3 instance [12] features two 64-core AMD EPYC 7V13 "Milan" CPUs for a total of 128 physical "Zen3" cores. SMT is disabled on HBv3. These 128 cores are divided into 16 sections (8 per socket), each section containing 8 processor cores with uniform access to a 32 MB L3 cache. The HBv3 VM reserves 8 hypervisor host cores symmetrically across both CPU sockets, taking the first 2 cores from specific Core Complex Dies (CCDs) on each NUMA domain. As a result, the remaining 120 cores are available for use. When the NPS is two, the server boots with 4 NUMA domains (2 per socket) each 32-cores in size. Each NUMA has direct access to 4 channels of physical DRAM operating at 3200 MT/s.

3.2 Software

We selected a suite of scientific applications for benchmark closely matching the most used applications on both Purdue's community clusters and XSEDE platforms. Representative applications in the field of molecular dynamics, computation fluid dynamics and weather forecasting are selected to provide broader performance assessment. A well-known benchmark, HPCG [7], was also used.

CentOS 7.8 and CentOS 8.1 are used on Bell and HBv3 VMs, respectively. To ensure fast and reliable application build, we set up the latest Spack [6] version 0.16.1 on the Azure instance to make the installations whenever possible. On both systems, we used the GCC compiler version 9.3 and OpenMPI version 3.1.4. All test code and datasets used by the experiments in this paper are available in Appendix A.

3.2.1 DB12 - DIRAC Benchmark 2012 is a Python-based self-contained benchmark that provides an estimate of CPU processing capabilities for High Energy Physics codes. Considered as a short version of HS06 [11], it iteratively samples a sequence of pseudo-random numbers from a Gaussian distribution. The number of iterations is fixed at 12.5x106, which corresponds to 250 HS06 seconds. We used version 0.1 of DB12 from the GitHub repository.

3.2.2 HPCG - High Performance Conjugate Gradient is a benchmark designed to test a handful of common HPC algorithms. It is intended as a complement to HPL [15], however, it gives a more complete picture of the capabilities of any specific machine.

3.2.3 GROMACS [1] – GROningen MAchine for Chemical Simulations is a molecular dynamics package designed for biomolecuar simulations. In our benchmark runs, we used version 2019.2 and performed an 500ps MD simulation of a molecular system with 206,220 atoms.

3.2.4 LAMMPS [17] – Large-scale Atomic/Molecular Massively Parallel Simulator is a molecular dynamics program from Sandia National Laboratories. Version 21Jul20 was used in our benchmark runs. We performed 2000 MD steps of a system build from the Rhodopsion benchmark by replicating four times along each direction. The final system contains 2,048,000 atoms.

3.2.5 NAMD [16] – Nanoscale Molecular Dynamics is a parallel molecular dynamics code designed for high-performance simulation of large biomolecular systems. We used version 2.13. The F1ATPase dataset in the benchmark has 327,506 atoms and employs periodic boundary condition and Particle Mesh Ewald (PME) for treating electrostatic interactions.

3.2.6 WRF [18] - The Weather Research and Forecasting Model is a mesoscale numerical weather prediction system designed for both atmospheric research and operational forecasting applications. The WRF data set is a real domain over the continental US (CONUS) February 19th, 2019 with a timestep of 18 seconds, with a 3-kilometer grid spacing with 50 vertical levels and approximately 325,000 tiles. The benchmark period is 12 hours. We used WRF V3.9.1 as well as WPS V3.9.1 from UCAR for pre-processing.

3.2.7 OpenFOAM [21] - Open source Field Operation And Manipulation is a free, open source C++ CFD toolbox with over sixty customized solvers and pre-/post- processing utilities that can perform simulations on basic CFD, combustion, turbulence modeling, electromagnetics, heat transfer, multiphase flow and stress analysis. In our testing, we used OpenFOAM 6 to run a modified case based on depthCharge3D example that comes with the source code.

4 RESULTS AND ANALYSIS

Here we present the single node performance for the two platforms of Bell and Azure. We summarized our test results in Table 1 below to show the performance enhancement for different benchmarks and applications. To account for the different number of available computing cores, we adjusted the results obtained on Bell when all the cores on a node was fully subscribed by a scaling factor 0.9375 (120/128). In addition, the two computing systems we tested were not set up in the same way, including the NUMA nodes per socket parameters. Such differences could make the quantitative comparisons in the study less straightforward, but would still provide insight into how to properly build and optimize the application environment on the latest AMD CPUs.

Most applications have a 10%-30% performance improvement which is consistent with the vendor's expectation and our projection in the designing phase of Anvil. In addition, we have observed that process pinning to core in general help the performance and properly mapping by NUMA or L3cache is better than the default binding, resulting a small but noticeable performance enhancement on both test systems.

Table 1: Single node performance data of Bell-Rome and Azure-Milan system for different benchmarks and applications.

Application	Performance on Rome	Performance on Milan	Performance Improvement	unit
GROMACS	49.3	67.2	1.36	ns/day
LAMMPS	0.81	0.88	1.09	ns/day
NAMD	4.13	4.80	1.16	ns/day
OpenFOAM	4754	4048	1.17	seconds
WRF	1852	1469	1.26	seconds
DB12	1800.3	2509.1	1.39	events/s
HPCG	38.8	42.4	1.09	GFLOPs

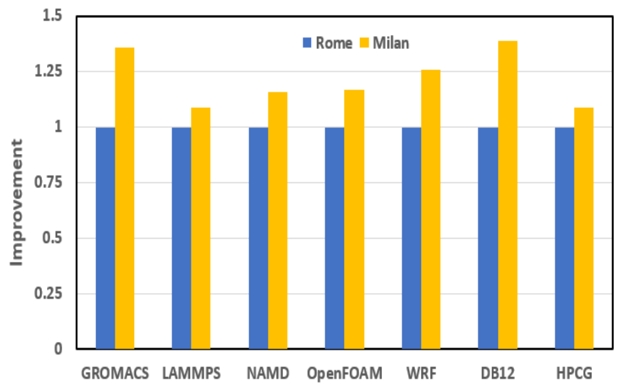

Figure 1: Single node performance enhancement between the Bell-Rome and Azure-Milan systems for each application and benchmark suite. The results obtained from Bell and Rome are referenced to be 1 for easy comparisons.

We also explored the performance impact of cache and memory bandwidth on the AMD CPUs. WRF is known to be very sensitive to memory bandwidth, so different process and tiling schemes were tested, using both MPI version as well as the hybrid parallel configuration option with OpenMP (see Table 2). The portion of the computational domain to a process could be too large to fit into the CPU cache. It is possible to decompose the domain into tiles. Significant speedup is observed when the number of tiles is increased from 1 to 8 on both test systems. We also evaluate the effect of hybrid MPI+OpenMP parallelization. Similar performance could be achieved by a judicious combination of processes and tiles as well as the careful placement of MPI processes onto CPU cores. For instance, we are able to get better performance results using 1 or 2 threads than 4 threads on HBv3 VM. It is also noted that the MPI communication overhead could be reduced by using hybrid MPI+OpenMP parallelization depending on the specific use case.

Using Bell and the Azure instances up to 64 cores provided us expected results but performance decreased when scaling up. We continue to work to understand the problem.

Table 2: Single node performance of WRF in seconds with different parallel process and tiling schemes on both Bell and Azure HBv3.

parallelization	1 tile	2 tiles	4 tiles	8 tiles	platform
120 tasks	1461	1348	1278	1250	Azure
128 tasks	1736	1616	1540	1520	Bell
60 tasks/2 threads		1386	1299	1243	Azure
30 tasks/4 threads			1437	1347	Azure

5 CONCLUSION

In this paper we have tested single node performance of Rome processors and Milan processors which, at the time of writing this paper, were only available on the Microsoft Azure platform. We compare these numbers with the benchmarking extrapolation from the Anvil NSF proposal.

At the Anvil cluster designing stage, the proposal team estimated approximately 20% performance enhancement moving top SKUs of Rome to the Milan platform. As shown in this paper we are able to obtain the expected performance gain on Milan vs the Rome architecture at a single node level.

We will continue to explore interesting issues on the Azure instances, such as the HPL benchmark beyond a single socket as well as carry out multi-node studies including strong scaling of the applications. We expect that the early experiences on the Milan platform will be crucial for us to successfully deliver the Anvil cluster to the national science and engineering community.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. OAC-2005632, and the Purdue University/Azure HPC & AI Collaboration Center funded by Microsoft in partnership with AMD .

REFERENCES

[1] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C Smith, Berk Hess, and Erik Lindahl. 2015. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* 1 (2015), 19–25.
[2] Purdue Research Computing. 2020. *Bell Cluster.* https://www.rcac.purdue.edu/compute/bell/
[3] Advanced Micro Devices. 2018. *NUMA Topology for AMD EPYC Naples Family Processors.* https://developer.amd.com/wp-content/resources/56308-NUMA%20Topology%20for%20AMD%20EPYC%E2%84%A2%20Naples%20Family%20Processors.PDF
[4] Advanced Micro Devices. 2021. *AMD EPYC™ 7H12.* https://www.amd.com/en/products/cpu/amd-epyc-7h12
[5] Richard Todd Evans, Amit Ruhela, Zhao Zhang, Matthew Cawood, Ian Wang, and Si Liu. 2021. Optimizing GPU-enhanced HPC System and Cloud Procurements for Scientific Workloads. In *ISC High Performance 21.*
[6] Todd Gamblin, Matthew LeGendre, Michael R Collette, Gregory L Lee, Adam Moody, Bronis R de Supinski, and Scott Futral. 2015. The Spack package manager: bringing order to HPC software chaos. In *SC'15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis.* IEEE, 1–12.
[7] M. Heroux, J. Dongarra, and P. Luszczek. 2013. *HPCG Technical Specification.* Technical Report SAND2013-8752. Sandia National Laboratory.
[8] Joel Hruska. 2021. *AMD's Milan Brings Zen 3 to Epyc, With Mostly Positive Results - ExtremeTech.* https://www.extremetech.com/computing/320851-amds-milan-brings-zen-3-to-epyc-with-mostly-positive-results
[9] Intel Corporation. 2019. *Intel Xeon Platinum 8280 Processor (38.5M Cache, 2.70 GHz) Product Specifications.* https://ark.intel.com/content/www/us/en/ark/products/192478/intel-xeon-platinum-8280-processor-38-5m-cache-2-70-ghz.html
[10] Gerry McCartney, Thomas Hacker, and Baijin Yang. 2014. Empowering Faculty: A Campus Cyberinfrastructure Strategy for Research Communities. *Educause Review* (2014).
[11] Michele Michelotto, Manfred Alef, Alejandro Iribarren, Helge Meinhard, Peter Wegner, Martin Bly, Gabriele Benelli, Franco Brasolin, Hubert Degaudenzi, Alessandro De Salvo, et al. 2010. A comparison of HEP code with SPEC1 benchmarks on multi-core worker nodes. In *Journal of Physics: Conference Series*, Vol. 219. IOP Publishing, 052009.
[12] Microsoft Azure. 2021. HBv3-series virtual machine overview. https://docs.microsoft.com/en-us/azure/virtual-machines/workloads/hpc/hbv3-series-overview/
[13] Timothy Prickett Morgan. 2019. *A Deep Drive Into AMD's Rome EPYC Architecture.* https://www.nextplatform.com/2019/08/15/a-deep-dive-into-amds-rome-epyc-architecture/
[14] Timothy Prickett Morgan. 2021. *Deep Dive Into AMD's "Milan" Epyc 7003 Architecture.* https://www.nextplatform.com/2021/03/26/deep-dive-into-amds-milan-epyc-7003-architecture/
[15] A Petitet, RC Whaley, J Dongara, and A Cleary. 2014. HPL-A Portable Implementation of the High-Performance Linkpack Benchmark for Distributed-Memory Computers, Sept. 10, 2008.
[16] James C Phillips, David J Hardy, Julio DC Maia, John E Stone, João V Ribeiro, Rafael C Bernardi, Ronak Buch, Giacomo Fiorin, Jérôme Hénin, Wei Jiang, et al. 2020. Scalable molecular dynamics on CPU and GPU architectures with NAMD. *The Journal of chemical physics* 153, 4 (2020), 044130.
[17] Steve Plimpton. 1995. Fast parallel algorithms for short-range molecular dynamics. *Journal of computational physics* 117, 1 (1995), 1–19.
[18] WC Skamarock. 2008. *A description of the advanced research WRF version 3.* Technical Report NCAR/TN-475+STR. National Center for Atmospheric Research. https://doi.org/10.5065/D68S4MVH
[19] Carol Song, Preston Smith, Xiao Zhu, and Rajesh Kalyanam. 2020. *NSF Award Search: Award #2005632 - Category I: Anvil - A National Composable Advanced Computational Resource for the Future of Science and Engineering.* https://www.nsf.gov/awardsearch/showAward?AWD_ID=2005632
[20] John Towns, Maytal Dahan, Ian Foster, Kelly Gaither, Andrew Grimshaw, Victor Hazlewood, Scott Lathrop, Dave Lifka, Gregory D Peterson, Ralph Roskies, et al. 2014. XSEDE: accelerating scientific discovery. *Computing in Science & Engineering* 16, 5 (2014), 62–74.
[21] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. 1998. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics* 12, 6 (1998), 620–631. https://doi.org/10.1063/1.168744

A REPOSITORIES

A.1 Application Code

A.1.1 Spack. - https://github.com/spack/spack

A.1.2 DB12 - DIRAC Benchmark 2012. - https://github.com/DIRACGrid/DB12

A.1.3 HPCG. - https://github.com/hpcg-benchmark/hpcg

A.1.4 GROMACS. - https://github.com/gromacs/gromacs

A.1.5 LAMMPS. - https://github.com/lammps/lammps/

A.1.6 NAMD. - https://www.ks.uiuc.edu/Research/namd/

A.1.7 WRF. - https://github.com/NCAR/WRFV3/

A.1.8 OpenFOAM. - https://github.com/OpenFOAM/OpenFOAM-6/

A.2 Application Test Code

A.2.1 Benchmark Repo. - https://github.itap.purdue.edu/wu979/azure_milan_benchmark.git