LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-TR-847128

Working with Multiple MPIs: Overcoming ABI Incompatibility

E. A. Leon, M. Joos, N. Hanford, C. Fontenaille, A. Cotte

April 4, 2023

Disclaimer

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Working with Multiple MPIs

Overcoming ABI Incompatibility

Edgar A. León[⋆], Marc Joos[†], Nathan Hanford[⋆], Clement Fontenaille[◇], and Adrien Cotte[◇]
[⋆]Lawrence Livermore National Laboratory
[†]French Alternative Energies and Atomic Energy Commission
[◇]AS+ Groupe Eolen

Abstract

Scientific applications rely on third-party libraries, which may have multiple implementations. While sharing an application programming interface (API), many of these implementations do not have a shared application binary interface (ABI) and require recompiling. Recompiling can be a long and complex process and sometimes not even an option when the application is shipped binary only. ABI incompatibility strikes at the heart of portability, productivity, and performance by (1) impeding application execution across different HPC and Cloud systems; (2) adding developer hours rebuilding an application; and (3) not taking advantage of host-optimized libraries.

This tutorial teaches attendees a portable way to address ABI incompatibility in MPI using the Wi4MPI library. Wi4MPI translates the ABI dynamically from the MPI library used to build the application to a different MPI library available at run time. With Wi4MPI, HPC practitioners can break the portability barrier imposed by ABI incompatibility, potentially increase performance, and increase user productivity. The tutorial is broken down in three components: (1) Understanding ABI compatibility in MPI; (2) Translating MPI libraries dynamically; and (3) Applying dynamic translation to key use cases in HPC, including Containers.

If you use more than one MPI library or supercomputer, this tutorial is for you.

Contents

1 Description

1.1 Overview and goals

ABI incompatibility between MPI implementations has long been a barrier to code portability in HPC. The widely-accepted solution when wanting to use a different MPI, is to simply rebuild code with that MPI implementation. However, this can be prohibitive in a variety of scenarios. For example: (1) Suppose a scientific application, consisting of millions of lines of code and integrating tens of libraries, encountered a bug in an MPI library and needs to run with a different MPI implementation; and (2) Suppose a container is deployed on a system with an application built with one MPI, but the target system has an MPI that is not ABI-compatible with the container MPI. This tutorial covers scenarios such as these and shows participants how they can use MPI dynamic translation to increase productivity without fully rebuilding an application.

The goal of the tutorial is to teach attendees a portable way to address ABI incompatibility in MPI using the Wi4MPI library. Wi4MPI translates the ABI dynamically from the MPI library used to build an application to a different MPI library available at runtime. With this *hot, plug and play* capability for MPI, users can move their applications freely from a given MPI library to another (without recompilation) within a system or across systems.

When coupled with containerization and other performance portability layers, Wi4MPI can help shield application developers from proprietary network lock-in by allowing containers to bind-mount the host system MPI, and translate the container MPI transparently. This allows developers the ability to leverage performance advantages of underlying fabrics and accelerated transports without recompilation and the ensuing technical debt–With Wi4MPI, an MPI application can be built at one HPC center and run at another, or even a Cloud provider, with no recompilation and minimal performance overhead, even if the underlying network fabrics are completely different, e.g., from different vendors.

A successful tutorial will provide attendees with a clear understanding of ABI compatibility in MPI and the tools to address it portably and without recompilation. In particular, attendees will learn:

1. How to approach ABI incompatibility in MPI using dynamic translation.
2. How to change MPI libraries at run time and without recompilation using Wi4MPI.
3. How to apply Wi4MPI to key areas in HPC, including Containers.

1.2 Target audience

The tutorial is geared toward MPI practitioners including application writers, developers, and system administrators. Application writers and developers will learn how to use MPI dynamic translation to improve productivity, portability, and testing without recompilation on HPC clusters. The range of applications provided in this tutorial would cover most of these users' common use cases. System administrators will learn how to successfuly deploy the Wi4MPI library on HPC clusters and expose the library to users using popular package managers such as Spack.

1.3 Outline

Begin	End	Activity
9:00	9:45	Module 1: *A general approach to ABI translation*
9:45	11:00	Module 2: *Changing MPIs dynamically with Wi4MPI*
11:00	11:30	Morning break
11:30	13:00	Module 3: *Applying Wi4MPI to key use cases in HPC*

This half-day tutorial consists of the following three modules. We would prefer the morning slot.

1. A general approach to ABI translation.

   Attendees will learn about the problem of ABI incompatibility in MPI, the benefits of using multiple MPI libraries without recompilation, how Wi4MPI addresses this problem, and Wi4MPI's design architecture. At the conclusion of this module, attendees will be familiar with key concepts in MPI ABI compatibility and the Wi4MPI library.

(a) Working with multiple MPI libraries: Challenging but necessary

(b) ABI incompatibility in MPI

(c) Wi4MPI: How it works

(d) Use cases: The power of dynamic translation

(e) Dynamic translation overhead

2. Changing MPI libraries dynamically with Wi4MPI.

   Attendees will learn how to change MPI libraries at run time using Wi4MPI. We will demonstrate Wi4MPI's two modes of operation, Interface and Preload, on simple benchmarks and mini-applications. Interface and Preload modes give users the option of either compiling the application once with Wi4MPI and then running with any other MPI library, or using existing application binaries to switch between MPI libraries at run time. Time-permitting, we will also provide guidance on installing, using, and maintaining Wi4MPI in production. At the conclusion of this module, attendees will know how to use Wi4MPI and adapt it to meet their needs.

   (a) Changing MPI libraries dynamically and without recompilation

   (b) Preload mode: Using Wi4MPI on existing binaries

   (c) Interface mode: A stub library to rule them all

   (d) Demonstration with benchmarks and mini-applications

   (e) Wi4MPI in production

3. Applying Wi4MPI to key use cases in HPC.

   Attendees will learn how to apply Wi4MPI to two out of three important use cases in HPC (depending on audience interest): Choosing the best MPI for one's application at run time; untangling specific MPI library dependencies of distributed Python codes; and bridging ABI incompatibility between container-MPIs and optimized host-MPIs portably to efficiently use containers at scale. At the conclusion of this module, attendees will leverage Wi4MPI to address non-trivial and important use cases in HPC.

   (a) Choosing the best MPI for my application

   (b) Distributed Python: Untangling specific MPI library dependencies

   (c) Containers: Bridging container MPI and host MPI portably to unleash containers performance at scale

1.4 Hands-on activities

We will use the Wi4MPI library to demonstrate a solution to ABI incompatibility in MPI. Wi4MPI is a robust library and conducive to a successful hands-on tutorial: (1) Wi4MPI is built and tested on laptops and workstations as well as production supercomputers at the French Alternative Energies and Atomic Energy Commission (CEA); (2) the testing environment includes a variety of compilers, operating systems, environments such as modules, and MPI libraries; (3) Wi4MPI, a collaboration between CEA and Lawrence Livermore National Laboratory (LLNL), is also tested and deployed on several supercomputing systems at LLNL, which have a different computing environment than CEA's.

Our plan for ISC23 is to use an Amazon AWS Parallel instance to create an HPC cluster with Wi4MPI, several MPI implementations, and several container images. Participants will begin compiling and translating MPI on simple applications and more complex codes. From there, we will move on to the Python use case and the containers use case. This AWS setup is useful in a hybrid environment and avoids requiring tutorial participants to download large virtual images or containers onto their own systems.

We devised several hands-on exercises for Modules 2 and 3 of the tutorial:

Module 2: Changing MPI libraries dynamically with Wi4MPI

1. Build a simple MPI program with MVAPICH2 and, using Wi4MPI, run it with Open MPI without recompilation

2. Using Wi4MPI, compare the performance of selected OSU micro-benchmarks built with MVAPICH2 and ran under two different MPIs (Open MPI and MPICH)
3. Perform a similar experiment with SW4lite, an earthquake ground motion simulation code.
4. Build and install Wi4MPI (to teach attendees how to do this, but not necessary to complete the exercises since there will be a Wi4MPI library already installed).

Module 3: Applying Wi4MPI to key use cases in HPC

1. *Best MPI for my binary*: Bring your MPI application (or choose from a presenter-provided suite of MPI codes) and run it under MVAPICH2, Open MPI, and MPICH without recompilation using Wi4MPI. Compare performance.
2. *Flexible distributed Python*: Using Wi4MPI, run a distributed Python application with mpi4py under MVAPICH2 and Open MPI without rebuilding mpi4py.
3. *Lack of portability and lack of flexibility–no dynamic translation*: Run a given Docker container matching the container MPI to the host MPI.
4. *Portability and ease-of-use*: Using Wi4MPI, run a Docker container with MVAPICH2 on an optimized host MPI.

Finally, several codes we plan on using in the tutorial have been run successfully on a wide variety of supercomputers with different processors, accelerators, networks, and MPI libraries. These experiments have been documented in the IEEE Cluster 2021 proceedings [1].

1.5 Updates from previous presentations

While this would be the first edition of this tutorial, we have hosted and organized training sessions focused on Wi4MPI at the CEA supercomputing center in collaboration with Eolen/AS+. In particular, we facilitated the 2018, 2019 and 2022 PRACE training courses: `https://events.prace-ri.eu/event/740/`, `https://events.prace-ri.eu/event/890/`, and `https://events.prace-ri.eu/event/1404/`. In these sessions, we presented an overview of ABI incompatibility challenges, basic Wi4MPI usage on key application examples, and a subset of the hands-on activities described in this proposal.

Unlike the PRACE training, the ISC23 tutorial has a broader focus on addressing ABI incompatibility in MPI and presents key use cases in the fields of containers and Python-based frameworks. In addition, we will leverage some of the previously developed hands-on exercises for this tutorial.

1.6 Number of attendees from previous tutorials

We had about 25 people attend each of the PRACE training sessions described above (75 people total).

1.7 URLs to sample slides and other material

`https://github.com/cea-hpc/wi4mpi/tree/tutorials/doc/tutorials/isc2023`

2 Logistics

2.1 Length

This is a half-day tutorial of 3.5 hours.

2.2 Content split

35% Beginner

Goals: Understand ABI incompatibility in MPI and leverage pragmatic dynamic translation to address this problem.

Activities: Load Wi4MPI with modules and use it to translate MPI codes at run time using the Preload and Interface modes.

65% Intermediate

Goals: Apply MPI dynamic translation to achieve portability, potentially increase performance, and leverage optimized host MPIs from within a container and from distributed Python applications with specific MPI library dependencies.

Activities: Using dynamic translation, compare performance between MPIs with benchmark applications and SW4lite. Run container and distributed Python applications with and without Wi4MPI.

2.3 Requirements and prerequisites

Attendees will need a laptop equipped with Wi-Fi, a shell terminal, and the ssh program. Users will be provided accounts to access Amazon AWS parallel instances required for demonstrations and hands-on exercises.

To successfully follow the tutorial, attendees should be familiar with compiling and running parallel codes on HPC systems. They should also have basic knowledge of MPI allowing them to launch and run MPI programs.

2.4 Estimated number of attendees

ABI incompatibility in MPI and the Wi4MPI library are gaining momentum and were the subject of discussions in several HPC conferences in 2022: the High Performance Container Workshop (HPCW) at ISC 2022, the Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE) at SC 2022, and EuroMPI 2022.

Therefore, we believe this tutorial will be of interest to ISC attendees and we expect between 50 and 80 attendees.

3 Resume of presenters

Edgar A. León

Computer Scientist, Lawrence Livermore National Laboratory

Education

Ph.D. Computer Science	University of New Mexico	2009
M.S. Computer Science	University of New Mexico	2003
B.S. Computer Science	Universidad Nacional Autónoma de México	2001

Professional Experience (Since 2009)

2011–Present, Computer Scientist, **Lawrence Livermore National Laboratory (LLNL)**
Leading research and cross-functional team collaboration in HPC; providing technical requirements for future National Nuclear Security Administration (NNSA) supercomputers of the U.S. Department of Energy.
2018, Visiting Scientist, **French Alternative Energies and Atomic Energy Commission (CEA)**
Set strategic direction and oversaw implementation of NNSA-CEA partnership.
2010–2011, Postdoctoral Researcher, **IBM Austin Research Laboratory**
Optimized IBM's Power7-IH, a supercomputer architecture developed under the PERCS program.
2009, Research Consultant, **Sandia National Laboratories**
Developed a scalable simulation env. to study the impact of novel architectures on applications.

Synergistic Activities

2022, **Tutorial Presenter**, ACM TAPIA Conference
Supercomputer Systems 101.
2022, **Tutorial Presenter**, CEA-EDF-Inria Computer Science Summer School
Affinity and Binding for Hybrid Applications.
2022, **Lecturer**, Parallel Computing Summer Lecture Series, Los Alamos National Laboratory
Machine Topology and Binding.
2021, Invited Speaker, Workshop on Resource Arbitration for Dynamic Runtimes (RADR)
Node Resource Management in Next-Generation Systems.
2019, Mini-Symposium Leader, ACM Platform for Advanced Scientific Computing (PASC)
Mapping Parallel Scientific Applications onto Complex Architectures Portably and Efficiently.
2018, Invited Panelist, Intel Panel, Supercomputing Conference
Exascale Computing Challenges.
2018, Keynote Speaker, INFOCOMP
Mapping Parallel Scientific Applications: A Memory-Driven Approach to Performance Portability.
2016, **Tutorial Presenter**, HPC Day–Department of Energy National Laboratories, ACM TAPIA Conference
Introduction to Parallel Programming with MPI.
2016, **Tutorials Chair**, Hot Interconnects
IEEE International Symposium on High-Performance Interconnects.
2016, Session Leader, DOE Centers of Excellence Performance Portability Meeting
Tools, Compilers, and System Software Requirements.
2015, Invited Speaker, University of California, Merced
Optimizing Explicit Hydrodynamics for Power, Energy, and Performance.
2006, **Instructor**, University of New Mexico
CS341 Computer Organization and Architecture.
2006, **Invited Instructor**, University of Costa Rica
Workshop on Building, Programming, and Administering a Linux High-Performance Cluster.

Selected Publications

[1] **Edgar A. León**, Marc Joos, Nathan Hanford, Adrien Cotte, Tony Delforge, François Diakhaté, Vincent Ducrot, Ian Karlin, and Marc Pérache. On-the-fly, robust translation of MPI libraries. In *International Conference on Cluster Computing*, CLUSTER'21, pages 504–515. IEEE, 2021. Acceptance rate: 48/163 (29%).

[2] **Edgar A. León**, Trent D'Hooge, Nathan Hanford, Ian Karlin, Ramesh Pankajakshan, Jim Foraker, Chris Chambreau, and Matthew L. Leininger. TOSS-2020: A commodity software stack for HPC. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, SC'20, pages 553–567. IEEE Computer Society, November 2020. Acceptance rate: 95/378 (25%).

[3] **Edgar A. León**, Balazs Gerofi, Julien Jaeger, Guillaume Mercier, Rolf Riesen, Masamichi Takagi, and Brice Goglin. Application-driven requirements for node resource management in next-generation systems. In *International Workshop on Runtime and Operating Systems for Supercomputers*, ROSS'20. IEEE, November 2020.

[4] J. P. Dahm, D. F. Richards, A. Black, A. D. Bertsch, L. Grinberg, I. Karlin, S. Kokkila-Schumacher, **Edgar A. León**, R. Neely, R. Pankajakshan, and O. Pearce. Lessons learned from the Sierra Center of Excellence. *IBM Journal of Research and Development*, December 2019.

[5] **Edgar A. León**, Brice Goglin, and Andres Rubio Proaño. M&MMs: Navigating complex memory spaces with hwloc. In *International Symposium on Memory Systems*, MEMSYS'19, Washington, DC, September 2019. ACM.

[6] **Edgar A. León** and Matthieu Hautreux. Achieving transparency mapping parallel applications: A memory hierarchy affair. In *International Symposium on Memory Systems*, MEMSYS'18, Washington, DC, October 2018. ACM.

[7] Bo Li, **Edgar A. León**, and Kirk W. Cameron. Understanding power measurement capabilities on Zaius Power9. In *International Conference on Advanced Communications and Computation*, INFOCOMP'18, Barcelona, Spain, July 2018. IARIA. **Best Paper Award**.

[8] Bo Li, **Edgar A. León**, and Kirk W. Cameron. COS: A parallel performance model for dynamic variations in processor speed, memory speed, and thread concurrency. In *International Symposium on High-Performance Parallel and Distributed Computing*, HPDC'17, Washington, DC, June 2017. ACM. **Karsten Schwan Best Paper Award**. Acceptance rate: 19%.

[9] **Edgar A. León**, Chris Chambreau, and Matthew L. Leininger. What do scientific applications need? An empirical study of multirail network bandwidth. In *International Conference on Advanced Communications and Computation*, INFOCOMP'17. IARIA, June 2017. **Best Paper Award**.

[10] **Edgar A. León**, Ian Karlin, Abhinav Bhatele, Steven H. Langer, Chris Chambreau, Louis H. Howell, Trent D'Hooge, and Matthew L. Leininger. Characterizing parallel scientific applications on commodity clusters: An empirical study of a tapered fat-tree. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, SC'16, Salt Lake City, UT, November 2016. IEEE/ACM. Acceptance rate: 82/446 (18%).

[11] **Edgar A. León**, Ian Karlin, and Adam T. Moody. System noise revisited: Enabling application scalability and reproducibility with SMT. In *International Parallel & Distributed Processing Symposium*, IPDPS'16, Chicago, IL, May 2016. IEEE. Acceptance rate: 114/496 (23%).

[12] **Edgar A. León**, Ian Karlin, and Ryan E. Grant. Optimizing explicit hydrodynamics for power, energy, and performance. In *International Conference on Cluster Computing*, Cluster'15, Chicago, IL, September 2015. IEEE. **Best Paper Award**. Acceptance rate: 38/157 (24%).

[13] **Edgar A. León**, Rolf Riesen, Kurt B. Ferreira, and Arthur B. Maccabe. Cache injection for parallel applications. In *International Symposium on High-Performance Parallel and Distributed Computing*, HPDC-20, San Jose, CA, June 2011. ACM. Acceptance rate: 22/170 (12%).

[14] **Edgar A. León**, Rolf Riesen, Arthur B. Maccabe, and Patrick G. Bridges. Instruction-level simulation of a cluster at scale. In *International Conference on High-Performance Computing, Networking, Storage and Analysis*, SC'09. ACM/IEEE, November 2009. Acceptance rate: 59/261 (22%).

Marc Joos

Computer Scientist,
French Alternative Energies and Atomic Energy Commission

Education

Ph.D. Astronomy & Astrophysics	Université Pierre et Marie Curie – Paris VI (France)	2012
M.S. Astronomy & Astrophysics	Université Denis Diderot – Paris VII, Observatoire de Paris	2009
B.S. Fundamental Physics	Université Denis Diderot – Paris VII (France)	2007

Professional Experience

2018–Present, Computer Scientist, **French Alternative Energies and Atomic Energy Commission (CEA, France)**
Level 3 application support team of CEA computing complex, project management of Wi4MPI development, design and hosting of trainings and workshops for CEA computing complex users

2017–2018, CEO, **Mindmatcher (France)**
Management of a startup developing solutions to harvest & analyze data in the HR sector

2015–2018, Head of R&D department, **Adoc Talent Managemnet (France)**
In charge of statistical studies on doctorate holders career paths & of software development projects (data analysis and visualization), hosting training for early career researchers (about 100 days of training hosted)

2012–2015, Computer Scientist, **French Alternative Energies and Atomic Energy Commission (CEA, France)**
Optimization and hybridation (shared memory and GPU) of a magnetohydrodynamics code, organization & hosting of a Python bootcamp for astrophysicists

2009–2012, Doctoral Researcher, **Radioastronomy Laboratory (LRA), ENS, Paris Observatory, Pierre et Marie Curie University (France)**
Simulations, analytical study & observation comparisons of star formation processes

2009–2012, Popularization Officer, **Pierre et Marie Curie University (France)**
Organization and hosting of scientific events

Relevant Teaching Experience

2022, **Instructor**, ETSN 2022 Summer School
Multi-GPU programming

2022, **Instructor**, CEA Very Large Computing Center (TGCC)
Leveraging the power of MPI libraries with Wi4MPI

2022, **Instructor**, CEA Research & Technology Computing Center (CCRT)
Introduction to GPU computing

2021, **Speaker**, CCRT
Novel architecture & good practices on CCRT supercomputer

2015-2018, **Instructor**, Adoc Talent Management
Course series for early career researchers in French universities and research organizations, including courses on project management and innovation awareness

2015, **Instructor**, Astrophysics Department, CEA
Python programming for performance

Publications

[1] T. Van Reeth, P. De Cat, J. Van Beeck, V. Prat, D. J. Wright, H. Lehmann, A. N. Chené, E. Kambe, S. L. S. Yang, G. Gentile, and **Marc Joos**. The near-core rotation of HD 112429: a gamma Doradus star with TESS photometry and legacy spectroscopy. *A&A*, March 2022. accepted for publication.

[2] Edgar León, **Marc Joos**, Nathan Hanford, Adrien Cotte, Tony Delforge, François Diakhaté, Vincent Ducrot, Ian Karlin, and Marc Pérache. On-the-fly, robust translation of MPI libraries. In *International Conference on Cluster Computing*, CLUSTER'21, pages 504–515. IEEE, 2021. Acceptance rate: 48/163 (29%).

[3] **Marc Joos**, S. Fromang, and H. Méheut. Magnetohydrodynamic Turbulence in Accretion Discs: A Test Case for Petascale Computing in Astrophysics. In N. V. Pogorelov, E. Audit, and G. P. Zank, editors, *Numerical Modeling of Space Plasma Flows ASTRONUM-2014*, volume 498 of *Astronomical Society of the Pacific Conference Series*, page 10, October 2015.

[4] Heloise Meheut, Sébastien Fromang, Geoffroy Lesur, **Marc Joos**, and Pierre-Yves Longaretti. Angular momentum transport and large eddy simulations in magnetorotational turbulence: the small Pm limit. *A&A*, 579:A117, July 2015.

[5] Marco Padovani, Daniele Galli, Patrick Hennebelle, Benoît Commercon, and **Marc Joos**. Cosmic-ray propagation at small scale: a support for protostellar disc formation. In *CRISM 2014*, March 2015.

[6] M. Padovani, D. Galli, P. Hennebelle, B. Commerçon, and **Marc Joos**. The role of cosmic rays on magnetic field diffusion and the formation of protostellar discs. *A&A*, 571:A33, November 2014.

[7] **Marc Joos**, Patrick Hennebelle, Andrea Ciardi, and Sébastien Fromang. The Early Era: How do protostellar discs form? In Mark Booth, Brenda C. Matthews, and James R. Graham, editors, *Exploring the Formation and Evolution of Planetary Systems*, volume 299, pages 163–164, January 2014.

[8] **Marc Joos**, Patrick Hennebelle, and Andrea Ciardi. Protostellar Disk Formation and Angular Momentum Transport During Magnetized Core Collapse. In *The Labyrinth of Star Formation*, volume 36 of *Astrophysics and Space Science Proceedings*, page 69, January 2014.

[9] **Marc Joos**, Patrick Hennebelle, Andrea Ciardi, and Sebastien Fromang. The Early Era: How do protostellar discs form? In *Protostars and Planets VI*, July 2013.

[10] **Marc Joos**, P. Hennebelle, A. Ciardi, and S. Fromang. The influence of turbulence during magnetized core collapse and its consequences on low-mass star formation. *A&A*, 554:A17, June 2013.

[11] **Marc Joos**, P. Hennebelle, and A. Ciardi. Protostellar disk formation and transport of angular momentum during magnetized core collapse. *A&A*, 543:A128, July 2012.

[12] P. Hennebelle, B. Commerçon, **Marc Joos**, R. S. Klessen, M. Krumholz, J. C. Tan, and R. Teyssier. Collapse, outflows and fragmentation of massive, turbulent and magnetized prestellar barotropic cores. *A&A*, 528:A72, April 2011.

[13] Nicolas Giraud, **Marc Joos**, Jacques Courtieu, and Denis Merlet. Application of a 1h δ-resolved 2d nmr experiment to the visualization of enantiomers in chiral environment, using sample spatial encoding and selective echoes. *Magnetic Resonance in Chemistry*, 47(4):300–306, 2009.

Nathan Hanford

Computer Scientist,
Lawrence Livermore National Laboratory

Education

B.S.E. Computer Science and Engineering	University of Connecticut	2011
Ph.D. Computer Science	University of California Davis	2018

Professional Experience

2020–present, Computer Scientist, **Lawrence Livermore National Laboratory (LLNL)**
Design and develop the ABI compatibility architecture for HPC systems at LLNL. Deploy and support MPI and messaging middleware.

2019–2020, Postdoctoral Researcher, **Lawrence Livermore National Laboratory (LLNL)**
Investigated performance bottlenecks between accelerators and fabrics on different HPC architectures.

Teaching Experience

ECS 32A: Introduction to Programming, **UC Davis**

Instructor of Record. 288 Registered Students in Winter, 299 Registered Students in Spring.
Developed all new lectures and assessments.
Integrated LTI between zyBooks and Canvas to implement automatic grading and assignment management.
Engaged campus resources to develop video walk-throughs.

ECS 132: Probability and Statistical Modeling for Computer Science, **UC Davis**

Instructor of Record. 104 Registered Students.
Revise and expand existing notes into a CS-applications-focused curriculum.
Increase focus of the curriculum on the R programming language.
Develop all new assessments: homework assignments, weekly quizzes, and exams.

Publications

[1] Edgar A. León, Marc Joos, **Nathan Hanford**, Adrien Cotte, Tony Delforge, François Diakhaté, Vincent Ducrot, Ian Karlin, and Marc Pérache. On-the-fly, robust translation of MPI libraries. In *International Conference on Cluster Computing*, CLUSTER'21, pages 504–515. IEEE, 2021. Acceptance rate: 48/163 (29%).

[2] Edgar A. León, Trent D'Hooge, **Nathan Hanford**, Ian Karlin, Ramesh Pankajakshan, Jim Foraker, Chris Chambreau, and Matthew L. Leininger. TOSS-2020: A commodity software stack for HPC. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, SC'20, pages 553–567. IEEE Computer Society, November 2020. Acceptance rate: 95/378 (25%).

[3] **Nathan Hanford**, Vishal Ahuja, Matthew K. Farrens, Brian Tierney, and Dipak Ghosal. A survey of end-system optimizations for high-speed networks. *ACM Computing Surveys*, 51(3):1–36, jul 2018.

[4] Fatma Alali, **Nathan Hanford**, Eric Pouyoul, Raj Kettimuthu, Mariam Kiran, Ben Mack-Crane, Brian Tierney, Yatish Kumar, and Dipak Ghosal. Calibers: A bandwidth calendaring paradigm for science workflows. *Future Generation Computer Systems*, 89:736–745, dec 2018.

[5] **Nathan Hanford**, David Fridovich-Keil, **Nathan Hanford**, Margaret P. Chapman, Claire J. Tomlin, Matthew K. Farrens, and Dipak Ghosal. A model predictive control approach to flow pacing for tcp, 2017.

[6] **Nathan Hanford**, Vishal Ahuja, Matthew Farrens, Dipak Ghosal, Mehmet Balman, Eric Pouyoul, and Brian Tierney. Improving network performance on multicore systems: Impact of core affinities on high throughput flows. *Future Generation Computer Systems*, 56:277–283, mar 2016.

[7] **Nathan Hanford**, Brian Tierney, and Dipak Ghosal. Optimizing data transfer nodes using packet pacing. In *Proceedings of the Second Workshop on Innovating the Network for Data-Intensive Science - INDIS '15*. ACM Press, 2015.

[8] **Nathan Hanford**, Vishal Ahuja, Matthew Farrens, Dipak Ghosal, Mehmet Balman, Eric Pouyoul, and Brian Tierney. Analysis of the effect of core affinity on high-throughput flows. In *2014 Fourth International Workshop on Network-Aware Data Management*. Institute of Electrical & Electronics Engineers (IEEE), nov 2014.

[9] **Nathan Hanford**, Vishal Ahuja, Matthew K. Farrens, Dipak Ghosal, Mehmet Balman, Eric Pouyoul, and Brian Tierney. Impact of the end-system and affinities on the throughput of high-speed flows. In *Proceedings of the tenth ACM/IEEE symposium on Architectures for networking and communications systems*. ACM, oct 2014.

[10] **Nathan Hanford**, Vishal Ahuja, Mehmet Balman, Matthew K. Farrens, Dipak Ghosal, Eric Pouyoul, and Brian Tierney. Characterizing the impact of end-system affinities on the end-to-end performance of high-speed flows. In *Proceedings of the Third International Workshop on Network-Aware Data Management - NDM '13*. ACM Press, 2013.

4 Acknowledgment

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-TR-847128.

References

[1] E. A. León, M. Joos, N. Hanford, A. Cotte, T. Delforge, F. Diakhaté, V. Ducrot, I. Karlin, and M. Pérache. On-the-fly, robust translation of MPI libraries. In *International Conference on Cluster Computing*, CLUSTER'21, pages 504–515. IEEE, 2021.