Platform Readiness

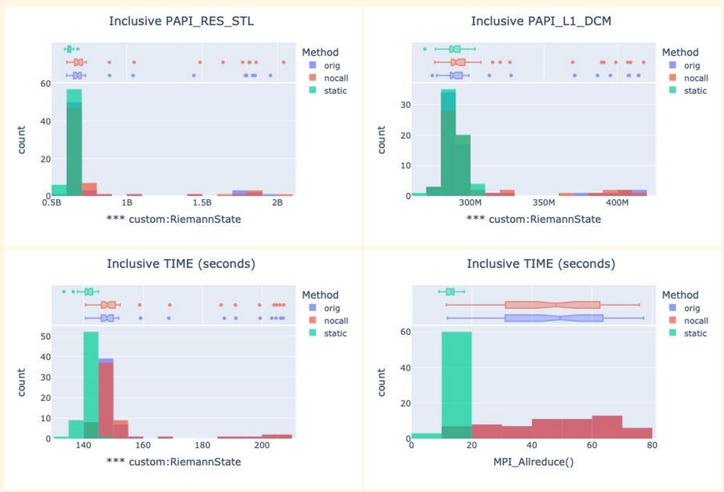
Lead: Samuel Williams (Argonne National Laboratory)
Co-lead: Kevin Huck (University of Oregon)

RAPIDS

Performance Modeling, Analysis, and Optimization

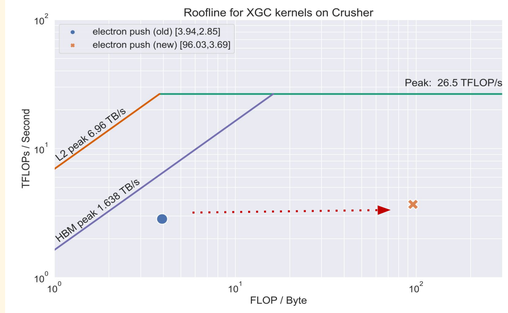
Understanding and optimizing the performance of applications on heterogeneous architectures is a significant challenge; we develop performance modeling tools to guide the process and performance analysis tools to capture empirical data.

Recent accomplishments:
- Measured, analyzed and tuned XGC kernels for GPU using TAU and APEX
- Developed a data dependency aware performance model for sparse triangular solvers
- Improved the performance of a loop amplitude code by replacing quadruple precision with double double



Right: FLASH application executed on ALCF Theta system, showing performance distributions of counters and time across MPI ranks. The original code performance (orig), the performance after refactored code was removed (nocall), and with the improved static data allocations (static) are shown.

Below: Using APEX to capture Roofline metrics from the main push kernel, we identified opportunities for improvement. Changing the launch bounds for the Kokkos kernel from the default of <1024,1> to <256,2> at compile time increased computational intensity 25x and reduced time by 22% on Frontier (AMD/HIP).



Key Publications:

Weile Wei, Eduardo D'Azevedo, Kevin Huck, Arghya Chatterjee, Oscar Hernandez, and Hartmut Kaiser. 2021. Memory reduction using a ring abstraction over GPU RDMA for distributed quantum Monte Carlo solver. In Proceedings of the Platform for Advanced Scientific Computing Conference (PASC '21). Association for Computing Machinery, New York, NY, USA, Article 14, 1–9. https://doi.org/10.1145/3468267.3470618

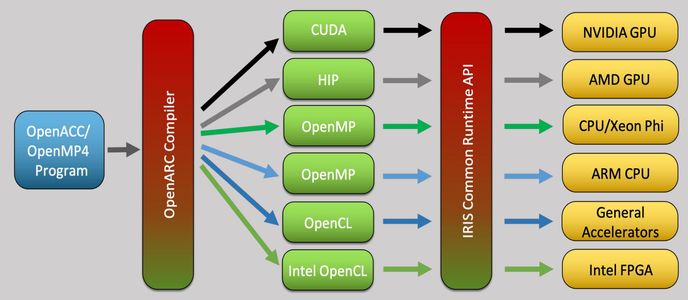
K. Huck et al., "Performance Debugging and Tuning of Flash-X with Data Analysis Tools," 2022 IEEE/ACM Workshop on Programming and Performance Visualization Tools (ProTools), Dallas, TX, USA, 2022, pp. 1-10. doi: 10.1109/ProTools56701.2022.00009.

Portable Programming

Effective application development for heterogeneous architectures requires portable programming models. We develop tools and techniques to support selection of OpenMP directives and migration to descriptive constructs, topology-aware memory management, and integrated translation and optimization of OpenACC and OpenMP.

Recent accomplishments:
- IRIS: portable runtime system exploiting multiple heterogeneous programming systems
- Explored use of OpenACC and OpenMP to portably accelerate the Fortran-based EFIT code (FES)
- Developed an algorithm-centric performance portability metric

The IRIS heterogeneous runtime system exploits multiple heterogeneous programming systems (e.g., CUDA, Hexagon, HIP, Level Zero, OpenCL, OpenMP) so that a single application can use multiple, heterogeneous types of devices available on the target system. Multi-programming model (bottom) can improve the observed performance portability across a wide range of configurations.

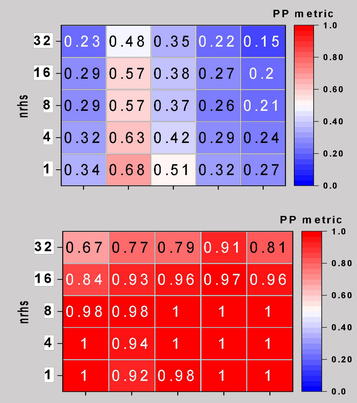


Key Publications:

A. Cabrera, S. Hitefield, J. Kim, S. Lee, N. R. Miniskar, and J. S. Vetter, Toward Performance Portable Programming for Heterogeneous System-on-Chips: Case Study with Qualcomm Snapdragon SoC, The IEEE High Performance Extreme Computing Conference (HPEC '21), 2021.

K. Z. Ibrahim, C. Yang, P. Maris. Performance Portability of Sparse Block Diagonal Matrix Multiple Vector Multiplications on GPUs, 2022 International Workshop on Performance, Portability and Productivity in HPC (P3HPC), SC22.

A. P. Dieguez, M. Choi, X. Zhu, B. M. Wong and K. Z Ibrahim. "ML-based Performance Portability for Time-Dependent Density Functional Theory in HPC Environments" 2022 International Workshop on Performance Modeling, Benchmarking and Simulation of High-Performance Computer Systems (PMBS), Supercomputing 2022.



Kokkos (top) performance portability metric for a sparse matrix multi-vector computation varies significantly with problem configuration.

Multi-programming model (bottom) can improve the observed performance portability across a wide range of configurations.

The platform readiness thrust focuses on preparing scientific codes for current and upcoming systems through the application of best-in-class expertise and tools, addressing concerns such as performance portability and correctness across multiple parallel programming paradigms.

Autotuning

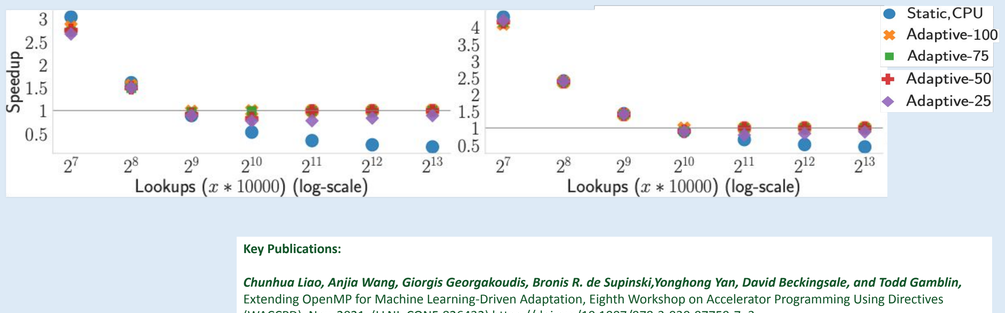
Achieving and maintaining correctness of applications with multiple programming models is formidable. Verification tools can help identify potential bugs and ensure that new bugs are not introduced.

Recent accomplishments:
- Flexible autotuning framework including Bayesian optimization and model-based estimation
- OpenMP language extensions for runtime adaptation
- Framework for transfer-learning in autotuning

```
1 void vecAdd(double *A, double *B, double *C, size_t N) {
2   #pragma omp begin adaptation \
3     model_name(by_len) features(N) \
4     variants(cpu, gpu)
5
6   #pragma omp metadirective \
7     when(user={adaptation(by_len==cpu)} : parallel for) \
8     when(user={adaptation(by_len==gpu)} : \
9       target map(to: A[0:N], B[0:N]) map(from:C[0:N]) \
10      teams distribute parallel for)
11  for(size_t i = 0; i < N; ++i)
12      C[i] = A[i] + B[i];
13
14  #pragma omp end adaptation model_name(by_len)
15 }
```

Left: New OpenMP directives supporting CPU-GPU execution adaptation: leveraging directive variants defined by metadirective

Below: Results on CPU-GPU execution adaptation: XSbench on Power9+v100 (left) and Intel+P100 (right)



Key Publications:

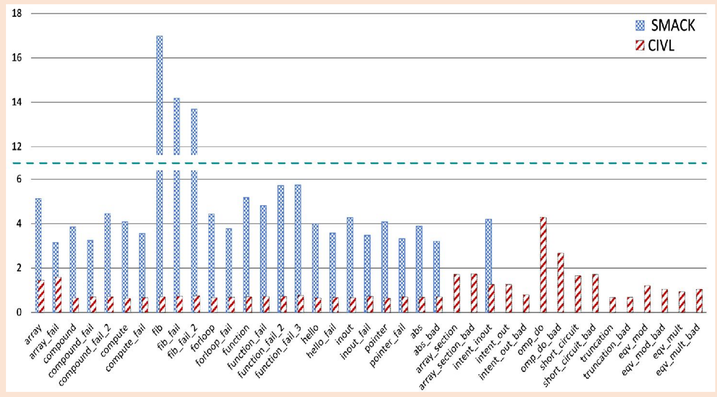
Chunhua Liao, Anjia Wang, Giorgis Georgakoudis, Bronis R. de Supinski,Yonghong Yan, David Beckingsale, and Todd Gamblin, Extending OpenMP for Machine Learning-Driven Adaptation, Eighth Workshop on Accelerator Programming Using Directives (WACCPD), Nov. 2021. (LLNL-CONF-826432) https://doi.org/10.1007/978-3-030-97759-7_3

Giorgis Georgakoudis, Konstantinos Parasyris, Chunhua Liao, David Beckingsale, Todd Gamblin, Bronis de Supinski, Machine Learning-Driven Adaptive OpenMP For Portable Performance on Heterogeneous Systems, ACM Transactions on Architecture and Code Optimization (submitted), Feb. 2023

Correctness

Achieving and maintaining correctness of applications with multiple programming models is formidable. Verification tools can help identify potential bugs and ensure that new bugs are not introduced.

Recent accomplishments:
- Support for verification of Fortran in CIVL framework through correctness-preserving transformations to CIVL language
- New model checking technique that can be used to verify race-freedom for all sequentially-consistent executions



Left: Verification time (seconds) for CIVL (red striped) vs. SMACK (blue) on a Fortran verification benchmark suite. Lower is better. No bar is shown when a tool could not complete a verification task. Each time is the mean over 5 of 7 executions after dropping the shortest and longest.

Below: CIVL detects a data race in a DataRaceBench benchmark, reporting precise diagnostic information to user.

Key Publications:

Wenhao Wu, Jan Hückelheim, Paul D. Hovland, and Stephen F. Siegel, Verifying Fortran Programs with CIVL. TACAS 2022. doi: 10.1007/978-3-030-99524-9_6. Badges: *EAPLS Artifacts Available* and *EAPLS Artifacts Evaluated and Reusable*, v1.1.

Wu, W., Hückelheim, J., Hovland, P.D., Luo, Z., Siegel, S.F. (2023). Model Checking Race-Freedom When "Sequential Consistency for Data-Race-Free Programs" is Guaranteed. In: Enea, C., Lal, A. (eds) Computer Aided Verification. CAV 2023. Lecture Notes in Computer Science, vol 13965. Springer, Cham. https://doi.org/10.1007/978-3-031-37703-7_13

```
int main() {
  int i, len=100, a[100];
  for (i=0; i<len; i++) a[i]=i;
#pragma omp parallel for
  for (i=0; i<len-1; i++) a[i+1]=a[i]+1;
  printf("a[50]=%d\n", a[50]);
  return 0;
}


> civl verify -input_omp_thread_max=10 \
  DRB029-truedep1-orig-yes.c

a[50]=50
Data-race detected:
  {a[1], a[3], ..., a[97]}
read by thread 1 intersects
  {a[1], a[3], a[5], ..., a[99]}
written by thread 0.
```

SciDAC Institute for Computer Science, Data, and Artificial Intelligence

Argonne NATIONAL LABORATORY · Brookhaven National Laboratory · BERKELEY LAB · University of Delaware · kitware · Los Alamos NATIONAL LABORATORY · UF University of Florida · Lawrence Livermore National Laboratory · The Ohio State University · Northwestern University · University of Oregon · Oak Ridge National Laboratory · SCI www.sci.utah.edu · U.S. Department of Energy · Office of Science