Argo: Then and Now

ANL: Pete Beckman (*PI*), Idriss Daoudi, Rinku Gupta, Kamil Iskra, John-Luke Navarro, Swann Perarnau, John Tramm, Brice Videau, Kazutomo Yoshii,
LLNL: Maya Gokhale (*co-PI*), Eric Green, Keita Iwabuchi, Roger Pearce, Ivy Peng, Abhik Sarkar; Tapasya Patki (*co-PI*), Stephanie Brink, Aniruddha Marathe, Barry Rountree, Kathleen Shoga
University of Arizona: David Lowenthal and team

https://web.cels.anl.gov/projects/argo/

Developing vendor-neutral, open-source software for OS/R improvements

Argo improves or augments existing OS/R components for use in production HPC systems, providing portable, open source software that improves the performance and scalability and that provides increased functionality to exascale applications.

AML

Overview

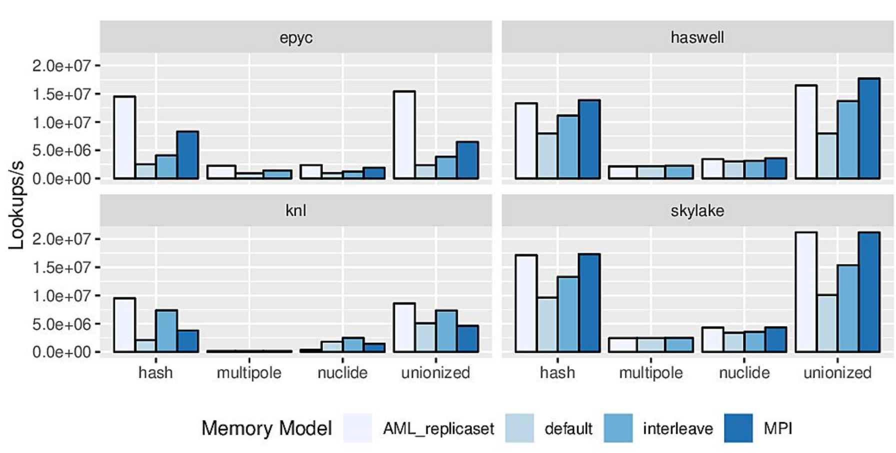
- A library for application-aware management of byte-addressable memory devices
  - Explicit placement and movement of data
- Designed as a collection of building blocks
  - Users can create custom memory management policies for allocation and placement of data across devices
- Designed for deep, heterogeneous memory systems, featuring NUMA, HBM, or GPU memory

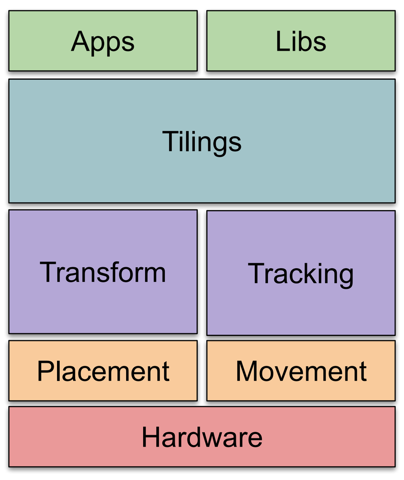
Impact

- Improved performance of applications regarding memory usage on the complex compute nodes of exascale systems
- Improved performance portability of applications across exascale systems

Before ECP

- A proof-of-concept library then called *DeepRAM*
- Focus on multilevel DRAM hierarchy *on CPU*
  - Software-managed scratchpad in MCDRAM
- Exploration of different migration mechanisms
  - User-space, kernel-space, hardware
  - Asynchronous using dedicated CPU threads





Key Components

- *Topology & hardware management*
- *Data layout descriptions (application-specific)*
- *Tiling schemes*
- *Data movement facilities*
- *Pipelining helpers (asynchronous requests)*

Now

- Production-quality implementation
- Major refocus on GPUs, given the eventual architectures of first exascale systems
- Integration into ExaSMR's XSBench
- Interface to build custom memory mapping policies that are application-focused, on top of any GPU interface (OpenCL, CUDA, HIP, oneAPI)
- Duplication of latency-sensitive data across devices
- Transformation, optimization of data layout on target accelerators

Future

- Continuous improvements to application performance, including better use of GPU memory capacity, leading to better scaling
- Towards a vendor-neutral, programming model agnostic memory management layer for future production systems
- Increased use across exascale applications, and more portable performance across complex architectures

Explicit data replication in low-latency memory
● improved performance compared to OpenMP data sharing
● performance on par with tuned MPI process pinning
Integration into ExaSMR's XSBench using the replicaset feature

NRM

Overview

Node Resource Manager is a node-local userspace client-server daemon for managing scientific applications

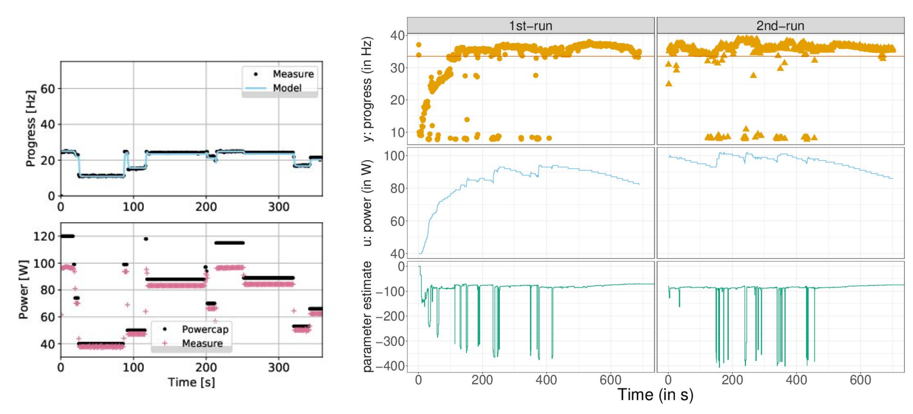
- Compose an application in resource-constrained slices
- Monitor performance, power use, and application progress
- Arbitrate resources at the node level between application and runtime services
  - CPU cores, NUMA nodes, power budget

Impact

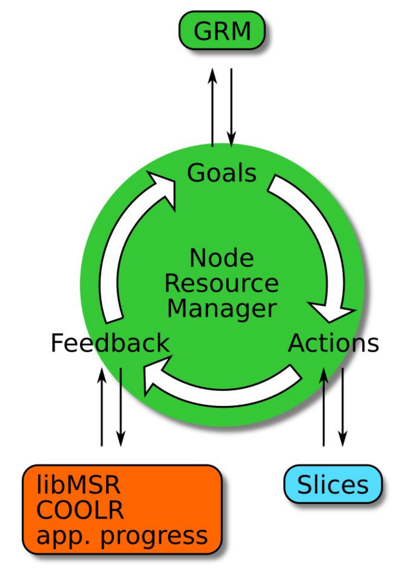
- Better energy efficiency across facilities, with users involved in the process
  - Facilities can make user workloads more energy efficient
  - Users can make flexible improvements to their use of compute nodes

Before ECP

- A collection of experimental components
  - COOLR: monitoring and control of CPU power, temperature, and frequency
  - Compute Containers: performance isolation through partitioning of physical resources



Now



- Integrated, production-quality implementation
- Abstracted resource accounting in the form of "sensors" and "actuators", allowing for flexible control design
  - Monitor application and hardware
  - Actuators act on hardware/application
  - Collection of control loops as functions of available sensors, actuators, and user-defined goal
- Dynamic resource management infrastructure available on production systems
- Power/energy efficiency optimization control loops for exascale systems
- Integration with vendor/facility stack
  - Variorum, PAPI, GeoPM, vendor APIs

Future

- Continuous increase in the quality and applicability of the resource policies available to users
- Moving towards more runtime-reconfigurable software components on compute nodes
- Improvements to the performance and energy efficiency of complex application workloads (workflows)
- Expected improvement to the energy efficiency of facilities

Modelization of application progress and power consumption, and adaptation of the controller parameters during the two-runs initialization phase.

UMap

Overview

- A library that enables user-space optimizations for memory mapping NVM devices into the complex memory hierarchy
- Facilitates direct access to large data sets through virtual address spaces
- Provides application-specific configurations suited to massive observational and simulation data sets
- High-performance design features I/O decoupling, dynamic load balancing, and application-level controls

Impact

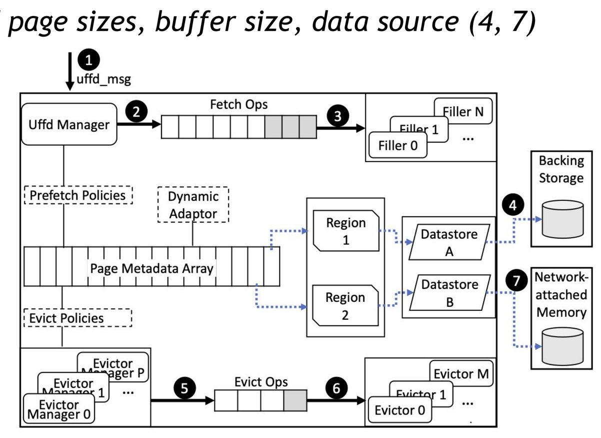
- The UMap memory mapping abstraction is important for the blurring of the memory/storage hierarchy. UMap enables accessing file-resident data as memory
- UMap breaks the dichotomy between memory and storage by providing a unified virtual memory interface and simplifying application code
- UMap enables application-specific tailoring of the in-memory page cache and page size in user space
- Successful use cases demonstrated in graph processing, database, metagenomics, and file compression applications

Before ECP

- A proof-of-concept library then called *PERMA*
- Focus on NVM memories
- Requires kernel modifications and root privileges

Key Components

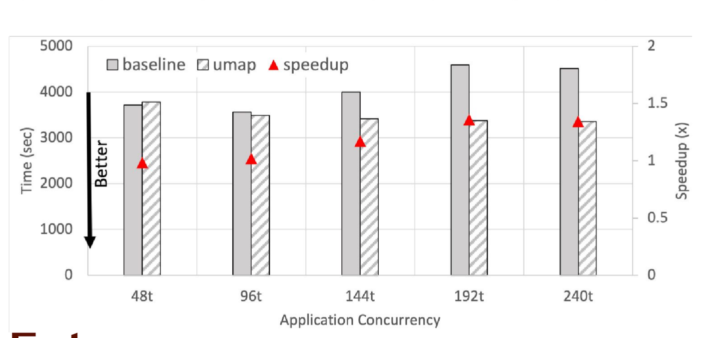
- *Asynchronous message-based API (1—3)*
- *Resolves page faults in regions by fetching/flushing data from datastores following user-defined policies (4—6)*
- *Customized page sizes, buffer size, data source (4, 7)*



Now

- Decoupled page fetch and eviction queues
- Concurrency-aware adaptation
- Dynamic load balancing
- Support for persistent memory allocator
- Supports network-attached memory

Searching out-of-core K-mer database shows UMap with 1.8X speedup over system memory map at high query concurrency



Future

- Continuous improvements to application performance, including ligra graph processing
- Continue improve caching policies for integrating remote memory on future memory servers into application

PowerStack

Overview

- Holistic System Power Management for exascale with production-quality software
  - Kernel-level module for safe access to low-level registers with msr-safe
  - Node-level: CPUs, GPUs Memory with a vendor-neutral open-source library, Variorum, which supports
  - Application-level performance optimizations with a task-aware runtime: Intel GEOPM and Conductor, as well as Kokkos support
  - Power-aware resource management and scheduling with SLURM and Flux
  - Large-scale power telemetry with LDMS
- HPC PowerStack Initiative: Community-wide and international effort with industrial partners (Intel, AMD, IBM, NVIDIA, ARM), academic partners, and national labs

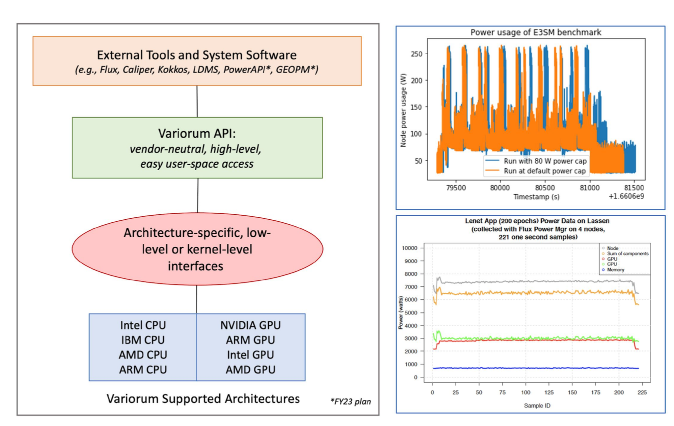
Impact

- Improved performance and energy efficiency across facilities, applications, as well as node-level, with users involved in the process
  - Facilities can make user workloads more energy efficient and performant
  - Users can make flexible improvements to their use of compute nodes

Before ECP

- Sparse efforts existed with msr-safe and libmsr, which were Intel-specific implementations
- Power-aware scheduling prototype based on SLURM simulator
- Runtime system prototype for optimization which was a research code, early version of Intel GEOPM

Variorum v0.6 (Sept 2022) allows for vendor-neutral power management for more than 15 diverse architectures, including El Capitan and Aurora CPUs and GPUs. Right figure shows results from LDMS and Flux on two clusters (Intel and IBM) with E3SM and LBANN workflows at scale, including CPU/memory/GPU data



Now

- Production-quality power management software at all levels, ranging from the node-level all the way through system resource managers, across 15 architectures
- Variorum integrations with Caliper, Kokkos, LDMS, Flux, Intel GEOPM allowing users and administrators to manage power easily at various levels in a vendor-neutral manner
- Power management of large-scale workflows

Future

- Policies to mitigate power swings to be integrated into resource managers
- Optimization of science workflows and dependency graphs using integration with workload managers
- Power management with elastic scheduling
- Additional support for upcoming architectures and performance counters

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering and early testbed platforms, in support of the nation's exascale computing imperative.
Part of 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-POST-803567