Containerization in HPC Environments: Challenges in Optimizing Performance and Security in Scientific Computing

SC24 Atlanta, GA | hpc creates.

Presenter: Vijayalakshmi Saravanan

Conference: SC24 CANOPIE

Project: An Efficient Storage-Driven Machine Learning Model for Performance in the Era of Multimodal Scientific Data.

DOE-funded ASCR Project (Award No: DE-SC0024492)

PI - Saravanan Vijayalakshmi Co-I : Ibrahim Z Khaled

Facility: NERSC PERLMUTTER Supercomputer

UTTyler THE UNIVERSITY OF TEXAS AT TYLER

BERKELEY LAB Lawrence Berkeley National Laboratory

U.S. DEPARTMENT OF ENERGY

Outline

- Overview
- Performance optimization
- Docker Challenges
- Singularity Containers
- Sample scientific use case

Overview/Performance Optimization Challenges

- High data volumes and velocities from sensors and simulations demand efficient data ingestion and processing.
- High performance is crucial for data-intensive and compute-intensive scientific workflows.
- Heterogeneous hardware (CPUs, GPUs, FPGAs) and diverse software dependencies complicate deployment.
- Optimizing container resource utilization (CPU, GPU, memory, network) is essential for efficiency.
- Balancing performance isolation with shared resource access is key to maximizing system performance.

Why Containerization?

Containerizing Scientific Simulations in HPC

- Real-time streaming of scientific data in HPC environments enables rapid data processing and insights

- Cross-System Portability: Containers enable easy transfer of applications and storage setups across HPC systems.

- I/O Optimization: Containers can be tailored for the I/O needs of scientific workloads.

- Streamlined Data Management: Containers support efficient and portable data workflows in HPC.

Efficient Use of Storage Resources

- Containers can use specific storage volumes only when necessary, reducing the load on storage resources. For example, we can mount storage volumes only at runtime, which is beneficial in managing high-demand HPC storage resources, particularly in shared environments.

Challenges in Docker

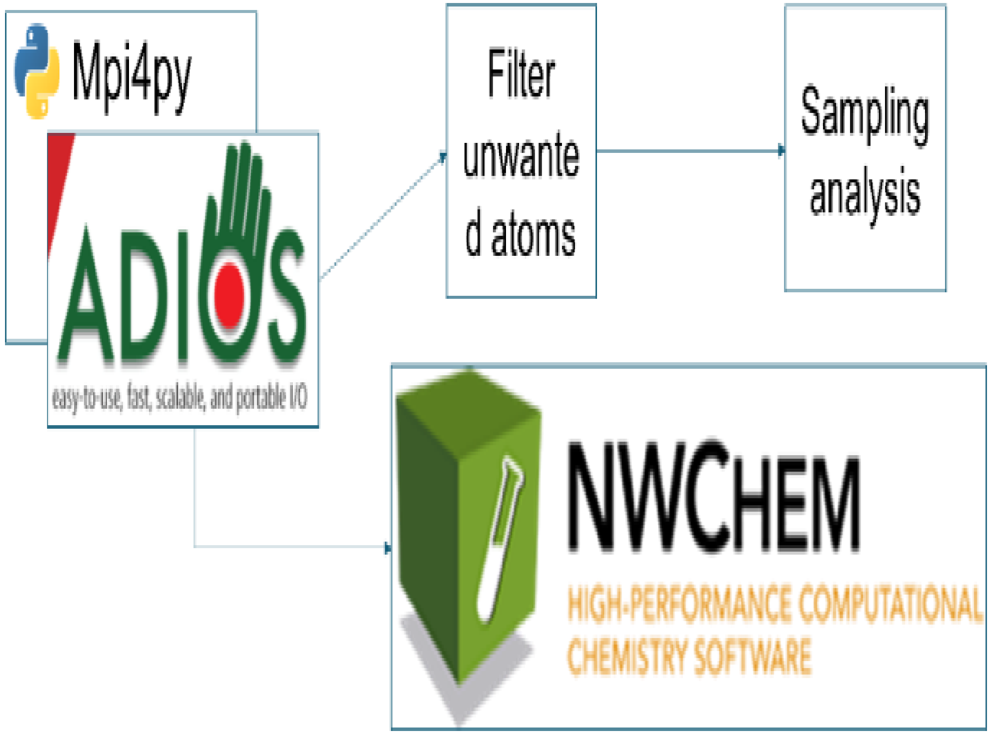
Why Singularity? - Docker lacks in meeting HPC-specific needs for security, scalability, and performance.

- HPC environments present unique challenges compared to traditional workflow deployments
- HPC Storage Compatibility: Singularity integrates well with Lustre, GPFS, and BeeGFS storage.
- Enhanced Security: Unlike Docker, Singularity avoids root access issues, providing greater security for HPC applications.
- Scalability and Efficiency: Singularity delivers better performance and scalability in HPC workloads compared to Docker.

Containerizing the Analysis of Online Streaming Change Point Detection for Molecular Dynamics

Scientific Simulation and Data Handling- Our simulations target systems with one million atoms, generating vast amounts of molecular dynamics (MD) trajectory data, potentially reaching tens of petabytes (PB).

Proposed System: We propose a **containerized online in-situ sampling framework** designed to evaluate existing methods while incorporating an **Online Adaptive Sampling (OAS)** algorithm for improved efficiency and scalability.



Key Goals

Containerizing Change Point Detection: We package the change point detection model in containers, enabling a producer-consumer architecture with multiple container nodes.

Data Flow & Processing: Molecular data acts as the producer, while the OAS container consumes and processes the data using multiple processors.

Efficient Data Sharing: We integrate ADIOS (Adaptable I/O System) for seamless, efficient, and reliable data exchange between molecular data containers and OAS containers.

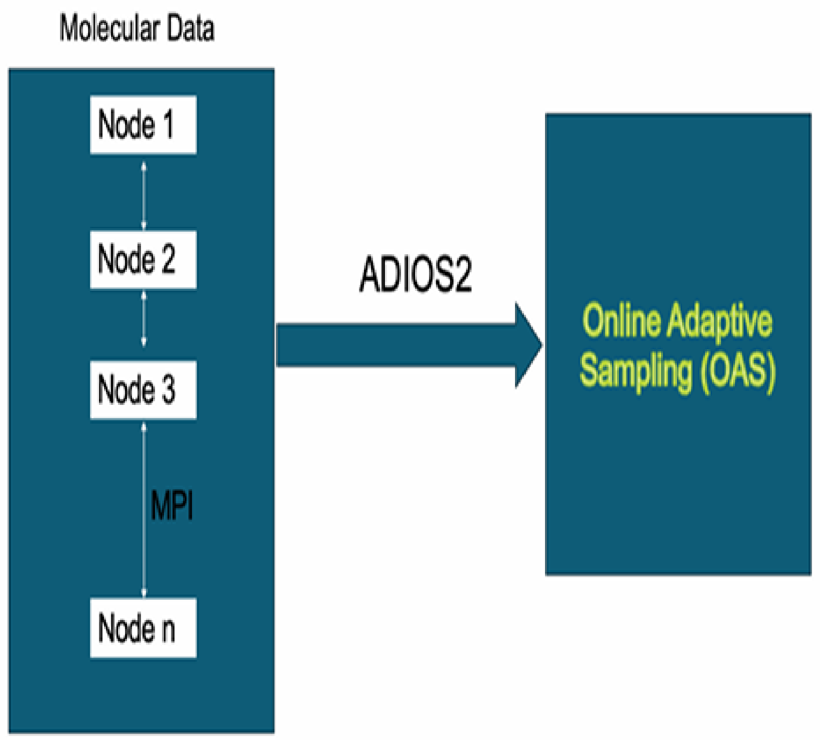
Dockerization of Scientific Computing Workflow

- Producer-Consumer Architecture: We extend the existing online adaptive sampling algorithm using a producer-consumer model based on unsupervised, randomized streaming.

- Enhanced Sampling Framework: We aim to improve the framework with features like atom filtering and integration of additional sampling algorithms for greater flexibility.

- Statistical Sampling Analysis: We apply matrix sketches and the Kernel CUSUM algorithm for effective statistical analysis and change point detection.

- Multi-Container Deployment: We plan to deploy the system in HPC with multiple containers, integrating Change Point Detection (CPD) algorithms into the Online Adaptive Sampler (OAS) framework for real-time molecular simulation and adaptive sampling.



Preliminary Results

- Containerized HPC Workflow for MDT: Achieve reduction in compute time and improvement in resource efficiency.
- Containerized Synthetic Data with GPU Acceleration: Deliver faster predictions and reduction in infrastructure costs.

Best Practices and Recommendations

Performance Optimization

- CPU Pinning: Optimize resource allocation by assigning specific CPUs to containers.
- GPU Virtualization: Efficiently share GPU resources across containers for enhanced performance.

Security & Best Practices

- Secure Container Image Creation: Use minimal base images and enforce least-privilege access to reduce security risks.
- Vulnerability Scanning: Continuously scan containers for vulnerabilities to ensure a secure environment.

Monitoring & Auditing

- Monitoring and Logging: Track performance and resource usage in real time.
- Auditing: Implement auditing mechanisms for security, compliance, and data access tracking.

Future/Work-in-Progress

- Advancements in Container Runtimes, Networking, and Storage for Scientific Workloads
- Further optimizing Singularity for diverse HPC scientific applications.
- Expanding to additional simulations and ML frameworks to enhance HPC utility.

Thank you