

Jarvis: Towards a Shared, User-Friendly, and Reproducible, I/O Infrastructure.

1st Jaime Cernuda

Gnosis Research Center

Illinois Institute of Technology

jcernudagarcia@hawk.iit.edu

2nd Luke Logan

Gnosis Research Center

Illinois Institute of Technology

llogan@hawk.iit.edu

3rd Noah Lewis

Department of Computer Science and Engineering

The Ohio State University

lewis.3621@buckeyemail.osu.edu

4th Suren Byna

Department of Computer Science and Engineering

The Ohio State University

byna.1@osu.edu

5th Xian-He Sun

Gnosis Research Center

Illinois Institute of Technology

sun@iit.edu

6th Anthony Kougkas

Gnosis Research Center

Illinois Institute of Technology

akougkas@iit.edu

Abstract—Hardware is becoming increasingly heterogeneous in modern high-performance computing clusters. However, computing environments for developing tools to harness these technologies are not easily available to researchers. This work showcases the need for a new high-pace, heterogeneous I/O research cluster and presents a novel software deployment framework named *Jarvis* to manage its hardware diversity. *Jarvis* is an extensible Python framework that allows users to create *Packages* that deploy, manage, and monitor software, including complex applications (e.g., scientific simulations), support tools (e.g., Darshan, GDB), and storage systems (e.g., Lustre, DAOS). These packages can be combined to form complex deployment *Pipelines*. To ensure pipelines are portable across hardware, *Jarvis* defines a novel *Resource Graph* schema file, which is a snapshot of a cluster’s machine-specific information. This schema can be queried by *Jarvis* packages to deploy software across diverse hardware compositions with minimal user effort.

Index Terms—deployment, HPC, hardware abstraction, resource management, I/O, Python

I. EXTENDED ABSTRACT

There has been a rapid innovation in hardware for storage (e.g., node-local NVMe, PMEM), processors (TPUs, SmartNICs), and networks (Infiniband, Slingshot) to address the growing diversification of scientific and AI applications. While these technologies offer great potential to improve energy-efficiency and performance, researching methods to leverage them is difficult for several reasons. First, due to high costs, many new technologies (e.g., Compute Express Link) are not widely available, limiting research throughput. Second, hardware acquisition is often tied to building entirely new clusters, increasing the time between proposal and implementation. Third, the need for reliable storage in production clusters hampers the deployment and testing of new software systems at scale. Finally, researchers are often restricted from administrative control, limiting the ability to test advanced I/O tools or modify network and storage configurations for comprehensive experiments.

To address these problems, we are currently developing a new I/O cluster that overcomes four essential challenges: First, it must support the rapid installation of new I/O hardware; Second, it should embrace heterogeneity by offering diverse hardware combinations across nodes. Third, it must grant researchers full administrative access to all devices, allowing them to deploy, modify, and test novel technologies without the

restrictions imposed by typical system administration. Fourth, it should support rapid deployment of software over this diverse hardware. This combination of features would enable faster experimentation and innovation in HPC environments.

However, rapidly deploying and testing software in this highly heterogeneous environment remains difficult. First, applications are oftentimes complex and require expertise to deploy. They tend to have limited documentation, large parameter spaces, limited input verification, and many environment variables. Second, experiments compound this complexity, requiring several applications to be configured and orchestrated (e.g., first deploy a storage system and then a simulation storing data there). Due to the diversity of configuration, experiments are either conducted manually or with one-off bash scripts, limiting reproducibility and portability. Third, the hardware composition of the cluster is dynamic, requiring further changes to configuration to ensure compatibility and use of available hardware.

To address these problems, we propose **Jarvis**, which is a software deployment framework designed specifically for heterogeneous HPC environments. To reduce the complexity of application configuration, *Jarvis* allows users to create custom deployment **packages**, which expose only relevant parameters to users and provide procedures to automatically generate application-specific configuration files, set necessary environment variables, verify input parameters in addition to executing, terminating, and debugging the application. Packages can include various software such as storage systems (e.g. DAOS, Hermes), debugging tools (e.g. Darshan, AddressSanitizer), and workflows (e.g. Gray-Scott, WRF, DeepDriveMD, CM1).

These packages can be combined to form **pipelines** representing end-to-end deployment workflows. Pipelines simplify the orchestration of multiple stages of deployment, testing, and monitoring, automating repetitive tasks and ensuring that deployments scale efficiently across different environments. Additionally, *Jarvis* integrates with popular job schedulers such as SLURM and PBS, ensuring *Jarvis* can manage job allocations in multiple HPC clusters.

To ensure that packages are portable across machine architectures, *Jarvis* introduces the **Resource Graph**, which provides a unified view of the underlying hardware details of a cluster, such as block devices (e.g., capacity, location,

bandwidth, and latency) and networks (e.g., IP address, protocol). Jarvis packages can query this resource graph to automatically generate machine-optimized application configurations without requiring code modification, increasing flexibility and reducing administrative overhead.

We aim to leverage community-driven feedback on Jarvis to help us comprehensively address software deployment issues across machines. Jarvis has been tested across multiple clusters, from small university clusters to large-scale national labs, including Argonne, Pacific Northwest, Lawrence Livermore, and Sandia National Laboratories. However, with each new cluster, some new functionality was added to support their scientist's unique requirements. Jarvis is publicly available on GitHub¹ with a more than 20 packages and resources graphs for a few supercomputers.

¹ <https://github.com/grc-iit/jarvis-cd>