Utilizing Distributed Heterogeneous Computing with PanDA in ATLAS

Tadashi Maeno[1,*], *Aleksandr* Alekseev[2], *Fernando Harald* Barreiro Megino[2], *Kaushik* De[2], *Wen* Guan[1], *Edward* Karavakis[1], *Alexei* Klimentov[1], *Tatiana* Korchuganova[3], *FaHui* Lin[2], *Paul* Nilsson[1], *Torre* Wenaus[1], *Zhaoyu* Yang[1], and *Xin* Zhao[1]

[1]Brookhaven National Laboratory, Upton, NY, USA
[2]University of Texas at Arlington, Arlington, TX, USA
[3]University of Pittsburgh, Pittsburgh, PA, USA

Abstract. In recent years, advanced and complex analysis workflows have gained increasing importance in the ATLAS experiment at CERN, one of the large scientific experiments at LHC. Support for such workflows has allowed users to exploit remote computing resources and service providers distributed worldwide, overcoming limitations on local resources and services. The spectrum of computing options keeps increasing across the Worldwide LHC Computing Grid (WLCG), volunteer computing, high-performance computing, commercial clouds, and emerging service levels like Platform-as-a-Service (PaaS), Container-as-a-Service (CaaS) and Function-as-a-Service (FaaS), each one providing new advantages and constraints. Users can significantly benefit from these providers, but at the same time, it is cumbersome to deal with multiple providers, even in a single analysis workflow with fine-grained requirements coming from their applications' nature and characteristics. In this paper, we will first highlight issues in geographically-distributed heterogeneous computing, such as the insulation of users from the complexities of dealing with remote providers, smart workload routing, complex resource provisioning, seamless execution of advanced workflows, workflow description, pseudo-interactive analysis, and integration of PaaS, CaaS, and FaaS providers. We will also outline solutions developed in ATLAS with the Production and Distributed Analysis (PanDA) system and future challenges for LHC Run4.

1 Introduction

In recent years, the ATLAS experiment [1], one of the large scientific experiments at LHC [2], has placed increasing importance on advanced and complex analysis workflows. These workflows have proven invaluable in overcoming limitations on local resources and services by enabling users to leverage remote computing resources and service providers distributed worldwide. The spectrum of computing options available in ATLAS has also expanded significantly, encompassing the Worldwide LHC Computing Grid (WLCG) [3], volunteer

[*]e-mail: tmaeno@bnl.gov

computing, high-performance computing (HPC), commercial clouds, and emerging service levels such as Platform-as-a-Service (PaaS), Container-as-a-Service (CaaS), and Function-as-a-Service (FaaS). Each resource and service offers unique advantages and constraints, and users can benefit significantly from utilizing them. However, working with diverse resource and service providers spread across different locations can be difficult, especially when trying to incorporate specific detailed requirements into an analysis workflow that aligns with the unique nature and characteristics of their applications.

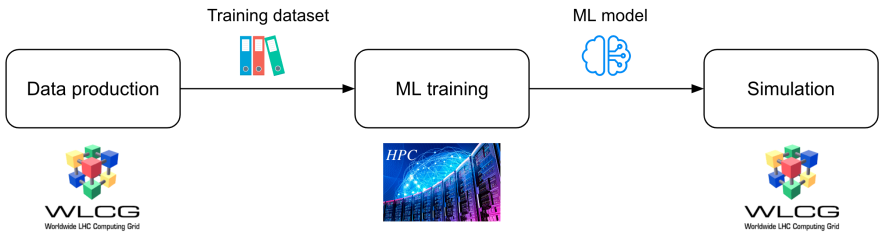

Figure 1. A simple linear workflow involving multiple providers, generating Monte-Carlo (MC) samples with a Machine Learning-based (ML-based) detector simulation.

Figure 1 shows a simple linear workflow involving multiple providers, generating Monte-Carlo (MC) samples with a Machine Learning-based (ML-based) detector simulation. The workflow comprises three tasks:

1. In the first task, training datasets are produced on WLCG.
2. The second task involves training an ML model using the training datasets.
3. Finally, the third task executes MC production utilizing the ML model again on the WLGC grid.

Each task presents different resource and service requirements. The first and third tasks are traditional High Energy Physics (HEP) workloads, which are CPU/IO intensive and highly compatible with the resources provided by WLCG. On the other hand, the workload in the second task necessitates extensive matrix multiplication and convolution, making it suitable for GPU resources available in HPC, commercial cloud, or PaaS/FaaS environments.

We present various challenges associated with geographically-distributed heterogeneous computing, including

- insulating users from the complexities of managing remote resources and services,
- smart workload routing,
- complex resource provisioning,
- seamless execution of advanced workflows,
- workflow description,
- pseudo-interactive analysis, and
- integration of PaaS, FaaS and CaaS providers.

We also outline the solutions developed in ATLAS, with the Production and Distributed Analysis (PanDA) system [4] both in the present and in preparation for LHC Run4.

2 Challenges in distributed heterogeneous computing

2.1 Insulating users from details of remote resources and services

The goal is to insulate users from the complexities of dealing with various remote resources and services, while maintaining transparency and diagnostic capabilities.

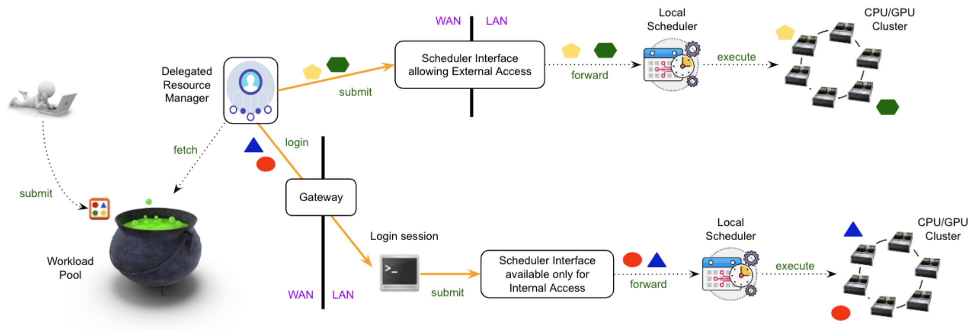

Figure 2. A centralized approach to insulate users from the complexities of dealing with various remote resources and services, combining a central workload pool with delegated resource access.

The PanDA system combines a central workload pool with delegated resource access, as shown in Figure 2. Users submit their workflows to the central workload pool, which then decomposes them into individual tasks. This decomposition enables efficient and fine-grained processing, optimizing the overall execution of the workflows. Delegated resource managers access remote resources and services on behalf of users, utilizing either common credentials or users' own credentials. When workload schedulers in the remote provider allow external access, the delegated resource managers establish interaction with them over the Wide Area Network (WAN) utilizing the specific protocol or access method for each provider. However, in cases where the workload schedulers forbid external access, the delegated resource managers access the providers' Local Area Network (LAN) by logging in through gateway services like sshd and Jupyter Hub, enabling interaction with the schedulers in the LAN environment. By delegating resource and service access, users are relieved from the complexities associated with direct resource and service management, allowing focus on their analysis.

The approach offers further advantages in addition to the user's insulation: Fair-share and priorities are centrally managed among users, which ensures equitable resource allocation and fairness across multiple users. Also, workloads can be routed intelligently using a central knowledge of resource and service characteristics, such as performance, availability, and cost, as well as fine-grained requirements of tasks.

2.2 Smart workload routing

Tasks should be assigned to resources and services by considering their requirements and specifications. The aim is to achieve an optimal assignment so that each task is matched with the most appropriate resources and services based on their specific needs, capabilities, and characteristics, maximizing efficiency and effectiveness in the overall workflow execution.

The workflow description incorporates specific requirements for each task to define its needs precisely. The system utilizes a matching algorithm to select providers that align with the specified resource and service requirements. This selection process takes into account published and real-time information about providers, such as hardware specifications, service descriptions, current slot availability, data locality, allocation status, costs, electricity prices, and carbon footprint. It is crucial that resource providers maintain up-to-date information to avoid misleading the selection process.

One caveat is users may inadvertently specify incorrect resource requirements, e.g. overestimating or underestimating RAM requirements. This issue is addressed through a two-phased dynamic correction of the requirements: In the initial phase, a small subset of the task is executed to assess the actual resource usage. Based on the resource usage observed during this phase, the system dynamically corrects the requirements before processing the remaining subset. This correction comes into play when tasks can be partitioned for partial execution, such as traditional HEP workloads.

2.3 Complex resource provisioning

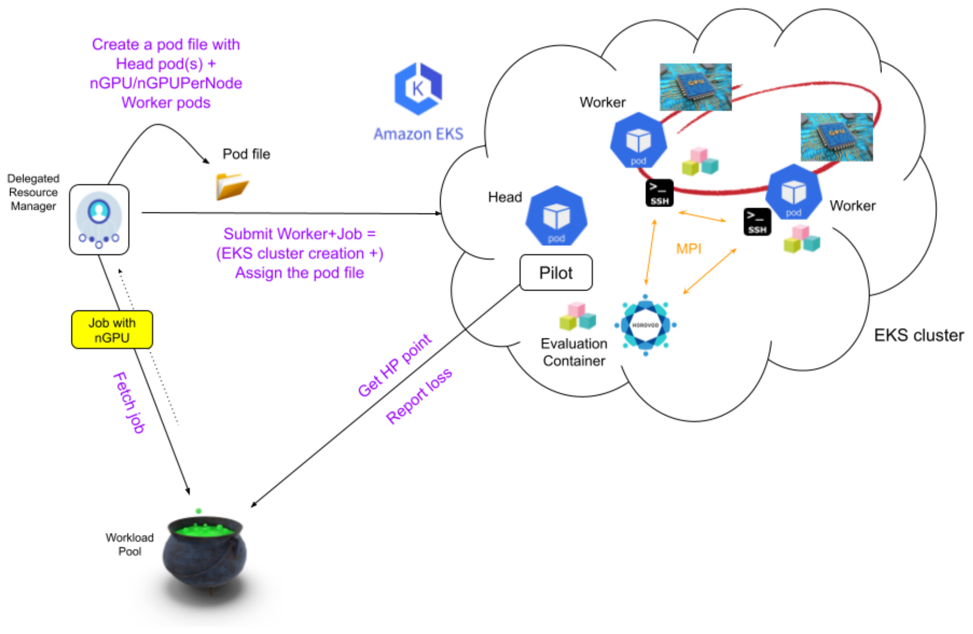

Figure 3. A resource provisioning scheme for elastic distributed ML training on Amazon EKS.

Complex resource provisioning is necessary to support emerging workloads. Figure 3 shows a resource provisioning scheme for elastic distributed ML training on Amazon EKS [5]. This scheme involves launching a hybrid cluster with both CPU and GPU instances to facilitate elastic training with Horovod [6]. The hybrid cluster comprises a head pod running on an on-demand CPU-only instance, while the worker pods are deployed on spot GPU instances. The utilization of spot instances for worker pods is advantageous due to their cost-effectiveness compared to on-demand instances.

During a training session, the head pod remains persistent, while worker pods may come and go based on the availability of GPU instances in the spot market. Consequently, the number of GPU pods in the cluster is subject to change. Horovod allows the cluster to dynamically scale up or down the number of workers without requiring a restart or resume from checkpoints stored in durable storage. This flexibility enables complex resource provisioning and contributes to cost-saving measures.

2.4 Workflow description

As discussed earlier, it is crucial to accurately describe each task's resource and service requirements. However, describing complex workflows is not straightforward for several reasons. For instance, there is no common language that spans across ML training on HPC, MC production on WLCG, and analysis on a cloud-based analysis platform. Each task may involve different software applications, making it difficult to establish a standardized framework. Furthermore, a sophisticated language syntax is required to uniformly specify resource and service requirements that can be translated into diverse hardware specifications and service descriptions across various providers. Exposing provider-specific terminology and formats to users should be avoided. In addition, users need to explicitly specify parallelizable or critical sections within the workflow and define the relationships among tasks, unless the tasks are embarrassingly parallel and can be described declaratively.

Two languages are supported, offering intuitive interfaces and standardized approaches to simplify the description of complex workflows. Users can describe workflows using directed acyclic graphs with either YAML+CWL [7] or Python+Snakemake [8]. These languages provide a structured syntax capturing the dependencies and relationships between tasks in the workflow and resource requirements of the individual tasks. PanDA monitoring developers are actively developing a graphical user interface (GUI) to enhance the monitoring system, transforming it into an interactive platform in several months. The GUI will provide a friendly interface that abstracts away the technical complexities, enabling users to interact with and manage the workflows more easily.

2.5 Seamless execution of advanced workflows

Active Learning (AL) [9] employs iterative regression on limit setting to increase analysis efficiency. It deviates from the traditional method of a brute-force grid/random search. Instead, it incorporates a search session consisting of multiple iterations. In each iteration, the search space for the next iteration is defined based on the results obtained from previous iterations. This iterative process allows for a more focused exploration of the parameter space, homing in on regions of interest.

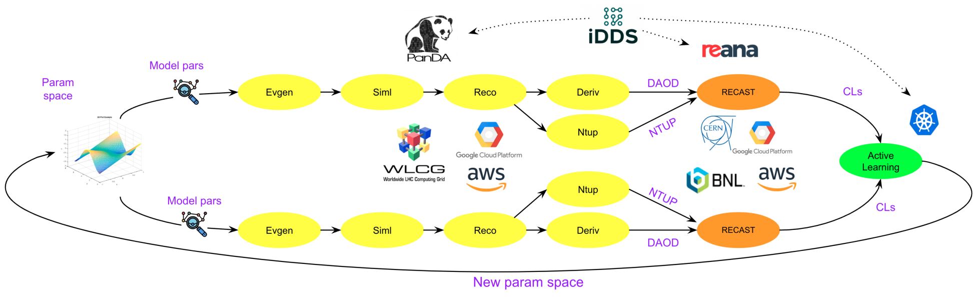

Figure 4. The dark-Z search [10, 11] workflow comprises parallel execution of production pipelines on the WLGC grid, calculation of confidence limits on REANA [12], and definition of the search space for the next iteration.

This approach is implemented as a single workflow with multiple loops, encompassing various tasks distributed across different resource and service providers. The transitions between providers are fully automated, requiring no human intervention. The AL-based analysis is specifically applied to the mono-Hbb exclusion limit reinterpretation [13], dark-Z search [10, 11], and heavy Higgs analysis [11] in the ATLAS experiment. By leveraging AL techniques, these analyses aim to enhance the efficiency and effectiveness of the search. Figure 4 shows the workflow for the dark-Z search, which comprises parallel execution of production pipelines on the WLGC grid, calculation of confidence limits on REANA [12], and definition of the search space for the next iteration. The intelligent Data Delivery Service (iDDS) [14] is a workflow execution engine playing a crucial role in executing such complex workflows. iDDS utilizes workflow descriptions and dependencies to orchestrate the execution of tasks, ensuring proper sequencing and orchestration.

2.6 Pseudo-interactive analysis

Making distributed analysis more interactive has emerged as a significant focus within the ATLAS community. Interactivity is a system's ability to respond promptly to user actions, which involves more than just an interactive user interface. Users may feel a lack of interactivity when combining an interactive user interface with a slow processing backend. For instance, the overall user experience could be hindered when running a large payload synchronously on a sluggish backend through an interactive Jupyter notebook. In this context, the response time of the processing backend holds greater importance. The primary resources utilized by ATLAS are provided through WLCG. These resources operate asynchronously and hold inherent processing latencies, making it challenging to achieve interactivity on top of them.

A pseudo-interactive system is currently under development in PanDA, to achieve a reasonable response time and provide an interactive user experience for processing complex workflows, while still leveraging asynchronous resources distributed geographically. This system incorporates smart workload routing to minimize processing latencies in various processing resources. Additionally, it encompasses an interactive user interface, and real-time streaming of the system and workload logging to keep users updated.

The pseudo-interactive system is a centralized one-stop service that enables the utilization of diverse providers and features all the advantages described earlier, such as users' insulation from the complexities of remote providers, central accounting, fair share, and portability. However, there are certain disadvantages to consider. Primarily, remote resources typically exhibit larger latencies compared to local resources, which could imply an unavoidable ceiling of system performance. Moreover, the development efforts involved in creating a sophisticated interface and monitoring are amplified due to the lack of direct access to the underlying computing resources. Nevertheless, the development of the pseudo-interactive system is actively addressing those issues, forming a part of the effort to transform the monitoring system into an interactive platform.

2.7 PaaS, FaaS, and CaaS integration

PaaS and FaaS have their strengths and limitations. While they may lack versatility, they offer significant computational power for suitable workloads, such as those in the AI/ML field.

Figure 5 shows the integration scheme of PaaS, FaaS, and CaaS into the PanDA system, highlighting the use of funcX [15] as an example. funcX is a high-performance FaaS system to orchestrate scientific workloads across diverse resources. A user submits a workflow comprising two tasks: a HEP data production task and an AI/ML task with Parsl [16]. funcX acts as a gateway to interface with a HPC center, enabling seamless routing of Parsl-based payloads to the underlying resources. The integration employs two plugins to facilitate communication and coordination with relevant resources. One plugin establishes

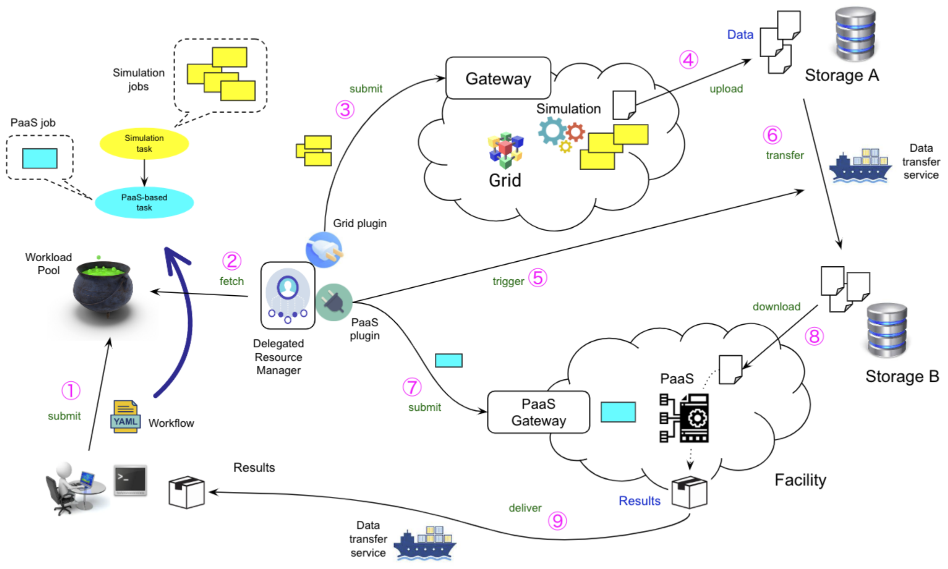

Figure 5. PassS, FaaS and CaaS integration scheme. funcX is used as an example.

a connection with WLCG, while the other is responsible for interacting with funcX. This integration ensures a cohesive and efficient execution of tasks in the workflow, streamlining the overall computational process.

3 Conclusions

The overarching objective of distributed heterogeneous computing is to streamline users' workflows by harnessing the available large-scale resources. This necessitates abstracting the complexities associated with geographically distributed resources and services, while providing transparency, diagnostics, and a near-real-time interactive interface. The landscape of resource and service providers is diverse, as are the emerging workflows. Supporting all possible combinations of resources, services, and workflows poses a significant challenge.

However, progress in this domain is fueled by real-world use cases, which the PanDA system and its development programs are addressing. While many issues have been addressed, numerous challenges remain. Ongoing efforts are dedicated to broadening the scope of user applications and expanding the range of supported resources and services both in the present and in preparation for LHC Run4.

Acknowledgments

This manuscript has been authored by employees of Brookhaven Science Associates, LLC under Contract No. DE-SC0012704 with the U.S. Department of Energy. The publisher by accepting the manuscript for publication acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government

purposes.

References

[1] ATLAS Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, J. Inst. **3**, S08003 (2008)

[2] L. Evans and P. Bryant (editors), *LHC Machine*, J. Inst. **3**, S08001 (2008)

[3] *WLCG*, https://wlcg.web.cern.ch/

[4] T. Maeno et al., *PanDA: Production and Distributed Analysis System*, Computing and Software for Big Science (to be submitted)

[5] *Amazon EKS*, https://aws.amazon.com/eks/

[6] *Horovod*, https://horovod.ai/

[7] *CWL*, https://www.commonwl.org/

[8] *Snakemake*, https://snakemake.readthedocs.io/en/stable/

[9] B. Settles, *Active Learning*, Synthesis Lectures on Artificial Intelligence and Machine Learning (Morgan Claypool Publishers, 2012)

[10] ATLAS Collaboration, *Demonstrating an active learning driven pipeline for optimised analysis reinterpretation: an extended search for Higgs bosons decaying into four-lepton final states via an intermediate dark Z boson*, https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-PHYS-PUB-2023-010/

[11] C. Weber et al., *An Active Learning application in a dark matter search with ATLAS PanDA and iDDS,* (in these proceedings)

[12] *REANA*, https://reanahub.io/

[13] ATLAS Collaboration, *Active Learning reinterpretation of an ATLAS Dark Matter search constraining a model of a dark Higgs boson decaying to two b-quarks*, https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-PHYS-PUB-2022-045/

[14] W. Guan et al., *An intelligent Data Delivery Service for and beyond the ATLAS experiment*, EPJ Web Conf. **251**, 02007 (2021)

[15] *funcX*, https://funcx.readthedocs.io/en/latest/

[16] *Parsl*, https://parsl-project.org/