Workflow Models for Heterogeneous Distributed Systems

Iacopo Colonnelli

Università degli Studi di Torino, Department of Computer Science, Corso Svizzera 185, 10149, Torino, Italy

Abstract
This article introduces a novel hybrid workflow abstraction that injects topology awareness directly into the definition of a distributed workflow model. In particular, the article briefly discusses the advantages brought by this approach to the design and orchestration of large-scale data-oriented workflows, the current level of support from state-of-the-art workflow systems, and some future research directions.

Keywords
Scientific Workflows, HPC, Cloud Computing, Distributed Computing, Hybrid Workflows

1. Hybrid workflow models

When considering data-oriented workflows, all the aspects of data management become crucial for performance optimisation, privacy preservation and security. The data locality principle, i.e., moving computation close to the data, inspired the foundational algorithms [1] and data structures [2] of modern Big Data analysis frameworks and became an unwaivable requirement of federated learning approaches [3]. On the other hand, there are scenarios in which it is worth, or even unavoidable, to transfer data between different modules of a complex application. The modular nature of modern applications and the heterogeneity in contemporary hardware resources and their features, further exacerbated by the end-to-end co-design approach [4], require Workflow Management Systems (WMSs) to support a large ecosystem of execution environments (from HPC to cloud, to the Edge), optimisation policies (performance vs. energy efficiency) and computational models (from classical to quantum). For these reasons, modern workflow models and tools need to be topology-aware, allowing an explicit mapping of workflow steps onto (families of) processing elements. This mapping can be either manual, driven by the combined experience of domain experts and computer scientists, or (semi-)automatic, using advanced learning algorithms to infer the best-suited execution environment for each step.

A hybrid workflow can be defined as a workflow whose steps can span multiple, heterogeneous, and independent computing infrastructures [5]. Each of these aspects has significant implications. Support for multiple infrastructures implies that each step must potentially target a different deployment location in charge of executing it. Locations can be heterogeneous, exposing different methods and protocols for authentication, communication, resource allocation and job...
execution. Plus, they can be independent of each other, meaning that direct communications and data transfers among them may not be allowed. A suitable model for hybrid workflows must then be composite, enclosing a specification of the workflow dependencies, a topology of the involved locations, and a mapping relation between steps and locations.

2. State of the art and future directions

Grid-native WMSs [6, 7, 8] typically support distributed workflows out of the box, providing automatic scheduling and data transfer management across multiple execution locations. However, all the orchestration aspects are delegated to external, grid-specific libraries and frameworks, limiting the spectrum of supported execution environments.

Recently, a new class of topology-aware WMSs is starting to be designed and implemented, bringing advantages in performance and costs of workflow executions on top of heterogeneous distributed environments. StreamFlow [9] augments the Common Workflow Language (CWL) [10] open standard with a topology of deployment locations and relies on a set of connectors to support several execution environments, from HPC queue managers to container orchestrators. DagOnStar [11] allows users to model hybrid workflows as pure Python scripts, scheduling each task on an HPC facility, a cloud VM, or a software container. Jupyter Workflow [12] transforms a sequential computational notebook into a hybrid workflow by treating each cell as a workflow step, semi-automatically extracting inter-cell data dependencies from the code, and mapping each cell into one or more execution locations. Mashup [13] automatically maps each workflow step onto the best-suited location, choosing between Cloud VMs and serverless platforms.

Hybrid workflows proved themselves flexible enough to efficiently model and orchestrate large-scale applications from a diverse set of domains, including bioinformatics [9, 14], large-scale scientific simulations [15, 12], and deep learning [16, 17], on top of hybrid cloud-HPC environments. Nevertheless, the syntax and semantics used to model and execute distributed workflows are still product-specific, hindering the portability and reusability of both workflow models and orchestration strategies. Hybrid workflow models [5] represent a first step toward a vendor-agnostic way to incorporate topology awareness directly in the workflow definition, and further research efforts are ongoing to distil a formal representation of hybrid workflows, enabling optimisation strategies with theoretical correctness and consistency guarantees [18]. Another promising research direction involves relying on topology information to improve the overall workflow execution plan, e.g., developing location-aware scheduling algorithms or transparently injecting streaming capabilities in file-based workflows [19].

Acknowledgments

This work has been partially supported by the ACROSS project, “HPC Big Data Artificial Intelligence cross-stack platform toward exascale,” and the EUPEX project, “European Pilot for Exascale,” which have received funding from the EuroHPC JU under grant agreements No. 955648 and 101033975, respectively. Plus, it has been partially supported by the ICSC – Centro Nazionale di Ricerca in High Performance Computing, BigData and Quantum Computing, funded by European Union – NextGenerationEU.
References


