

Workflow Models for Heterogeneous Distributed Systems

Iacopo Colonnelli¹

¹Università 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], 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


ITADATA2023: The 2nd Italian Conference on Big Data and Data Science, September 11–13, 2023, Naples, Italy

 iacopo.colonnelli@unito.it (I. Colonnelli)

 <https://alpha.di.unito.it/iacopo-colonnelli/> (I. Colonnelli)

 0000-0001-9290-2017 (I. Colonnelli)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

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

- [1] J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, in: 6th Symposium on Operating System Design and Implementation (OSDI 2004), USENIX Association, San Francisco, California, USA, 2004, pp. 137–150.
- [2] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. J. Franklin, S. Shenker, I. Stoica, Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing, in: Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, San Jose, CA, USA, April 25-27, 2012, USENIX Association, 2012, pp. 15–28.
- [3] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.
- [4] D. A. Reed, D. Gannon, J. J. Dongarra, Reinventing high performance computing: Challenges and opportunities, CoRR abs/2203.02544 (2022). doi:10.48550/arXiv.2203.02544. arXiv:2203.02544.
- [5] I. Colonnelli, Workflow models for heterogeneous distributed systems, Ph.D. thesis, Università degli Studi di Torino, 2022. doi:10.5281/zenodo.7135483.
- [6] I. J. Taylor, M. S. Shields, I. Wang, A. Harrison, The Triana workflow environment: Architecture and applications, in: Workflows for e-Science, Scientific Workflows for Grids, Springer, 2007, pp. 320–339. doi:10.1007/978-1-84628-757-2_20.
- [7] T. Fahringer, R. Prodan, R. Duan, J. Hofer, F. Nadeem, F. Nerieri, S. Podlipnig, J. Qin, M. Siddiqui, H. L. Truong, A. Villazón, M. Wiczorek, ASKALON: A development and grid computing environment for scientific workflows, in: Workflows for e-Science, Scientific Workflows for Grids, Springer, 2007, pp. 450–471. doi:10.1007/978-1-84628-757-2_27.
- [8] E. Deelman, K. Vahi, M. Rynge, R. Mayani, R. F. da Silva, G. Papadimitriou, M. Livny, The evolution of the Pegasus workflow management software, Computing in Science and Engineering 21 (2019) 22–36. doi:10.1109/MCSE.2019.2919690.
- [9] I. Colonnelli, B. Cantalupo, I. Merelli, M. Aldinucci, StreamFlow: cross-breeding cloud with HPC, IEEE Transactions on Emerging Topics in Computing 9 (2021) 1723–1737. doi:10.1109/TETC.2020.3019202.
- [10] M. R. Crusoe, S. Abeln, A. Iosup, P. Amstutz, J. Chilton, N. Tijanic, H. Ménager, S. Soiland-Reyes, C. A. Goble, Methods included: Standardizing computational reuse and portability with the Common Workflow Language, Communication of the ACM (2022). doi:10.1145/3486897.
- [11] D. D. Sánchez-Gallegos, D. D. Luccio, S. Kosta, J. L. G. Compeán, R. Montella, An efficient pattern-based approach for workflow supporting large-scale science: The DagOnStar experience, Future Generation Computer Systems 122 (2021) 187–203. doi:10.1016/J.FUTURE.2021.03.017.
- [12] I. Colonnelli, M. Aldinucci, B. Cantalupo, L. Padovani, S. Rabellino, C. Spampinato, R. Morelli, R. Di Carlo, N. Magini, C. Cavazzoni, Distributed workflows with Jupyter, Future Generation Computer Systems 128 (2022) 282–298. doi:10.1016/j.future.2021.10.007.
- [13] R. B. Roy, T. Patel, V. Gadepally, D. Tiwari, Mashup: making serverless computing

- useful for HPC workflows via hybrid execution, in: PPOPP '22: 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, ACM, 2022, pp. 46–60. doi:10.1145/3503221.3508407.
- [14] A. Mulone, S. Awad, D. Chiarugi, M. Aldinucci, Porting the variant calling pipeline for NGS data in cloud-hpc environment, in: 47th IEEE Annual Computers, Software, and Applications Conference, COMPSAC 2023, IEEE, Torino, Italy, 2023, pp. 1858–1863. doi:10.1109/COMPSAC57700.2023.00288.
- [15] R. Montella, D. Di Luccio, S. Kosta, DagOn*: Executing direct acyclic graphs as parallel jobs on anything, in: IEEE/ACM Workshop on Workflows in Support of Large-Scale Science, WORKS@SC 2018, IEEE, 2018, pp. 64–73. doi:10.1109/WORKS.2018.00012.
- [16] I. Colonnelli, B. Cantalupo, R. Esposito, M. Pennisi, C. Spampinato, M. Aldinucci, HPC application cloudification: The StreamFlow toolkit, in: 12th Workshop on Parallel Programming and Run-Time Management Techniques for Many-core Architectures and 10th Workshop on Design Tools and Architectures for Multicore Embedded Computing Platforms, PARMA-DITAM 2021, volume 88 of *OASIScs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Budapest, Hungary, 2021, pp. 5:1–5:13. doi:10.4230/OASIScs.PARMA-DITAM.2021.5.
- [17] I. Colonnelli, B. Casella, G. Mittone, Y. Arfat, B. Cantalupo, R. Esposito, A. R. Martinelli, D. Medić, M. Aldinucci, Federated learning meets HPC and cloud, in: *Astrophysics and Space Science Proceedings*, volume 60, Springer, Catania, Italy, 2023, pp. 193–199. doi:10.1007/978-3-031-34167-0_39.
- [18] D. Medić, M. Aldinucci, Towards formal model for location aware workflows, in: 47th IEEE Annual Computers, Software, and Applications Conference, COMPSAC 2023, IEEE, Torino, Italy, 2023, pp. 1864–1869. doi:10.1109/COMPSAC57700.2023.00289.
- [19] A. R. Martinelli, M. Torquati, I. Colonnelli, B. Cantalupo, M. Aldinucci, CAPIO: a middleware for transparent I/O streaming in data-intensive workflows, in: 30th IEEE International Conference on High Performance Computing, Data, and Analytics, HiPC 2023, IEEE, Goa, India, 2023.