Adapting the Common Workflow Scheduler to Argo's Workflow Model

Béla Anton Paulus

May 2024

To facilitate the analysis of big scientific data, scientific workflow management systems like Argo Workflows (Argo) and resource managers like Kubernetes are used. In typical setups, the resource manager lacks information about the workflow structure leading to suboptimal scheduling decisions. In this paper, we build on prior work by Lehmann et al. to improve scheduling in setups using Argo and Kubernetes. To achieve this, we adapt the Common Workflow Scheduler (CWS) API to Argo's workflow model, we implement the CWS API in Argo, and we extend the custom Kubernetes scheduler by Lehman et al. To evaluate our work, we conduct experiments using workflows generated by the WfCommons framework.

1 Introduction

Scientists analyze increasing amounts of data. In Remote Sensing, for instance, Earth observation satellites are generating multiple terabytes of image data per day [1]. In Genomics, sequence databases such as the Sequence Read Archive or the European Nucleotide Archive store tens of petabytes of sequencing data [2], [3].

To analyze such large amounts of data, the required analysis tasks and their interdependencies are described as workflows [4]–[7]. Typically, a workflow is defined as a Directed Acyclic Graph (DAG) consisting of vertices representing tasks and edges representing dependencies between tasks. To facilitate the execution of such analysis workflows, Scientific Workflow Management Systems (SWMS) such as Nextflow [8] and Argo Workflows[1] are used.

Commonly, workflows are executed on a compute cluster using a resource manager to distribute the work to be done over the available resources. Popular resource managers

[1] https://argo-workflows.readthedocs.io/en/latest/

include SLURM [9] and Kubernetes[2]. In typical setups, a scientist provides the SWMS with a workflow definition and input data. The SWMS parses the workflow definition and passes ready-to-run tasks to the resource manager. The resource manager in turn schedules the given tasks on free compute resources. This setup has a drawback. Since the resource manager has no information about the workflow structure, it can not make well-informed scheduling decisions [10].

Previous work by Lehmann et al. approaches this problem by proposing an API that allows the SWMS to pass additional information to the resource manager [10]. They implement the proposed API for Nextflow and Kubernetes and show that the additional information can be used to make better scheduling decisions and reduce workflow execution times.

Our work builds on the contributions by Lehmann et al. and our goal is to improve scheduling in setups using Argo Workflows (Argo) and Kubernetes. Argo is a popular open-source Kubernetes-native workflow engine. In contrast to Nextflow, Argo's workflow model allows workflows with loops. In general, the number of repetitions of a loop is only determined at runtime. This makes Argo incompatible with the current CWS API and the custom Kubernetes scheduler implemented by Lehmann et al. Consequently, our contributions are as follows: First, we adapt the CWS API and custom Kubernetes scheduler to Argo's workflow model. Second, we implement the CWS API in Argo. Finally, we conduct experiments to evaluate our work.

2 Related Work

In general, workflow scheduling relates to the problem of mapping a partially ordered set of tasks onto a set of resources and times such that the task order is preserved and the total makespan is minimal. This problem has been extensively studied and is proven to be NP-hard, so there exist no fast optimal solutions [11]. Nonetheless, scientists have come up with heuristics, like the HEFT scheduling heuristic, that provide good results within a reasonable amount of time [12].

In practice, however, workflow scheduling systems face a variety of additional challenges:

- Task execution times are often unknown in advance [13].
- In typical setups, the structure of the workflow is not known by the scheduler [10].
- The execution environment can change dynamically [14].

There has been extensive research regarding workflow scheduling and scientists have come up with solutions to some of these problems [15], [16].

[2]https://kubernetes.io/

Regarding the problem that the scheduler has no information about the workflow structure, Lehmann et al. propose the Common Workflow Scheduler (CWS) API, a REST API that allows the SWMS to pass additional information to the scheduler [10]. The authors implement the CWS API in a custom Kubernetes scheduler that uses the additional information to improve scheduling decisions. It does so by applying scheduling heuristics, which do not need knowledge of task execution times in advance and can handle dynamic workflows.

3 Goals and Evaluation

This work aims to improve the scheduling in setups using Argo and Kubernetes, by leveraging previous work by Lehmann et al. [10]. To connect Argo to the custom Kubernetes scheduler, we take the following steps:

First, we investigate ways to reconcile the workflow model of Argo and the CWS. In contrast to the workflow model of the CWS, the workflow model of Argo includes loops. Although it might be theoretically possible to schedule workflows with loops using the CWS API and the custom Kubernetes scheduler by abusing their support for dynamic workflows, it would go against the spirit of providing the scheduler with as much information as possible. Second, we build a prototype. This includes extracting Argo's internal workflow representation, translating it, and sending it to the CWS, as well as implementing scheduling heuristics for workflows with loops in the custom Kubernetes scheduler. Third, we evaluate our prototype with synthetic workflows generated by the WfCommons framework [17]. We compare different approaches and scheduling heuristics to show how well setups using Argo and Kubernetes benefit from the CWS API and the custom Kubernetes scheduler.

Generally, we expect similar results to what Lehmann et al. achieved in setups using Nextflow and Kubernetes. Our main contributions will be: a) Adapting the CWS API and custom Kubernetes scheduler to Argo, including scheduling heuristics for workflows with loops. b) Implementing the CWS API in Argo. c) Show how well setups using Argo and Kubernetes can benefit from the CWS approach.

References

[1] M. Sudmanns, D. Tiede, H. Augustin, and S. Lang, "Assessing global sentinel-2 coverage dynamics and data availability for operational earth observation (EO) applications using the EO-compass," *International Journal of Digital Earth*, vol. 13, no. 7, pp. 768–784, Feb. 5, 2019. DOI: 10.1080/17538947.2019.1572799.

[2] R. Leinonen, H. Sugawara, M. Shumway, and on behalf of the International Nucleotide Sequence Database Collaboration, "The sequence read archive," *Nucleic Acids Research*, vol. 39, pp. D19–D21, suppl_1 Jan. 1, 2011. DOI: 10.1093/nar/gkq1019.

[3] J. Burgin, A. Ahamed, C. Cummins, R. Devraj, K. Gueye, *et al.*, "The european nucleotide archive in 2022," *Nucleic Acids Research*, vol. 51, pp. D121–D125, D1 Jan. 6, 2023. DOI: 10.1093/nar/gkac1051.

[4] M. Garcia, S. Juhos, M. Larsson, P. Olason, M. Martin, *et al.*, "Sarek: A portable workflow for whole-genome sequencing analysis of germline and somatic variants," *F1000Research*, vol. 9, p. 63, 2020. DOI: 10.1101/316976.

[5] J. Yates, T. Lamnidis, M. Borry, A. Valtueña, Z. Fagernäs, *et al.*, "Reproducible, portable, and efficient ancient genome reconstruction with nf-core/eager," *PeerJ*, vol. 9, p. 10 947, 2021. DOI: 10.1101/2020.06.11.145615.

[6] S. Groth, F. Fichtner, M. Wieland, N. Mandery, T. Riedlinger, *et al.*, "Enabling global processing of reference water products for flood mapping using kubernetes and STAC," presented at the IGARSS 2023 - 2023 IEEE International Geoscience and Remote Sensing Symposium, Jul. 2023. DOI: 10.1109/IGARSS52108.2023.10283106.

[7] N. Maxwell, J. Papadakis, A. Ilie, S. C. Stutts, J. Bates, *et al.*, "Country-scale photogrammetry and true orthorectification using very-high-resolution (VHR) multi-view satellite imagery," presented at the IGARSS 2023 - 2023 IEEE International Geoscience and Remote Sensing Symposium, Jul. 2023. DOI: 10.1109/IGARSS52108.2023.10281555.

[8] P. Di Tommaso, M. Chatzou, E. W. Floden, P. P. Barja, E. Palumbo, *et al.*, "Nextflow enables reproducible computational workflows," *Nature Biotechnology*, vol. 35, no. 4, pp. 316–319, Apr. 2017. DOI: 10.1038/nbt.3820.

[9] A. Yoo, M. Jette, and M. Grondona, "SLURM: Simple linux utility for resource management," in *Job Scheduling Strategies for Parallel Processing*, Springer Berlin Heidelberg, 2003, pp. 44–60. DOI: 10.1007/10968987_3.

[10] F. Lehmann, J. Bader, F. Tschirpke, L. Thamsen, and U. Leser, "How workflow engines should talk to resource managers: A proposal for a common workflow scheduling interface," presented at the 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid), May 2023. DOI: 10.1109/CCGrid57682.2023.00025.

[11] H. El-Rewini, H. Ali, and T. Lewis, "Task scheduling in multiprocessing systems," *Computer*, vol. 28, no. 12, pp. 27–37, Dec. 1995. DOI: 10.1109/2.476197.

[12] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260–274, Mar. 2002. DOI: 10.1109/71.993206.

[13] J. Bader, F. Lehmann, L. Thamsen, U. Leser, and O. Kao, "Lotaru: Locally predicting workflow task runtimes for resource management on heterogeneous infrastructures," *Future Generation Computer Systems*, vol. 150, pp. 171–185, Jan. 2024. DOI: 10.1016/j.future.2023.08.022.

[14] Z. Yu and W. Shi, "An adaptive rescheduling strategy for grid workflow applications," presented at the 2007 IEEE International Parallel and Distributed Processing Symposium, Mar. 2007. DOI: 10.1109/IPDPS.2007.370305.

[15] W. Khallouli and J. Huang, "Cluster resource scheduling in cloud computing: Literature review and research challenges," *The Journal of Supercomputing*, vol. 78, no. 5, pp. 6898–6943, Apr. 1, 2022. DOI: 10.1007/s11227-021-04138-z.

[16] F. Wu, Q. Wu, and Y. Tan, "Workflow scheduling in cloud: A survey," *The Journal of Supercomputing*, vol. 71, no. 9, pp. 3373–3418, Sep. 1, 2015. DOI: 10.1007/s11227-015-1438-4.

[17] T. Coleman, H. Casanova, L. Pottier, M. Kaushik, E. Deelman, *et al.*, "WfCommons: A framework for enabling scientific workflow research and development," *Future Generation Computer Systems*, vol. 128, pp. 16–27, Mar. 1, 2022. DOI: 10.1016/j.future.2021.09.043.