How Workflow Management Systems address Kubernetes - A Taxonomy

Exposé

Lennart Seiffer

November 9, 2021

Abstract

Data analysis workflows are an integral part of modern research in the natural sciences. Yet, several issues are slowing down the process of creating and sharing workflows. One issue is the lack of portability and reproducibility of workflows across different environments. Several workflow management systems aim to achieve these properties by abstracting their execution over various infrastructures. One common attempt to enable this abstraction is the use of resource managers like Kubernetes. Kubernetes manages Cloud resources in a uniform way, simplifying the migration process of workflows across different Clouds and datacenters.

In this thesis, we give an overview of the utilization of Kubernetes by multiple state-of-the-art workflow management systems. We aim to categorize the systems, based on the way they address Kubernetes. To discuss aspects of workflow implementation, performance, and distribution of responsibilities, we adapt a simplified word count workflow to all systems. For evaluation, we run each workflow in a Kubernetes cluster hosted in the Cloud, measure the execution time, and track the task scheduling decisions. Based on these results, we discuss advantages and disadvantages of the approaches.

1 Introduction

Nowadays, many research departments in the natural sciences heavily rely on data analysis workflows (DAW), processing large data volumes [13]. Various approaches for creating, managing, and executing DAWs exist, ranging from custom-built tools to complex workflow management systems (WMS). Researchers dealing with DAWs face several challenges, such as:

- Each research lab has its own infrastructure, and transferring a DAW from one to another is time-consuming and error-prone.

- Researchers creating and managing the DAWs are often no computer scientists and need to spend significant time to learn the tools for creating pipelines.
- Adapting a DAW from one WMS to another may require fundamental adjustments.

For these reasons, research teams can hardly reuse other institutions' workflows to reproduce their results. Modern WMSs aim to achieve portability and reusability of data analysis workflows by abstracting their execution environment, while supporting a range of different resource managers and executors. A common resource manager that we examine in this paper is Kubernetes, a state-of-the-art, Cloud-native infrastructure orchestration system [21].

This bachelor thesis aims to give an overview of how existing WMSs address Kubernetes to enable abstract DAW definitions and portable workflows. We have limited the scope of this thesis to Kubernetes to enable a more direct comparison of WMSs than considering the complete range of supported resource managers would allow. We will focus on the Cloud as the execution infrastructure.

For the comparison, we consider the following open-source workflow management systems, which support DAW execution via Kubernetes:

- Apache Airflow [7]
- Apache OODT [6]
- Argo [3]
- Arvados [18]
- Galaxy [19]
- Nextflow [11]
- Pegasus [25]
- Snakemake [9]
- Toil [10]

2 Related work

There are several overviews of WMSs that highlight different aspects of these systems, such as the survey of Barker and Hemert [1], who present various early scientific WMSs and discuss key aspects of future research with scientific workflows. Also, Mork et al. [17] survey WMSs, how they operate on research Clouds, and how this has been utilized in papers across various domains. An overview of more recent WMSs, which also utilize Kubernetes, can be found in Spjuth et al. [20]. Bux and Leser [2] and Liu et al. [14] give an overview of parallelization techniques in WMSs and how they are implemented in specific systems. Furthermore, Fjukstad and Bongo [5] investigate scalability aspects of WMSs in the bioinformatics realm, also mentioning Toil. In addition, Wratten et al. [26] and Leser et al. [13] discuss general requirements for WMSs on a broader scope.

Several articles examine specific WMSs, such as Galaxy [16], Toil [23], Nextflow [4], and Snakemake [15]. The workflow management systems Apache Airflow, Apache OODT, Arvados, and Snakemake do not appear in the overview papers mentioned above. In contrast, Galaxy and Pegasus have been discussed extensively in older literature, while Nextflow is often mentioned in more recent papers.

Various papers specifically focus on requirements for running DAWs in the Cloud and provide example migrations. Vöckler et al. [24] demonstrate the deployment of a Pegasus workflow from the astronomy domain onto various Clouds simultaneously. Similarly, Zhao et al. [27] discuss general requirements for running workflows in the Cloud and provide a migration of the WMS Swift into the Cloud. Klop [8] investigates scheduling of workflow jobs over Kubernetes and Docker Swarm. Moreover, Lehmann et al. [12] discuss the adaption process of tailor-made data analysis workflows to workflow management systems by the example of porting a DAW for earth observation to Nextflow. Bux and Leser [2] also mention operational models and cost aspects of deploying WMSs in the Cloud.

We have found little material that discusses the implementation of Kubernetes executors of specific WMSs in more detail, this is where this thesis can establish a foundation of knowledge. It can also be considered as an update to the overviews of state-of-the-art WMSs. However, this kind of information is outdated quite fast, as the DAW management and execution field is still evolving.

3 Goals

In this thesis we aim to give an overview of state-of-the-art workflow management systems that utilize the resource manager Kubernetes. We plan to investigate how these systems are making use of resource managers to achieve portability and reproducibility of workflows. More precisely, we will analyze the distribution of responsibilities between Kubernetes resources for DAW execution, the means used to transfer data between jobs, and the scheduling of the jobs.

Based on our findings, we aim to categorize the different workflow systems by their approaches to use Kubernetes. In addition, we compare the systems in terms of performance and highlight under which circumstances one system may be preferable over the other.

As the number of existing systems supporting execution via Kubernetes exceeds the scope of a Bachelor Thesis, we have limited ourselves to at most nine systems listed above. We may adjust the exact number of systems depending on the amount of work and available information.

In the first chapters of the thesis, we plan to introduce general concepts, such as data analysis workflows, workflow management systems, Cloud, and Kubernetes. We will give a brief overview of current research concerning DAW execution in the Cloud, focusing on Kubernetes. To convey the context for assessing the analyzed WMSs, we will highlight requirements of WMSs, that are associated with their infrastructure.

4 Methodology

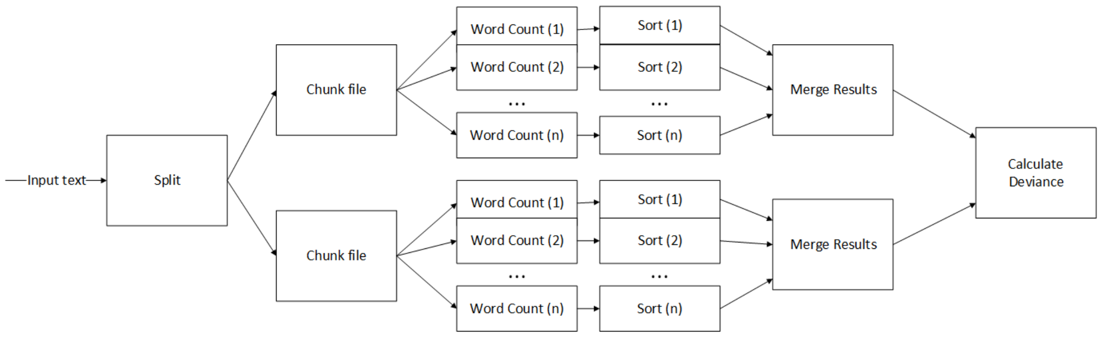

Figure 1: A distributed workflow

In the following chapter, we describe the planned proceeding of writing the thesis.

We will gather information about how the WMSs utilize Kubernetes from various sources. Specifically, we expect the WMS documentation to give a brief overview of the deployment and execution procedure. More in-depth information will be taken from papers, if available, or derived from the WMS's source code. Additionally, monitoring the execution of the evaluation-workflow may yield further details.

To evaluate the systems' performance and demonstrate differences in DAW specification, we will create a simple test workflow that we adapt to each WMS (Figure 1). The evaluation-workflow splits a text file into two equally sized pieces and counts word frequencies for each of those. Furthermore, it sorts the resulting word count lists and compares them, returning the word with the largest deviance among the two lists. To make the workflow scalable, the text pieces get chunked and passed to replicas of the counting and sorting jobs. Thus, the jobs can be distributed among multiple machines, so that they are processed in parallel.

We will test the workflow on large amounts of public data, like a dump of the German Wikipedia [22], that is publicly available in XML format and has a size of about 23 GB. The data will be pre-processed, so that the input of the chunking job is unformatted text without special characters. However, the pre-processing is not part of the workflow itself.

The amount of test data used should ensure that the running time of the workflow is not too short. This should reduce the impact of little variances and side effects on the execution time. Concerning resources, we will use 5-10 virtual machines with at least two cores. With this configuration, we want to ensure that differences in the use of the resource manager will have a significant impact on the execution time.

We plan to execute the DAW in an Azure Kubernetes Service (AKS) cluster. For this purpose, we will use a trial Azure subscription with limited budget. However, the choice of the Cloud provider could still change during the process, as long as all workflow runs are executed in the same Cloud environment. For monitoring, we will either use services of the Cloud provider or deploy a monitoring stack alongside the WMS in the Kubernetes cluster. The requirement is, that the evaluation procedure for each WMS is the same. For each system, we will run the workflow three times and take the median for comparison. This should eliminate strong deviations, which could appear due to over allocation of Cloud resources. However, we still need to decide on the exact method for measuring the execution time of a DAW execution.

References

[1] Adam Barker and Jano van Hemert. Scientific workflow: A survey and research directions. In *Parallel Processing and Applied Mathematics*, pages 769–776, 2008.

[2] Marc Bux and Ulf Leser. Parallelization in scientific workflow management systems. *arXiv preprint arXiv:1303.7195*, 2013.

[3] Cloud Native Computing Foundation (CNCF). Argo workflows - the workflow engine for kubernetes, September 2021. URL: `https://argoproj.github.io/argo-workflows/`.

[4] Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo, and Cedric Notredame. Nextflow enables reproducible computational workflows. *Nature biotechnology*, 35(4):316–319, 2017.

[5] Bjørn Fjukstad and Lars Ailo Bongo. A review of scalable bioinformatics pipelines. *Data Science and Engineering*, 2(3):245–251, 2017.

[6] The Apache Foundation. Apache OODT - distributed data management, September 2021. URL: `https://oodt.apache.org/`.

[7] The Apache Software Foundation. Apache airflow, September 2021. URL: `https://airflow.apache.org/`.

[8] Isaac Klop. Containerized workflow scheduling. 2018.

[9] Johannes Köster. Snakemake - a framework for reproducible data analysis, September 2021. URL: `https://snakemake.github.io/`.

[10] UCSC Computational Genomics Lab. Toil, September 2021. URL: `http://toil.ucsc-cgl.org/`.

[11] Seqera Labs. Nextflow - a DSL for parallel and scalable computational pipelines, September 2021. URL: `https://www.nextflow.io/`.

[12] Fabian Lehmann, David Frantz, Sören Becker, Ulf Leser, and Patrick Hostert. FORCE on nextflow: Scalable analysis of earth observation data on commodity clusters. *1st Int. Workshop on Complex Data Challenges in Earth Observation*, 2021.

[13] Ulf Leser, Marcus Hilbrich, Claudia Draxl, Peter Eisert, Lars Grunske, Patrick Hostert, Dagmar Kainmüller, Odej Kao, Birte Kehr, Timo Kehrer, Christoph Koch, Volker Markl, Henning Meyerhenke, Tilmann Rabl, Alexander Reinefeld, Knut Reinert, Kerstin Ritter, Björn Scheuermann, Florian Schintke, Nicole Schweikardt, and Matthias Weidlich. The collaborative research center FONDA. *Datenbank Spektrum*, 2021.

[14] Ji Liu, Esther Pacitti, Patrick Valduriez, and Marta Mattoso. A survey of data-intensive scientific workflow management. *Journal of Grid Computing*, 13(4):457–493, 2015.

[15] Felix Mölder, Kim Philipp Jablonski, Brice Letcher, Michael B Hall, Christopher H Tomkins-Tinch, Vanessa Sochat, Jan Forster, Soohyun Lee, Sven O Twardziok, Alexander Kanitz, et al. Sustainable data analysis with snakemake. *F1000Research*, 10, 2021.

[16] Pablo Moreno, Luca Pireddu, Pierrick Roger, Nuwan Goonasekera, Enis Afgan, Marius van den Beek, Sijin He, Anders Larsson, Daniel Schober, Christoph Ruttkies, et al. Galaxy-kubernetes integration: scaling bioinformatics workflows in the cloud. *Preprint*, 2018.

[17] Ryan Mork, Paul Martin, and Zhiming Zhao. Contemporary challenges for data-intensive scientific workflow management systems. In *Proceedings of the 10th Workshop on Workflows in Support of Large-Scale Science*, pages 1–11, 2015.

[18] Arvados Project. Arvados, September 2021. URL: `https://arvados.org/`.

[19] The Galaxy Project. Galaxy community hub, September 2021. URL: `https://galaxyproject.org/`.

[20] Ola Spjuth, Marco Capuccini, Matteo Carone, Anders Larsson, Wesley Schaal, Jon Ander Novella, Oliver Stein, Morgan Ekmefjord, Paolo Di Tommaso, Evan Floden, et al. Approaches for containerized scientific workflows in cloud environments with applications in life science. 2020.

[21] Various. Kubernetes, November 2021. URL: `https://kubernetes.io/`.

[22] Various. Wikipedia:Technik/Datenbank/Download, October 2021. URL: `https://de.wikipedia.org/wiki/Wikipedia:Technik/Datenbank/Download#Herunterladen_aller_Seiten_als_XML-Dump`.

[23] John Vivian, Arjun Arkal Rao, Frank Austin Nothaft, Christopher Ketchum, Joel Armstrong, Adam Novak, Jacob Pfeil, Jake Narkizian, Alden D Deran, Audrey Musselman-Brown, et al. Toil enables reproducible, open source, big biomedical data analyses. *Nature biotechnology*, 35(4):314–316, 2017.

[24] Jens-S. Vöckler, Gideon Juve, Ewa Deelman, Mats Rynge, and Bruce Berriman. Experiences using cloud computing for a scientific workflow application. In *ScienceCloud'11*, 2011.

[25] Pegasus WMS. Pegasus WMS - automate, recover, and debug scientific computations, September 2021. URL: `https://pegasus.isi.edu/`.

[26] Laura Wratten, Andreas Wilm, and Jonathan Göke. Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers. *Nature Methods*, pages 1–8, 2021.

[27] Yong Zhao, Youfu Li, Ioan Raicu, Shiyong Lu, Wenhong Tian, and Heng Liu. Enabling scalable scientific workflow management in the cloud. *Future Generation Computer Systems*, 46:3–16, 2015.