Adapting scientific workflows to changing infrastructures

Ninon De Mecquenem[1], Supervised by Ulf Leser[1]

[1]Humboldt-Universität zu Berlin, Unter den Linden 6, Berlin, DE 10099

Abstract

Scientific workflows are increasingly popular for large-scale data analyses as they promise better documentation, increased reproducibility, and easier scalability of complex analysis pipelines. However, reproducibility is severely reduced when a given workflow is optimized for a specific infrastructure, as it would require other scientists to access the same computing environment. Hence, it is important to develop techniques that automatically adapt a given workflow to changes in the underlying infrastructure or characteristics of the analyzed data, for instance, by using different data partitions or different tools for individual steps of the analysis. Automatic workflow adaptation requires a cost model setting properties of different tools, data set sizes, and characteristics of the given infrastructure into perspective. As a first step in this direction, we here study in detail the performance of an important analysis in genomics, namely RNASeq, in different settings. We experimentally measured the runtime of different RNAseq workflows implemented in Nextflow on different infrastructures (stand-alone or distributed), composed of different tool chains, using different data set sizes. As different tools also lead to (slightly) different outputs, we additionally compared the output of different workflow variants. We show that workflow variants designed for a given infrastructure perform much worse in other settings and that rewritings sometimes keep and sometimes change the output, even when tools are only replaced by others with the same purpose. We see these experiments as an important first step toward automatically adapting workflows to different infrastructures.

Keywords

Data Analysis Workflows, Bioinformatics, Distributed infrastructures, Portability

1. Introduction

Data Analysis Workflows (DAWs) are used to solve a specific data analysis problem using a chain of tools connected by input/output dependencies. In bioinformatics, the usage of DAWs is critical to perform reproducible analyses [1]. However, porting DAWs to different infrastructures or using them for different input data sizes can cause severe problems. For example, if the new infrastructure has fewer resources, the workflow can crash due to insufficient memory, or time outs as computations take longer than anticipated at workflow design time. On the other hand, also with more resources scaling problems can affect the runtime of the analysis [2, 3, 4]. In our research, we hypothesize that knowledge of the infrastructure, the input, the DAW itself and the particular tools it is made of can be used to automatically adapt a given DAW such that it performs gracefully also in a new environment. The rewriting of chains of interdependent commands has a long tradition, especially in the database [5] and the big data world [6]. However, rewriting DAWs for scientific data analysis differs from these settings in two regards. First, DAWs are typically designed and executed in a black box model both for the data and the operations - the executing infrastructure cannot make any assumptions regarding the functionality of the operations nor the format of the data [7]. Second, DAWs for complex scientific data analysis consist of many steps that are heuristics, which means that the "correct" result of an analysis actually is not known and that different DAWs for the same purpose on the same data might produce diverging results [8]. Therefore, DAW adaptation may consider a wide range of valid primitive operations, such as: the replacement a tool of the DAW by another one with the same purpose (1), the change of tool/DAW parameters (2), the modification of the DAW structure (3), or the adjustment of the sizes of data partitions (4). Before implementing such functionalities, it is crucial to understand the impact of a given adaptation for a given setting on the workflow runtime and the output. In this paper, we study this problem for DAWs performing an RNA sequencing (RNAseq) analysis. RNASeq is particularly interesting as it is a widespread of analysis used to understand gene expression and regulation under certain conditions or diseases such as cancer. RNASeq DAWs take a large set of short strings as input, which are sequenced fractions of mRNA, the transient molecules generated during gene expression as an intermediate step to protein sequences. DAWs next map each string to a reference genome to then cluster sets of strings stemming from the same transcript. Real-life DAWs also include further steps, such as data pre-processing, quality filtering, or computation of different quality metrics. Each task of these DAWs can be performed by several tools that serve the same purpose, but use different heuristics

Published in the Workshop Proceedings of the EDBT/ICDT 2023 Joint Conference (March 28-March 31, 2023, Ioannina, Greece)

mecquenn@informatik.hu-berlin.de (N. D. Mecquenem); leser@informatik.hu-berlin.de (S. b. U. Leser)
0000-0003-3052-6129 (N. D. Mecquenem); 0000-0003-2166-9582 (S. b. U. Leser)

leading to different resource requirements and results. A particularly complex step is the mapping, for which a plethora of possible tools exist [8].

Here, we studied the behaviour of three RNAseq DAWs on two different infrastructures using two different data sets. We created the DAWs tool-chains based on the tools' popularity and compatibility with each other. Each DAW was implemented in two versions: one is designed for a stand-alone server, and another one is designed for a distributed infrastructure. Adaptation consists of splitting the load of resource-demanding tasks across several nodes of the cluster by splitting the input files - which is not supported equally well by all tools. We ran the two versions of these three workflows on two different infrastructures and measured the runtime and output differences between the workflow versions depending on several parameters. We consider this work as a base to better understand the impact of DAW rewritings. Ultimately, we aim at abstracting these findings into a set of rules that lead to an automatic DAW rewriting according to a given input and context specifications.

2. Related work

Several strategies have already been used to optimize DAW execution in a given context. For instance, [9] surveys genomic workflows for the Map-Reduce infrastructure. Zaid Al-Ars et al. created a version of the popular RNAseq GATK workflow adapted to Spark [10]. Yakeen et al. describe a large-scale variant caller optimized for execution on a commercial cloud [11]. Roy et al. studied the influence of different Hadoop parameters on a specific genomics DAW [12]. However, all these works focused on optimization for a specific target infrastructure; in contrast, we research methods that can adapt DAWs to any execution infrastructure. Another line of related research is concerned with optimization of (relational) queries with user-defined functions, especially in big data processing pipelines [6]. These, however, typically are built on the paradigm that (a) individual operations (tasks in a workflow setting) have pre-defined semantics, (b) data follows a relational model, and (c) all rewritings preserve exactly the results of a query. These assumptions do not hold in the realm of DAWs for scientific analysis - data can have arbitrary formats, tasks are typically exchanged as binaries without any guarantees, and operations are partly computationally so complex that they can only be approached using heuristics, leading to different results for different concrete physical implementations. Accordingly, we envision that DAW rewriting makes up for these more complex settings by relying on knowledge provided by workflow designers that want to support portability and re-usability of their workflows. Finally, there is some commonality to recent studies in the field of AutoML [13]. The main differences are that in AutoML (1) pipelines are linear and only exchanges of tasks are considered and (2) the results of different workflow variants may also vary, but that typically a notion of ”best” is defined (e.g. highest accuracy on a test data set), which often is not the case in scientific data analysis.

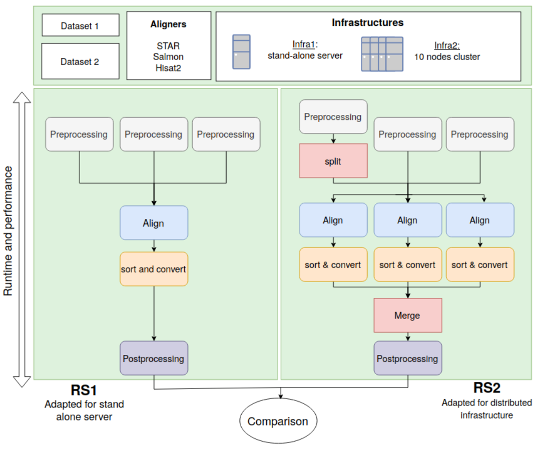

Figure 1: Design of the experiments. RS1 was created with three tool-chains (Salmon, STAR, Hisat2). A variation of these workflows (RS2) was created. We ran it on two infrastructures (infra1 and infra2) and two datasets (D1 and D2). Their runtime was measured, and their output was compared.

3. Experiments

As described in Figure 1, we created two RNASeq workflows performing the same operations but optimized for different infrastructures. Both have as central and most time-consuming task the *alignment*, which matches short stretches of genomic sequences to a reference genome. The alignment is also fundamental in other bioinformatics workflows studying genomic sequences [14]. RS1 follows a pipeline structure, which we assume fits better to a stand-alone server. RS2 is optimized for a distributed infrastructure, as it splits the input to allow for a distributed computation of the alignment step across several nodes of a cluster. From each workflow, we furthermore created three variants according to the specific tool used to compute alignments, i.e., STAR [15], Salmon [16], and Hisat2 [17]. We implemented all DAWs using Nextflow [18], a workflow engine of increasing popularity in the Bioinformatics community. Nextflow workflows are implemented in a specific DSL which allows for automatic parallelization and distributed execution of tasks. For local execution, NextFlow uses its own execution engine; for a distributed setup, it can work together with Kubernetes resource managers. The goal of this experiment is

Data and Infrastructures

Two RNAseq-paired input datasets of different sizes were considered. Both data sets were obtained by sequencing the transcriptome of *Drosophila melanogaster*. The difference in size allows us to understand better the impact of the input size on the decision to rewrite the workflow. Dataset 1 consists of two paired files of 13GB each, and Dataset 2 is two files of 48G.

The DAWs were run on two different infrastructures: one stand-alone server and one cluster. The stand-alone server (infra1) consists of 32 Intel Xeon CPU E5-2667 v2 Octa Core, with a memory of 387 GB and a SATA SSD 1,9TiB Raid 5. The cluster Infra2 in our experiments consists of 10 homogeneous nodes, each with a Quadcore Intel Xeon CPU E3-1230 V2 3.30GHz; Memory: 16 GB; Disks: 3x1TB, connected by a network of 2x 1GBit. The stand alone server has way more resources than the cluster. Therefore, we expect all the runtimes to be faster on Infra1.

4. Results

Runtime comparison

Table 1 shows the runtimes of the pipeline version (RS1) and the distributed version (RS2) of the DAWs on the two infrastructures with both datasets. Note that each value displayed in this table was obtained from a single run. We are currently generating duplicate runs to acquire more robust values. However, as we had exclusive usage of the infrastructures during the measurements, we are confident about our experiments not being perturbed by other computations. Furthermore, the duplicated runs that were already computed are consistent with the results presented in the table.

We observe notable runtime differences that show tendencies but not an entirely consistent picture. In general, RS1 and RS2 show similar runtimes on the stand-alone server Infra1, with the notable exception of Hisat2 on the large dataset D2. For this case, time reduction is almost 50%, while runtimes for the smaller dataset D1 are very similar. We attribute this behaviour to the low resource usage of Hisat2. In a non-distributed setting, Nextflow parallelizes the tasks over the different CPUs available, which makes the runtime on Infra1 overall smaller. Almost no difference is observed for STAR on infra1 as it requires a lot of RAM to run over a single chunk of the input data. On Infra2, runtimes differ considerably. In almost all cases, RS2 (designed for distributed computation) achieves much lower runtimes than RS1, with reductions up to 66%. Again, there is one exception: Salmon actually takes longer with RS2 than with RS1. This runtime difference is due to the task splitting the input files.

DAW	Dataset	Infrastructure	RS1	RS2
STAR	D1	Infra1	58 m	48 m
		Infra2	364 m	134 m
	D2	Infra1	401 m	321 m
		Infra2	2383 m	720 m
Hisat2	D1	Infra1	60 m	47 m
		Infra2	175 m	85 m
	D2	Infra1	569 m	232 m
		Infra2	935 m	437 m
Salmon	D1	Infra1	8 m	15 m
		Infra2	68 m	51 m
	D2	Infra1	32 m	51 m
		Infra2	186 m	270 m

Table 1
Runtimes of the three RNAseq DAWs, compared across DAW versions and datasets for both infrastructures. The DAWs are named by the aligner used in their tool-chain. In bold, we highlight large reduction in runtimes from RS1 to RS2.

Interestingly, the DAWS require very different runtimes depending on which tool was used for the alignment step. On Infra1 with D2, the runtime of Salmon is approx. 20 times faster than HiSat2 on RS1 and five times faster on RS2. In almost all cases, workflows profited from the plus in resources in the distributed Infra2 when switching from RS1 to RS2, but to varying degrees.

However, recall that our intention is not to find the best DAW for a given infrastructure, but to develop algorithms that can rewrite a given DAW developed for a setting A to adapt it to a new setting B - which might simply have a slow network, such as Infra2. For instance, imagine a researcher developed RS1 on Infra2 using HiSat2 for data sets of the size of DS1. Now, she wants to run it on larger datasets yet avoid that 5-fold increase in runtime. An adaptation an optimizer could propose is to rewrite the workflow into RS2, which would only lead to a 2-fold runtime. Or imagine another user who wants to reuse this workflow, but is forced to use STAR as aligner because it is the lab-internal standard. Runtime would be doubled, or even increased by a factor of 13 when also switching to larger files. An optimizer could recognize that switching to RS2 would decrease the expected increase by 65%.

Quality comparison

In scientific data analysis, different DAWs for the same problem often lead to (slightly) different results due to the usage of different heuristics for solving complex subproblems. Sometimes, avoiding such changes can be mandatory, for instance, when a certain analysis method is defined as an organizational standard. However, often such changes are acceptable, for instance, in the early phases of a data analysis project in which different trade-offs are explored, such as runtime, result quality, analysis cost etc. In any case, users need to be informed about the

	STAR	Salmon	Hisat2
Dataset 1	100 %	82,06 %	97,39 %
Dataset 2	100 %	71,71 %	99,65 %

Table 2
Percentage of similarity of transcript counts between the pipeline version (RS1) and the scatter/gather version (RS2) of each DAW.

expected degree of changes a DAW rewriting would incur. To this end, we compared the results of the different DAW versions to understand how much the DAG structure modification impacts analysis results. We measured the similarity between the results of the RS1 and RS2 versions of each DAW in Table 2. Clearly, the DAWs using Hisat2 and STAR are very robust to this rewriting, while the one using Salmon produces largely different results.

5. Conclusion and future work

We presented the results of an initial study on the impact of DAW adaptations to different infrastructures, considering both the replacement of central tools as well as changing the workflow structure. The main purpose is to show that such rewritings impact performance considerably and that certain variants are more suitable for certain infrastructures and that suitability also depends on the input size. Relationships are overall complex and certainly will vary with different analysis problems, different DAWs for solving them, and different infrastructures. We are consolidating these results with experiment replicates and more workflows and dataset sizes. In future work, we will focus on languages to provide descriptions of core aspects of infrastructures, methods to derive properties of tools on different infrastructures, annotation schemes to describe the equivalence of tools in genomics and a cost model as a basis for a rule-based DAW adaptation algorithm that takes these properties into account. We will then develop an automatic DAW rewriting that implements this algorithm.

Acknowledgements

Funded by the Deutsche Forschungsgemeinschaft – Project-ID 414984028 – SFB 1404 FONDA

References

[1] L. Wratten, A. Wilm, J. Göke, Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers, Nat Methods (2021).
[2] C. Schiefer, M. Bux, J. Brandt, et al., Portability of Scientific Workflows in NGS Data Analysis: A Case Study, arXiv:2006.03104 (2020).
[3] F. Lehmann, D. Frantz, S. Becker, et al., FORCE on Nextflow: Scalable Analysis of Earth Observation data on Commodity Clusters, CIKM Workshops (2021).
[4] M. Hanussek, F. Bartusch, J. Krüger, Performance and scaling behavior of bioinformatics applications in virtualization environments to create awareness for the efficient use of compute resources, PLOS Computational Biology 17(7): e1009244 (2021).
[5] H. Garcia-Molina, J. D. Ullman, J. Widom, Database systems - the complete book, Pearson (2009).
[6] A. Rheinländer, A. Heise, F. Hueske, et al., Sofa: An extensible logical optimizer for udf-heavy data flows, Information Systems 52 (2015) 96–125.
[7] R. Mork, P. Martin, et al., Contemporary challenges for data-intensive scientific workflow management systems, Workshop on Workflows in Support of Large-Scale Science (2015).
[8] A. Schaarschmidt, A. Fischer, E. Zuther, et al., Evaluation of seven different rna-seq alignment tools based on experimental data from the model plant arabidopsis thaliana, Int J Mol Sci (2020).
[9] Z. Quan, L. Xu-Bin, J. Wen-Rui, et al., Survey of MapReduce frame operation in bioinformatics, Briefings in Bioinformatics 15 (2013) 637–647.
[10] A.-A. Zaid, W. Saiyi, M. Hamid, Sparkra: Enabling big data scalability for the gatk rna-seq pipeline with apache spark, Genes 11 (2020).
[11] S. Yakneen, S. Waszak, M. Gertz, et al., Butler enables rapid cloud-based analysis of thousands of human genomes, Nat Biotechnol (2020).
[12] A. Roy, Y. Diao, U. Evani, et al., Massively parallel processing of whole genome sequence data: An in-depth performance study, SIGMOD (2017).
[13] K. He, Xin amd Zhao, X. Chu, Automl: A survey of the state-of-the-art, Knowledge-Based Systems 12 (2021).
[14] R. Musich, L. Cadle-Davidson, M. Osier, Comparison of short-read sequence aligners indicates strengths and weaknesses for biologists to consider, Front Plant Sci. (2021).
[15] A. Dobin, C. A. Davis, F. Schlesinger, et al., STAR: ultrafast universal RNA-seq aligner, Bioinformatics 29 (2012) 15–21.
[16] R. Patro, G. Duggal, M. Love, Salmon provides fast and bias-aware quantification of transcript expression, Nat Methods (2017).
[17] D. Kim, J. Paggi, C. Park, Graph-based genome alignment and genotyping with hisat2 and hisat-genotype, Nat Biotechnol (2019).
[18] P. D. Tommaso, M. Chatzou, E. Floden, Nextflow enables reproducible computational workflows, Nat Biotechnol (2017).