

Classification of Scientific Workflows Based on Reproducibility Analysis

A. Bánáti¹, P. Kacsuk^{2,3} and M. Kozlovsky^{1,2}

¹ Óbuda University, John von Neumann Faculty of Informatics, Biotech Lab
Bécsi str. 96/b., H-1034, Budapest, Hungary

² MTA SZTAKI, LPDS, Kende str. 13-17, H-1111, Budapest, Hungary

³ University of Westminster, 115 New Cavendish Street, London W1W 6UW
{banati.anna, kozlovsky.miklos}@nik.uni-obuda.hu,
kacsuk@sztaki.mta.hu

Abstract - In the scientist's community one of the most vital challenges is the reproducibility of a workflow execution. The necessary parameters of the execution (we call them descriptors) can be external which depend on for example the computing infrastructure (grids, clusters and clouds), on third party resources or it can be internal which belong to the code of the workflow such as variables. Consequently, during the process of re-execution these parameters may change or become unavailable and finally they can prevent to reproduce the workflow. However in most cases the lack of the original parameters can be compensated by replacing, evaluating or simulating the value of the descriptors with some extra cost in order to make it reproducible. Our goal in this paper is to classify the scientific workflows based on the method and cost how they can become reproducible.

I. INTRODUCTION

In large computational challenges scientific workflows have emerged as a widely accepted solution for performing in-silico experiments. In general these in-silico experiments consist of series of particularly data and compute intensive jobs, and in most cases their executions require parallel and distributed infrastructure (super/hypercomputers, grids, clusters, clouds).

The successive steps of an experiment are chained to a so called workflow, which can be represented by a directed acyclic graph (DAG). The nodes are so called jobs, which includes the experimental computations based on the input data accessed through their input ports. In addition, these jobs can product output data, which can be forwarded through their output ports to the input port of the next job. The edges of a DAG represent the dataflow between the jobs (Figure 1.).

An essential part of the scientific method is to repeat and reproduce the experiments of other scientist and test the outcomes themselves even in a different execution environment. A scientific workflow is reproducible, if it can be re-executed without failures and gives the same result as the first time. In this approach the failures do not mean the failures of the Scientific Workflow Management System (SWfMS) but the correctness and the availability of the inputs, libraries, variables etc. Different users for different purposes may be interested in reproducing the workflow, for example the

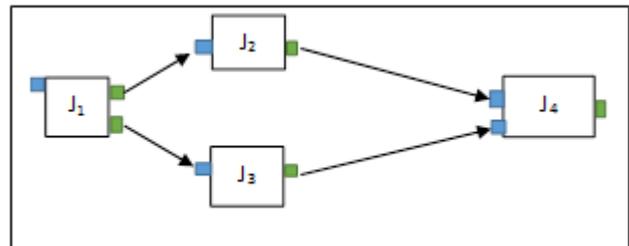


Figure 1. Workflow example with four jobs (J1, J2, J3, J4)

authors of the workflow(s) in order to prove their results, readers or other scientists in order to reuse the results or reviewers in order to verify the correctness of the results [1]. Additionally, nowadays scientific workflow repositories are already available and in this way the scientists can share their results with each other and even they can reuse the existing workflows to create new ones.

The two most significant obstacles of reproducing a workflow are the dependencies of workflow execution and the rich collection of provenance data. The former can be perceived as the necessary and the latter one as the satisfactory requirements of the reproducibility. The dependencies of the execution mean those resources which require external (out of the scientific workflow management system, SWfMS) services or resources such as third party services, special hardwares/software or random value generator [2]. Elimination of these dependencies in most cases is not possible, so they have to be handled in some other way: different methods should be set up to make the workflows reproducible.

To achieve our goal we have defined the descriptor space and the decay-parameters of the jobs that give us the possibility to analyze the workflow from a reproducibility perspective. The descriptor space contains all the parameters (call descriptors), which are necessary to reproduce the workflow. There are descriptors, which are constant and do not change in time. Other descriptors are continuously changing (for example a database which continuously get more and more data from sensor networks). Also descriptors based on external services (such as third party services) may exist which can be unavailable after a few years. Finally there are descriptors which are unknown and its behavior is unpredictable. In

this case the workflow is non-reproducible. The decay-parameter describes the type and the measure of the change of the descriptor. With the help of the decay-parameter we have determined five categories of the workflows: reproducible, reproducible with extra cost, approximately reproducible, reproducible with probability P and non-reproducible.

The goal of our investigation to find different methods to make reproducible the workflow in the different categories even if it requires extra costs or compromises. In certain cases this goal is implementable but often the result of the workflow is only evaluative with the help of simulations. If there is no method to make the workflows reproducible, our goal is to provide the scientists with useful information about the conditions and probability of the reproducibility of his workflows.

The rest of the paper is organized as follows: In the next section we provide a short background and overview about works related to our research. Section 3 presents the mathematical model of our reproducibility analysis. In section 4 we give the classification of the scientific workflows based on our analysis. In section 5 based on our model we define the general measures of the reproducibility analysis. Finally we summarize our conclusions and reveal the potential future research directions.

II. STATE OF THE ART

Currently the reproducibility of scientific workflows is a burning question which the scientist community has to face with and has to solve. Accordingly in the latter one-two years many researchers investigate this issue. One part of the literature analyzes the requirements of reproducibility and the other part deals with the implementation of such tools or frameworks.

The first group agree on the importance of the careful design [3], [4], [5], [6], [7] which on one hand means the increased robustness of the scientific code, for example with a modular design and detailed description of the workflow, and of the input and output data examples, and consequent annotations [8]. On the other the careful design includes the careful usage of volatile third party or special local services. In these cases two solutions exist, but reproducibility is unfeasible: 1. taking a digital copy of the entire environment using a system virtual machine/hardware virtualization approach capturing and storing metadata about the code and environment that allows it to be recreated later [8].

Zhao et al. [9] in their paper investigate the cause of the so called *workflow decay*, which means that year by year the ability and success of the re-execution of any workflow significantly reduces. They examined 92 Taverna workflows submitted in the period from 2007 to 2012 and found four major causes: 1. Missing volatile third party resources 2. Missing example data 3. Missing execution environment (requirement of special local services) and 4. Insufficient descriptions about workflows. Hettne et al. [10] in their paper list ten best practice to prevent the workflow decay. Grothe et al. [11] analyze the characteristic of applications used by workflows and list the requirements in order to enable the reproducibility of results and

determination of provenance. To the former mentioned requirements they assumed the deterministic feature of applications in order to perform appropriate provenance collection.

There exist available tools, VisTrail, ReProZip or PROB [12], [13], [14], which allow the researcher and scientist to create reproducible workflow. With help of VisTrail [12], [15] reproducible paper can be created, which includes not only the description of scientific experiment, but all the links for input data, applications and visualized output which always harmonizes with the actually applied input data, filter or other parameters. ReProZip [13] is another tool, which stitches together the detailed provenance information and the environmental parameters into a self-contained reproducible package.

The Research Object (RO) approach [16], [17] is a new direction in this research field. RO defines an extendable model, which aggregates a number of resources in a core or unit. Namely a workflow template; workflow runs obtained by enacting the workflow template; other artifacts which can be of different kinds; annotations describing the aforementioned elements and their relationships. Accordingly to the RO, the authors in [18] also investigate the requirements of the reproducibility and the required information necessary to achieve it. They created ontologies, which help to uniform these data. These ontologies can help our work and give us a basis to perform our reproducibility analysis and make the workflows reproducible despite their dependencies.

Piccolo et al [19] collected the tools and techniques and proposed six strategies which can help the scientist to create reproducible scientific workflows.

Santana-Perez et al [20] proposed an alternative approach to reproduce scientific workflows which focused on the equipment of a computational experiment. They have developed an infrastructure-aware approach for computational execution environment conservation and reproducibility based on documenting the components of the infrastructure.

To sum up the results mentioned above, we can conclude that the general approach is that the scientist has to create reproducible workflows with careful design, appropriate tools and strategies. But none of them intended to solve the problem related to the dependencies rather they suggested to bypass them. Moreover, they did not deal with the following question: How an existing workflow can be made reproducible?

III. THE MODEL

In our approach a scientific workflow consisted of N jobs can be written as a function of its job:

$$SWf(J_1, J_2, \dots, J_N) = \mathbf{R} \quad (1)$$

where \mathbf{R} is the vector of results.

In our investigation we assume, that a given workflow is executed at least one time and the provenance database of the workflow execution is available. In this case we can assign a so called descriptor space to every job of the given workflow.

$$D_{J_i} = \{d_{i1}, d_{i2}, \dots, d_{iK_i}\} \quad (2)$$

The elements of this descriptor space are called descriptors and they give all the necessary parameters to reproduce the job. These parameters can be for example variables of the infrastructure, variables of the code, parameters of system calls, inputs, outputs and partial data or access paths of external resources etc [21]. Every descriptor has a name and a value. In addition we also assign them a so called decay-parameter which describes the type and the measure of the change of the given value. The decay-parameter can be zero, which means that the value of this descriptor is not changing in time, in other word the availability of this descriptor (and its value) can be insured in one, two, ten or any years. In this case this descriptor does not cause dependency and the reproducibility of the job does not depend on this descriptor. The decay parameter can be infinite, if the descriptor's value is unknown. For example in case of random generated values. The value of the decay-parameter can be a distribution function $F(t)$ if the availability of the given resource varies in time according to this $F(t)$. The fourth option is that the value of the decay parameter is a function – vary(t, x) – depending on time, which determines the variation of the descriptor's value.

Formally:

$$decay(v_i) = \begin{cases} 0, & \text{if the value of the descriptor is} \\ & \text{not changing in time} \\ \infty, & \text{if the value of the descriptor} \\ & \text{is unknown} \\ F_i(t), & \text{distribution function of the} \\ & \text{availability of the given value} \\ Vary_i(t, v_i), & \text{if the value of the} \\ & \text{descriptor is changing} \\ & \text{in time} \end{cases} \quad (3)$$

The descriptors and its decay parameters can originate from three different sources: from the users, from the provenance database and it can be automatically generated by the SWfMS. [21]

With the help of these expressions we can define the reproducibility in the following way:

Definition: The J_i job is reproducible, if

$$\begin{aligned} JOB_i(t_0, v_{i1}(d_{i1}), v_{i2}(d_{i2}), \dots, v_{iK_i}(d_{iK_i})) = \\ = JOB_i(t_0 + \Delta t, v_{i1}(d_{i1}), v_{i2}(d_{i2}), \dots, v_{iK_i}(d_{iK_i})) = \mathbf{R}_i \end{aligned} \quad (4)$$

for every Δt .

In addition if a scientific workflow contains N jobs and the jobs are reproducible, the scientific workflow is also reproducible:

$$SWF(t_0, J_1, J_2, \dots, J_N) = SWF(t_0 + \Delta t, J_1, J_2, \dots, J_N) = \mathbf{R} \quad (5)$$

for every Δt .

Also we can assign a cost to the descriptors. This gives the measurement of the “work” or cost which is necessary

TABLE 1. THE DESCRIPTOR SPACE OF A JOB AND ITS MEASURES

Descriptor's name	Descriptor's value	Decay-parameter	Cost
d_1	$v_1(d_1)$	$decay(v_1)$	c_1
d_2	$v_2(d_2)$	$decay(v_2)$	c_2
...
d_K	$v_K(d_K)$	$decay(v_K)$	c_K

to make the job reproducible. For example, when the value of the descriptor is a large amount of data which cannot be stored even on extra storage. We can assign a cost to this extra storage. Or another example if the descriptor is changing in time and its decay-parameter is a so called “vary function”. In this case to reproduce this workflow we can apply simulation tools based on the sample set which also result an extra cost (see section IV.A).

IV. THE CLASSIFICATION

Analyzing the decay parameters of the descriptors we can classify the scientific workflows. First, we can separate the workflows which decay-parameters for all the jobs are zero. These workflows are reproducible at any time and any circumstance since they do not have dependencies. Than we can determine those ones which can influence the reproducibility of the workflow in other words which also have non-zero decay parameter(s). Four groups have been created:

1. At least one decay-parameter of the descriptor is infinite, but with the help of additional resources or tools this dependency of execution can be eliminated. In this case the cost of this descriptor indicates that there are possibility to reproduce the job with some extra cost.
2. At least one decay-parameter of the descriptor is infinite and the cost of this descriptor is also infinite. In this case the dependency of the workflow can not be eliminated and the workflow is non-reproducible.
3. At least one decay-parameter of the descriptor is a probability distribution function and the other ones are zero.
4. At least one decay-parameter of the descriptor is a vary function and the other ones are zero. (Table 2.)

A. Reproducible workflows

The first group represents the reproducible workflows. In this case all the decay-parameters of all the jobs belonged to a workflow are zero. These workflows are reproducible and they can be executed and re-executed at any time and any circumstance since they are not influenced by dependencies.

B. Reproducible workflow with extra cost

There are workflows, which have dependencies and infinite decay-parameters, but the appropriate cost is not infinite. In this case with the help of additional resources or tools these dependencies can be eliminated. For example, if

a computation is based on random generated value, this descriptor's value is unknown (infinite). In this case with the help of an extra, operation system level tool we can capture the return value of the system call and we can save it in the provenance database [22]. The third example is when a virtualization tool, such as a virtual machine have to be applied to reproduce the workflow.

C. Approximately reproducible workflows

In certain cases the workflow execution may depend on some continuously changing resource. For example there are continuously growing databases which get the data from sensor networks without intermission. If the computation of a workflow use some statistical parameters of this database, the statistical values never will be the same. In this case the appropriate descriptor's value of the given job may change on occasion of every re-execution, consequently the reproducibility of this workflow could be failed.

If the workflow was executed S times and the provenance database is available, we can create a sample set which contains the S different values of the changing descriptors and the S results of the workflow. In this case we can analyze the change of the descriptor's value, we can write its function and even, we can determine a general evaluating method of the result. On occasion of a later re-execution, if reproducing is not possible, this evaluating method can be applied and an evaluated result can be done with a given probability [22].

D. Reproducible workflows with a given probability

Many investigations revealed the problem caused by volatile third party resources [...], when the reproducibility of workflows became uncertain. The third party services or any external resources can be unavailable during the years. If we know this decay of the resources and if we can determine its probability distribution function we can predict the behavior of the workflow on occasion of a re-execution at a later time. Sometime the users may have to know the chance of the reproducibility of their workflow. Assuming that the probability distribution of the third party service is known or assumable we can inform the users about the expected probability of the reproducibility.

To formalize the problem, first, we have separated the M_i descriptors of a given job J_i which depend on external or third party resources and its decay-parameter, which is a probability distribution function given as follows: $F_{i1}(t), F_{i2}(t), \dots, F_{iM_i}(t)$. The rest of the descriptors have zero decay-parameter. In this case, at time t_0 , a given descriptor's value $v_{ij}(d_{ij})$ is available with a given probability (for the sake of the easier comprehensibility hereafter we omitted the i index referred to the i th job of a given scientific workflow):

$$F_1(t_0) = p_1^{(t_0)}, F_2(t_0) = p_2^{(t_0)}, \dots, F_M(t_0) = p_M^{(t_0)} \quad (6)$$

Let us assign to the job J_i a state vector $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iM_i}) \in \{0,1\}^{M_i}$, in which the $y_{ij} = 1$, if the j th descriptor of the job J_i is unavailable. In this way the probability of a given \mathbf{y}_i state vector can be computed as follows:

$$p(\mathbf{y}) = \prod_{j=1}^M p_j^{y_j} (1 - p_j)^{1-y_j} \quad (7)$$

TABLE 2. CLASSIFICATION OF SCIENTIFIC WORKFLOWS

decay-parameter	cost	category
$decay(v)=0$	$cost = 0$	reproducible
$decay(v) = \infty$	$cost = \infty$	non-reproducible
$decay(v) = \infty$	$cost = C_1$	reproducible with extra cost
$decay(v) = F(t)$	$cost = C_2$	reproducible with probability P
$decay(v) = \text{vary}(t,v)$	$cost = C_3$	approximately reproducible

In addition a time interval can be given during which the descriptor is available with a given probability P .

Since we assume the independency of the descriptors the cumulative distribution function of the job J_i can be written as follows:

$$\mathbf{F}_i(t) = \prod_{j=1}^M F_{ij}(t) \quad (8)$$

E. Non-reproducible workflows

There is no method to make the workflow reproducible. In this case the scientific workflow probably contains non-deterministic job or jobs.

V. REPRODUCIBILITY ANALYSIS

It may be important to inform the user about the reproducibility of his workflow or even the cost of the reproducibility. Based on our mathematical model we can determine two measures according to the expected cost: the average cost and the reproducibility probability.

1. Average Cost (AC) expressed as

$$E(g(\mathbf{y})) = \sum_{\mathbf{y} \in Y} g(\mathbf{y})p(\mathbf{y}) \quad (9)$$

where $g(\mathbf{y}) = \sum_{i=1}^K c_i$.

2. Reproducibility Probability (RP)

$$P(g(\mathbf{y}) > C) = \sum_{Y: g(\mathbf{y}) > C} p(\mathbf{y}) \quad (10)$$

where C is a given level of the reproducibility cost.

VI. CONCLUSION

In this paper we investigated the possible types of the scientific workflows from a reproducibility perspective. The basis of our analysis is the decay-parameter which describes the type and the measure of the change of the descriptor's values. According to this parameter we determined a cost function which means the "work" required to reproduce the given job or workflow. In this way we could classify the scientific workflows, how they can be reproduced at a later time. In the different categories we set up methods to make the workflows reproducible or we gave the probability and the extra cost of the reproducibility. Finally we gave two general measure to evaluate the expected cost of the reproducibility.

The goal of our research is to support the scientists with methods to make their experiment reproducible and to provide information about the possibility to reproduce their workflows.

REFERENCES

- [1] D. Koop, E. Santos, P. Mates, T. Vo Huy, P. Bonnet, B. Bauer, M. Troyer, D.N. Williams, J.E. Tohline, J. Freire, C.T. Silva, „A Provenance-Based Infrastructure to Support the Life Cycle of Executable Papers”, International Conference on Computational Science, ICCS 2011. [Online]. Available: <http://www.sciencedirect.com>.
- [2] A. Banati, P. Kacsuk, M. Kozlovsky, M. Four level provenance support to achieve portable reproducibility of scientific workflows. In Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on (pp. 241-244). IEEE.
- [3] P. Missier, S. Woodman, H. Hiden, és P. Watson, „Provenance and data differencing for workflow reproducibility analysis”, Concurrency and Computation: Practice and Experience, 2013
- [4] R. D. Peng, „Reproducible Research in Computational Science”, Science, köt. 334, sz. 6060, o. 1226–1227, dec. 2011
- [5] J. P. Mesirov, „Accessible Reproducible Research”, Science, köt. 327, sz. 5964, o. 415–416, jan. 2010.
- [6] D. De Roure, K. Belhajjame, P. Missier, J. M. Gómez-Pérez, R. Palma, J. E. Ruiz, K. Hettne, M. Roos, G. Klyne, C. Goble, és others, „Towards the preservation of scientific workflows”, in Procs. of the 8th International Conference on Preservation of Digital Objects (iPRES 2011). ACM, 2011.
- [7] S. Woodman, H. Hiden, P. Watson, és P. Missier, „Achieving reproducibility by combining provenance with service and workflow versioning”, in Proceedings of the 6th workshop on Workflows in support of large-scale science, 2011, o. 127–136.
- [8] A. Davison, „Automated Capture of Experiment Context for Easier Reproducibility in Computational Research”, Computing in Science & Engineering, köt. 14, sz. 4, o. 48–56, júl. 2012.
- [9] J. Zhao, J. M. Gomez-Perez, K. Belhajjame, G. Klyne, E. Garcia-Cuesta, A. Garrido, K. Hettne, M. Roos, D. De Roure, és C. Goble, „Why workflows break—Understanding and combating decay in Taverna workflows”, in E-Science (e-Science), 2012 IEEE 8th International Conference on, 2012, o. 1–9.
- [10] K. M. Hettne, K. Wolstencroft, K. Belhajjame, C. A. Goble, E. Mina, H. Dharuri, D. De Roure, L. Verdes-Montenegro, J. Garrido, és M. Roos, „Best Practices for Workflow Design: How to Prevent Workflow Decay.”, in SWAT4LS, 2012
- [11] P. Groth, E. Deelman, G. Juve, G. Mehta, és B. Berriman, „Pipeline-centric provenance model”, in Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science, 2009, o. 4.
- [12] J. Freire, D. Koop, F. S. Chirigati, és C. T. Silva, „Reproducibility Using VisTrails”, 2014. [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.369.9566>
- [13] F. S. Chirigati, D. Shasha, és J. Freire, „ReproZip: Using Provenance to Support Computational Reproducibility.”, in *TaPP*, 2013.
- [14] V. Korolev, A. Joshi, V. Korolev, M. A. Grasso, A. Joshi, M. A. Grasso, D. Dalvi, S. Das, V. Korolev, Y. Yesha, és others, „PROB: A tool for Tracking Provenance and Reproducibility of Big Data Experiments.”, *Reproduce'14. HPCA 2014*, köt. 11, o. 264–286, 2014.
- [15] D. Koop, J. Freire, és C. T. Silva, „Enabling Reproducible Science with VisTrails”, *arXiv preprint arXiv:1309.1784*, 2013.
- [16] O. Belhajjame, K. Corcho, D. Garijo, J. Zhao, P. Missier, D. R. Newman, R. Palma, S. Bechhofer, G. C. Esteban, J. M. Gomez-Perez, G. Klyne, K. Page, M. Roos, J. E. Ruiz, S. Soiland-Reyes, L. Verdes-Montenegro, D. De Roure, and C. Goble. Workflow-centric research objects: First class citizens in scholarly discourse. In Proceedings of the ESWC2012 Workshop on the Future of Scholarly Communication in the Semantic Web, 2012
- [17] Bechhofer S., De Roure D., Gamble M., Goble C., Buchan I.: Research objects: Towards exchange and reuse of digital knowledge. In: he Future of the Web for Collaborative Science, 2010.
- [18] Belhajjame K., Zhao J., Garijo D., Gamble M., Hettne K., Palma R., Goble C.: Using a suite of ontologies for preserving workflow-centric research objects. In: Web Semantics: Science, Services and Agents on the World Wide Web, 2015.
- [19] Piccolo S.R., Lee A.B., Frampton M.B.: Tools and techniques for computational reproducibility. In: bioRxiv, vol. 022707, 2015.
- [20] Santana-Perez I., Prez-Hernandez M.S.: Towards Reproducibility in Scientific Workflows: An Infrastructure-Based Approach. In: Scientific Programming, vol. 2015, p. 11, 2015.
- [21] Banati A., Kacsuk P., Kozlovsky M.: Minimal sufficient information about the scientific workflows to create reproducible experiment. In: IEEE 19th International Conference on Intelligent Engineering systems (INES), Slovakia, 2015.
- [22] Banati A., Kacsuk P., Kozlovsky M.: Reproducibility analysis of scientific workflows; Acta Politechnica Hungarica, unpublished