Four level provenance support to achieve portable reproducibility of scientific workflows

A. Bánáti[1], P. Kacsuk[2,3] and M. Kozlovszky[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, kozlovszky.miklos}@nik.uni-obuda.hu,
kacsuk@sztaki.mta.hu

Abstract - In the scientist's community one of the most vital challenges is the issue of reproducibility of workflow execution. In order to reproduce the results of an experiment, on one hand provenance information must be collected and on the other hand the dependencies of the execution need to be eliminated. Concerning the workflow execution environment we have differentiated four levels of provenance: infrastructural, environmental, workflow and data provenance. During the re-execution at all levels the components can change and capturing the data of each levels targets different problems to solve. For example storing the environmental and infrastructural parameters enables the portability of workflows between the different parallel and distributed systems (grid, HPC, cloud). The describers of the workflow model enable tracking the different versions of the workflow and their impacts on the execution. Our goal is to capture the most optimal parameters in number and type as well and reconstruct the way of data production independently from the environment. In this paper we investigate the necessary and satisfactory parameters of workflow reproducibility and give a mathematical formula to determine the rate of reproducibility. These measurements allow the scientist to make a decision about the next steps toward the creation of reproducible workflows.

I. INTRODUCTION

Nowadays in many scientific research fields the experiments are especially compute and data intensive processes. The execution of these experiments in most cases requires parallel and distributed infrastructures.

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 *job*s, 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.).

Different users for different purposes may be interested in reproducing the workflow, for example scientists (author of workflow) in order to prove their results, readers or other scientists in order to reuse results or reviewers in order to verify the correctness of the results [1]. The scientist community has to face two important challenges related on the reproducibility of results: on one hand more and more metadata and provenance data are necessary about the infrastructure, the environment, the changes and the partial results of an execution in order to reconstruct the execution at a later time. The collected data called provenance help to store the actual environments, the partial and final data products and system variables [2], [3]. The saved information can be used in many fields concerning workflows (failure tolerance, optimization, user steering and dynamism [4], [5]). During the processing of scientific workflows four components can change: infrastructure, environment, abstract workflow model and input data. The provenance information collected at the different levels can support different features, services and goals. On the other hand the preparing reproducible workflows require careful and precise design which eliminates the dependencies and includes detailed description about the model and input/output data. Examples of dependencies can be a special hardware requirements, demand of third party or special local services or the usage of random based computing.

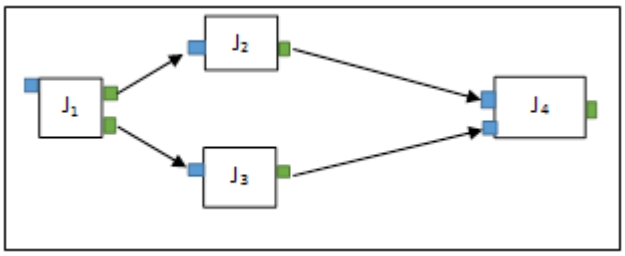

Figure 1. Workflow example with four jobs (J_1, J_2, J_3, J_4)

In this paper we investigate the necessary and satisfactory parameters of reproducibility, we examine the possibility and advantages of a four level provenance. We also give a mathematical model to measure the rate of reproducibility (RoR) to support the scientist in designing and creating reproducible workflow.

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 necessary and satisfactory requirements of scientific workflow reproducibility including our four level provenance structure. In section 5 we give a mathematical model and define the dependency functions to determine

the rate of reproducibility of a given workflow. Finally we summarize our conclusions and reveal the potential future research directions.

II. RELATED WORKS

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 [6], [7], [8], [9], [10] 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 [11]. 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 uninsurable: 1. taking a digital copy of the entire environment using a system virtual machine/hardware virtualization approach 2. capturing and storing metadata about the code and environment that allows it to be recreated later [11].

Zhao et al. [12] 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. [13] in their paper list ten best practice to prevent the workflow decay. Grothe et al. [14] analyze the characteristic of applications used by workflows and list the requirements in order to enable the reproducibility of results and determination of provenance. Compared 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 [15], [16], [17], which allow the researcher and scientist to create reproducible workflow. With help of VisTrail [15], [18] 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 [16] is another tool, which stitches together the detailed provenance information and the environmental parameters into a self-contained reproducible package.

III. NECESSARY AND SATISFACTORY REQUIREMENTS

The two main difficulties of reproducing a workflow are dependencies of workflow execution and collection of provenance data. The former can be determined as necessary requirements and the latter one gives the satisfactory requirements.

A. *Necessity – Dependencies*

The execution of a workflow may require many resources, such as third party or local services, database services or even special hardware infrastructure. These resources are not constantly available, they can change their location, their access condition or the provided services from time to time. These conditions, which we refer to as dependencies, significantly complicate the chances of reproducibility and repeatability. We have classified the dependencies into three categories: infrastructural dependency, data dependency and job execution dependency as shown in table 1.

TABLE I. CATEGORIES OF DEPENDENCIES OF WORKFLOW EXECUTION

infrastructural	data	job execution
• spec. hardware demand	• changing • TP demand • local spec demand	• deterministic • dependency between jobs • Third Party demand • Local spec demand

By infrastructural dependency we mean special hardware requirements, which are available solely on the local system or not evidently provided by other systems, such a special processing unit (GPU, GPGPU).

In the group of data dependency we listed the cases which does not guarantee the accessibility of the input dataset in another time interval. The causes can be that the data is provided by a third party or special local services. Occasionally the problem origins from the continuously changing and updated database that stores the input data. These changes are impossible to restore from provenance data.

The job execution can also depend on a third party or local services, but the main problem arise when the job execution is not deterministic. The operation of GPU or GPGPU are based on random processes consequently the results of re-executions may differ. More over if the dependency factor is too high between the jobs the reproducibility is harder to guarantee.

These conditions are all necessary to perform reproducibility of workflow execution. In the next section (IV) we give a mathematical formula to determine the rate of reproducibility of a given workflow. With help of this measurement the scientist can see how much part of the workflow can be reproducible with 100 percent at a later period of time. Knowing this information the scientist can decide to apply for example an extra provenance policy with extra resource requirement, which stores the whole third party data or apply virtual machine towards the reproducibility.

B. *Satisfying – Provenance*

We defined four levels of provenance data because during the execution of workflow four components can change that would affect the reproducibility: the infrastructure, the environment, the data and the workflow model.

The first is a system level provenance, which stores the type of infrastructure, the variables of the system and the timing parameters. At this level happens the storing the details of the mapping process as a results we can answer the question of what, where, when and how long has been executed. This information supports the portability of the workflow which is a crucial requirement of reproducibility.

The environmental provenance stores the actual execution details which includes the operating system properties (identity, version, updates, etc), the system calls the used libraries and the code interpreter properties. The execution of a workflow may rely on a particular local execution environment, for example, a local R server or a specific version of workflow execution software, which also has to be captured as provenance data or virtual machine snapshot.

The third category is data provenance. In the literature the provenance often refers to data provenance, which deals with the lineage of a data product or with origin of a result. With this data provenance we can track the way of the results and dependency between the partial results. This information can support the visualization, the deep and complete troubleshooting of the experimental model, the proving of the experiment but first of all the reproducibility. In addition in one of our previous paper [Krakko] we investigated the possibility and the need of user steering. We found that some parameters, filter criteria and input data set need to be modified during execution, which rely on data provenance.

The last provenance level tracks the modifications of the workflow model. The scientist during the workflow lifecycle often performs minor changes, which can be undocumented and later it is difficult to identify or restore. This phenomenon is usually referred as workflow evolution. Provenance data collected at this level can support the workflow versioning.

These structured provenance information of a workflow can support reproducibility at different levels if it meets the requirements of independency discussed before. In addition extra provenance information can be stored in that cases, in which however the workflow contains some dependencies but these dependencies can be eliminated with usage of extra resources.

IV. MEASUREMENT OF REPRODUCIBILITY

A. *The model*

The *workflow* (Wf) is represented by the set of job, the *job* (*J*) is defined by the series of input and output ports and the *dataflow* is considered between the given output port of a job and input port of another job:

- $WF=\{J_1, J_2, \ldots, J_N\}$,
- $J_i(p_1, p_2, \ldots, p_{Ki}, P_1, P_2, \ldots, P_{Li})$
- $dataflow(J_i, P_l;J_j;p_k)$

where N, K_i, L_i are natural numbers and denote the number of jobs in a given workflow, the number of input ports (p_i) and output ports (P_i) in a given J_i job.

B. *Dependency functions*

In order to determine the measure of reproducibility of a given workflow we need to define six binary dependency functions. These functions can indicate whether a given job or port depends on any services, input data, random value or each other, in which cases the reproduction of workflow is not possible or especially difficult. We call these criteria together *non-repro* criteria.

$$Dep_{service}(J_i) = \begin{cases} 1, \text{ if the } J_i \text{ job calls a Third party or special local service} \\ 0, \text{ else} \end{cases} \quad (1)$$

$$Dep_{infra}(J_i) = \begin{cases} 1, & \text{if the } J_i \text{ job depends on a special hardware} \\ 0, & \text{else} \end{cases} \quad (2)$$

$$Dep_{input}(J_i, p_j) = \begin{cases} 1, \text{ if the } p_j \text{ input port of the } J_i \text{ job depends on third party, local or changing data} \\ 0, \text{ else} \end{cases} \quad (3)$$

$$Dep_{random}(J_i) = \begin{cases} 1, & \text{if the } J_i \text{ job depends on true random generator} \\ 0, & \text{else} \end{cases} \quad (4)$$

$$Dep_{all}(J_i) = \sum_{i=1}^{N} \left(Dep_{service}(J_i) + Dep_{infra}(J_i) + Dep_{random}(J_i) \right) + \\ + \sum_{i=1}^{N} \sum_{j=1}^{K_i} Dep_{input}(J_i, p_j) \quad (5)$$

$$Dep_{job}(J_i, J_j) = \begin{cases} 1, & \text{if the } J_i \text{ job depends on } J_j \text{ and } Dep_{all}(J_j) \geq 1 \\ 0, & \text{else} \end{cases} \quad (6)$$

The functions (1), (2), (3), decide whether a given job or input port use third party services, local services, or the execution of the job requires special hardware. The function (4) examines the determinism of the results of a given job. A result of a job is deterministic, if during repeated executions the same input data give the same results. For example a computation is non deterministic, if it based on random process and random values or the job requires human intervention. The function (5) returns 1, if at least one of the above mentioned criteria is true. The last function examines the relation between a given job and all other jobs, and it returns 1, if the given job is based on a job which depends on a non-repro criteria (one or more of the functions (1), (2), (3), (4), (5) return 1).

C. *Rate of Reproducibility*

After determining the dependency functions we can define the *Decay*(J_i) function, which returns 1 if the job is not reproducible.

$$Decay(J_i) = \begin{cases} 1 \ \textit{if} \ Dep_{all}(J_i) + \sum_{j=1}^{N} Dep_{job}(J_i, J_j) \geq 1 \\ 0 \ \textit{else} \end{cases} \quad (7)$$

With help of function (7) we can define the rate of reproducibility, RoR:

$$RoR(Wf) = \frac{\sum_{i=1}^{N} Decay(J_i)}{N} \quad (8)$$

The RoR function says how big part of a given workflow is surely reproducible. At this point, the user – before finally submits his workflow – can think over the model, after viewing the results he can modify the model and eliminates certain dependencies or he can decide to apply extra provenance or virtualization tools to preserve the workflow.

V. CONLUSION

In this paper we investigated the necessary and satisfactory requirements of scientific workflow reproducibility. We approached the problem from two perspectives. On one hand we deal with the design and preparation of a workflow. Many best practices for appropriate designing exist in the literature, but one of the most important issue investigates the dependencies of execution (infrastructural dependency, services dependency or random based compute dependency), which add more difficulties to the reproducibility of a workflow at a later time. To eliminate these dependencies there are the necessary requirements of reproducibility. We gave a mathematical model to describe the execution dependencies and we defined the rate of reproducibility. Our goal is to support the scientists to make their experiment reproducible or to support in using some other tools (extra provenance data collection or virtual machine) to achieve reproducibility.

The second perspective is the possible structures of provenance. We determined a four level provenance, which gives the satisfactory requirements of the reproducibility. The information captured at the different levels can support the different levels of reproducibility, such portability or repeatability.

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”, Internatioonal Conference on Computational Science, ICCS 2011. [Online]. Available: http://www.sciencedirect.com.

[2] J. Cheney, A. Finkelstein, B. Ludäscher, és S. Vansummeren, „Principles of provenance (dagstuhl seminar 12091)”, Dagstuhl Reports, köt. 2, sz. 2, 2012

[3] S.M.S da Cruz, Maria Luiza M. Campos, M. Mattoso. „Towards a Taxonomy of Provenance in Scientific Workflow Management Systems”, 259–66. IEEE, 2009. doi:10.1109/SERVICES-I.2009.18.

[4] Bánáti, A., Kacsuk, P., Kozlovszky, M.: Towards Flexible Provenance and Workflow Manipulation in Scientific Workflows. In Proceedings of CGW 14 (2014)

[5] Kail, E., Bánáti, A., Kacsuk, P., Kozlovszky, M.: Provenance based adaptive and dynamic workflows. In: 15th IEEE International Symposium on Computational Intelligence and Informatics, pp 215-219, IEEE Press, Budapest (2014).

[6] P. Missier, S. Woodman, H. Hiden, és P. Watson, „Provenance and data differencing for workflow reproducibility analysis”, Concurrency and Computation: Practice and Experience, 2013

[7] R. D. Peng, „Reproducible Research in Computational Science”, Science, köt. 334, sz. 6060, o. 1226–1227, dec. 2011

[8] J. P. Mesirov, „Accessible Reproducible Research”, *Science*, köt. 327, sz. 5964, o. 415–416, jan. 2010.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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

[14] 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.

[15] 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

[16] F. S. Chirigati, D. Shasha, és J. Freire, „ReproZip: Using Provenance to Support Computational Reproducibility.”, in *TaPP*, 2013.

[17] 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.

[18] D. Koop, J. Freire, és C. T. Silva, „Enabling Reproducible Science with VisTrails”, *arXiv preprint arXiv:1309.1784*, 2013.