UniProv - Provenance Management for UNICORE Workflows in HPC Environments

André Giesler[0000-0002-7929-4341], Myriam Czekala[0000-0002-7389-2623], Björn Hagemeier[0000-0003-1528-0933]

Juelich Supercomputing Centre, Forschungszentrum Juelich GmbH, Juelich, Germany
{a.giesler,m.czekala,b.hagemeier}@fz-juelich.de

Abstract. The goal of comprehensive provenance tracking in the scientific environment should be the inclusion of the entire life cycle of data management. Thus, the data collection process begins with the registration of lab-generated or sensor-generated data, continues to organize and manage data in the storage repositories, processing analysis and simulation data on clusters and HPC systems, and finally referencing and verifying computational results in scientific publications. In the associated provenance tracking life cycle, UniProv initially concentrates on the processing and simulation of data in scientific workflows used in particular on supercomputers in the HPC environment. In this context, UniProv aims to create the core of a provenance management framework that can be extended in order to integrate different sources of the scientific provenance cycle. Here UniProv should facilitate the creation, the standardized formalization, the storage and the retrieval of Provenance Information.

Keywords: Provenance, Scientific Workflows, Interoperability, PROV-O, Neo4j, UNICORE.

1 Introduction

Scientific workflows are an integral part of using High-performance Computing (HPC) systems in data centers. Some users rely on workflows that consist of a sequence of logically concatenated programs and scripts between which intermediate results are transferred and processed. Another group of users applies Workflow Management Systems (WfMS) to enable complex, coherent simulations and calculations. These tools provide a higher level of abstraction in terms of predefined structures, the channeling of well-defined input and output data, standardization of analysis methods and monitoring of all tasks within a workflow. The latter allows a continuous automated provenance tracking of scientific workflows by capturing information from the processing logic of the WfMS.

2 Interoperable Provenance Framework for UNICORE

The UNICORE[1] federation software suite includes a generic WfMS [1], which is currently being used increasingly in the field of neuroscience [2]. UNICORE does not provide proprietary provenance management. It is thus the initial goal of the UniProv framework to process provenance information in UNICORE-based workflows to allow UNICORE users to track their submitted workflows. For this purpose, UniProv was initially integrated into the existing UNICORE service architecture. Based on the UNICORE implementation UniProv is to be converted later to an autonomous framework, to which further provenance providers can be attached. In this context, UniProv pursues the following concept:

- Monitoring and extracting provenance information from UNICORE Job and Workflow management services
- Processing of collected provenance information into an interoperable provenance model according to W3C-PROV[2]
- Storing the provenance graph in a Neo4j graph database and allowing arbitrary queries on the data

UniProv takes an interoperable approach in two respects: firstly, the potential processing of the resulting provenance data should be enabled in other standard compliant provenance services. Secondly, the later integration of such provenance providers should be facilitated, which are capable of generating standard compliant provenance data. For this reason, all provenance data is modeled based on the recognized W3C-PROV standard. UniProv uses the PROV-O ontology to express generated provenance information in compatibility with PROV. Furthermore, PROV acts as a reference model to create application-specific ontologies for UniProv.

For instance, the general complexity of scientific workflows and processing WfMS cannot be mapped sufficiently with the capabilities of the generic PROV standard. For this reason, UniProv integrates the ProvONE ontology developed in the context of the DataONE[3] project and used in the context of the Pbase workflow provenance repository [3]. The advantage of ProvONE lies in its variable specification of the cyclical sequences of workflow logic, so that it can be very well modeled on UNICORE based workflows. UniProv benefits from the conceptual decoupling of workflow provenance into a prospective, a retrospective and a data-specific provenance view. In terms of UNICORE-based workflows, this has the advantage that the static workflow definition persisted in an XML file as well as the runtime information captured from the UNICORE workflow engine can now be mapped in a common provenance graph. Additionally, the appropriate data flow of the workflow trace can be mapped and added accordingly.

However, even the ProvONE ontology is not sufficient to map the entire resources, elements and structures of a UNICORE workflow running on an HPC machine. To

1 https://www.unicore.eu/
2 https://www.w3.org/TR/prov-overview/
3 https://www.dataone.org/

achieve this, ProvONE[4] was specialized in the form of the UniPROV[5] ontology. It introduces the vocabulary that UNICORE uses for specifying structures such as workflow loops, groupings, or synchronizations. For example, in ProvONE, the logical dependency of two consecutive computational jobs in a workflow definition is described by the class `Controller`. In UNICORE, the class `Transition` inheriting from `Controller` describes the transition from one job to the next, and contains an additional condition describing that a job will not be executed until the associated condition is met. In addition, the UniProv ontology provides the ability to include annotations from users in the provenance graph. The UniProv ontology was serialized in the OWL2 format such as PROV-O and the ProvONE ontology. All three ontologies form the vocabulary of the UniProv framework to map UNICORE based workflows. Using the Apache Jena RDF and Ontology API, Java interfaces were generated from the ontologies so that compliant provenance data can be modeled in the UniProv framework when processing UNICORE workflows.

3 Capturing and Storing Provenance Data

For extracting and processing provenance information from UNICORE, loggers have been implemented and integrated into the UniProv framework. The Job Logger hooks into the respective job management service of a UNICORE installation on a supercomputer and tracks the status of running computation jobs, the respective program code, defined environment variables, the job properties (number of cores per node, RAM, walltime, etc.), the use of imported files, and the generation of result data. In addition, UNICORE job monitoring provides runtime information about the status of the submitted job, hostnames, or the runtime of jobs.

The UNICORE workflow engine controls the logical sequence and synchronization of interdependent jobs as well as the flow of data within a workflow. This information is captured by the UniProv Workflow Logger, which is triggered by status messages from the workflow engine. It then traverses all nodes and edges of the workflow graph to capture current runtime information. Once a single job of the workflow is successfully completed, the provenance information is prepared by the Job Logger of the corresponding UNICORE instance, sent to the Workflow Logger, and added to the internal UniProv workflow model. In this process, the WfModelOntProcessor module processes the workflow model to the overall provenance graph by using the Apache Jena RDF and Ontology API to generate ontology compliant provenance data. The provenance graph is serialized by default in Turtle RDF, but can easily be managed to a different syntax like XML or JSON.

UniProv supports the storage of provenance data in repositories based on a Neo4j graph database, which was inspired by the scientific work in [4] and [5]. Such a database model allows a natural mapping of graph structures as modeled for workflow provenance data in UniProv. One advantage of Neo4j is the support of highly con-

[4] https://purl.dataone.org/provone/2015/01/15/ontology#
[5] https://datapub.fz-juelich.de/uniprov

nected data as given in provenance graphs, particular as application-specific provenance data is often subject to evolutionary changes. In this context, the schema-less character of a graph database allows more flexibility than for example a relational database where schema redesigning can be a time-consuming process. In Neo4j, it is possible to add new data or relationships based on revised ontologies on the fly. On the other hand, the connected graph structures of provenance data, especially of workflow provenance, can cause complex nested queries along the nodes and edges of a graph. A graph database such as Neo4j allows easy traversing of the data due to its graph-oriented Cypher query language, while complex JOINs would be expected for traversal queries in a relational database.

Since UniProv generates provenance output in PROV-based Turtle RDF syntax for interoperability reasons, the created provenance graph must be migrated into the Labeled Property Graph (LPG) model of Neo4j. For that reason, the UniProv framework has been supplemented by the Neosemantics module representing an extension point of Neo4j that is used to import existing RDF provenance data into the graph database without the loss of its graph nature. The UniProv Neosemantics extension can be installed in any Neo4j 3.x database as a plugin.

As future work, we intend to support Persistent Identification[6] (PID) of data in UniProv in order to provide a more consistent data flow tracking. We plan to support and integrate other WfMS in UniProv such as Snakemake[7], which is also a tool frequently used in the domain of neuroscience.

Acknowledgements. The authors wish to thank all people and institutions involved in LSDMA. We also thank the German Helmholtz Association for funding.

References

1. Demuth, B., Schuller, B., Holl, S., Daivandy, J., Giesler, A., and Huber, V., Sild, S.: The UNICORE Rich Client: facilitating the automated execution of scientific workflows. In: 2010 IEEE Sixth International Conference on e-Science (e-Science), pp. 238–245, Brisbane, Australia (2010)
2. Amunts K., Bücker O. and Axer M. Towards a Multiscale, High-Resolution Model of the Human Brain (Cham: Springer International Publishing) 3-14, Cetraro, Italy (2013)
3. Cuevas, V.,Kianmajd, P., Ludäscher, B. et.al.: The PBase Scientific Workflow Provenance Repository, In International Journal of Digital Curation, 9(2):28–38. (2014)
4. Brauer P.C., Fittkau F., Hasselbring W. The Aspect-Oriented Architecture of the CAPS Framework for Capturing, Analyzing and Archiving Provenance Data. In: Ludäscher B., Plale B. (eds) Provenance and Annotation of Data and Processes. IPAW 2014. Lecture Notes in Computer Science, vol 8628. Springer, Cham (2014)
5. Heinis, T., Chapman, A.: Provenance Storage. In, Liu, Ling and Ozsu, M. Tamer (eds.) Encyclopedia of Database Systems. Springer New York (2017)

6 https://eudat.eu/services/userdoc/pids-in-eudat
7 https://snakemake.readthedocs.io/en/stable/