A Graph Data Model-based Micro-Provenance Approach for Multi-level Provenance Exploration in End-to-End Climate Workflows

Sandro Fiore
Department of Information Engineering and Computer Science
University of Trento
Trento, Italy
sandro.fiore@unitn.it

Mattia Rampazzo
Department of Information Engineering and Computer Science
University of Trento
Trento, Italy

Donatello Elia
Advanced Scientific Computing Division
CMCC Foundation
Lecce, Italy

Ludovica Sacco
Department of Information Engineering and Computer Science
University of Trento
Trento, Italy

Fabrizio Antonio
Advanced Scientific Computing Division
CMCC Foundation
Lecce, Italy

Paola Nassisi
Advanced Scientific Computing Division
CMCC Foundation
Lecce, Italy

***Abstract*— Open Science is a vital part in the current and future research agenda worldwide. In order to meet Open Science goals, it is of paramount importance to fully support the research process, which includes also properly addressing provenance and reproducibility of scientific experiments. Indeed, provenance and reproducibility are two key requirements for analytics workflows in Open Science contexts. Handling provenance at different levels of granularity and during the entire experiment lifecycle becomes key to properly and flexibly managing lineage information related to large-scale experiments as well as enabling reproducibility scenarios. To this end, this work introduces the micro-provenance concept, and it provides an in-depth description of its design, implementation and exploitation in the context of a multi-model climate analytics workflow.**

Keywords*— *micro-provenance, graph data model, reproducibility, Open Science, climate domain

I. INTRODUCTION

Open Science is a vital part in the current and future research agenda worldwide. One of the most prominent initiatives run by the EC is the European Open Science Cloud (EOSC) which *"aims to create a trusted environment for hosting and processing research data to support EU science in its global leading role"* [1]. Open Science leverages new ways to perform research and share the results through open digital technologies and collaborative tools [2] embracing the concepts of Open Access, Open Data, Open Source software and Open reproducible research [3][4]. Besides accessibility, an Open Science research culture fosters transparency, enabling trust and recognition as well as providing basic foundations for reproducibility [5]. Several efforts in literature have been made towards reproducibility [6][7][8]. Moreover, reproducibility implicitly fosters re-usability, which is one of the FAIR guiding data principles [9][10].

In such a context, provenance can be a crucial element to enable scientific experiments reproducibility.

This paper focuses on a multi-level *provenance* approach in *climate workflows*, as a way to explore data lineage across multiple dimensions and to provide in-depth knowledge and new tools to inspect, document and navigate climate experiments. The yProv micro-provenance service, a new core component within an Open Science-enabled research data lifecycle, is introduced in this work by highlighting the design and implementation aspects as well as it graph-based data model.

The rest of the paper is organized as follows: Section I provides the motivation and definitions about the micro-provenance paradigm, whereas Section II discusses the current state of art in the area; Section III presents the design of the yProv micro-provenance service, while Section IV discusses the implementation aspects. Section V presents an in-depth discussion of a micro-provenance use case addressing multiple aspects such as: scientific context, use case design and implementation, input data, workflow strategies and micro-provenance results. Finally, Section VI draws the final conclusions, highlighting future work.

II. MICRO-PROVENANCE: MOTIVATION AND DEFINITIONS

Provenance, i.e., the description of the different stages data has undergone during the analysis process, from its origin to the outcome, provides crucial information for scientific experiments in the climate domain and beyond. Indeed, a complete provenance record can enable reproducibility scenarios, which are becoming a mandatory target within Open Science contexts, particularly in the climate domain.

Climate experiments are usually documented in terms of provenance from a very high-level perspective, without

delving into the internals of a specific workflow task. As climate experiments are rapidly evolving towards end-to-end workflows including simulation runs, data pre-post processing, data-driven and data-intensive applications, task heterogeneity can lead to different needs from scientists about the exploration of specific provenance information. For instance, someone could be interested in the overall set of macro-tasks in the workflow to provide the provenance metadata of the final data products, while others could be more intrigued by the data lineage information of a data-intensive step consisting of a Directed Acyclic Graph (DAG) of hundreds or thousands of data analytics kernels.

It becomes clear that a more articulated and multifaceted end-to-end approach for scientific experiments requires a new vertical dimension to navigate and explore the provenance metadata, beyond the usual horizontal one.

This concept leads to the **micro-provenance** definition as a "*multi-dimensional metadata description to navigate within the provenance space across different axes".*

The present paper addresses the horizontal and vertical dimensions of provenance, by providing a first implementation of the multi-level concept. Related to the previous definition, we define the **micro-provenance service** as *"the software component in charge of handling the micro-provenance metadata".*

It is worth mentioning the micro-provenance concept can contribute towards (i) stronger traceability, transparency and trust, through a richer and detailed set of metadata); (ii) multi-dimensional exploration/navigation of provenance metadata (by definition), (iii) knowledge extraction, through data mining techniques) on the micro-provenance database, and finally (iv) fine-grained tasks understanding as a means/tool to address the explainability of complex processes (i.e., Machine Learning/Deep Learning).

III. State of the Art

A. Main research projects

Two major projects address linking of resources via graph data model approaches, namely: Freya[1] and OpenAire[2].

Freya [16] is a European Commission-funded project established with the aim of interconnecting resources with persistent identifier (PID), a unique identifier for entities of scientific interest such as publications, datasets, or people. These PIDs also carry metadata that make it possible to unambiguously relate resources belonging to the same type or to different types, thus allowing the creation of a graph (PID Graph [11]) that is used as a basis for other services.

OpenAire (Open Access Infrastructure for Research in Europe) is a digital infrastructure funded by the European Union that offers a variety of services for Open Science. By exploiting a graph data model approach, it aims to offer services about scientific publications and results, such as search, publication management, validation, and more [12].

While the proposed work shares the graph database approach for storing information about connected resources with the two projects above, it also differs from them for two main reasons. First, it focuses on tracking detailed provenance information about scientists' workflows and experiments, and provides a rich set of lineage metadata in terms of (i) tasks performed, (ii) data (input, output, but also intermediate products), (iii) type of storage (persistent vs in-memory data structures), and (iv) the set of steps that connect input to output datasets, therefore laying the ground for a reproducibility framework. Second, our work also introduces the multi-level concept as a way to navigate through the different levels of provenance, and so distinguishing between coarse and fine-grain (micro-provenance) steps within the same climate analytics workflow.

B. W3C PROV implementations

The W3C PROV [13] family of standards plays a crucial role in addressing interoperability. At present, the best implementations of W3C PROV are the following:

- ProvStore[3]: an online repository for storing provenance documents both privately and publicly. It features a Web interface and a REST API [14].
- Validator[4]: a web service from openprovenance.org for validating provenance descriptions against the PROV specification.
- Translator[5]: a web service from openprovenance.org for translating between the various representations supported by PROV.
- ProvToolbox[6]: a Java library that allows the creation and conversion of PROV model representations.
- ProvJS[7]: an implementation of the PROV standard for JavaScript.
- Prov Python[8]: an open source Python library created according to the PROV data model specification. It allows the creation of in-memory representations of provenance documents and the gradual addition of records to them. It also offers support for serialization and deserialization in PROV-O, PROV-XML and PROV-JSON and export to various graphical formats.
- prov2neo[9]: a Python library for efficiently importing PROV documents to Neo4j [15].
- Git2PROV[10]: an application for exporting provenance data from a Git repository into the PROV standard.

IV. yProv Service Design

This section deals with several design aspects of the yProv service, such as the requirements elicitation, the architectural design, the service modes and the API. The challenge is to fully address micro-provenance in end-to-end scientific workflows, through a lightweight and interoperable service, based on a simple architecture as well as a graph data model and a well-defined API.

[1] https://www.project-freya.eu/en
[2] https://www.openaire.eu
[3] https://openprovenance.org/store/
[4] https://openprovenance.org/services/view/validator
[5] https://openprovenance.org/services/view/translator
[6] https://github.com/lucmoreau/ProvToolbox
[7] https://bitbucket.org/provenance/provjs/src/master
[8] https://github.com/trungdong/prov
[9] https://github.com/DLR-SC/prov2neo
[10] https://github.com/IDLabResearch/Git2PROV

A. Requirements elicitation

Starting from the main motivation reported in Section I, the key requirements for a micro-provenance service are listed below:

- Scalability: running complex end-to-end scientific workflows implies tracking a myriad of tasks/sub-tasks, spanning across multiple provenance levels. There is a strong need to provide scalable solutions to properly implement the micro-provenance paradigm.
- Multi-level support: according to the micro-provenance definition, the related service must properly deal with the provenance space and the different levels.
- Interoperability: the micro-provenance service must expose an interoperable interface and be compliant with the W3C PROV family of standards.
- Data model: the micro-provenance service must adopt a data model that values relationships as a first-class citizen. In this respect, graph data models can turn out to be very flexible in modelling relationships, incorporating new classes of information over time, and helping an agile development of the system.
- Consistency: community-based vocabularies should be integrated into the micro-provenance service to ensure consistent tracking of provenance information.

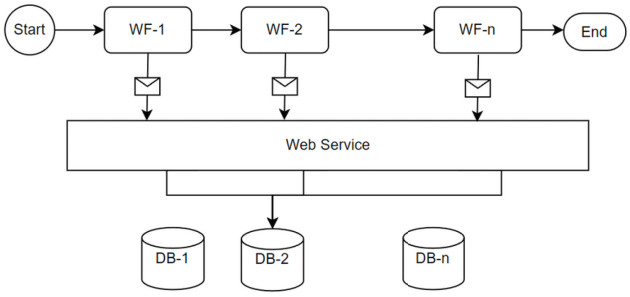

Figure 1: yProv service chunked mode.

B. yProv service architectural design

The yProv service architecture follows a simple web-service paradigm. A client connects to the web service front-end through the exposed API.

The web service in turn acts as a client for the graph-database back-end, thus implementing all the needed CRUD (Create, Read, Update, Delete) logic to manage provenance information. Fig. 1 shows how a generic workflow can track lineage information step-by-step by calling the provenance web service API.

Provenance of different workflows can then be stored into different graph databases, thus ensuring isolation with respect to the provenance management of separate experiments.

C. yProv service modes

The yProv service has been designed to support two different modes, i.e., *one-shot* and *chunked*. In the former case, provenance information is entirely sent to the web service at the end of the workflow through a single call, whereas in the *chunked* mode, provenance information is sent in separate chunks to the web service during workflow execution. While the first mode limits (to one) the number of interactions with the web service and it is well-suited for small-medium workflows, the second one streams provenance information at run-time in separate chunks to the web service, thus enabling real-time inspection of provenance information by the end-users, while the workflow is running.

	GET	PUT	POST	DELETE
/documents	Get names of all documents already stored			
/documents/<doc_id>	Get a json representation of document <doc_id>	Create or Update document <doc_id>		Delete document <doc_id>
/documents/<doc_id>/ entities			Create entity <e_id> on document <doc_id>	
/documents/<doc_id>/ entities/<e_id>	Get a json representation of entity <e_id> of document <doc_id>	Create or Replace entity <e_id> of document <doc_id>		Delete entity <e_id> and its relations from document <doc_id>
/documents/<doc_id>/ activities			Create activity <ac_id> on document <doc_id>	
/documents/<doc_id>/ activities/<ac_id>	Get a json representation of activity <ac_id> of document <doc_id>	Create or Replace activity <ac_id> of document <doc_id>		Delete activity <ac_id> and its relations from document <doc_id>
/documents/<doc_id>/ activities			Create activity <ac_id> on document <doc_id>	
/documents/<doc_id>/ agents/<ag_id>	Get a json representation of agent <ag_id> of document <doc_id>	Create or Replace agent <ag_id> of document <doc_id>		Delete agent <ag_id> and its relations from document <doc_id>
/documents/<doc_id>/ activities			Create agent <ag_id> on document <doc_id>	
/documents/<doc_id>/ relations/<r_id>	Get a json representation of relation <r_id> of document <doc_id>	Create or Replace relation <r_id> of document <doc_id>		Delete relation <r_id> from document <doc_id>

Table 1: yProv service API and mapping with HTTP verbs.

D. yProv service API design

The yProv API has been designed by following the REST principles. A simplified list of methods and the corresponding action for each HTTP verb is reported in Table 1.

More in detail, five classes of resources have been identified, namely *document*, *entity*, *activity*, *agent,* and *relation*.

Document is the abstraction of full provenance information associated to a workflow. For each abstract document there is a one-to-one association with a graph database. *Entity*, *activity,* and *agent* are sub-resources of the document resource, and they allow performing the CRUD operations on the three types of elements stored into a provenance graph according to the W3C PROV data model, thus ensuring a fine-grain control on each node stored in the graph database; similarly, relation allows acting on each individual relationship of the graph.

V. yPROV SERVICE IMPLEMENTATION

From an implementation perspective, the yProv service consists of the following components:

- yProv Command Line Interface (CLI)
- yProv Web Service front-end
- Graph database engine back-end

The yProv CLI provides a set of commands which are basically Python wrappers to the RESTful API calls.

The yProv web service is implemented in Python by using Flask micro-framework, which is based on the Jinja2 Template Engine and Werkzeug WSGI Toolkit. Flask has been preferred to other solutions like Django since it is rather lightweight and easy-to-use to rapidly build RESTful web-services in Python. Additionally, two libraries have been used to develop the API methods, namely Prov Python and py2neo3. More in detail:

- **Prov Python** is an open source Python library created according to the W3C PROV specification. In particular, yProv uses the *prov.model* module that defines the *ProvDocument, ProvRecord, ProvElement, and ProvRelation* classes to provide a very good in-memory representation of a provenance document. This representation is very useful not only for the serialization and deserialization methods but also because it operates checks on the validity of the document against PROV constraints.
- **py2neo3** is a Python library that provides support for connecting and querying Neo4j databases. Basic functionality is provided by the *Graph* class that represents a graph database exposed by an instance of Neo4j. The *Node, Relationship,* and *Subgraph* classes, on the other hand, are the basic types of library that offer methods for interacting with the elements of a graph.

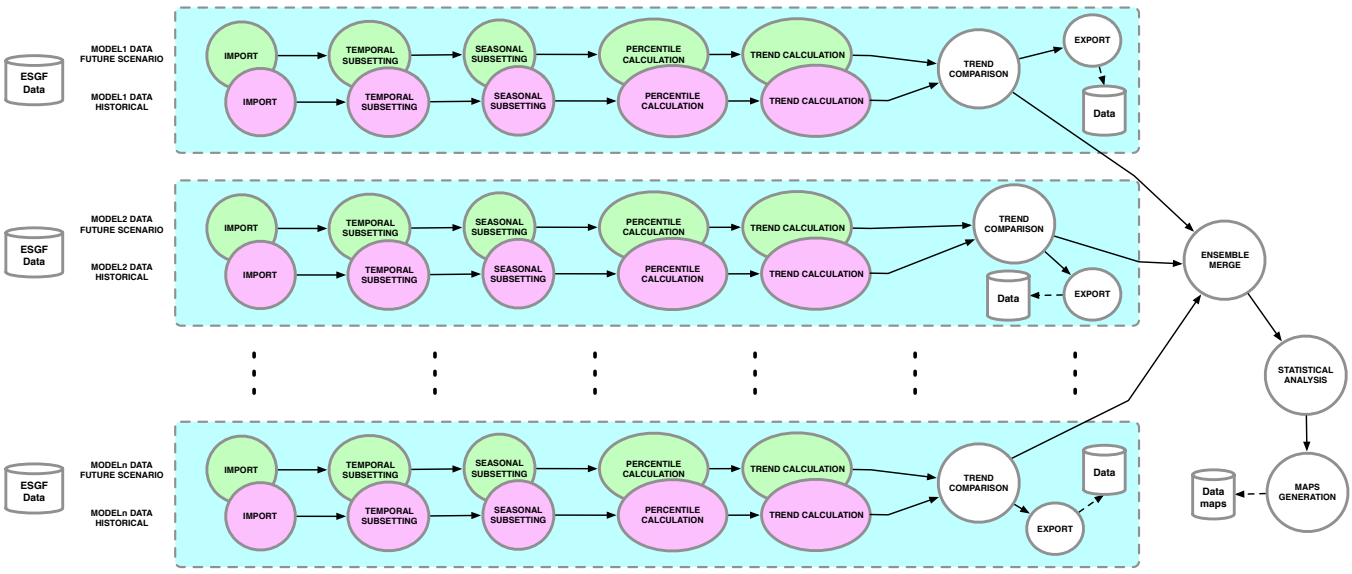

Figure 2. Design (abstract workflow) of the precipitation trend analysis use case

VI. Micro-provenance use case

This section delves into the use case design. It provides some context in the first sub-section to introduce the main source of data for the proposed use case, and it also illustrates the abstract workflow in the second section.

A. Background: CMIP and ESGF

Multi-model data analysis requires access to data from multiple climate models simulations like those produced in the context of the Coupled Model Intercomparison Project (CMIP) [36]. These data are available through the Earth System Grid Federation (ESGF) data archive [17][18], which provides production-level support for search & discovery, browsing and access to climate simulation data and observational data products.

ESGF puts together: (i) a few tens of data nodes from multiple climate modelling centers around the world, (ii) tens of petabytes of data published (most of which relate to the CMIP5 [34] and CMIP6 [20][35] experiments), (iii) tens of thousands of users worldwide, and (iv) more than one billion downloads over the last five years. CMIP data are key for the IPCC Assessment Reports [19].

Data of course are just the input, whereas the infrastructure and the software stack provide respectively the proper capacity and the enabling technologies.

In the CMIP context, *multi-model analyses* are clearly among the most relevant exercises that can be run by scientists. As an example, the multi-model precipitation trend analysis was considered. This use case was also selected as a pilot case in the context of the H2020 INDIGO-Datacloud project [21-23] [33] since it is scientifically relevant and general enough to extrapolate outcomes, patterns and lessons learnt on multi-model analysis, beyond the specific example.

B. Use case design: abstract workflow

Fig. 2 shows the design (abstract workflow) of the multi-model precipitation trend analysis in the CMIP context [29].

The inputs, in NetCDF [26] format, are *precipitation data* related to the historical run and a future scenario of a set of models from the CMIP experiment. The outputs are some *statistical indicators* (maximum, minimum, average, standard deviation, and variance) stored into NetCDF format; maps are also generated for visualization purposes.

Additionally, intermediate products are also stored and retained, for later analysis and re-use.
In terms of workflow, the analysis consists of two major steps (single-model and multi-model), as described below.

1) Workflow step1: single model

The *single-model* step consists in the difference between the precipitation trends evaluated with respect to both the historical and future scenario data on the same model. The step1 steps are shown within blue rectangles. The tasks related to historical data processing are in pink circles, whereas the tasks that process data resulting from the future scenario are in green circles. More in detail, starting from the input data, the trend is calculated by performing the following sub-tasks:

a) Importing the data from files to memory structures.
b) Sub-setting over a 30-year timeframe (2071-2100 for the future scenario and 1976-2005 for the historical).
c) Sub-setting over JJA (June-July-August).
d) 90th percentile calculation (on the JJA data).
e) Linear regression and calculation of the trend coefficient.
f) Comparison (difference) between the historical and future trends.

It should be noted that spatial sub-setting is not performed on the input data, as the analysis is run at a global scale. Moreover, as it can be seen from the Fig. 2, five operators of Step1 are unary (a, b, c, d, e), whereas only one is binary (f).

2) Workflow step2: multi-model

The multi-model step consists in the ensemble analysis on the trend comparison data gathered from the different models as well as computation and storage of the output (five statistical indicators).
More in detail, the multi-model step performs the following sub-tasks:

a) Ensemble analysis on the n outputs gathered from the single model step.
b) Calculation of five statistical indicators
c) Maps generation
d) Output storage on the file system.

As it can be seen from Fig. 2, three operators of Step2 are unary (b, c, d), while one (a) is n-ary.

C. Use case implementation

The micro-provenance use case on the precipitation trend analysis has been implemented by using the Ophidia High Performance Data Analytics framework [37] and the yProv service on the HPC cluster hosted at the SuperComputing Centre of the Euro-Mediterranean Center on Climate Change (CMCC). More info about Ophidia are provided in sub-section E.

D. Input data

Data have been collected through ESGF and downloaded at a central location at the CMCC SuperComputing Center. The details of the datasets gathered from ESGF are reported in Table 2.

E. Ophidia HPDA and workflow strategies

Ophidia provides support for the implementation of *analytics workflows* [30], a collection of an arbitrary set of parallel operators (analytical kernels) organised in the form of a direct acyclic graph (DAG). Each kernel is responsible for addressing a specific core task like data sub-setting, data aggregation, statistical analysis (e.g., max, min, avg) etc. When combined into an analytics workflow, they can address Science use cases like, among others, extreme events analysis, climate indicators computation and anomalies detection. It is worth mentioning that each kernel is based on the hybrid parallelization approach (MPI+OpenMP) and runs over a cluster taking advantage of the in-memory analytics support offered by Ophidia.

Acronym	Expansion	Lat x Lon	Institute
CCSM4	Community Climate System Model, v4	0.9° x 1.5°	National Center for Atmospheric Research (NCAR)
CMCC-CMS	CMCC - Coupled Modeling System	1.9° x 1.9°	Euro-Mediterranean Center on Climate Change (CMCC)
CMCC-CM	CMCC - Climate Model	0.8° x 0.8°	Euro-Mediterranean Center on Climate Change (CMCC)
CNRM-CM5	CNRM - Coupled Global Climate Model, v5	1.4° x 1.4°	Centre National de Recherches Météorologiques (CNRM)/Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique (CERFACS)
CSIRO Mk3.6.0	CSIRO Mark, v3.6.0	1.9° x 1.9°	Commonwealth Scientific and Industrial Research Organisation (CSIRO) in collaboration with Queensland Climate Change Centre of Excellence (QCCCE)
CanESM2	Second Generation Canadian Earth System Model	2.8° x 2.8°	Canadian Centre for Climate Modelling and Analysis (CCCma)
GFDL-CM3	GFDL Climate Model, v3	2.0° x 2.5°	National Oceanic and Atmospheric Administration (NOAA)/Geophysical Fluid Dynamics Laboratory (GFDL)
GFDL-ESM2G	GFDL Earth System Model with Generalized Ocean Layer Dynamics (GOLD) component	2.0° x 2.5°	National Oceanic and Atmospheric Administration (NOAA)/Geophysical Fluid Dynamics Laboratory (GFDL)
GFDL-ESM2M	GFDL Earth System Model with Modular Ocean Model 4 (MOM4) component	2.0° x 2.5°	National Oceanic and Atmospheric Administration (NOAA)/Geophysical Fluid Dynamics Laboratory (GFDL)
HadGEM2-CC	Hadley Centre Global Environment Model, v2 (Carbon Cycle)	1.2° x 2.8°	Met Office (UKMO) Hadley Centre (HC)
HadGEM2-ES	Hadley Centre Global Environment Model, v2 (Earth System)	1.2° x 2.8°	Met Office (UKMO) Hadley Centre (HC)
INM-CM4.0	INM Coupled Model, v4.0	1.5° x 2.0°	Institute of Numerical Mathematics (INM)
IPSL-CM5A-MR	IPSL Coupled Model, version 5, coupled with NEMO, mid resolution	1.2° x 2.5°	L'Institut Pierre-Simon Laplace (IPSL)

Table 2: List of models, with full name, spatial resolution and institute, used for the precipitation trend analysis use case.

Based on that, the use case described in the previous section has been implemented with Ophidia by using two different strategies, i.e., (i) workflow-based and (ii) notebook-based, in both cases exploiting the API made available by the HPDA framework.
More specifically, in the former case, a single JSON file with the entire workflow definition has been submitted to the HPDA framework via a single Ophidia Workflow API call. In the notebook-based strategy, instead, the concrete workflow has been developed as a Jupyter Notebook, by using multiple Python calls to the operators through the PyOphidia API [31].
For the sake of completeness, it is rather interesting to analyze how the two different approaches differ according

to the following dimensions of analysis: approach, mode, library, code, lines of code (LoC) (Table 3).

	Workflow	Notebook
Approach	SSSC*	MSSC*
Mode	Batch	Interactive
Library	Ophidia WF	PyOphidia
Code	JSON	Python
LoC	544	199

Table 3: Main differences between the two workflow implementation solutions for the precipitation trend analysis use case.

To sum up, the native workflow approach requires more coding (544 vs 199 LoC) and exhibits higher complexity (JSON vs Python) with respect to the notebook approach, although it represents a machine-readable format; yet, it requires a single batch server-side call (SSSC) with Ophidia as opposed to the multiple interactive server-side calls (MSSC), one per operator, needed in the Jupyter Notebook solution.
Provenance-wise, the two approaches lead to the same metadata information, so it is up to the developer to choose the most suitable option for any specific data analytics workflow. The following section describes the provenance result obtained in the aforementioned implementations.

F. Micro-provenance result and outcomes discussion

Every time an Ophidia analytics operator is executed, the Ophidia framework stores the related lineage information into a memory structure; at the end of the workflow execution, a JSON representation (based on the W3C PROV standard) is inferred and sent to the yProv service via the yProv document API call.
As a result, the yProv service parses the JSON document, validates it, and adds nodes and edges to the associated provenance graph database running on the Neo4J back-end.
For the precipitation trend analysis workflow, the resulting graph is provided in Figure 3, which represents the output of a web application built on top of the yProv service, to query, retrieve and visualize micro-provenance information. As it can be noted, the output (full graph) is quite articulated and deserves further description to capture all the information it provides.
As it can be seen from the figure, the provenance graph consists of several repeated patterns with two branches each. One of the patterns is represented in box (a). These patterns describe the single-model steps of the use case; more specifically, the two branches represent the historical and future scenario paths in the Step1 of the workflow. For the sake of clarity, the design of the use case is also reported in the same figure (top-right box).
Box (a) contains blue, yellow and red circles, which respectively relate to I/O files, tasks, and in-memory results. According to that, and consistently with the design of the use case, box (a) shows:

- Two input files as well as an (intermediate) output (blue circles), respectively at the beginning and at the end of Step1.
- Several tasks (yellow circles), mostly unary (i.e., JJA subsetting, 90th percentile JJA) and one binary task (the trend comparison sub-task in Step1).
- Several in-memory data (red circles) acting as the output of one task and the input of the subsequent one. Indeed, not all the partial results of the workflow need to be stored on the file system, so they are managed in-memory by default, unless differently specified by the user through a specific "export-to-file" Ophidia operator.

Box (b) represents the first part of Step2, more precisely the one related to the n-ary operator that performs the ensemble computation. As it can be seen, such operator takes as input a set of files (blue circles) which are (i) imported in memory into separated data cubes (red circles) and then (ii) merged by the n-ary operator (central yellow circle) into an ensemble data cube in-memory structure, i.e., the red circle on the right hand-side of box (b).
This kind of data structure represents then the input for the statistical analysis, i.e., box (c), which performs the computation of the five statistical indicators (one branch each). Box (c) also includes five blue nodes which represent the five maps produced as final output of Step2.
As a final note, the provided example represents just one query run against the yProv service, and many other could be performed to retrieve sub-graph information, thanks to the flexible query language support provided by the graph-database technology running in the back-end. Some other examples include:

- A Step1 sub-graph related to a specific climate model (the equivalent of box (a) in Fig. 3).
- A Step2 sub-graph related to the ensemble analysis.
- The input and output data (including PIDs information) to build the data-chain without considering the in-depth details of the performed workflow tasks.

Of course, many other types of queries can be performed, including running more advanced algorithms on the provenance graph, filtering on the provenance level of interest in complex workflows, etc. However, for page limit constraints, advanced queries and knowledge extraction through graph mining algorithms will be further discussed in a future work.
Finally, as general remarks, it is important to point out that (i) the output graph can be queried at run-time during workflow execution, thus providing preliminary exploratory and inspection capabilities to scientists; (ii) the outputs of the queries are provided in a W3C PROV-compliant JSON representation, which ensures interoperability with other services and software components built on top of the same family of provenance standards; (iii) the interoperable formats for inputs and outputs (i.e., JSON) ensure a straightforward integration of the yProv service API into data science applications like Jupyter Notebooks.

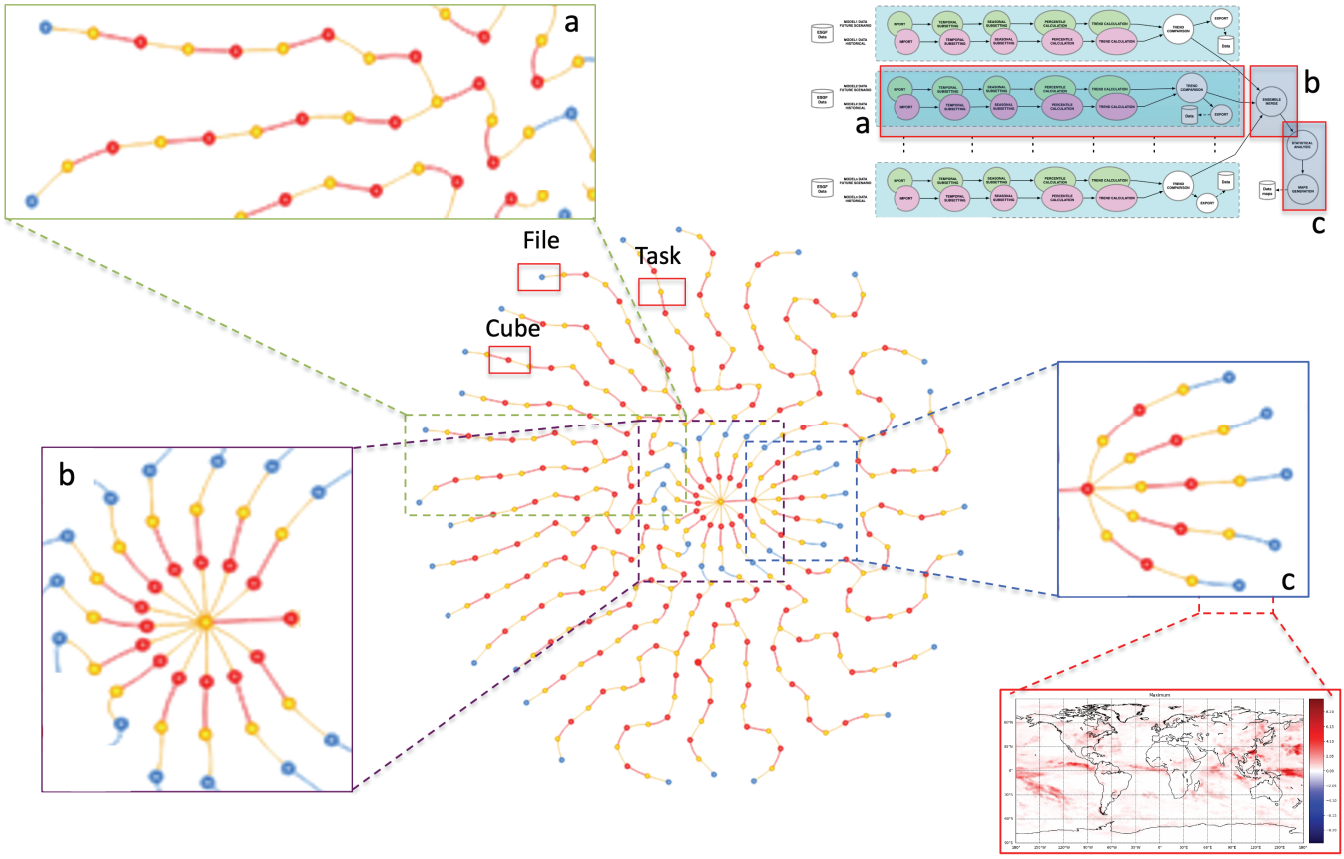

Figure 3: Provenance graph for the precipitation trend analysis use case.

VII. Conclusions and Future Work

This paper presents yProv, a new micro-provenance service developed at the University of Trento to address multi-level provenance as well as reproducibility challenges in climate analytics experiments. Presently, the yProv is mainly explored for retrospective provenance through the native, available query support. More precisely, the current support allows end-users to perform CRUD operations on the available resources, thus providing complete provenance graph management. The service has been deployed at CMCC and tested on several use cases in the climate domain.

Future work on this topic concerns (i) the knowledge extraction part, by focusing on graph mining algorithms, including learning aspects, (ii) the development of a micro-service (container-based) version of yProv for cloud/HPC-enabled environments (i.e., ENES Data Space [24][28][32]), as well as (iii) the service registration into the EOSC Marketplace for a wider adoption both within the climate community and across scientific communities. Finally, (iv) the extension of the graph data model is also envisaged to include system-level provenance resources (i.e., containers) that will be essential to better address provenance (in terms of more complete documentation), computational reproducibility (in terms of virtualized resources running in cloud environments) and to spot issues related to the surrounding software ecosystem (rather than the specific task or application) [27].

Acknowledgment

This work was partially funded by the EU InterTwin project (Grant Agreement 101058386) and the EU Climateurope2 project (Grant Agreement 101056933). Moreover, this work was partially funded under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.4 - Call for tender No. 1031 of 17/06/2022 of Italian Ministry for University and Research funded by the European Union – NextGenerationEU (proj. nr. CN_00000013).

References

[1] European Open Science Cloud (EOSC). European Commission. [Online]. Available: https://ec.europa.eu/research/openscience/index.cfm?pg=open-science-cloud.

[2] Open innovation, open science, open to the world – A Vision for Europe. Directorate-General for Research and Innovation (European Commission), 05 2016. [Online]. Available: https://publications.europa.eu/en/publication-detail/-/publication/3213b335-1cbc-11e6-ba9a-01aa75ed71a1/language-en

[3] B. Fecher and S. Friesike, Open Science: One Term, Five Schools of Thought. Cham: Springer International Publishing, 2014, pp. 17–47. [Online]. Available: https://doi.org/10.1007/978-3-319-00026-82

[4] N. Pontika, P. Knoth, M. Cancellieri, and S. Pearce, "Fostering open science to research using a taxonomy and an elearning portal," in iKnow: 15th International Conference on Knowledge Technologies and Data Driven Business, 2015. [Online]. Available: http://oro.open.ac.uk/44719/.

[5] B. A. Nosek, G. Alter et al., “Promoting an open research culture,” Science, vol. 348, no. 6242, pp. 1422–1425, 2015. [Online]. Available: http://science.sciencemag.org/content/348/6242/1422

[6] J. Freire, P. Bonnet, and D. Shasha, “Computational reproducibility: State-of-the-art, challenges, and database research opportunities,” in Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD ’12. New York, NY, USA: ACM, 2012, pp. 593–596. [Online]. Available: http://doi.acm.org/10.1145/2213836.2213908

[7] Z. Yuan, D. H. T. That, S. Kothari, G. Fils, and T. Malik, “Utilizing provenance in reusable research objects,” Informatics, vol. 5, no. 1, p. 14, 2018. [Online]. Available: https://doi.org/10.3390/informatics5010014

[8] F. S. Chirigati, D. E. Shasha, and J. Freire, “Reprozip: Using provenance to support computational reproducibility,” in 5th Workshop on the Theory and Practice of Provenance, TaPP’13, Lombard, IL, USA, April 2-3, 2013, 2013. [Online]. Available: https://www.usenix.org/conference/tapp13/technical-sessions/presentation/chirigati

[9] Barend Mons, Cameron Neylon, Jan Velterop, Michel Dumontier, Luiz Olavo Bonino da Silva Santos, and Mark D. Wilkinson. 2017. Cloudy, increasingly FAIR; revisiting the FAIR Data guiding principles for the European Open Science Cloud. Inf. Serv. Use 37, 1 (2017), 49–56. https://doi.org/10.3233/ISU-170824.

[10] M. D. Wilkinson, M. Dumontier et al., “The fair guiding principles for scientific data management and stewardship,” Scientific data, vol. 3, 2016.

[11] Helena Cousijn, Ricarda Braukmann, Martin Fenner, Christine Ferguson, René van Horik, Rachael Lammey, Alice Meadows, Simon Lambert, Connected Research: The Potential of the PID Graph, Patterns, Volume 2, Issue 1, 2021, 100180, ISSN 2666-3899, https://doi.org/10.1016/j.patter.2020.100180.

[12] Atzori, C., Bardi, A., Manghi, P., Mannocci, A. (2017). The OpenAIRE Workflows for Data Management. In: Grana, C., Baraldi, L. (eds) Digital Libraries and Archives. IRCDL 2017. Communications in Computer and Information Science, vol 733. Springer, Cham. https://doi.org/10.1007/978-3-319-68130-6_8.

[13] W3C PROV standards: https://www.w3.org/TR/prov-overview/

[14] Roy Thomas Fielding and Richard N. Taylor. 2000. Architectural styles and the design of network-based software architectures. Ph.D. Dissertation. University of California, Irvine. Order Number: AAI9980887.

[15] Neo4J Graph database technology: https://neo4j.com/

[16] EU Freya Project Deliverable - D4.2 Using the PID Graph: Provenance in Disciplinary Systems, Public Report, 31 May 2019.

[17] Earth System Grid Federation - http://esgf.llnl.gov

[18] Luca Cinquini, et al., The Earth System Grid Federation: An open infrastructure for access to distributed geospatial data, Future Generation Computer Systems, Volume 36, 2014, Pages 400-417, ISSN 0167-739X, https://doi.org/10.1016/j.future.2013.07.002.

[19] Intergovernmental Panel on Climate Change – http://www.ipcc.ch

[20] CMIP6 - https://pcmdi.llnl.gov/CMIP6/

[21] INDIGO-DataCloud - https://www.indigo-datacloud.eu

[22] Salomoni, D., Campos, I., Gaido, L. et al. INDIGO-DataCloud: a Platform to Facilitate Seamless Access to E-Infrastructures. J Grid Computing 16, 381–408 (2018). https://doi.org/10.1007/s10723-018-9453-3

[23] Marcin Płóciennik, Sandro Fiore, Giacinto Donvito, Michał Owsiak, Marco Fargetta, Roberto Barbera, Riccardo Bruno, Emidio Giorgio, Dean N. Williams, Giovanni Aloisio, Two-level Dynamic Workflow Orchestration in the INDIGO DataCloud for Large-scale, Climate Change Data Analytics Experiments, Procedia Computer Science, Volume 80, 2016, Pages 722-733, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2016.05.359.

[24] ENES - European Network for Earth System Modelling

[25] IPCC Assessment Reports - https://www.ipcc.ch/reports/

[26] Rew, R. K. and G. P. Davis, “The Unidata netCDF: Software for Scientific Data Access”, 6th International Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology, Anaheim, California, American Meteorology Society, pp. 33-40, February 1990.

[27] J. Freire and F. S. Chirigati, “Provenance and the different flavors of reproducibility,” IEEE Data Eng. Bull., vol. 41, no. 1, pp. 15–26, 2018.

[28] D. Elia, F. Antonio, S. Fiore, P. Nassisi, G. Aloisio, A data space for climate science in the european open science cloud, Computing in Science & Engineering (01) (2023) 1–10. doi:10.1109/MCSE.2023.3274047.

[29] S. Fiore et al., "Distributed and cloud-based multi-model analytics experiments on large volumes of climate change data in the earth system grid federation eco-system," 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, USA, 2016, pp. 2911-2918, doi: 10.1109/BigData.2016.7840941.

[30] C. Palazzo et al., "A workflow-enabled big data analytics software stack for escience," 2015 International Conference on High Performance Computing & Simulation (HPCS), Amsterdam, Netherlands, 2015, pp. 545-552, doi: 10.1109/HPCSim.2015.7237088.

[31] D. Elia, C. Palazzo, S. Fiore, A. D’Anca, A. Mariello, G. Aloisio, “PyOphidia: A Python library for High Performance Data Analytics at scale”, *SoftwareX*, Volume 24, 2023, doi: 10.1016/j.softx.2023.101538.

[32] S. Fiore, et al., "Towards an Open (Data) Science Analytics-Hub for Reproducible Multi-Model Climate Analysis at Scale," in 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018 pp. 3226-3234. doi: 10.1109/BigData.2018.8622205

[33] S. Fiore, et al. 2017. Big Data Analytics on Large-Scale Scientific Datasets in the INDIGO-DataCloud Project. In Proceedings of the Computing Frontiers Conference (CF'17). Association for Computing Machinery, New York, NY, USA, 343–348. https://doi.org/10.1145/3075564.3078884.

[34] Taylor, K. E., Stouffer, R. J., & Meehl, G. A. (2012). An overview of CMIP5 and the experiment design. Bulletin of the American meteorological Society, 93(4), 485-498.

[35] Eyring, Veronika, et al. "Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization." Geoscientific Model Development 9.5 (2016): 1937-1958.

[36] CMIP: https://pcmdi.llnl.gov/mips/cmip/about-cmip.html

[37] D. Elia, S. Fiore and G. Aloisio, "Towards HPC and Big Data Analytics Convergence: Design and Experimental Evaluation of a HPDA Framework for eScience at Scale," in IEEE Access, vol. 9, pp. 73307-73326, 2021, doi: 10.1109/ACCESS.2021.3079139.