Analyzing Provenance across Heterogeneous Provenance Graphs

Wellington Oliveira[1,2], Paolo Missier[3], Kary Ocaña[4],
Daniel de Oliveira[1], Vanessa Braganholo[1]

[1]Instituto de Computação, Universidade Federal Fluminense (UFF), Brazil
[2]DACC, Instituto Federal do Sudeste de Minas Gerais – Rio Pomba Campus, Brazil
[3]School of Computing Science, Newcastle University, UK
[4]Laboratório Nacional de Computação Científica (LNCC), Brazil
{wellmor, danielcmo, vanessa}@ic.uff.br,
paolo.missier@ncl.ac.uk, karyann@lncc.br

Abstract. Provenance generated by different workflow systems is generally expressed using different formats. This is not an issue when scientists analyze provenance graphs in isolation, or when they use the same workflow system. However, when analyzing heterogeneous provenance graphs from multiple systems poses a challenge. To address this problem we adopt ProvONE as an integration model, and show how different provenance databases can be converted to a global ProvONE schema. Scientists can then query this integrated database, exploring and linking provenance across several different workflows that may represent different implementations of the same experiment. To illustrate the feasibility of our approach, we developed conceptual mappings between the provenance databases of two workflow systems (e-Science Central and SciCumulus). We provide *cartridges* that implement these mappings and generate an integrated provenance database expressed as Prolog facts. To demonstrate its usage, we have developed Prolog rules that enable scientists to query the integrated database.

1 Introduction

Workflow Management Systems (WfMS) facilitate the design and implementation of data-driven computational science experiments, through a high-level programming model and a middleware-based runtime environment. A number of WfMS also capture and store *retrospective* provenance [1], and provide query languages and other analytical tools to help scientists use the resulting provenance traces [2–4].

Consider a scenario where two or more collaborative research teams work independently on common scientific goals, adopting slightly different approaches and producing workflows that differ in design, implementation, and execution middleware, but are otherwise similar in terms of intent, using comparable tools and algorithms. The concrete example we use throughout the paper is that of two research groups, both interested in generating phylogenetic trees. The two groups independently design and implement two workflows, SciPhy [5] and ML[1], using different WfMS,

[1] http://eubrazilcloudconnect.eu/content/leishmaniasis-virtual-laboratory

namely SciCumulus [6] and e-Science Central [7]. Each of these has their specificities, but both are capable of collecting retrospective provenance traces from their workflow runs. Since both workflows use either the same or similar input data and produce similar outputs, it seems natural to try and use the provenance of their runs to compare and discuss the results. However, the two WfMS use different proprietary schemas and logical data models to represent their respective provenance (relational and graph-based, respectively) as well as to store it. Furthermore, the different nature of the WfMS middleware leads to different levels of details in the provenance traces.

Thus, while in theory it should be possible for researchers to ask questions on both provenance graphs seamlessly and transparently, the heterogeneity in the design, implementation, and execution of their workflows translates into provenance traces that are themselves heterogeneous, making it difficult to analyze them jointly. Ultimately, this lessens the role of provenance in facilitating scientific discourse.

Promoting provenance interoperability has been the goal of several recent community efforts in provenance modeling, starting with the Open Provenance Model (OPM) [8] and culminating with PROV [9], a W3C recommendation. Further, ProvONE [3] and PROV-Wf [10] independently extended PROV, adding explicit representation of *prospective* provenance [1] to the model.

Contributions. In this paper we build upon these efforts to show how provenance interoperability that includes integration of the traces and their seamless querying, can be achieved in a practical setting where we can assume a degree of similarity amongst the traces, as in the science scenario just outlined.

The paper offers the following specific contributions: (i) Firstly, we argue that, to be useful, an integration model should include both retrospective and prospective provenance (which we henceforth concisely refer to as *r-prov* and *p-prov*). We use a number of example queries to show the benefits of an integrated provenance database that accommodates both r-prov and p-prov traces from two or more heterogeneous workflow runs. In our proposed approach, we use ProvONE as the integration model, as it is fairly comprehensive including both p-prov and r-prov and allows for easy integration of terms from external vocabularies, including Dublin Core or WfMS. ProvONE is also fairly stable, and supported by a large data conservation project, DataONE (dataone.org); (ii) We then map the proprietary provenance models of SciCumulus and e-Science Central to ProvONE, showing that ProvONE is indeed a viable target integration model; (iii) Thirdly, we present an actual mapping of provenance traces, obtained from running the example workflow on the two WfMS, to the ProvONE model. This exercise also shows the limitations of each of the provenance capture systems, as both proprietary traces miss some of the provenance elements (entities, activities, actors, and relationships) that are available in ProvONE; (iv) As an illustration of system-level integration we have implemented provenance components, or *cartridges*, for SciCumulus and e-Science Central, which translate the traces to ProvONE and write them to an integrated provenance database, and (v) Finally, we implement the example queries mentioned in (i) to show provenance querying on our integrated model.

In our proof-of-concept implementation we have used Prolog as it allows great flexibility both in producing the integrated database, because provenance relationships

translate to Prolog facts, and in formulating powerful queries with inference capability, using Prolog rules. We should note that Prolog has been also used to query and analyze provenance generated from scripts in the noWorkflow approach [11], and that it is also a natural choice owing to its syntactic similarity to PROV-N [12].

Running Example: Phylogenetic Analysis Workflows. As anticipated, our running example is a phylogenetic analysis experiment designed by two research groups and executed in two different WfMS. This analysis aims at generating phylogenetic trees along with other statistics, which can then be used to infer the evolutionary ancestry of a set of genes, species, or other taxa. Each of the workflows presents different designs and specifications (*e.g.* number and name of activities), but they have similar goals, which makes useful to compare the achieved results. To clarify the use of specific parameter values in both workflows, domain experts from the two groups defined semantic mappings between pairs of workflow activities in the SciPhy and ML workflows, as shown in Table 1. We use this mapping to compare the provenance of similar activities from distinct and heterogeneous provenance graphs, and later to drive the design of cross-traces queries.

Table 1. Semantic relationships between activities of two scientific workflows

SciPhy	ML	Description
DataSelection	ImportFile and FilterDuplicates	Importing, filtering, and selection of data.
Mafft	ClustalW	Sequence alignment.
ReadSeq	-	Conversion of alignment format.
ModelGenerator	-	Choice of the evolutionary model.
RAxML	MEGA-Maximum Likelihood	Generation of the phylogenetic tree.
-	CSVExport	Exporting filtered sequences on CSV format.
RAxML	ExportFiles	Exporting of the phylogenetic tree.

Specifically, the SciPhy workflow consists of five activities: (i) DataSelection; (ii) Mafft; (iii) ReadSeq; (iv) ModelGenerator; and (v) RAxML. The ML workflow is composed of six activities: (i) ImportFile; (ii) FilterDuplicates; (iii) ClustalW; (iv) MEGA-Maximum Likelihood; (v) CSVExport; and (vi) ExportFiles. The two workflows were set up with similar input data and parameters. Although the number of their activities differs, two key activities appear in both, namely *sequence alignment* and *tree generation*. Their mappings: Mafft → ClustalW and RAxML → MEGA help us compare the critical elements of the workflows (the other activities are responsible for format conversions and some optional optimizations in the process).

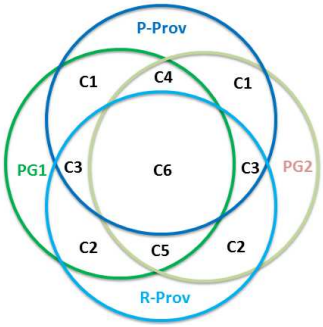
2 Provenance Analysis Across Heterogeneous Provenance Graphs

2.1 A reference classification of the provenance space and of its queries

We argue that, in the collaborative scenario just outlined, scientists will benefit from provenance graphs that (a) include both p-prov and r-prov, and (b) include traces from

both experiments. The case for combining p-prov and r-prov has been made before [10, 13, 14], namely that p-prov enables new types of queries to be made on r-prov, such as *find all data produced by any activity that occurs downstream from block X in the workflow*. Other interesting queries that span r-prov and p-prov are presented later in this section. The case for point (b) is that the ability to perform analysis on combined provenance graphs will help collaborative teams obtaining deeper understanding from related workflows with different levels of details. As we have seen, this is possible because these workflows typically share similarities on their activities, data flows, or input parameters. When detailed provenance graphs from similar workflows are available, scientists can use those sources to clarify their understanding and get more insights about the experiment.

Given two traces *PG1* and *PG2*, each from a different workflow run (from the same or different workflows), and each providing both r-prov and p-prov, we can categorize the set of all possible provenance queries as illustrated in Fig. 1. In this Venn diagram, queries are classified according to the provenance data needed to answer them. For instance, queries in class *C1* operate on p-prov only and on one graph at a time, while *C3* queries require both p-prov and r-prov, on one graph. Class *C6* is perhaps the most challenging, as it operates simultaneously on p-prov and r-prov, and on both graphs. Note that our classification is conceptual, and the actual fragment returned by a query is sensitive to the values of query parameters.



Class	P-Prov	R-Prov	Single	Both
C1	✓	✗	✓	✗
C2	✗	✓	✓	✗
C3	✓	✓	✓	✗
C4	✓	✗	✓	✓
C5	✗	✓	✓	✓
C6	✓	✓	✓	✓

Fig. 1. Classification of provenance fragments and corresponding queries

Example queries for each of the classes are listed in Table 2. Note that queries from classes C1, C2 and C3 are easily answered using provenance captured by most WfMS. However, queries of classes C4, C5 and C6 require additional mapping information that is not automatically provided by those systems. This mapping encompasses two aspects: (a) a syntactic mapping between heterogeneous schemas of provenance data and (b) a semantic mapping that informs the similarity or equivalence between p-prov elements. The syntactic mapping of local and global provenance schemas using ProvONE is described next, while a sample of a semantic mapping of two workflows specifications appears in Table 1. Note that the semantic mapping comes from the researchers/domain experts' mind and is used just as auxiliary information to perform queries by filtering results. Later, we will come back with the queries and classes presented here and we will demonstrate how an integrating architecture enables their implementation.

Table 2. Provenance queries on intersection classes

#	Queries	Class
Q1	Retrieve all programs with their input and output ports for the workflow *w'* and provenance graph *g'*.	C1
Q2	Retrieve all activity executions with their generated data for the workflow execution *w'* and provenance graph *g*'.	C2
Q3	Retrieve the time consumed by each activity execution for the workflow execution *w'* and provenance graph *g*'.	C2
Q4	Retrieve the complete activity execution trace that influenced the generation of the data *d'*.	C2
Q5	Retrieve the complete dataflow trace of the output data *d'* for the workflow execution *w'* and provenance graph *g*'.	C2
Q6	Retrieve all programs (plans) of each execution and their input parameters for the workflow execution *w'* and provenance graph *g*'.	C3
Q7	Retrieve the workflow version, and the time consumed by each workflow execution for the workflow *wf*' and provenance graph *g*'.	C3
Q8	Retrieve all programs with their input and output ports for each workflow specification.	C4
Q9	Retrieve all activity executions with their generated data for each workflow execution.	C5
Q10	Retrieve the time consumed by each activity execution for each workflow execution.	C5
Q11	Retrieve the ports, workflow executions, provenance graphs, and the complete activity execution trace that influenced the generation of all data.	C6
Q12	Retrieve the complete dataflow trace and workflow for each workflow execution.	C6
Q13	Retrieve the time consumed by each workflow execution for each workflow and provenance graph.	C6
Q14	Retrieve all programs (plans) of each activity execution and their input parameters for each workflow *wf*'.	C6

2.2 Mapping provenance models to ProvONE

Executing queries in each of these classes requires converting *PG1* and *PG2* to a common provenance model. We now illustrate the integration process using two WfMS, SciCumulus and e-Science Central. As mentioned before, SciPhy [5] and ML, which run on each of these WfMS respectively, share the common goal of generating phylogenetic trees. The two WfMS collect provenance data at different levels of detail and heterogeneity is present both in format as well as in content.

SciCumulus captures p-prov and r-prov and stores them in relational tables in a PostgreSQL database, while e-Science Central stores just r-prov as a graph in a Neo4J database. However, it maintains information about the workflow structure in a relational database (PostgreSQL) blended with several additional data related to the workflow viewing (*i.e.*, coordinates of each graph object) and exports it to JSON files.

We use ProvONE (Fig. 2) as the target global schema for integration of the provenance traces produced by the two systems. ProvONE extends the PROV model with an explicit representation of p-prov, thus capturing the most relevant information on scientific workflow processes, and is designed to accommodate extensions for specific scientific workflow systems [3].

Table 3 describes the logical mapping between elements of the two source provenance traces, and the corresponding ProvONE elements. Each relational table from SciCumulus and JSON element from e-Science Central, which hold p-prov, were mapped to the corresponding ProvONE entities and relationships. Furthermore, the nodes and edges of e-Science Central database (Neo4J) and also relational tables of

SciCumulus that hold r-prov were mapped to ProvONE entities and relationships. The gaps in the SciCumulus and e-Science Central column indicate missing information.

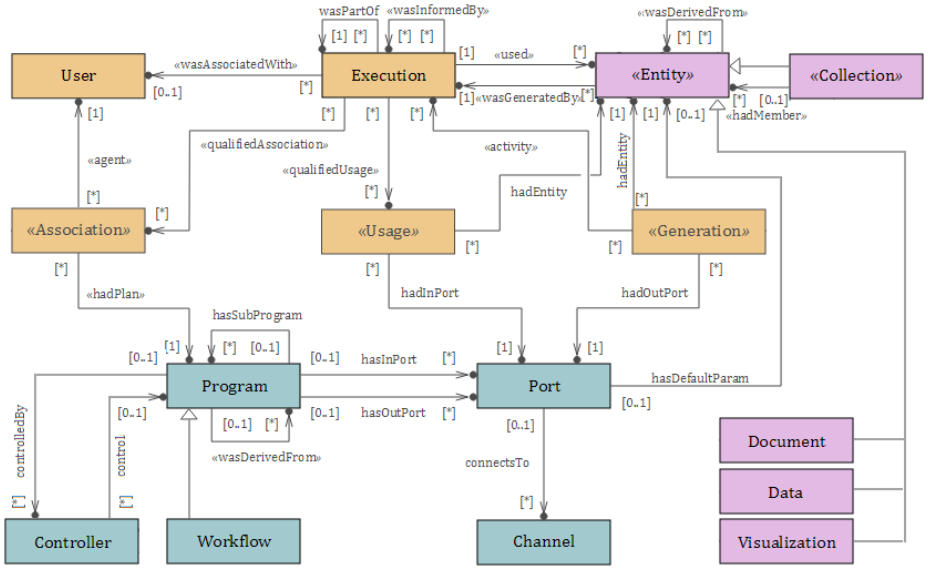


Fig. 2. ProvONE conceptual model, from the DataONE documentation[2]

As there is no previous relation between p-prov and r-prov in the e-Science Central database and the exported JSON files, we use some information such as invocations and blocks identifiers to unify them. The relation between p-prov and r-prov is straightforward in SciCumulus, since it first stores p-prov and then collects and stores r-prov during the workflow execution (*i.e.* at runtime).

2.3 ProvONE assertions as Prolog facts

We now show examples of how provenance traces from specific workflow executions are represented as Prolog facts. We have chosen Prolog as it allows great flexibility both in producing the integrated database (provenance relationships are translated to facts) and in formulating powerful queries with inference capability (rules).

Two fragments of provenance graphs for e-Science Central and SciCumulus, respectively, are depicted in Fig. 3 and Fig. 4, after mapping to ProvONE. Gray boxes represent p-prov, orange boxes correspond to r-prov, and light blue boxes are entities (p-prov and r-prov). Since both provenance graphs are represented using the same model, queries can easily traverse both provenance graphs. Table 4 presents examples of Prolog facts for these workflow fragments (the complete set of facts is available at GitHub at https://github.com/dew-uff/integrated-provenance-analysis). As Prolog facts syntax is similar to the PROV-N notation, each entity and activity was named

[2] http://jenkins-1.dataone.org/jenkins/view/Documentation%20Projects/job/ProvONE-Documentation-trunk/ws/provenance/ProvONE/v1/provone.html

and labeled in a similar style, using an identifier followed by a set of properties delimitated by brackets. Relationships use the identifiers for each ProvONE element. Furthermore, entity identifiers were modified to make them unique in the global schema and facts were created to identify the provenance graphs and relate them to their workflows.

Table 3. Mapping between ProvOne, SciCumulus, and e-Science Central provenance models

#	ProvONE	SciCumulus	e-Science Central
1	provone:workflow	cworkflow	invocation
2	provone:program	cactivity	blocks
3	provone:port	crelation	connections
4	provone:execution	eworkflow, eactivity, eactivation	Service Run, Workfow Run
5	provone:execution (Workflow Execution)	eworkflow, eactivity, eactivation	Service Run, Workfow Run
6	provone:user	emachine	-
7	provone:document	efile	DataVersion
8	provone:data	idataselection, odataselection, omafft, oreadseq, omodelgenerator, oraxml	properties
9	provone:hadPlan	eactivation, eactivity, cactivity, eworkflow, cworkflow	Service Run, blocks
10	prov:wasDerivedFrom (Data)	efile, cmapping	Used, DataVersion
11	prov:wasDerivedFrom (Program)	-	Run_Of, Instance_Of, Service Run, Service Version, Workflow Version
12	prov:used	efile, cmapping	Used, DataVersion, Service Run
13	prov:wasGeneratedBy	eafile	Was_Generated_By, DataVersion, Service Run
14	prov:wasAssociatedWith	eactivation, emachine	-
15	prov:wasInformedBy	cmapping	Used, Was_Generated_By, Service Run
16	provone:hasInPort	crelation, cmapping, cactivity	blocks, connections
17	provone:hasOutPort	crelation, cmapping, cactivity	blocks, connections
18	provone:hasSubProgram	cworkflow, cactivity	invocation, blocks
19	provone:hasDefaultParam	cfield	connections, properties
20	provone:wasPartOf	eworkflow, eactivity, eactivation	Contained, Service Run
21	provone:hadInPort	crelation, cmapping, cactivity, eactivity, eactivation	Service Run, connections
22	provone:hadOutPort	crelation, cmapping, cactivity, eactivity, eactivation	Service Run, connections

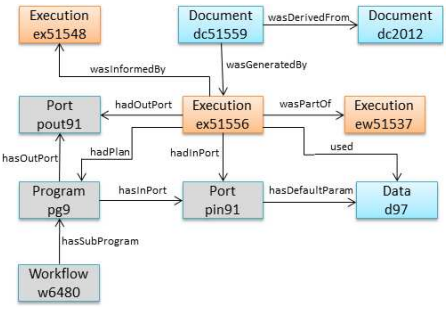


Fig. 3. Part of e-Science Central provenance

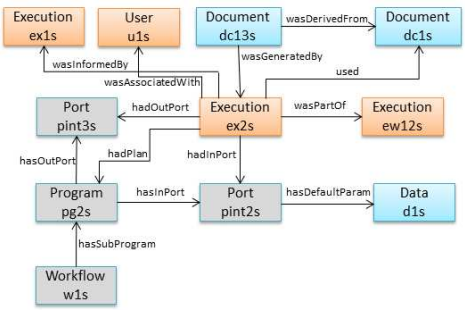


Fig. 4. Part of SciCumulus provenance

Table 4. Prolog instances for each ProvOne construct

#	Prolog Instances for e-Science Central	Prolog Instances for SciCumulus
1	entity(w6480,[prop(prov:type,['prov:plan', 'provone:workflow']),prop(prov:label,'ML Pipe-line')]).	entity(w1s,[prop(prov:type,['prov:plan', 'provone:workflow']),prop(prov:label,'sciphy')]).
2	entity(pg9,[prop(prov:type,['prov:plan','provone: program']),prop(prov:label,'CSVExport')]).	entity(pg2s,[prop(prov:type,['prov:plan', 'provone:program']),prop(prov:label,'mafft')]).
3	-	agent(u1s,[prop(prov:type,['provone:user']),prop(prov:label,'wellington-VirtualBox')]).
4	entity(dc51559,[prop(prov:type,['provone:document']),prop(prov:label,'sequence-map.csv'), prop(prov:type,'null'),prop(prov:value,'null')]).	entity(dc13s,[prop(prov:type,['provone: document']),prop(prov:label,'FILE13'), prop(prov:value,'ORTHOMCL256.mafft')]).
5	hadPlan(ex51556,pg9).	hadPlan(ex2s,pg2s).
6	wasDerivedFrom(dc51559,dc2012).	wasDerivedFrom(dc13s,dc1s).
7	wasDerivedFrom(pg9, pgV50025).	-
8	used(ex51556,d97).	used(ex2s,dc1s).
9	wasGeneratedBy(dc51559,ex51556).	wasGeneratedBy(dc13s,ex2s).
10	-	wasAssociatedWith(ex2s,u1s).

Regarding relative incompleteness, note that the e-Science Central provenance graph (rows 3 and 10 of Table 4) does not hold information about the agent, while the SciCumulus provenance graph does not store the program versions (row 7).

3 Implementation: adapters and global queries

Integration Architecture. Converting from SciCumulus and e-Science Central proprietary provenance to ProvONE requires the implementation of specialized adapters, or *cartridges*, one for each system. Provenance obtained from these cartridges is stored in a unified knowledge base as Prolog facts, as described earlier. The cartridges were implemented in Java using the mapping of ProvONE, SciCumulus, and e-Science Central provenance models presented in Table 3. The implementation is simple and is not time consuming. All code and some data are also available on GitHub.

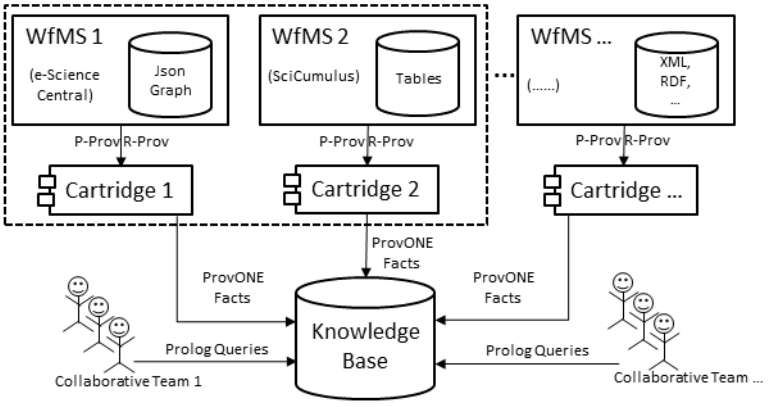


Fig. 5. Provenance integration architecture

Using the knowledge base, various teams may access provenance and work collaboratively on provenance analysis. They can use pre-defined logical rules to query provenance, and thus get more information about similar experiments. Fig. 5 gives an overview of the provenance gathering, conversion, integration and query processes. The example cartridges are specific to our case study.

Consistent with the mappings presented in Table 3, SciCumulus cartridge gets p-prov and r-prov from the relational database and converts them to Prolog. In turn, e-Science Central cartridge fetches r-prov from the graph database and extracts p-prov from JSON files. Clearly, extending the approach to integrating other provenance sources requires new cartridges to be developed. This effort is similar to database integration efforts that are well known in the literature [15].

Querying the integrated traces. Using our integration architecture, we are now able to express queries that span different types of provenance and different types of graphs. Queries over the integrated schema are expressed in Prolog as rules. To illustrate, we have implemented the queries listed in Table 2, which exemplify the intersection classes of Fig. 1. (Due to space restrictions we only present a subset of the queries). Specifically, the *dataTrace* and *dataFlow* rules implement queries *Q5* and *Q12*. Query *Q5* covers class *C2* and retrieves r-prov from either provenance graph *PG1* or provenance graph *PG2*, while *Q12* covers class *C6* and retrieves p-prov and r-prov from both *PG1* and *PG2*. Although these queries are quite similar, *Q12* retrieves the trace of data for all executions, while *Q5* considers only one of the workflow systems. The following rules were designed for retrieving all data that influenced the generation of a particular data product. The query result is a historical data trace that shows which input files influenced the generation of a given output.

```
dataTrace(DstName, WkfName, WExName, OutputId, InputId):-
  dataSet(DsId, DstName), hasDataSet(WkfId, DsId),
  activity(WExId,[prop(prov:type,['provone:execution']),
  prop(prov:label, WExName),_,_,_]),
  entity(WkfId,[prop(prov:type,['prov:plan', 'provone:workflow']),
  prop(prov:label,WkfName)]), hadPlan(WExId,WkfId),
  wasPartOf(ExId, WExId), wasGeneratedBy(OutputId, ExId),
  dataFlow(OutputId, InputId).
dataFlow(Output, Input):- wasDerivedFrom(Output, Input).
dataFlow(Output, Input):- wasDerivedFrom(Output, X), dataFlow(X, Input).
```

Table 5 shows the query calls (and their results) with the parameters used to query the data trace for a specific result generated by SciCumulus and e-Science Central respectively. Query Q5 retrieves the input files that influenced the generation of the *dc19s* output file on the *scyphy-1* execution of the *sciphy* workflow that was executed in SciCumulus, while Q12 does the same for the *dc51559* output of the *ML Pipeline* workflow run on e-Science Central. These query instances hide the complexity of the Prolog rules and become suitable for non-experts in the Prolog language. Note that the user may bind none, one, or multiple parameter values. For example, if one specifies no parameter values, the query will return the graph name, workflow name, execution name, along with the input and output data for both datasets. This makes Prolog queries a flexible resource to retrieve provenance according to specific requirements.

Table 5. Prolog queries and results on SciCumulus and e-Science Central provenance graphs

SciCumulus	`dataTrace('SciCumulus', 'sciphy', 'sciphy-1', dc19s, InputId).`
	`InputId = dc6s; InputId = dc12s; InputId = dc13s; InputId = dc14s; InputId = dc1s; InputId = dc1s; InputId = dc1s; InputId = dc1s;`
e-Science Central	`dataTrace('e-Science Central', 'ML Pipeline', 'Testing ML Pipeline', dc51559, InputId).`
	`InputId = dc2012;`

4 Related Work

Working on the integration of provenance models, Ellqvist *et al.* [16] propose an architecture based on a generic mediator that blends different provenance data sources. In this approach, a global schema is presented to the user who specifies a generic query that is converted into specific queries for each database. Wrappers access the data from the data source and convert them to the mediator model. Apart from this, it uses a proprietary mediation schema that is not compatible with the OPM or PROV models. Also based on interoperability issues that were exposed by the Third Provenance Challenge (PC3), Ding *et. al.* [17] approach provenance reuse using OPM, OWL and Linked Data. They argue that provenance trace reuse requires generic provenance and domain-specific data (*e.g.,* a classification of artifact types). Their OPM ontology (PC3OPM) was extended and modularized to cover interoperability gaps found in the PC3. Such approach allows one to import provenance from OPM/XML, export it to RDF, query, and improve provenance by creating new relations with SPARQL-based rule inferences. Similarly, Braun and Seltzer [18] propose a Common Provenance Framework to provide provenance interoperability. To develop the framework, they analyzed the problems and challenges encountered in importing PASS [19] data into the PLUS system [20]. Their framework includes concepts, constraints, and tools to provide semantics and structure to query provenance across different systems using the OPM model and XML Schema. Both [17] and [18] use OPM as mediator model, which does not consider p-prov as ProvONE does.

Missier *et al.* [21] present an approach to solve problems found in the implicit collaboration between different provenance systems that use the result from another workflow execution as part of their input. The local provenance is mapped to a common model and stored in a database with new global identifiers. This allows the tracking of provenance for workflows, systems and user group executions. Differently from our approach, their aim is to provide a data model to track the provenance across different workflows. Similarly, Altintas *et al.* [22] propose a data model based on collaborative views and develop QLP, a query language for provenance. QLP was designed to facilitate querying implicit collaboration in interoperable provenance datasets. In the same direction of the previous authors, they propose the union of several data sources into one single repository to be handled by one single query. On the other hand, they use OPM as the provenance model and cannot represent p-prov.

Aiming to facilitate publication, sharing, exchanging, and reuse of self-contained units of knowledge, Bechhofer *et al.* [23] introduce Research Objects (RO). These are semantic aggregations of resources (eg data, methods, metadata) that are produced

and consumed by common services. Similarly, SHIWA [24], a EU project to support workflow sharing, was designed to integrate the execution of different workflows that use different workflow systems, different workflow languages, and different distributed infrastructures. Although these approaches allow storing and sharing provenance from different sources, they do not enable provenance querying across different data sources.

5 Final Remarks

The integration of heterogeneous data sources can be a powerful tool for provenance analytics. In particular, it can provide considerable advantages for research teams that work collaboratively on similar experiments. In this paper, we have presented an approach that enables integrating and querying provenance data from similar workflows designed and implemented in different systems with different specifications. To achieve this, we use an integration model (ProvONE) that includes both p-prov and r-prov and create cartridges that convert different provenance databases to a global ProvONE schema of Prolog facts.

Our approach introduces classes that explore intersection between p-prov, r-prov, and heterogeneous provenance graphs and presents related queries that run across both provenance graphs and retrieve information with different contents and levels of detail. Prolog rules were developed for each pre-defined query, taking advantage of inference and unification facilities catered by Prolog. As a proof-of-concept, Prolog queries were executed and they could retrieve the data traces from both provenance graphs. New Prolog rules can easily be designed to accommodate new requirements., and new cartridges can be developed for other workflow systems using the proposed architecture.

As future work, we plan to develop a benchmark of completeness to evaluate provenance from different WfMSs. We also intend to investigate how to cover gaps in similar provenance graphs by using our intersection classes.

References

1. Freire, J., Koop, D., Santos, E., Silva, C.T.: Provenance for Computational Tasks: A Survey. Computing in Science Engineering 10, 11–21 (2008).
2. Lim, C., Lu, S., Chebotko, A., Fotouhi, F., Kashlev, A.: OPQL: Querying scientific workflow provenance at the graph level. Data Knowl. Eng. 88, 37–59 (2013).
3. Missier, P., Dey, S., Belhajjame, K., Cuevas-Vicenttín, V., Ludäscher, B.: D-PROV: Extending the PROV Provenance Model with Workflow Structure. In: TaPP (2013).
4. Dey, S., Köhler, S., Bowers, S., Ludäscher, B.: Datalog as a Lingua Franca for Provenance Querying and Reasoning. In: TaPP (2012).
5. Ocaña, K.A.C.S., Oliveira, D. de, Ogasawara, E., Dávila, A.M.R., Lima, A.A.B., Mattoso, M.: SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drug Targets in Protozoan Genomes. In: Advances in Bioinformatics and Computational Biology. (2011).
6. Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M.: SciCumulus: a lightweight cloud middleware to explore many task computing paradigm in scientific workflows. In: International Conference on Cloud Computing. (2010).
7. Watson, P., Hiden, H., Woodman, S.: e-Science Central for CARMEN: Science As a Service. Concurr Comput Pr. Exper. 22, 2369–2380 (2010).

8. Moreau, L., Freire, J., Futrelle, J., McGrath, R.E., Myers, J., Paulson, P.: The Open Provenance Model: An Overview. In: IPAW. (2008).
9. Moreau, L., Missier, P.: PROV-DM: The PROV Data Model, http://www.w3.org/TR/2013/REC-prov-dm-20130430/.
10. Costa, F., Silva, V., de Oliveira, D., Ocaña, K., Ogasawara, E., Dias, J., Mattoso, M.: Capturing and querying workflow runtime provenance with PROV: a practical approach. In: EDBT/ICDT Workshops. (2013).
11. Murta, L., Braganholo, V., Chirigati, F., Koop, D., Freire, J.: noWorkflow: Capturing and Analyzing Provenance of Scripts. In: IPAW. (2014).
12. Moreau, L., Missier, P.: PROV-N: The Provenance Notation, http://eprints.soton.ac.uk/356852/.
13. Missier, P., Sahoo, S.S., Zhao, J., Goble, C., Sheth, A.: Janus: From Workflows to Semantic Provenance and Linked Open Data. In: IPAW. (2010).
14. Belhajjame, K., Zhao, J., Garijo, D., Gamble, M., Hettne, K., Palma, R., Mina, E., Corcho, O., Gómez-Pérez, J.M., Bechhofer, S., Klyne, G., Goble, C.: Using a suite of ontologies for preserving workflow-centric research objects. Web Semant. Sci. Serv. Agents World Wide Web. 32, 16–42 (2015).
16. Batini, C., Lenzerini, M., Navathe, S.B.: A Comparative Analysis of Methodologies for Database Schema Integration. ACM Computing Surveys 18, 323–364 (1986).
17. Ellqvist, T., Koop, D., Freire, J., Silva, C., Strömbäck, L.: Using Mediation to Achieve Provenance Interoperability. In: IEEE World Conference on Services (2009).
17. Ding, L., Michaelis, J., McCusker, J., McGuinness, D.L.: Linked provenance data: A semantic Web-based approach to interoperable workflow traces. Future Gener. Comput. Syst. 27, 797–805 (2011).
19. Braun, U.J., Seltzer, M.I., Chapman, A., Blaustein, B., Allen, M.D., Seligman, L.: Towards Query interoperability: PASSing PLUS. (2011).
19. Muniswamy-Reddy, K.-K., Holland, D.A., Braun, U., Seltzer, M.I.: Provenance-Aware Storage Systems. Harvard University (2006).
21. Blaustein, B., Seligman, L., Morse, M., Allen, M.D., Rosenthal, A.: PLUS: Synthesizing privacy, lineage, uncertainty and security. In: International Conference on Data Engineering Workshops. (2008).
22. Missier, P., Ludascher, B., Bowers, S., Dey, S., Sarkar, A., Shrestha, B., Altintas, I., Anand, M.K., Goble, C.: Linking multiple workflow provenance traces for interoperable collaborative science. In: Workshop on Workflows in Support of Large-Scale Science (WORKS). (2010).
23. Altintas, I., Anand, M.K., Crawl, D., Bowers, S., Belloum, A., Missier, P., Ludäscher, B., Goble, C.A., Sloot, P.M.A.: Understanding Collaborative Studies through Interoperable Workflow Provenance. In: IPAW. (2010).
23. Bechhofer, S., De Roure, D., Gamble, M., Goble, C., Buchan, I.: Research Objects: Towards Exchange and Reuse of Digital Knowledge. Nat. Preced. (2010).
24. Terstyanszky, G., Kukla, T., Kiss, T., Kacsuk, P., Balasko, A., Farkas, Z.: Enabling scientific workflow sharing through coarse-grained interoperability. Future Gener. Comput. Syst. 37, 46–59 (2014).