SWPM 2009

Workshop on Semantic Web
and Provenance Management

The First International Workshop on the role of Semantic Web in Provenance Management

(at the 8th International Semantic Web Conference ISWC-2009)

October 25 2009, Westfields Conference Center, Washington D.C., USA.

Organization

Chairs

Amit Sheth
Vassilis Christophides

Organizing Committee/PC Co-Chairs

Juliana Freire
Paolo Missier
Satya S. Sahoo

Program Committee

Aleksander Slominski, IBM Research
Bertram Ludäscher, University of California Davis
Beth Plale, Indiana University
Claudio Silva, University of Utah
Francisco Curbera, IBM Research
Giorgos Flouris, FORTH-ICS, Greece
Ilkay Altintas, San Diego Supercomputer Center, UCSD
James Cheney, University of Edinburgh
Jun Zhao, Oxford University
Kei Cheung, Yale University
Krishnaprasad Thirunarayan, Wright State University
Luc Moreau, University of Southampton
Nirmal Mukhi, IBM Research
Olivier Bodenreider, National Library of Medicine, NIH
Paulo Pinheiro da Silva, University of Texas at El Paso
Peter Fox, Tetherless World Research Constellation, RPI
Roger Barga, Microsoft Research
Sarah Cohen-Boulakia, Universite Paris-Sud
Sudha Ram, Arizona State University
Val Tannen, University of Pennsylvania
Yogesh Simmhan, Microsoft Research

Introduction

The growing eScience infrastructure is enabling scientists to generate scientific data on an industrial scale. Similarly, the Web 2.0 paradigm is enabling Web users to create applications that combine data from multiple sources, popularly referred to as “mashups”, on a large scale. The importance of managing various forms of apparently ancillary metadata, in addition to the primary data products of eScience, Web, and business applications is increasingly being recognized as critical for the correct interpretation of the data. In this workshop we focus specifically on metadata that describes the origins of the data. The term *provenance* from the French word “provenir”, meaning “to come from", describes the *lineage* or origins of a data entity. Provenance metadata is essential to correctly interpret the results of a process execution, to validate data processing tools, to verify the quality of data, and to associate measures of trust to the data. The *proof layer* in the Semantic Web layer cake, corresponding to provenance information, has been identified as an important component for the implementation of “trust mechanisms” and effective information extraction from the Web.

The primary objective of this workshop is to explore the role of Semantic Web in addressing some of the critical challenges facing provenance management, namely:

1. Efficiently capturing and propagating provenance information as data is processed, fragmented and recombined across multiple applications on a Web scale.
2. A common representation model for provenance for processing and analysis by both agents and humans.
3. Interoperability of provenance information generated in distributed environments such as myGrid.
4. Tools leveraging the Semantic Web for visualization of provenance information.

We thank the keynote speakers, all members of the program committee, authors, invited speakers, participants and local organizers for their efforts.

We look forward to a successful workshop!

Juliana Freire, Paolo Missier, Satya S. Sahoo

Semantic Provenance for Science Data Products: Application to Image Data Processing

Stephan Zednik, Peter Fox, Deborah L. McGuinness, Paulo Pinheiro da Silva, Cynthia Chang

Abstract—A challenge in providing scientific data services to a broad user base is to also provide the metadata services and tools the user base needs to correctly interpret and trust the provided data. Provenance metadata is especially vital to establishing trust, giving the user information on the conditions under which the data originated and any processing that was applied to generate the data product provided.

In this paper, we describe our work on a federated set of data services in the area of solar coronal physics. These data services provide a particular challenge because there is decades of existing data whose provenance we will have to reconstruct, and because the quality of the final data product is highly sensitive to data capture conditions, information which is not currently propagated with the data.

We describe our use of semantic technologies for encoding provenance and domain knowledge and show how provenance and domain ontologies can be used together to satisfy complex use cases. We show our progress on provenance search and visualization tools and highlight the need for semantics in the user tools. Finally, we describe how our methods are applicable to generic data processing systems.

1 Introduction

We aim to create a next-generation virtual observatory[1] with extensive provenance support. Provenance is a first-class concept in our system; with full support in our search, explanation, and visualization tools. We require a general provenance model that is applicable in a wide array of domains, and integrable with domain models, so that domain concepts can be modeled along with provenance concepts. For these reasons we have chosen to use the Proof Markup Language (PML) [3], [7] family of OWL ontologies as our provenance model. We show how PML can be used to model provenance causality chains, introduce our domain model, and show how the PML provenance model and our science domain models can be integrated in a manner that provides a rich provenance infrastructure, able to model complex scientific provenance relations.

We have chosen to test our system in the domain of solar coronal physics, using the Advanced Coronal Observing System (ACOS) as a testbed. The ACOS data products are the result of several data ingest pipelines, processing observations from three imaging instruments located at the Mauna Loa Solar Observatory (MLSO). The ACOS data pipelines are distributed data pipelines, operated in part at MLSO in Hawaii and the National Center for Atmospheric Research High Altitude Observatory (NCAR/HAO) in Boulder, CO. ACOS has been operational for over a decade, and has produced terabytes of data.

ACOS was chosen because its data pipelines are typical of data ingest systems and the vast quantity of existing data ACOS has generated in its decades of operation provides the opportunity to design a system geared to reconstruct, as well as capture, provenance. One of the ACOS data pipelines, the Chromospheric Helium-I Imaging Photometer (CHIP) Intensity Image pipeline, is illustrated in Figure 1. This high-level diagram is designed to show not just the process/artifact flow of the pipeline, but domain concepts that could

- *Stephan Zednik, Peter Fox, Deborah L. McGuinness, and Cynthia Chang are with the Rensselaer Polytechnic Institute, Tetherless World Constellation.*
- *Paulo Pinheiro da Silva is with the University of Texas at El Paso, Department of Computer Science.*

[1] A virtual observatory is a collection of interoperating data archives and software tools which utilize the internet to form a scientific research environment in which research programs can be conducted.

and should be captured and represented in the provenance. In the pipeline, data (square boxes) passes through a number of stages (ovals) each of which can contain a number of complex processing, analysis, human interaction, and decision steps. Each of these stages contains domain-specific information (dotted-lined boxes) related to the data product provenance.

Of particular interest is information in the pipeline that is not a direct or inferred result of the data capture event. The Observer Log is a human-generated account of weather conditions and system status during the instrument observing schedule for the day. Bad weather conditions or instrument instability, noted in the log, can have significant negative effect on the quality of the data observations. This information is currently not propagated in the data pipeline nor do the data products reference it in any way, but is an invaluable reference in determining why an image has been given a low-grade quality assessment. This information is an important component of the origin of the data image and should be represented in its provenance.

The motivation for this project arose from our experiences designing and deploying a solar terrestrial physics virtual observatory system [1], [2], and from numerous discussions with the data providers (i.e. ’roles’ in Figure 1). Among their remarks were the following:

- Data is being used in new ways and we frequently do not have sufficient information on what happened to the data along the processing stages to determine if it is suitable for a use we did not envision.
- We often fail to capture, represent, and propagate manually generated information that needs to go with the data flows.

Further, when science data and visual representations of the data (such as the CHIP Quick Look images) are made available to the end-user, the product has often gone through a number of data filtration and processing steps. If thorough provenance metadata and processing documentation is not captured, propagated, and made available to the end-user; the data system is in effect a ’black box’, and the end-user must blindly trust the science quality of the data product and long-term consistency of the pipeline processing.

Virtual Observatories are particularly prone to this information gap. This project traces the entire pipeline and accounts for all roles, processes, and metadata as they relate to use cases which require provenance.

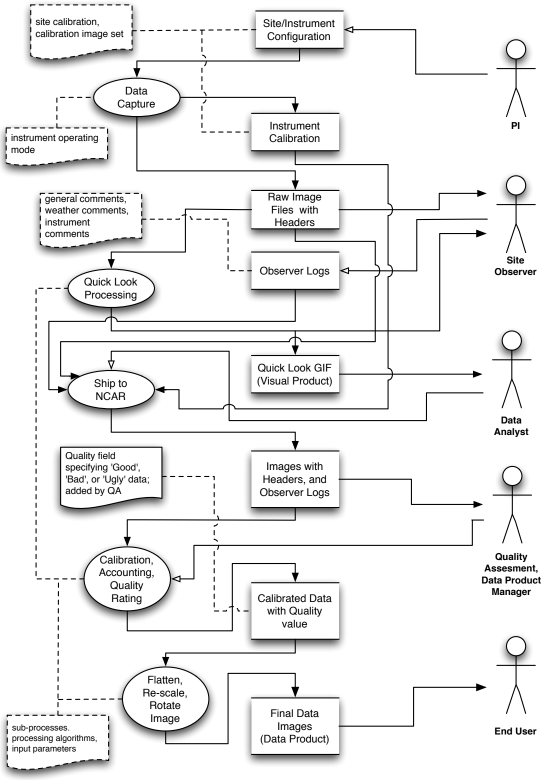


Fig. 1.
Chromospheric Helium-I Imaging Photometer (CHIP) Intensity Image pipeline

2 USE CASES

During discussions with science project participants, we developed an initial set of use cases which reflect real user questions that cannot presently be answered in any routine or automated manner:

- What were the cloud cover and seeing conditions during the observation period of *this* data product artifact?
- What calibrations have been applied to *this* data product artifact?
- Who (person or program) added the comments to the science data file for the best vignetted rectangular polarization brightness image from January 26, 2005 18:49:09UT taken by the ACOS Mark IV polarimeter?

- Find all *good* CHIP He 1083 nm intensity images on March 21, 2008.
- Why does *this* data look *bad*?

These use case scenarios mix domain and provenance terms in a manner that would make the question difficult to answer if the provenance and domain models are independent. The cloud cover and seeing conditions during observation periods is recorded, but it is never directly associated with nor propagated with processed data. We could adjust the data processing pipeline to pull this information from the observation records and propagate it with the data as metadata; but this would be a very heavy-handed approach to take for any and all information associated with a data product's generation and processing that a user may want to see. Instead, we intend to build the link in the provenance representation of the data product, so that by following the data product's causality graph the system can find the weather condition records, calibrations applied, quality control information, etc. that are associated with the data product's generation or processing.

These use case scenarios are representative of many different question types that are routinely asked for all data products produced by the ACOS data ingest pipeline, and we believe these are representative of use cases that are common in any science data pipeline application.

3 Provenance Requirements

We require a provenance infrastructure that supports queries, filtering, and reasoning by domain concepts. A design requiring the hardwiring of domain concepts into the provenance model, or into the system logic that accesses the provenance store, is undesirable because it will be difficult to maintain and extend, and furthermore, such a hardwired application will make interoperation more challenging.

The provenance infrastructure must also support existing data systems, requiring little to no modification of the processing pipeline; ACOS is a production system, and we do not have the opportunity to re-engineer it. The system should also support generating some amount of provenance for existing processed data. It is not feasible to reprocess all existing data, and doing so with the current pipeline may introduce discrepancies between the newly processed products and archived products processed on a earlier and different version of the pipeline. The provenance capture should be configurable such that as much provenance as possible can be generated based on our understanding of an earlier version of the pipeline, without forcing us to re-run data processing.

Finally, since ACOS is a distributed system the provenance infrastructure must also work as a distributed system. Provenance should be gathered where processing occurs and made available as part of a distributed provenance store.

4 Data Model

4.1 Provenance Representation

To support our provenance requirements we have elected to use OWL ontologies to model both domain and provenance concepts. The provenance and domain base ontologies are independent, but the system's individuals reference both models (via multiple-inheritance), so queries, filtering and reasoning by either domain or provenance concepts are supported. This design supports our desire to build a maintainable system that refrains from hardcoding solar terrestrial concepts into the base provenance model or provenance logic.

We have chosen as our provenance model the Inference Web [6] Framework's Proof Markup Language [3], [7] (PML) because of its capabilities in representing conclusions, justifications (inference and source usage), and explanations. Another particularly useful aspect of the PML model is its separation of the process engine and process rule concepts. By defining these concepts separately, PML can represent both the process that was executed (PML InferenceEngine) and the rule (PML InferenceRule) by which the executed process operated. Another way to view this concept separation is that PML can capture both execution history and execution purpose. Inference rules are a pivotal concept of the justification aspect of PML and provide a clear mechanism for relating domain concepts to a provenance causality graph. While other provenance models such as Open Provenance Model (OPM) could have provided some of the foundation provided in PML, we found some of the core representational constructs such as those mentioned above to be well suited for our applications. For further analysis of the relative benefits of PML and OPM, see 'Towards Usable and Interoperable Workflow Provenance: Emperical Case Studies using PML' [4] and 'Domain Knowledge and Provenance in Science Data Systems' [5].

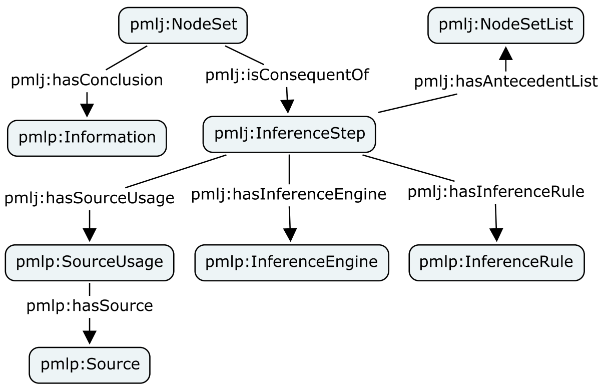


Fig. 2.
Basic PML NodeSet

In PML, a piece of information (the conclusion) and its justification(s) are modeled as a nodeset. A nodeset justification, known as an inference step, is used to describe the engine or source and the rule used to generate the nodeset conclusion. Each inference step may specify a list of nodesets, known as antecedents, whose conclusions it is dependent upon. The antecedent relations between nodesets can be used to build a causality graph, or explanation, for the conclusion of the nodeset. The fundamental classes and properties of a PML nodeset are shown in Figure 2.

4.2 Domain Representation

The VSTO[2] solar-terrestrial ontology, developed during our previous experience deploying a semantic virtual observatory [1], [2] will be used as one of our core science domain models. The VSTO ontology provides a model for data products, instruments, and parameters related to solar-terrestrial data systems. The VSTO ontology does not currently describe the processing that occurs in a typical science data ingest pipeline (calibrations, transformations, data filtering, quality control processes, etc.) so we are developing our own science data processing ontology based on experience gained during this[3] project and similar work with the MDSA[4] project.

Figure 3 illustrates some domain model concepts from the VSTO and (in-development) science data processing ontologies that we will be integrating with the ACOS provenance. Of particular interest in this example is the vsto:InstrumentOperatingMode concept, which is defined as the *configuration and process that allows an instrument to produce the required signal*. In the solar terrestrial domain terminology, an operating mode is not treated as an input configuration to a process, not as an artifact, but as a description of the state of the instrument and the entailing process for data capture. It is a description of how an instrument performs data capture of a specific parameter type. In fact, vsto:InstrumentOperatingMode was originally modeled as a subclass of the class vsto:AbstractProcess. The VSTO InstrumentOperatingMode and science data processing ontology DataCapture concepts relate to each other in much the same way the PML InferenceRule and InferenceEngine relate, and in the next section we will show how they can be integrated.

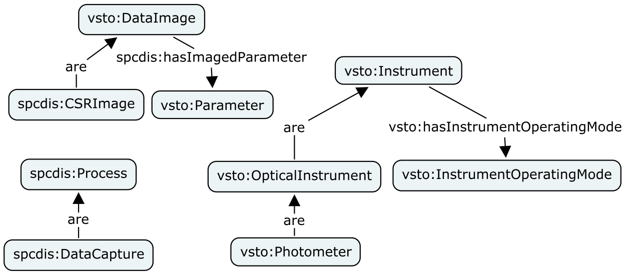


Fig. 3.
VSTO (vsto) and Science Data Processing (spcdis) domain concepts

4.3 Provenance and Domain Model Integration

The provenance and domain ontology concepts are integrated not in the model definitions, but in the individuals' declarations by taking advantage of OWLs natural support for multiple-inheritance. Where it is deemed beneficial to express both domain and provenance concepts, individuals (ontology class instances) are defined with multiple types, one type from the provenance model and at least one type from the domain ontologies. As an example, the science data processing ontology may define an individual spcdis:FlatFieldCalibration of type spcdis:Calibration and type pmlp:InferenceRule. The use case *'What calibrations have been applied to*

[2] Virtual Solar Terrestrial Observatory, http://vsto.org/

[3] Semantic Provenance Capture in Data Ingest Systems

[4] Multi-Sensor Data Synergy Advisor, http://tw.rpi.edu/portal/MDSA

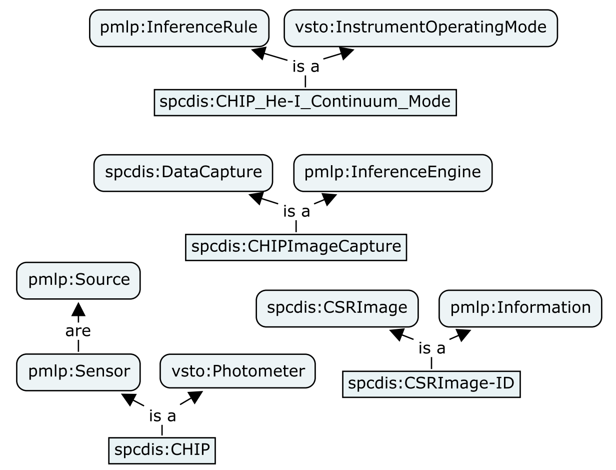


Fig. 4.
Individuals integrating provenance and domain models

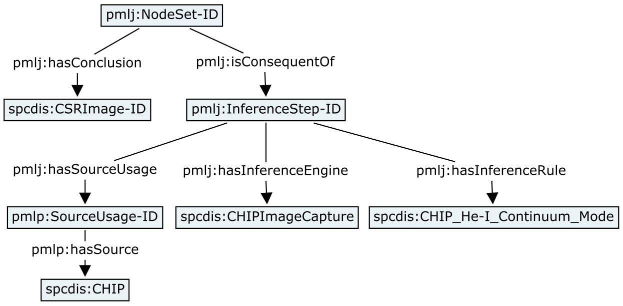


Fig. 5.
PML NodeSet comprised of domain model integrated individuals

this data product artifact?' may now be answered by querying the data product's causality graph for inference rule's of type spcdis:Calibration used by any nodeset's justification.

Figure 4 shows how individuals of the domain concepts illustrated in Figure 3 may be mapped to PML provenance concepts. The instrument operating mode instance is modeled not as the conclusion of a some nodeset's justification that acts as an antecedent to the data capture nodeset's justification, but as the rule by which data capture occurs. The individual spcdis:CHIP is defined as both a vsto:Photometer and a pmlp:Sensor, allowing it to be both the source of the conclusion of the data capture justification as well as assert any properties for which vsto:Photometer is in the domain.

We can now model a PML NodeSet using individuals that have type and properties from domain ontologies, as shown in Figure 5. Integrating domain types and properties with the provenance-based causality graph allows us to answer complex use case scenarios such as *'What calibrations have been applied to this data product artifact?'* by performing reasoning on domain concepts integrated with the individuals in the provenance.

5 Provenance Capture

To assist in PML generation, we describe a type of program referred to as a PML data annotator. A PML data annotator is a simple program whose sole purpose is to capture the provenance of a single decision/process in a decision system and encode that provenance as a PML nodeset. PML data annotator programs are run as components of a workflow; either as part or separate to the actual decision processing. When run as part of the data processing, the PML data annotator invokes the inference engine directly; extracting required processing inputs from antecedent nodesets and passing this information during inference engine invocation.

For the ACOS provenance capture we will utilize a parallel workflow, where PML data annotators do not directly invoke inference engines but reconstruct the processing of the existing data ingest pipelines. This architecture also supports our need for provenance generations for archived or pre-existing data products without preprocessing of the data. The PML data annotators are in in a workflow that simulates the processing of the data pipeline using analysis of existing artifacts and information about the data processing encoded in the PML data annotator configuration to reconstruct provenance. The PML data annotator workflow can be reconfigured to simulate different variations of the data processing pipeline to generate provenance from data processing pipelines that are no longer active.

6 Provenance Search

We will utilize the search and explanation capabilities of the Inference Web toolset to provide both a free text and guided search on provenance and domain concepts. Guided searches generate a SPARQL query on the provenance + domain RDF and free text searches currently performs a standard full text index search on the same. Search results

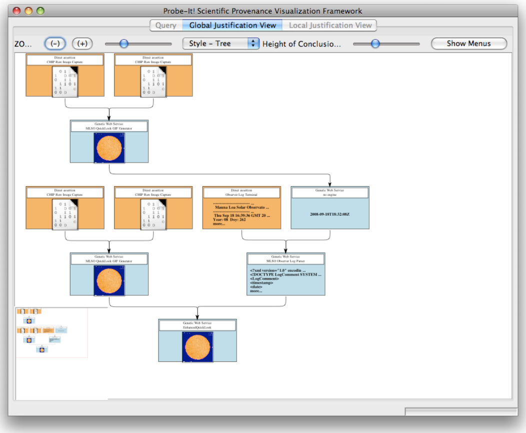


Fig. 6.
Probe-It visualizing the provenance of a CHIP QuickLook Visual Product

can be viewed using a number of tools including the Inference Web browser, where the user can explore the provenance encoding in detail, or in Probe-It!, a second generation PML provenance visualization tool using applet technology for greater graph functionality, introduced in the next section.

7 Provenance Visualization

Probe-It! [8] is graphical browser of PML-based provenance, developed by Cyber-ShARE at the University of Texas at El Paso. Probe-It! generates a causality graph for all antecedents to a specified nodeset and generates a visual representation (where applicable) for the conclusion of all nodesets in the graph. A Probe-It! visualization of the CHIP QuickLook[5] Visual Product is shown in Figure 6. Our primary interest in Probe-It for the ACOS provenance is in enabling scientists to better understand imperfections in and processing consequences upon science data images.

8 Discussion and Conclusion

To date we have reconstructed provenance for the Quick Look visual product from the CHIP Intensity data ingest and shown how weather condition information, previously not propagated, can provide value added to the data product. We have introduced our work constructing a domain-aware provenance store built using a generic, extensible provenance model and a solar-terrestrial domain model.

We described how this provenance store can be used to represent the relationships expressed in our use cases, and why these use cases are important in increasing user trust in the data products. While we did this in one workflow setting, since the use cases are representative of those in many scientific workflow settings, we believe this work provides a foundation for scientific workflow provenance applications. We have described in brief how we intend to use a parallel workflow to reconstruct and generate provenance, and the semantically-enabled provenance search, explanation, and visualization tools we will provide for the end users.

The next stage of our work will involve further modeling of data pipeline concepts in the ACOS provenance ontology, further documentation of the ACOS data pipelines, and construction of PML wrappers for newly documented sections of the data pipelines. We also intend to prototype semantic provenance faceted-search interfaces, move our free text search to the Apache Lucene text search engine, and develop new visual representations of nodeset conclusions in our visual provenance browser. Following the completion and testing of the ACOS application, we will separate our extensions that are ACOS-specific from those that are general to science applications and release scientific provenance module extensions for VSTO and PML ontologies and related wrapper support tools.

Acknowledgments

The SPCDIS[6] project is funded by NSF, Office of Cyber Infrastructure under the SDCI program, grant number OCI-0721943, and in collaboration with UTEP CyberShare Center, funded by NSF under the CREST program, grant number HRD-0734825.

References

[1] McGuinness, D., Fox, P., Cinquini, L., West, P., Garcia, J., Benedict, J., Middleton, D.: The Virtual Solar-Terrestrial Observatory: A Deployed Semantic Web Application Case Study for Scientific Research. In the proceedings of the 19th Conference on Innovative Applications of Artificial Intelligence (IAAI). Vancouver, BC, Canada, July 2007, 1730-1737 and AI magazine, 29, #1, 65-76.

[5] QuickLook images are lightly-calibrated visual approximations of the data image, generated in real-time and used primarily as quality checks on instrument operation

[6] Semantic Provenance Capture in Data Ingest Systems

[2] Fox, P., McGuinness, D., Cinquini, L., West, P., Garcia, J., Benedict, J., Middleton. D.: Ontology-supported Scientific Data Frameworks: The Virtual Solar-Terrestrial Observatory Experience. Computers and Geosciences. Vol. 35, Issue 4, pp 724-738.

[3] McGuinness, D., Ding, L., Pinheiro da Silva, P., Chang, C.: PML 2: A Modular Explanation Interlingua. In ExaCt pp. 49-55 Also Stanford KSL Tech Report KSL-07-07 (2007)

[4] Michaelis, J., Ding, L., Shangguan, Z., Zednik, S., Huang, R., Pinheiro da Silva, P., Del Rio, N., McGuinness, D.: Towards Usable and Interoperable Workflow Provenance: Empirical Case Studies using PML. To appear in the Proceedings of the First International Workshop on the role of Semantic Web in Provenance Management, Chantilly, VA. (2009)

[5] Zednik, S., Fox, P., McGuinness, D.: Domain Knowledge and Provenance in Science Data Systems. To appear in Emerging Issues in e-Science: Collaboration, Provenance, and the Ethics of Data (IN13), AGU Fall 2009 Meeting, San Francisco, CA. (2009)

[6] McGuinness, D., Ding, L., Pinheiro da Silva, P.: Explaining Answers from the Semantic Web: The Inference Web Approach. Web Semantics: Science, Services and Agents on the World Wide Web Special issue: International Semantic Web Conference 2003 - Edited by K.Sycara and J. Mylopoulous. 1(4). Fall, 2004. Also, Stanford KSL Tech Report KSL-04-03.

[7] Pinheiro da Silva, P., McGuinness, D., Fikes, R.: A Proof Markup Language for Semantic Web Services. Information Systems, 31(4-5), June-July 2006, pp 381-395. Prev. version, KSL Tech Report KSL-04-01 (June 2006)

[8] Del Rio, N., Pinheiro da Silva, P.: Probe-It! Visualization support for provenance. Proceedings of the Second International Symposium on Visual Computing (ISVC 2), Lake Tahoe, NV, USA. Volume 4842 of LNCS, pages 732-741, Springer (2007)

Reasoning With Provenance, Trust and all that other Meta Knowlege in OWL

Simon Schenk*, Renata Dividino* and Steffen Staab*
* ISWeb Research Group University of Koblenz-Landau
Email: sschenk, dividino, staab@uni-koblenz.de

***Abstract*—For many tasks, such as the integration of knowledge bases in the semantic web, one must not only handle the knowledge itself, but also characterizations of this knowledge, e.g.: (i) where did a knowledge item come from (i.e. provenance), (ii) what level of trust can be assigned to a knowledge item, or (iii) what degree of certainty is associated with it. We refer to all such kinds of characterizations as *meta knowledge*. Approaches for providing meta knowledge for query answers in relational databases and RDF repositories, based on algebraic operations, exist. As query answering in description logics in general does not boil down to algebraic evaluation of tree shaped query models, these formalizations do not easily carry over. In this paper we propose a formalization of meta knowledge, which is still algebraic, but allows for the computation of meta knowledge of inferred knowledge in description logics, including reasoning with conflicting and incomplete meta knowledge. We use pinpointing to come up with meta knowledge formulas for description logics, which then can be evaluated algebraically. We describe and evaluate our prototypical implementation.**

I. Introduction

When exploiting explicit/inferred knowledge in the semantic web, one must not only handle the knowledge itself, but also characterizations of this knowledge, e.g.: (i) where did a knowledge item come from (i.e. provenance), (ii) what level of trust can be assigned to a knowledge item, or (iii) what degree of certainty is associated with it. We refer to all such kinds of characterizations as *meta knowledge*. On the semantic web, meta knowledge needs to be computed along with each reasoning task.

Meta knowledge can come in various, complex dimensions. Many simplifications done today, such as assuming trust to be measured on a scale from 1 to 10, are not justified. In contrast, actual information sources, modification dates, etc. should be tracked to establish trust [1]. We propose a flexible mechanism for tracking meta knowledge, which meets these requirements.

Various approaches to this problem have been proposed. They can be grouped in to three clusters: First, we have extensions of logical formalisms, e.g. description logics, to deal with a particular kind of meta knowledge. Most prominent are extensions for reasoning with uncertainty, such as fuzzy and probabilistic [2] or possibilistic [3] description logics. Other proposals exist, which are tailored to specific meta knowledge such as trust [4]. Second, for systems allowing for algebraic query evaluation (such as relational databases and SPARQL engines), more flexible mechanisms such as [5] and [6] have been proposed, which allow for many kinds of meta knowledge, but are limited to lower expressiveness of the underlying logical formalism. Third, the expressive system proposed by [7] has a rather ad-hoc semantics, which is partially defined in constructors in queries and hence can differ in each query evaluation.

To come up with a flexible mechanism, which at the same time supports expressive logics and multiple kinds of meta knowledge, a suitable formalization of meta knowledge in a semantically precise manner is needed. Moreover, such a mechanism must be supported with a suitable operationalization. From the existing approaches it is clear, that integrating an expressive meta knowledge language with an expressive base knowledge representation language is a non-trivial task, mainly because of the different foundations, i.e. algebra vs. logics, of the meta knowledge and base languages.

Expressive descriptions of meta knowledge in less expressive languages (such as SPARQL based on RDF) have been founded on a tree-based algebraic formalization. Reasoning frameworks, however, frequently have non-tree-based derivations used for consistency checking and querying. In order to be able to reason with meta knowledge, which we formalize as algebraic structure, on top of expressive base languages, we propose a reasoning framework for meta knowledge based on *pinpointing*. Pinpointing summarizes explanations for axioms in a single boolean formula, which then can be evaluated using a meta knowledge algebra. We provide a blackbox algorithm for reasoning with meta knowledge, and describe our prototypical implementation. The algorithm uses an existing description logic reasoner for entailment checks. Hence, the supported expressivity is that of the underlying description logic.

As a motivation, we first explain a short use case, before laying foundations and defining the semantics of meta knowledge. Afterwards we briefly discuss the complexity and our prototypical implementation. We review the related work and conclude the paper.

II. Use Case

In a common scenario for collaborative ontology editing we have public, living ontologies, for which users can propose changes [8] and which are possibly interlinked through imports or views. Applications include large medical and biological ontologies such as SNOMED or the Gene Ontology. The example in [8] is based on a use case at the UN's Food and Agricultural Organization FAO. A change can be the addition, change, or removal of an axiom. Users have different levels of expertise and hence their knowledge items are assigned different degrees of trustworthiness. Moreover, there may be conflicting changes or modifications, which make the ontology inconsistent. When answering queries and inferring knowledge in such systems, users need to know for example

- who contributed to axioms used to infer new knowledge,
- when they were last modified, and
- how trustworthy they are.

The derivation of meta knowledge can happen dynamically, in a completely open system comparable to today's wikis, where the change history is available for every user.

III. Foundations

As we use pinpointing as a vehicle for computing meta knowledge, we introduce pinpointing as a foundation for the rest of the paper and give some information of existing algorithms for finding pinpoints.

A. Pinpointing

The term pinpointing has been coined for the process of finding explanations for concluded axioms or for a discovered inconsistency. An explanation is a minimal set of axioms, which makes the concluded axiom true (or the theory inconsistent, respectively). Such an explanation is called a pinpoint. While there may be multiple ways to establish the truth or falsity of an axiom, a pinpoint describes exactly one such way.

Definition 1: Pinpoint.
A pinpoint for a entailed axiom A wrt. an ontology O is a set of axioms $\{A_1, ..., A_n\}$ from O, such that $\{A_1, ..., A_n\} \models A$ and $\forall A_i \in \{A_1, ..., A_n\} : \{A_1, ...A_{i-1}, A_{i+1}, ..., A_n\} \not\models A$. Analogously, a pinpoint for a refuted axiom A wrt. an ontology O is a set of axioms $\{A_1, ..., A_n\}$ from O, such that $\{A, A_1, ..., A_n\}$ is inconsistent and $\forall A_i \in \{A_1, ..., A_n\} : \{A, A_1, ...A_{i-1}, A_{i+1}, ..., A_n\}$ is not.

Hence, finding pinpoints for a refuted axiom corresponds to finding the Minimum Unsatisfiable Subontologies (MUPS) for this axiom [9].

Pinpointing is the computation of all pinpoints for a given axiom and ontology. The truth of the axiom can then be computed using the *pinpointing formula* [10].

Definition 2: Pinpointing Formula.
Let A be an axiom, O an ontology and $P_1, ..., P_n$ with $P_i = \{A_{i,1}, ..., A_{i,m_i}\}$ the pinpoints of A wrt. O. Let lab be a function assigning a unique label to an axiom. Then $\bigvee_{i=1}^{n} \bigwedge_{j=i}^{m_i} lab(A_{i,j})$ is a pinpointing formula of A wrt. O.

A pinpointing formula of an axiom A describes, which (combination of) axioms need to be true in order to make A true or inconsistent respectively.

B. Finding all Pinpoints

Algorithms for finding Pinpoints can be grouped into three groups:

a) Finding one pinpoint: Algorithms to find one pinpoint can either derive a pinpoint by tracking the reasoning process of a tableaux reasoner, or use an existing reasoner as a black box. In the latter case, a pinpoint is searched by subsequently growing (shrinking) a subontology until it starts (stops) entailing the axiom under question. Based on the so derived smaller ontology the process is refined, until a pinpoint has been found. The advantage of blackbox algorithms is that they can support any description logic, for which a reasoner is available [9]. Extending a tableaux reasoner on the other hand is complicated, but yields better performance, as a pinpoint can be generated in parallel to a usual subsumption check with low overhead [10].

b) Finding all Pinpoints using a Tableaux Reasoner: Baader and Peñaloza have shown that forest tableaux with equality blocking (and hence, reasoners for the web ontology language OWL) can be extended to find pinpointing formulas [10]. In this approach a tableaux reasoner is extended to find not only one, but all pinpoints. Special care needs to be taken in order to ensure termination of the tableaux algorithm. As an advantage, the overhead for pinpointing is lower compared to a blackbox algorithm. Moreover, this approach can derive a compact representation of the pinpointing formula, which might have worst-case exponential size in conjunctive normal form. To the best of our knowledge none of the standard reasoners for complex description logics has been extended in this direction yet.

c) Finding all Pinpoints using Blackbox Algorithms: The most performant black-box algorithms for finding all justifications first extract a relevant module from the overall ontology, ensuring that this module yields the same inferences with respect to the axiom on interest. Then, starting from a single pinpoint, which is computed using an algorithm discussed in paragraph III-B0a, Reiter's Hitting Set Tree algorithm [11] is used to compute all pinpoints by iteratively removing one axiom from the pinpoint at hand and growing it to a full pinpoint again [12], [13]. Using this kind of algorithm, a lot of subsumption checks in the underlying description logic are needed.

For both, tableaux based and black box algorithms, the worst case complexity of finding all pinpoints is rather high, as there can be exponentially many pinpoints for any given ontology. However, recent work has shown that in the average case, the number is significantly lower [10].

IV. Syntax of Meta Knowledge

Meta knowledge can be expressed as annotations on axioms. Annotations are of main importance for the management of ontologies as annotations may be used to support analysis during collaborative engineering.

We associate ontology axioms with meta knowledge through axiom annotations. Basically, an axiom annotation assigns an annotation object to an axiom e.g. "(brokenLimb subClass Limb) was created by Crow on 15.01.2008". A meta knowledge annotation consists of an annotation URI and a meta knowledge object specifying the value of the annotation. In our case, the meta knowledge object is a constant-value

TABLE I
EXAMPLE OF META KNOWLEDGE ASSOCIATED WITH AXIOMS.

ID	Relevant Facts	Meta Knowledge
$\#_1$	[limb1 Limb]	statedBy Crow; modified 14-01-2008
$\#_2$	[limb2 Limb]	statedBy Crow; modified 14-01-2008

$\#_3$	[limb1 isBroken true]	statedBy House;
		modified 15-01-2008;

$\#_4$	[limb2 isWrenched true]	statedBy House; modified 15-01-2008

representing who asserted/modified the axiom, when the axiom was last modified, or the uncertainty degree of the axiom, or a combination thereof. The grammar for meta knowledge annotations as an extension of OWL 2 annotations[1] is as follows:

OWLAxiomAnnotation := ’OWLAxiomAnnotation’
’(’**OWLAxiom** **OWLAnnotation**$^+$’)’
OWLAnnotation := **OWLConstantAnnotation**
OWLConstantAnnotation := **MetaKnowledgeAnnotation**
MetaKnowledgeAnnotation := ’MetaKnowledgeAnnotation’
’(’**AnnotationURI** **MetaKnowledge**$^+$’)’
MetaKnowledge := **CertaintyAnnotation** | **DateAnnotation** | **SourceAnnotation** | **AgentAnnotation**
CertaintyAnnotation := ’CertaintyAnnotation’
’(’**AnnotationValue**’)’
SourceAnnotation := ’SourceAnnotation’ ’(’**AnnotationValue**’)’
DateAnnotation := ’DateAnnotation’ ’(’**AnnotationValue**’)’
AgentAnnotation := ’AgentAnnotation’ ’(’**AnnotationValue**’)’

In our scenario we assume that we are looking for meta knowledge information about all limbs which are either broken or wrenched. Our ontology contains the axioms and meta knowledge annotations summarized in Table I.

An example of how meta knowledge is represented and associated with OWL axioms is presented below.

```
OWLAxiomAnnotation(ClassAssertion(limb1 Limb)
  MetaKnowledgeAnnotation(
    annot1 AgentAnnotation(Crow)))
OWLAxiomAnnotation(
  PropertyAssertion(limb1 isBroken true)
  MetaKnowledgeAnnotation(
    annot2 AgentAnnotation(House)))
```

Annotations, however, have no semantic meaning in OWL 2. All annotations are ignored by the reasoner, and they may not themselves be structured by further axioms. For this reason, as next step, we first define the semantics of meta knowledge, later we describe how meta knowledge can be combined with reasoning.

V. SEMANTICS OF META KNOWLEDGE

Meta knowledge can have multiple dimensions, e.g. uncertainty, a least recently modified date or a trust metric. For this paper, we assume that these (and possible further) dimensions are independent of each other.

Definition 3: Knowledge dimension. A knowledge dimension D is an algebraic structure $(B_D, \vee_D, \wedge_D)$, such that $(B_D, \vee_D)$ and $(B_D, \wedge_D)$ are complete semilattices.

B_D represents the values the meta knowledge can take, e.g. all valid dates for the least recently modified date or a set of knowledge sources for provenance. As $(B_D, \vee_D)$ and $(B_D, \wedge_D)$ are *complete* semilattices, they are, in fact, also lattices. Hence, there are minimal elements in the corresponding orders.

As an example, let I be the meta knowledge interpretation[2] that is a partial function mapping axioms into the allowed value range of a meta knowledge dimension, and A and B be axioms of an ontology such that $A \neq B$. Provenance, i.e. the set of knowledge sources a piece of knowledge is derived from, can be modeled as:

- $I(A \vee B) = I(A) \cup I(B)$
- $I(A \wedge B) = I(A) \cup I(B)$

The least recently modified date could be modeled as:

- $I(A \vee B) = min(I(A), I(B))$
- $I(A \wedge B) = max(I(A), I(B))$

Axioms can be assigned meta knowledge from any of the meta knowledge dimensions. Within a single assignment, the meta knowledge must be uniquely defined.

Definition 4: Meta Knowledge Assignment.
A meta knowledge assignment M is a set $\{(D_1, d_1 \in D_1), ..., (D_n, d_n \in D_n)\}$ of pairs of meta knowledge dimensions and corresponding truth values, such that $D_i = D_j \Rightarrow d_i = d_j$.

In our running example, the meta knowledge assignment for *PropertyAssertion(limb1 isBroken true)* is $\{$*(agent, Crow), (date, 15.01.2008)*$\}$

Without loss of generality we assume a fixed number of meta knowledge dimensions. As a default value for D_n in a meta knowledge assignment we choose $\bot_D$.

To allow for reasoning with meta knowledge, we need to formalize, how meta knowledge assignments are combined. *How provenance* [14] is a strategy, which describes how an axiom A can be inferred from a set of axioms $\{A_1, ..., A_n\}$, i.e. it is a boolean formula connecting the A_i. We call a logical formula expressing how provenance a *meta knowledge formula*. For example the following query finds all limbs, that are either broken or wrenched:

$$x : \text{Limb} \wedge (\langle x, \text{true}\rangle : \text{isBroken} \vee \langle x, \text{true}\rangle : \text{isWrenched}).$$

The results of this query and the corresponding meta knowledge formulas are:

$$\text{limb1} \mid \#_1 \wedge \#_3 \quad \text{and} \quad \text{limb2} \mid \#_2 \wedge \#_4$$

The operators for meta knowledge dimensions extend to meta knowledge assignments, allowing us to compute meta knowledge for entailed knowledge by evaluating the corresponding meta knowledge formula.

Definition 5: Operations on Meta Knowledge Assignments. Let A, B be axioms and meta$(A) = \{(D_1, x_1), ..., (D_n, x_n)\}$ and meta$(B) = \{(E_1, y_1), ..., (E_m, y_m)\}$ be meta knowledge

[1] OWL 2 Web Ontology Language: Spec. and Func.-Style Syntax: http://www.w3.org/TR/2008/WD-owl2-syntax-20081202

[2] The administrator defines the intended semantics of these properties in order to facilitate query processing with complex expressions and pattern combinations.

assignments. Let dim(A) be the set of meta knowledge dimensions of A. Then meta$(A) \vee$meta$(B) = \{(D, x \vee_D y) | (D, x) \in$ meta(A) and $(D, y) \in$ meta$(B)\}$. $\wedge$ is defined analogously.

Having defined the operations on meta knowledge assignments, we can define formulas using these operations.

Definition 6: Meta Knowledge Formula.
Let A be an axiom of an ontology O, *lab* a function assigning a unique label to each A_i from O and *lab(O)* the set of all labels of axioms in O. A meta knowledge formula ϕ for a axiom A wrt. an ontology O is boolean formula over the set of labels $\{lab(A_1), ..., lab(A_n)\}$ of axioms $\{A_1, ..., A_n\}$ from O, such that for each valuation $V \subset lab(O)$, which makes ϕ true, the following holds: $lab^{-}(V) \models A$.

The meta knowledge of an axiom A within a meta knowledge dimension is obtained by evaluating the corresponding meta knowledge formula after replacing axiom labels with the corresponding meta knowledge in the dimension under consideration.

Definition 7: Meta Knowledge of an Axiom.
Let *meta* be a function mapping from an axiom to a meta knowledge assignment in dimension D. The meta knowledge of an axiom A wrt. O in D is obtained by evaluating the formula obtained from A's meta knowledge formula wrt. O by replacing each $lab(A_i)$ with the corresponding $meta(A_i)$.

In our running example, if we model the agent dimension as where provenance, the meta knowledge of the query result for *limb1* is: *(agent, $\{$Crow$\}$) $\wedge$ (agent, $\{$House$\}$) = (agent, $\{$Crow$\}$ $\cup$ $\{$House$\}$) = (agent, $\{$Crow, House$\}$).*

In contrast to [5] we omit the $\neg$ operator in our formalization, as description logics are monotonic and $\neg$ in [5] allows for default negation. While axioms in the underlying description logic may contain negation, this negation is not visible on the level of meta knowledge.

VI. Extended Semantics for Conflicting Meta Knowledge

In the following we extend our model to support conflicting meta knowledge, which can arise from conflicting changes or meta knowledge assignments by multiple users in an axiom.

Definition 8: Extended knowledge dimension. A extended knowledge dimension D is an algebraic structure $(B_D, \vee_D, \wedge_D, \oplus_D)$, such that $(B_D, \vee_D)$, $(B_D, \wedge_D)$ and $(B_D, \oplus_D)$ are complete semilattices. The minimum of $(B_D, \oplus_D)$ is called $\perp_D$.

As an example, let I be the meta knowledge interpretation that is a partial function mapping axioms into the allowed value range of a meta knowledge dimension A be an axiom of an ontology, and I_1 and I_2 interpretations of multiple meta knowledge assertions to A. Provenance, i.e. the set of knowledge sources a piece of knowledge is derived from, can be modeled as:

- $I(A \oplus A) = I_1(A) \cup I_2(A)$

The least recently modified date could be modeled as

- $I(A \oplus A) = max(I_1(A), I_2(A))$

Consider the following example presented in Table II and assume that two users assert the same axiom at different times into the example ontology:

TABLE II
Extension of our scenario where we assume two users assert the same axiom at different times

ID	Relevant Facts	Meta Knowledge
$\#_1$	[limb1 Limb]	statedBy Crow; modified 14-01-2008
$\#_2$	[limb2 Limb]	statedBy Crow; modified 14-01-2008

$\#_3$	[limb1 isBroken true]	statedBy House;
		modified 15-01-2008;

$\#_4$	[limb2 isWrenched true]	statedBy House; modified 15-01-2008
. . .		

$\#_{10}$	[BrokemLimb subClassOf (isBroken true)]	statedBy Crow;
		modified 14-01-2008
		statedBy House;
		modified 15-01-2008

In our running example, the meta knowledge assignment for axiom $\#_{10}$ is $\{$*(agent, Crow), (date, 14.01.2008), (agent, House), (date, 15.01.2008)*$\}$

In our running example, if we model the least recently modified date dimension, the meta knowledge of the axiom $\#_{10}$ is: *(date, $\{$14.01.2008$\}$) $\oplus$ (date, $\{$15.01.2008$\}$) = (date, max($\{$14.01.2008$\}$, $\{$15.01.2008$\}$)) = (date, $\{$15.01.2008$\}$).*

Consider the extended semantics of meta knowledge, we need to describe a different way of finding a meta knowledge formula. We redefine the *meta* function of Definition 7, such that it computes $\bigoplus$ of all meta knowledge assignments available for a statement.

Definition 9: Meta Knowledge of an Axiom. Extended Definition.
Let allmeta: axioms $\rightarrow 2^{\text{MKAssignments}}$ be a function mapping from an axiom to all meta knowledge assignments to that axiom in a meta knowledge dimension D. Then $meta(A)$ is defined as $\bigoplus$allmeta(A).

This definition of *meta* not only allows to aggregate meta knowledge from multiple sources, but also to gracefully handle unknown meta knowledge, i.e. situations where a knowledges source does not provide a truth value for some meta knowledge dimension.

For example, we want to model the agent dimension as where provenance, the meta knowledge of the query result for: *ClassAssertion(BrokenLimb limb1).* The axiom is satisfiable, so the corresponding pinpointing formula is $\#_1 \wedge \#_3 \wedge \#_{10}$ = *(agent, $\{$Crow$\}$) $\wedge$ (agent, House$\}$) $\wedge$ ((agent, $\{$Crow$\}$) $\oplus$ (agent, $\{$House$\}$)) = (agent, $\{$Crow, House$\}$).*

VII. Computing Meta Knowledge using Pinpoints

In order to allow for an *algebraic* evaluation of meta knowledge dimensions, we need a *single* boolean formula. In meta knowledge mechanisms like [5], it is derived from queries in relational algebra. When reasoning with description logics, however, such a rather simple algebraic foundation of the basic language does not exist. Instead, multiple axioms may be needed to establish the truth or falsity of inferred knowledge. For this purpose, we have defined the *meta knowledge* formula in definitions 6 and 9.

As we can see above, definitions 2, 6 and 9 are quite similar. In fact, a pinpointing formula provides exactly what we need for a meta knowledge formula: All combinations of axioms, which can be used to establish the truth or falsity of inferred knowledge.

For this reason, when reasoning in a logic, where a pinpointing algorithm is known, we can compute a pinpointing formula and then derive meta knowledge as usual.

VIII. Complexity

The complexity of this rather naive approach for computing meta knowledge is equivalent to the computation of pinpointings. Due to the algebraic specification of meta knowledge the complexity of the meta knowledge formula is polynomial. If the meta knowledge formula is in conjunctive normal form, however, we might encounter an exponential blowup. Approaches for computing pinpointings like [10] which, rather than representing pinpoints formula in a conjunctive normal form, derive a compact representation of the pinpoints formula benefit the computation of meta knowledge since they avoid exponential blowup.

IX. Experiments

In this section, we present the evaluation results of our algorithm. The experiments were performed on a Windows XP SP3 System and 512MB maximal heap space was set. Sun's Java 1.5.0 Update 6 was used for Java-based tools.

Reasoning with Meta knowledge The framework for reasoning with meta knowledge is is available as a Java prototype and is available as an open source implementation at <http://isweb.uni-koblenz.de/Research/MetaKnowledge> together with example of ontologies extended with meta knowledge. The aggregation of meta knowledge is computed based on the model presented in Section VI and Section VII.

Reasoning with Pinpointing The framework for reasoning with pinpointing is implemented with the OWL API and the OWL-DL reasoner, Pellet[3]. Pellet provides the axiom pinpointing service for debugging ontologies that, for any arbitrary entailment derived by a reasoner from an OWL-DL knowledge base, returns the minimal set (explanations) of source axioms that cause an inconsistency and the relation between unsatisfiable concepts. The algorithm is black box based. In the following experiments we compare the processing time of our approach with reasoning with pinpointing approach.

Data Our sample data consists of 7 typical existing OWL ontologies used for debugging. This dataset has already been used for tests the computing time of laconic justifications in [15]. Table III shows the number of entailments that hold in them and provide the range of expresivity. Each ontology was classified in order to determine the unsatisfiable classes. This classes were selected as input (query) to compute the meta knowledge degree and pinpoints. For each query the time to compute all pinpoints and the meta knowledge degree was recorded.

TABLE III
Ontologies uses in experiment. Table taken from [15]

ID	Ontology	Expressivity	Axioms	No. Entailments
1	Economy	$\mathcal{ALCH}(\mathcal{S})$	1625	51
2	People+Pets	$\mathcal{ALCHOIN}$	108	33
3	MiniTambis	$\mathcal{ALCN}$	173	66
4	Transport	$\mathcal{ALCH}$	1157	62
5	University	$\mathcal{SOIN}$	52	10
6	Chemical	$\mathcal{ALCHF}$	114	44
7	EarthRealm	$\mathcal{ALCHO}$	931	543

Evaluation Results Table IV displays the times for reasoning with meta knowledge and reasoning with pinpointing. For each ontology, we have computed all pinpoints for all unsatisfiable classes and reported the overall computing time. The experiments was done 10 times and the average time was considered. We can observe that the time for computing the meta knowledge degree takes longer than the computation of pinpointing (in average 4,9 ms longer). This is to be expected since the computation of meta knowledge degree is done once all justifications are already computed as we have shown in Section VIII. In all in all, the processing times presented in Table IV are still acceptable for interactive applications, and thus this approach can be used for solutions in real time.

TABLE IV
Times (in ms) to compute pinpointing vs. meta knowledge degree

ID	Ontology	Pinpointing	Meta Knowledge
1	Economy	347,63	348,24
2	People+Pets	328	329,12
3	MiniTambis	152,78	158,69
4	Transport	864,75	874,83
5	University	95,48	98,96
6	Chemical	3770,33	3781,17
7	EarthRealm	3030,06	3032,50

We expect optimizations to reduce the processing time to less than a second in the average case also for the more complex ontologies. As we are only interested in computing the meta knowledge, we can direct the pinpointing algorithm to only compute those pinpoints resulting in the highest meta knowledge values. The optimization will be reported in future work.

X. Related Work

Related work can be grouped into the following categories: (i) Extensions of description logics with a particular meta knowledge dimension, especially uncertainty. (ii) General meta knowledge for query answering with algebraic query languages. (iii) Extensions of description logics with general meta knowledge and (iv) meta knowledge for other logical formalisms.

ad (i) Several multi-valued extensions of description logic have been proposed: [2] propose fuzzy and probabilistic extensions of the DLs underlying the web ontology language OWL. [3] describe an extension towards a possibilistic logic. Another extension towards multi valued logic is presented by [4]. They target at trust and paraconsistency instead of uncertainty. OWL 2 is extended to reasoning over logical

[3] Pellet Reasoner: http://clarkparsia.com/pellet/

bilattices. Bilattices which reflect the desired trust orders are then used for reasoning. [16] provide an extension to reasoning in OWL with paraconsistency.

All of these approaches have in common, that they modify the character of models in the underlying description logic, e.g. to fuzzy or possibilistic models. In our approach in contrast, we reason on a meta level: While the underlying model remains unchanged, we compute consequences of annotations on axioms. This meta level reasoning is not possible in the approaches proposed above. Unlike general meta knowledge, these approaches are more tailored to a specific need and hence reasoning is cheaper for some. Particularly for fuzzy, possibilistic and paraconsistent description logics, the complexity of the underlying logic carries over, while in our case additional complexity is introduced through pinpointing.

ad (ii) Meta knowledge to algebraic languages has been proposed by various authors, for example for the Semantic Web Query Language SPARQL [5] and for relational databases [6]. In [17] the authors have propose a framework for meta knowledge management with support for querying and updating RDF/S graphs that takes into account both RDF named graphs and RDFS inference. While the actual meta knowledge formalisms are comparable to ours, the underlying languages are of lower expressivity, typically Datalog. Meta knowledge formulas in these language can directly be derived from the tree shaped representation of a query, which is not possible in description logics.

ad (iii) [7] propose a meta knowledge extension of OWL, which is also based on annotation properties. Even though meta knowlege can be expressed in ways comparable to ours, it has a rather ad-hoc semantics, which may differ from query to query. In our approach, meta knowledge and classical reasoning take place in parallel. Hence, we can answer queries such as "Give me all results with a confidence degree of $\geq x$". In contrast, reasoning on the ontology and meta level in [7] is separated. As a result, queries such as the following can be answered: "Give me all results, which are based on axioms with a confidence degree of $\geq x$". Although this difference might seem quite subtle, depending on the meta knowledge dimension, e.g. probabilistic confidence, these queries may have very different results.

ad (iv) [18] propose an extension of Datalog with weights, which are based on c-semirings and can be redefined to reflect various notions of trust and uncertainty. Our meta knowledge dimensions are similar to c-semirings, but additionally allow to handle conflicting meta knowledge using a third operator. As c-semirings have been investigated in great detail and have some desirable properties[4], a modification of our work towards similar algebraic structures might introduce additional interesting properties of meta knowledge.

XI. Conclusion

We have introduced a formalization of meta knowledge that allows to handle conflicting and incomplete meta knowledge on the Semantic Web. Meta knowledge per se cannot easily be built into a logical formalism such as description logics. Hence, we have provided an operationalization based on pinpointing, in order to derive a meta knowledge formula, which can easily be evaluated. Extensions of the approach beyond description logics are possible, based on pinpointing. Currently, we are working on the optimization of the algorithms for computing meta knowledge. The optimization are possible based on the observation, that we no longer need to compute all pinpointing formulas in oder to determine the meta knowledge but only computing a relevant subset of all pinpoints.

XII. Acknowledgements

This research was supported by the European Commission under contract IST-FP6-026978, X-Media and contract IST-2006-027595, Lifecycle Support for Networked Ontologies - NeOn. The expressed content is the view of the authors but not necessarily the view of the X-Media and NeOn projects.

References

[1] Harry Halpin: Provenance: The Missing Component of the Semantic Web, CEUR Workshop Proceedings, online CEUR-WS.org/Vol-447/paper1.pdf

[2] Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the Semantic Web. Journal of Web Semantics **6**(4) (2008) 291–308

[3] Qi, G., Pan, J.Z., Ji, Q.: Extending Description Logics with Uncertainty Reasoning in Possibilistic Logic. In: ECSQARU '07, Springer (2007) 828–839

[4] Schenk, S.: On the Semantics of Trust and Caching in the Semantic Web. In: ISWC2008. Volume 5313 of LNCS., Springer (2008) 533–549

[5] Schueler, B., Sizov, S., Tran, D.T.: Querying for Meta Knowledge . In: WWW2008, ACM (2008) 625–634

[6] Buneman, P., Khanna, S., Tan, W.C.: Why and Where: A Characterization of Data Provenance. In: ICDT. Volume 1973 of LNCS. (2001) 316–330

[7] Tran, D.T., Haase, P., Motik, B., Cuenca-Grau, B., Horrocks, I.: Metalevel Information in Ontology-Based Applications. In: AAAI'08. (2008) 1237–1242

[8] Palma, R., Haase, P., Corcho, Ó., Gómez-Pérez, A., Ji, Q.: An Editorial Workflow Approach For Collaborative Ontology Development. In: ASWC2008. Volume 5367 of LNCS. (2008) 227–241

[9] Kalyanpur, A., Parsia, B., Cuenca-Grau, B., Sirin, E.: Axiom pinpointing: Finding (precise) justifications for arbitrary entailments in OWL-DL. Technical report (2006)

[10] Baader, F., Peñaloza, R.: Axiom pinpointing in general tableaux. In: TABLEAUX '07: Proceedings of the 16th international conference on Automated Reasoning with Analytic Tableaux and Related Methods, Berlin, Heidelberg, Springer-Verlag (2007) 11–27

[11] Reiter, R.: A theory of diagnosis from first principles. Artif. Intell. **32**(1) (1987) 57–95

[12] Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of owl dl entailments. In: ISWC/ASWC. (2007) 267–280

[13] Ji, Q., Qi, G., , Haase, P.: A relevance-based algorithm for finding justifications of DL entailments. Technical report, University of Karlsruhe (2008)

[14] Green, T.J., Karvounarakis, G., Tannen, V.: Provenance Semirings. In: PODS. (2007) 31–40

[15] Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in owl. In: ISWC '08: Proceedings of the 7th International Conference on The Semantic Web, Berlin, Heidelberg, Springer-Verlag (2008) 323–338

[16] Ma, Y., Hitzler, P., Lin, Z.: Algorithms for Paraconsistent Reasoning with OWL. In: ESWC2007, Springer (2008) 399–413

[17] Pediaditis, P., Flouris, G., Fundulaki, I., Christophides, V.: On explicit provenance management in rdf/s graphs. In: TAPP'09: First workshop on on Theory and practice of provenance, Berkeley, CA, USA, USENIX Association (2009) 1–10

[18] Bistarelli, S., Martinelli, F., Santini, F.: A Semantic Foundation for Trust Management Languages with Weights: An Application to the RT Family. In: ATC '08, Springer (2008) 481–495

[4] Such as the fact that the cartesian product of two c-semirings again is a c-semiring.

Semantically Annotated Provenance in the Life Science Grid

Bin Cao, Beth Plale, and Girish Subramanian
School of Informatics and Computing
Indiana University, Bloomington, IN, USA
{plale, bincao, subramag}@cs.indiana.edu

Paolo Missier and Carole Goble
School of Computer Science
University of Manchester
Manchester, UK
{cgoble, pmissier}@cs.man.ac.uk

Yogesh Simmhan
Microsoft Research
One Microsoft Way
Redmond, WA, USA
yoges@microsoft.com

***Abstract*— Selected semantic annotation on raw provenance data can help bridge the gap between low level provenance events (e.g., service invocations, data creation, message passing) and the high-level view that the user has of his/her investigation (e.g., data retrieval and analysis). In this initial investigation we added semantically annotated provenance to the Life Science Grid, a cyber-infrastructure framework supporting interactive data exploration and automated data analysis tools, through (i) automated data provenance collection and (ii) automated semantic enrichment of the collected provenance metadata. We use a paradigmatic life sciences use case of interactive data exploration to show that semantically annotated provenance can help users recognize the occurrence of specific patterns of investigation from an otherwise low-level sequence of elementary interaction events.**

Keywords- life sciences, provenance, semantic annotation

I. Introduction

Cyber-infrastructure frameworks for experimental science are becoming an increasingly popular way of interacting with a variety of analysis tools and other computational and data resources on the Internet. Automated provenance [6] metadata, collected during the course of a scientist’s interaction with the framework during a data exploration session, can add value to the exploration process in a number of ways: it can be used to reproduce analyses and processes, identify the causality of a series of events, broaden sharing and reuse of data products, support the long-term preservation of scientific data, attribute ownership, and determine the quality of a particular data set. Raw provenance data, however, consists mainly of observations of a user’s interaction with some visual interface, as well as of system-level observations of system events (service invocations, data creation, message passing). Unlocking the potential of such provenance metadata requires bridging the gap between these low level events, and the view that the user has of his/her investigation, which is likely to be described in terms of high-level information processing, typically consisting of data retrieval and analysis steps that lead to some scientific finding. The work described in this paper stems from the hypothesis that augmenting raw provenance metadata with selected semantic annotations helps bridge this gap, and furthermore, that for the most part such annotations can be obtained automatically, i.e., with minimal user effort.

We explore this hypothesis in the specific context of the Eli Lilly open source Life Science Grid (LSG) [3], a cyber-infrastructure framework built from Microsoft .NET 2.0 Component Application Block (CAB) and Web Services that couples automated data visualization and display (through the CAB) with invocation of data sources and analysis tools (through Web Services). The LSG is in production use inside Eli Lilly with a more fully functioning open source version anticipated.

We approach the study by defining a paradigmatic use case for interactive exploration of life sciences data, and used it to drive the design of an architectural model that integrates LSG with (i) automated data provenance collection, using the Karma provenance framework [7] developed at Indiana University, and (ii) automated semantic enrichment of the collected provenance metadata, using the Semantic-Open Grid Service Architecture (S-OGSA) semantic annotation framework [1] developed at University of Manchester. The use case is based on the data playground idea, first proposed by Gibson et al. [2], which builds on the hypothesis that recognizable patterns of a complex data exploration process may emerge from the continuous observation of direct user interaction with data exploration and analysis tools.

The remainder of the paper describes an initial investigation into the potential for the use of provenance in this scenario, specifically to help users recognize the occurrence of specific patterns of investigation from an otherwise low-level sequence of elementary interaction events. Thus, in addition to describing the use case (Section II) and presenting the technical architecture that made this investigation possible (in Section III), we reflect upon the type of provenance metadata that can be usefully and inexpensively collected, semantically annotated, and exploited to add value to scientific findings.

II. Use Case

The use case driving our work, shown in Figure 1, describes a realistic scenario of exploratory analysis on genes and gene products. The example is representative of a typical investigation method in bioinformatics, where a small set of genes that are known to be involved in a particular disease, in this case human diabetes, is used as a seed to grow a larger collection of related genes, which will provide the scope for further and possibly more expensive lab analyses. The collection grows incrementally, in a series of iterations where

This work is sponsored by funding from the Eli Lilly Corporation and by the National Science Foundation award OCI-0721674 .

a gene pool, indicated as the *working set* in Figure 1, is updated by either adding or removing some of its elements. The iteration involves a combination of access and user interaction with public databases accessible through web services on the web (we use the NCBI Entrez service for searching gene details and the AmiGO browser for Gene Ontology associations), and the use of Basic Local Alignment Search Tool (BLAST) in order to reveal homologous genes in model organisms, typically the mouse. Genes obtained from BLAST are again inspected by the user, and can be selected for addition to a "working set" maintained on behalf of a user, or discarded based on the user's judgment. The process can repeat possibly multiple times, by again BLAST-ing some of the mouse genes, leading to a larger pool of human genes that are more or less directly related to each other through homology properties.

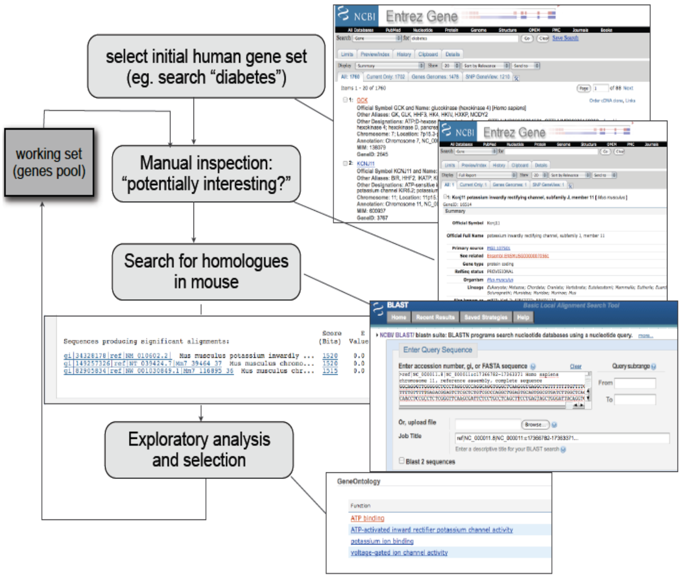


Figure 1. Illustration of use case

Throughout this process, users interact with a variety of interfaces, which LSG integrates into one single visual environment, as described in the next section. Although the iterations indicate a logical sequence of events, users are not constrained by any prescribed course of action; indeed, most of the steps can be performed in any sequence, making for a variety of different analysis paths. At the end of the process, it is important for users to understand how a certain final working set of genes was accumulated: certain genes were discovered but discarded, others were deemed worthy of further investigation, others were first added and then replaced by other, more promising elements. A combination of raw provenance metadata, user-provided and automatically added semantic annotations is used to support the explanation process. Raw provenance includes a trace of all the invocations to services through the LSG interface, as well as all UI interactions. User-provided annotations include optional descriptions that explain each update decision that affected the working set (addition, removal), and semantic annotations are obtained from various sources, for example a registry of semantically annotated Web Services, as described in the next Section.

III. System Architecture

We view the use case as an instance of a general user interaction model, where events and data products are recorded and associated to a *user session*, and various annotations are associated to both the events, for example a service invocation, and the data products, e.g., the result message from the service (we define a session as being delimited by user login and logout actions). Figure 2 gives an overview of the architecture used to support this interaction model. Individual users can configure their own personal LSG desktop environment by selectively enabling some of the available plugins, which control the interaction with specific services. In addition, LSG plugins interact amongst each other using a publish/subscribe model through an LSG event bus, providing users with an integrated, multi-panel interface. Thus, suppose for example that an NCBI Entrez[1] plugin accepts user gene lookup requests, and sends the corresponding gene descriptions onto the bus, while an AmiGO[2] plugin that is able to resolve Gene Ontology (GO) terms subscribes to those descriptions. When the user submits a request, the response triggers the AmiGO plugin, which responds by updating its own interface with the GO descriptions of the gene, while details of the latter are being displayed on the NCBI Entrez plugin interface.

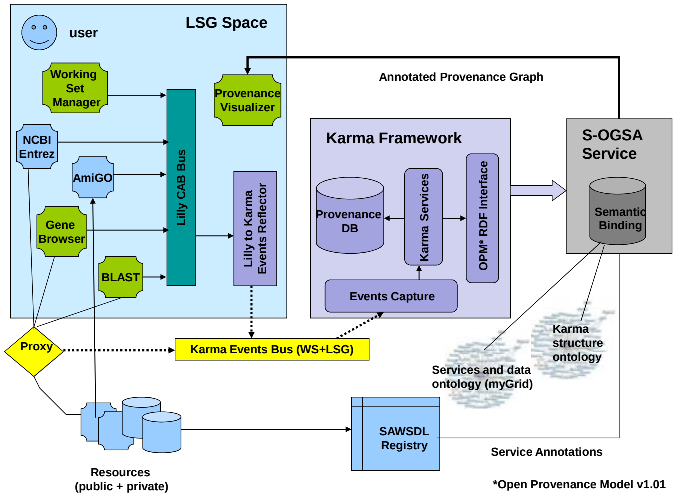


Figure 2. Integrated provenance management architecture

We have exploited this event model to generate elementary provenance events through the Karma component, which is configured to snoop on the LSG event bus, in addition to having its own instrumentation in the Web Service proxies that mediate LSG plugin interactions with the services. Karma structures these provenance events according to the Open Provenance Model (OPM) [5], a

[1] NCBI Entrez is a search engine for biomedical databases and available at http://www.ncbi.nlm.nih.gov/sites/entrez?db=gen.

[2] AmiGO is a web service to access Gene Ontology associations for genes and available at http://www.geneontology.org/.

community standard for describing causal graphs through a set of pre-defined types of nodes and their relationship.

Throughout a user session, fragments of OPM graphs representing single interactions are forwarded to the S-OGSA component, which performs two functions: firstly, it analyses the OPM graph and adds semantic annotations to some of its nodes, whenever possible and by using a variety of annotations sources. For instance, if a node represents a Web service invocation, and a semantically annotated description of the service is available, then S-OGSA augments the OPM graph by associating the annotations to that node (a more detailed description of the annotation architecture is described in Section C). Secondly, S-OGSA stores the pair <user session, OPM graph> in its own database, which the Provenance Visualizer can query to present semantic provenance to the user. By having the Provenance Visualizer implemented as a new LSG plugin itself, the combination of these components provides users with a seamlessly integrated feedback loop, by incrementally displaying the effect of their actions as a rendering of provenance metadata.

Next, we elaborate on the three main components of the integrated architecture.

A. LSG

Two main features make LSG an appealing platform for our experimentation: its openness, which made it possible to create provenance events simply by adding a subscriber to the LSG event bus, as described earlier; and its extensibility, which we have used to implement new plug-ins especially for our use case. Specifically, we have used two of the available plugins for the open source version of LSG, namely for searching the NCBI Entrez database and for resolving GO terms; and have implemented three new plugins:

- A BLAST plugin that interacts with one of the several publicly available BLAST services[3];
- A Working Set Manager, to manage the dynamic collection of data products, in this case genes, that represent the main outcome of the users' investigation;
- A Provenance Visualizer, which can display parts of the provenance graph to the users (see Figure 3).

The "LSG Space" in Figure 1 shows the relationship amongst these plugins. While we exploit the event model to automate much of the data flow across the plugins, we also identify points in the process where we felt that explicit, knowledge-intensive user input was desirable. Thus, for example, while it is possible to extract a DNA sequence in FASTA format from an NCBI gene description record, to be used as input to BLAST, expert users prefer to have control over the portion of the sequence, for example to include or exclude the gene promoter regions on either side of the sequence. This mix of automated data flow and explicit user input offers the additional opportunity for users to add their own notes as explanations of their actions, for example to comment on the choice of a wider region around a gene. This is particularly clear in the design of the Working Set Manager, which automatically accepts new elements, i.e., genes from BLAST, through the event bus, but also offer users the opportunity to examine (accept, reject, annotate) each of them individually.

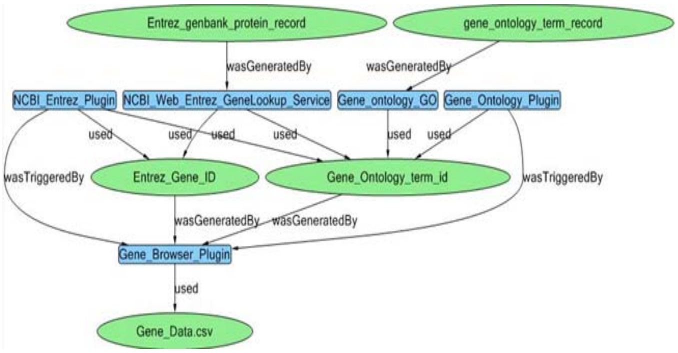
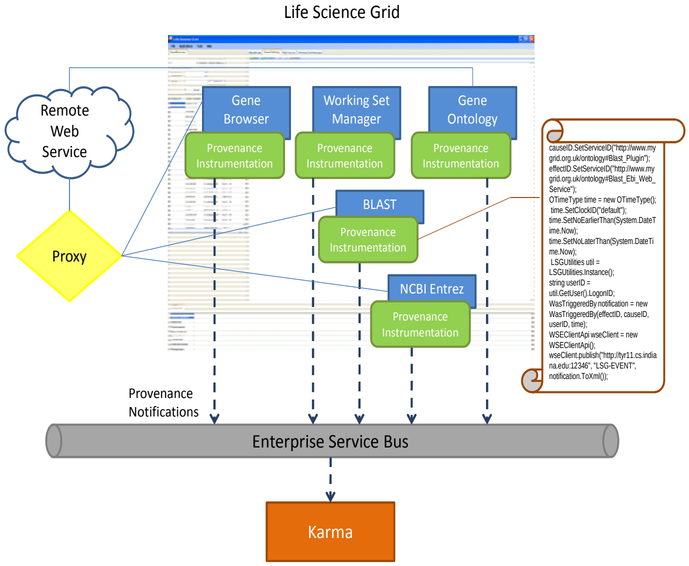


Figure 3. OPM graph fragment

B. Karma and OPM

The main functions of the Karma component in this setting are to capture raw provenance events, and to format them according to the Open Provenance Model specification. As Karma is a general provenance collection and management tool, it implements a generic provenance model and set of instrumentation tools that are independent of the application system. Instrumentation of the LSG required the use of several forms of instrumentation. For the web hosted data services and sources, we implemented proxy web services that utilize instrumentation handlers in Axis2 to collect provenance. Provenance of the CAB activity is captured by a listener on the CAB events bus. The listener forwards provenance relevant events to Karma. The high level view of capture is shown in Figure 4.



```
causeID.SetServiceID("http://www.my
grid.org.uk/ontology#Blast_Plugin");
effectID.SetServiceID("http://www.my
grid.org.uk/ontology#Blast_Ebi_Web_
Service");
OTimeType time = new OTimeType();
 time.SetClockID("default");
time.SetNoEarlierThan(System.DateT
ime.Now);
time.SetNoLaterThan(System.DateTi
me.Now);
 LSGUtilities util =
LSGUtilities.Instance();
string userID =
util.GetUser().LogonID;
WasTriggeredBy notification = new
WasTriggeredBy(effectID, causeID,
userID, time);
WSEClientApi wseClient = new
WSEClientApi();
wseClient.publish("http://tyr11.cs.india
na.edu:12346", "LSG-EVENT",
notification.ToXml());
```

Figure 4. Provenance instrumentation in LSG plugins

[3] http://www.ebi.ac.uk/Tools/blast/.

We distinguish between black box plugins, for which it may be possible to observe data exchange events that occur through the LSG bus, and white box plugins, where in addition, user interaction events that occur through a service interface can also be detected. In practice, black boxes are those where native web pages are displayed, so that access to the user click-throughs on the page is limited and can only be achieved by intercepting the HTTP requests using a proxy, for instance, but some of the context in which the request is made is missing. In white box components, on the other hand, the UI is part of the plugin design, and as a consequence we can capture user actions with full detail.

According to the black-box, white-box distinction, the Working Set manager is a white box, because all user events can be observed along with optional user annotations, while the native LSG plugins, the NCBI Entrez and AmiGO plugins, are black boxes. As for the BLAST plugin, making it a white box required the extra effort of encoding a bespoke web-based interface to interact with the service, in order to capture all of the important user interactions. Thus, Karma captures the user selection, de-selection, and annotations of genes in Working Set Manager, and the set of genes that transit on the LSG bus, including BLAST reports.

Karma maps provenance events to fragments of OPM graphs. In its simplest form, an OPM graph consists of two types of nodes, which represent Artifacts and Processes. These are shown as ovals and rectangles, respectively, in Figure 3. Nodes are connected using directed labeled arcs, which express properties that hold between two nodes. The set of all legal properties is fully described in [5], however the following three types of properties were found to be sufficient to express our provenance events:

- process P *used* artifact A, for example, NCBI_Entrez_Plugin used Entrez_Gene_ID,
- artifact A *wasGeneratedBy* process P, for example, Entrez_Gene_ID wasGeneratedBy Gene_Browser_Plugin, and
- process P1 *wasTriggeredBy* process P2, for example, NCBI_Entrez_Plugin wasTriggeredBy Gene_Browser_Plugin.

The first two properties express ordinary producer/consumer relationships, while the latter is useful in expressing the indirect interaction between two plugins that publish and subscribe to a data element, respectively. We also use the same property to express the fact that a plugin controls an underlying service, i.e., in the typical situation where a service invocation is triggered by a plugin.

Provenance events are published as notifications to the Web services-based message broker, *WS-Messenger* [8], where Karma is a subscriber. When a provenance notification arrives, the corresponding provenance handler picks it up, retrieves the raw provenance data, and stores these data into its own provenance database, a MySQL relational database. These raw provenance data can be used to answer general provenance questions as well as determine the artifact dependency and the process dependency during a user session. Meanwhile, these data is sent to S-OGSA for semantic annotation. Since OPM is an abstract process model with multiple concrete serialization formats for portability across applications, as indicated in Figure 2, we have used the RDF[4] serialization to transfer OPM graph fragments from Karma to the S-OGSA component.

C. Modular Semantic Annotations using S-OGSA

S-OGSA [1, 4] manages the persistent and stateful associations between Grid resources, i.e., data or services, and their annotations (or any form of related metadata), expressed primarily as RDF graphs. Such associations, known as semantic bindings, can be queried with SPARQL. S-OGSA mapping to this project has user sessions playing the role of resources, with OPM provenance graphs produced by Karma as their associated metadata. S-OGSA additionally augments the input graphs with semantic annotations. Here we focus on the latter part of the S-OGSA architecture[5].

The annotation architecture is based on the principle that annotations to nodes in the RDF OPM graph will depend on (i) the specific types of Artifact and Process nodes, and (ii) the availability of metadata sources that can be used to derive interesting metadata for those node types. To account for this flexibility, we designed a modular architecture based on the interceptor pattern, consisting of an extensible chain of annotators, each specialized to annotate specific types of nodes. Each annotator receives an input RDF graph, produces an augmented version of the same graph with annotations added to it, and forwards it to the next annotator down the chain. As no parts of the input graph are ever removed, annotators can be added incrementally to S-OGSA, in a monotonic fashion. The pattern is illustrated in Figure 5. As a proof of concept, we have implemented a chain consisting of two annotators, one for Process node of type Web Services, and one for Artifact nodes of type Blast report. We now describe how each of these two annotators uses a different metadata source to produce its annotations.

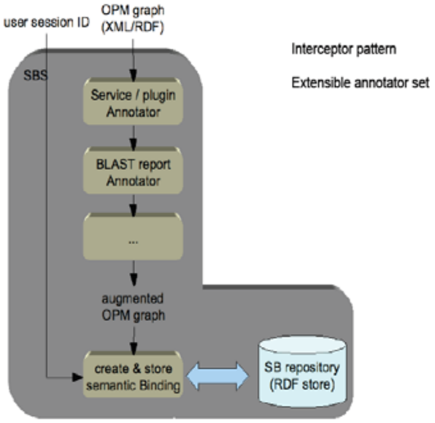


Figure 5. S-OGSA interceptors for incremental semantic annotations of OPM graphs

[4] http://www.w3.org/RDF/

[5] Technically, S-OGSA relies on the Anzo RDF API for storing its annotation graphs, and its functionality is exposed as a RESTful Web Service.

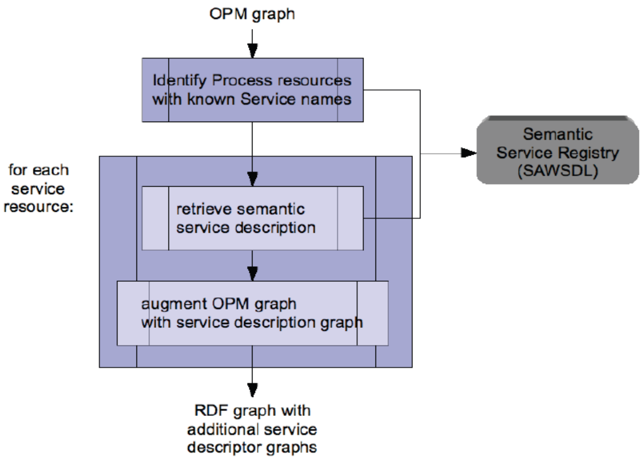

OPM graph
Identify Process resources with known Service names
Semantic Service Registry (SAWSDL)
for each service resource:
retrieve semantic service description
augment OPM graph with service description graph
RDF graph with additional service descriptor graphs

Figure 6. Service annotation

The *Service Annotator* relies on Process nodes that represent Web Services, to be labeled with a service name, for instance NCBI_Entrez, that can be matched against a local and bespoke registry of Web service descriptions. In this registry, service descriptions are semantically annotated using SAWSDL[6] (in a future version, the Biocatalogue service registry[7] will be used for this purpose). If a match is found, the corresponding SAWSDL annotations (i.e., the sawsdl:modelReference attribute values), are added to the RDF graph (see Figure 6). Since these annotations are references to concepts in some ontology (expressed as URIs), the standard rdf:type property is used to associate the annotation to the Process node. An example of SAWSDL-annotated service description for NCBI Entrez is shown below.

```
<wsdl:interface name="eFetchGeneService"
sawsdl:modelReference="http://www.mygrid.org.uk/ontology#E
ntrez_GenBank_protein">
  <wsdl:operation name="run_eFetch"
            pattern="http://www.w3.org/ns/wsdl/in-out"
            sawsdl:modelReference="http://www.owl-
ontologies.com/unnamed.owl#run_eFetch_dbGene">
    <wsdl:input element="nsef:eFetchRequest" />
    <wsdl:output element="nsef:eFetchResult" />
  </wsdl:operation>
</wsdl:interface>
```

Note that this entry annotates a generic NCBI eFetch service with concepts from the myGrid ontology[8], which qualify it as a gene lookup service.

The *Blast Report Annotator* is an example of data annotator that performs complex lookups in multiple public databases in order to semantically annotate a data entry of a specific type in the OPM graph. Its general structure is shown in Figure 7. The fragment above the line is part of LSG processing. The BLAST report is accessible to the annotator through a unique ID that is part of a RDF resource (a URI), and that is dereferenced against a persistent local data store.

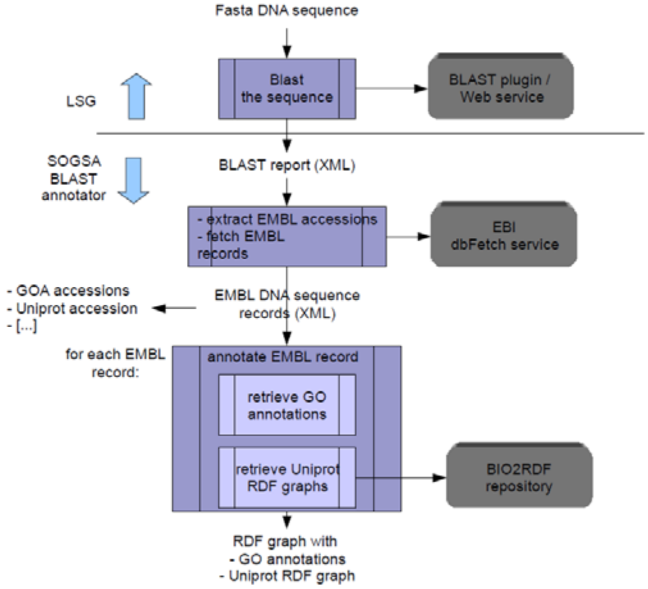


Figure 7. BLAST report annotation

An EBI BLAST report consists of a ranked list of matched DNA sequences, which may be parts of genes or proteins. Thus, some of these entries may optionally contain a variety of references to external databases; in our implementation we have focused on (i) Uniprot accession numbers, which appear whenever the matched DNA sequence is related to a protein, and (ii) GO annotations, i.e., references to entries in the Gene Ontology.

As the report is in a standard XML format, the annotator begins by extracting the EMBL accession numbers, which are then used to query the EMBL database, through the WSDbFetch Web Service[9] . This yields one XML document for each hit in the BLAST report, indicated as “EMBL DNA sequence records” in the figure. Then, for each of these records the annotator extracts both the set of GOA annotations (in the example: {A6NMX8, Q09428}), and the set of Uniprot accession numbers, if any (in the example: {Q09428}). The former is used to query the Gene Ontology to retrieve the associated descriptions, while we use the latter to query Bio2Rdf (using the dynamic URL http://bio2rdf.org/uniprot:Q09428). This is particularly interesting, as the Bio2Rdf project (http://bio2rdf.org/) exposes the content of entire Bioinfomatics databases, including Uniprot, as RDF graphs. Thus, associating the RDF entry for a specific protein, when available, is a very natural operation in the context of Blast report annotation.

Figure 8 shows a fragment of annotated RDF graph for a BLAST report. The content of each report resource is a bag of entries, i.e., a bag of b-node resources, each corresponding to one sequence hit in the report. All b-nodes have type EMBLRecord (a class in the myGrid ontology) and have an

[6] http://www.w3.org/2002/ws/sawsdl/

[7] http://www.biocatalogue.org/

[8] http://www.mygrid.org.uk/tools/service-management/mygrid-ontology/

[9] http://www.ebi.ac.uk/Tools/webservices/services/dbfetch

associated (a) EMBL accession number, (b) GOA accessions, if any, and (c) entire named graphs that resolve to the Uniprot records associated to the sequence, if available. Note that this type of graph could not be obtained automatically from an RDF-based provenance capture engine such as that of the Taverna workflow system[10].

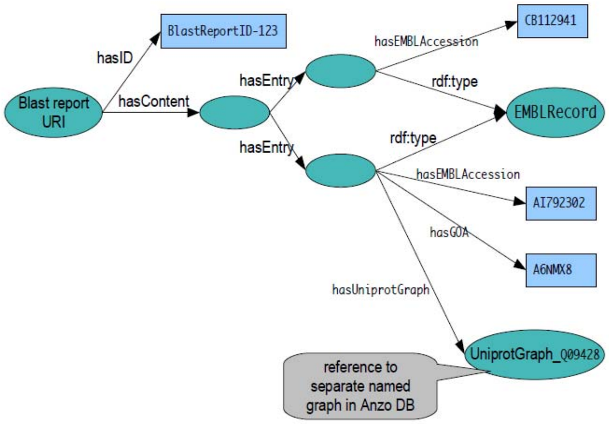


Figure 8. Annotated RDF graph for a BLAST report

IV. Lessons learnt and future work

This paper details the architectural complexity of collecting provenance data from LSG, augmenting it, semantically annotating it, and returning it to the user. The paradigmatic use case shows that semantically annotated provenance can help users recognize the occurrence of specific patterns of investigation from an otherwise low-level sequence of elementary provenance events.

There are several noteworthy outcomes that emerged in implementing the use case we describe in this paper. The Life Science Grid, as mentioned earlier, is built using the .NET Component Application Block “portal” with call-outs to web services. The CAB is multicast in that plugins drop events on a bus that are picked up and acted upon by all other plugins creating a stateless, shared-all medium. But once the architecture is extended to include web services that gather information from public databases and services on the web, even for our straightforward use case, the need for state sharing arises. State can be viewed as an artifact (in OPM terms), and one that is important to the provenance record particularly where long term preservation of data is the goal. We will more fully examine this impact in future work.

Substantial investigation remains in the visual presentation of provenance information. Lilly sees the historical, lineage nature of provenance as having significant potential to contribute to the drug discovery process. We are exploring visualization of higher levels of abstraction of provenance to more closely match user’s investigative process. It raises interesting questions on the implication to instrumentation as well. When can the underlying low level event collection be replaced with higher levels of abstraction and what form do these higher levels of instrumentation take? Moreover, a user study can assess the value that provenance collection brings to the daily research investigative process of the users. We are working with Eli Lilly to set this up.

Finally, provenance can be captured and semantically annotated for other grid systems such as caGrid [9], the service-based infrastructure that supports the cancer Biomedical Informatics (caBIG [11]). Unlike user-driven (streaming) workflows in LSG, caGrid users need to pre-define a workflow using a workflow orchestration tool before execution [10]. Since Karma can capture provenance from different service-based sources, through proper instrumentation in the workflow orchestration tool, raw provenance data can be captured and then semantically annotated by S-OGSA.

Acknowledgements

Our thanks to Katy Wolstencroft and Dr. Andy Gibson at University of Manchester for their help in the formulation of the usage scenario. Susie Stephens and Richard Bishop of Eli Lilly have been instrumental drivers in the project.

References

[1] O. Corcho, P. Alper, I. Kotsiopoulos, P. Missier, S. Bechhofer, C. Goble, “An Overview of S-OGSA: A Reference Semantic Grid Architecture,” Journal Web Semantic, vol. 4, no. 2, pp. 102–115, 2006.

[2] A. Gibson, M. Gamble, K. Wolstencroft, T. Oinn, C. Goble, K. Belajjame, P. Missier, “The Data Playground: An Intuitive Workflow Specification Environment,” Future Generation Computer Systems, vol. 25 no.4, pp. 453-459, April 2009.

[3] Life Sciences Grid, http://sourceforge.net/projects/lsg/

[4] P. Missier and P. Alper and O. Corcho and I. Dunlop and C. Goble, “Requirements and Services for Metadata Management, ” Journal IEEE internet Computing, Special issue on Semantic-Based Knowledge Management, Sept. / Oct., 2007.

[5] L. Moreau, B. Plale, S. Miles, C. Goble, P. Missier, R. Barga, Y. Simmhan, J. Futrelle, R. E. McGrath, J. Myers, P. Paulson, S. Bowers, B. Ludaescher, N. Kwasnikowska, Jan Van den Bussche, T. Ellkvist, J. Freire, P. Groth, “The Open Provenance Model (v1.01), ” July 17, 2008. http://eprints.ecs.soton.ac.uk/16148/1/opm-v1.01.pdf

[6] Y. Simmhan, B. Plale, and D. Gannon, “A Survey of Data Provenance in e-Science,” ACM SIGMOD Record, vol. 34, no. 3, pp. 31-36, 2005.

[7] Y. Simmhan, B. Plale, and D. Gannon, “Karma2: Provenance Management for Data-Driven Workflows,” International Journal of Web Services Research, vol. 5, no. 2, pp. 1-22, 2008.

[8] Y. Huang, A. Slominski, C. Herath, D. Gannon, “WS-Messenger: A Web Services-Based Messaging System for Service-Oriented Grid Computing,” Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID), pp. 166 – 173, 2006.

[9] J. Saltz, S. Oster, S. Hastings, S. Langella, W. Sanchez, M. Kher, P. Covitz, T. Kurc, K. Shanbhag, “caGrid: design and implementation of the core architecture of the cancer biomedical informatics grid”, Bioinformatics, vol. 22, no. 15, pp. 1910-1916, June 2006.

[10] W. Tan, P. Missier, R. Madduri, I. Foster, “Building Scientific Workflow with Taverna and BPEL: A Comparative Study in caGrid,” Service-Oriented Computing --- ICSOC 2008 Workshops, pp. 118-129, 2009.

[10] The Taverna provenance component is now being re-defined using a non-RDF data model.

[11] https://cabig.nci.nih.gov/

Towards Usable and Interoperable Workflow Provenance: Empirical Case Studies Using PML

James R. Michaelis*, Li Ding*, Zhenning Shangguan*, Stephan Zednik*, Rui Huang*,
Paulo Pinheiro da Silva†, Nicholas Del Rio† and Deborah L. McGuinness*
*Tetherless World Constellation, Rensselaer Polytechnic Institute, Troy, NY 12180
†Computer Science Department, University of Texas, El Paso, El Paso, TX, 79968

***Abstract*—In this paper, we describe how a semantic web-based provenance Interlingua called the Proof Markup Language (PML) has been used to encode workflow provenance in a variety of diverse application areas. We highlight some usability and interoperability challenges that arose in the application areas and show how PML was used in the solutions.**

I. Introduction

In scientific research, workflow systems are used to assemble steps (each corresponding to certain tasks) for processing scientific data. Provenance is a well-known and important component in these systems [1]. In particular, access to a workflow system's *data flow* has proven crucial for users to understand, validate, and reproduce its workflows [2], [3].

As workflow systems become more complex and distributed in nature, a number of provenance management challenges are known to emerge [1]. Within the scope of this paper, we focus on two particular challenges: *usability* and *interoperability*. To address the usability challenge, provenance information must be both sufficiently intuitive and expressive for end users to understand. Likewise, for the interoperability challenge, provenance representations must be capable of linking to, integrating, and reusing each other's content for unexpected purposes.

In this paper, we investigate how both challenges can be addressed through a domain independent provenance interlingua called the Proof Markup Language (PML) [4]. PML facilitates generation and sharing of provenance metadata for data derivation within and across intelligent systems, and acts as an enabler of trust by supporting explanations of information sources, assumptions, and learned information. As a critical part of the Inference Web (IW) [5] project, PML has been used in many domains [6], including: information extraction [7], logical reasoning [8], workflow processing [9], semantic eScience [10], and machine learning [11], [12]. Three workflow-based case studies we explore are as follows:

- **Case Study 1, Semantic Provenance Capture in Data Ingest Systems (SPCDIS)**: This project integrates provenance representations into scientific workflows in the fields of solar, solar-terrestrial, and space physics. These workflows include numerous scientific data products annotated by complex domain-specific ontologies. Here, provenance is needed to facilitate querying based on domain-knowledge (for instance, to list which scientific instruments were used to derive a certain type of data product).
- **Case Study 2, Generalized Integrated Learning Architecture (GILA)**: GILA is a multi-agent machine learning platform, which generates a workflow log about how a problem was resolved collaboratively by an ensemble of learning agents. Provenance in this system is implicitly encoded through domain-specific structuring, and needs to be normalized to allow basic querying.
- **Case Study 3, The Third Provenance Challenge (PC3)**: Unlike the former two case studies, this focuses on a workshop aimed at developing interoperable provenance. Here, multiple participants investigated a workflow from an astrometry/photometry-based system. Using individual approaches, everyone had to monitor this workflows execution and export the resulting provenance data for import, integration and querying by the other teams.

The remaining sections are organized as follows. Section 2 briefly reviews PML and shows its applicability in workflow provenance representations. Sections 3 through 5 detail the three case studies on SPCDIS, GILA and PC3 respectively. For each of these, we highlight: (i) examples of usability and interoperability challenges, (ii) how PML was used to address these challenges, and (iii) lessons learned from these efforts. Section 6 discusses related work with PML, and section 7 provides concluding remarks.

II. PML and Workflow Provenance

PML is a Semantic Web based provenance representation, defined through three core OWL ontology modules: the **Provenance module**(namespace: pmlp), which supports annotation of general provenance related entities, (such as agents, data products, and information sources); the **Justification module** (namespace: pmlj), which supports annotating derivation relations (pmlj:InferenceStep) among data products, represented by justification-based concepts (pmlj:NodeSet); and the **Trust module**(namespace: pmlt), which supports annotating complex trust relations on provenance and justification concepts. The modular design of PML facilitates future reuse and extension of these core ontologies.

In tracking workflow provenance, PML can be used to capture data flow by recording: (i) the sequence of operations taken to derive data products, and (ii) descriptions about these operations. Figure 1 depicts a simple workflow covering basic workflow concepts (above) and shows how these concepts are represented by PML (below). The workflow includes a sequence of processes $P_0 \ldots P_n$. Each process P_i(denoted by a rectangle and mapped to pmlj:InferenceStep), is defined as an execution of an operation O_i (denoted by a diamond and mapped to pmlj:InferenceRule) by an agent A_i (denoted by a person figure and mapped to pmlp:Person), and takes as input a data product D_i (denoted by an oval, and mapped to pmlp:Information) and derives another data product D_{i+1} as the output.

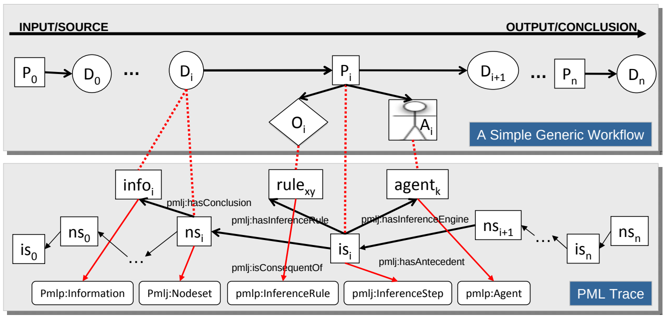


Fig. 1. Representing Execution of Workflow

There are some immediate benefits in representing workflow provenance using PML. Using Semantic Web representation strategies, PML-encoded data can be linked to domain ontologies supporting improved usability, and may be extended by (or mapped to) other provenance models for better interoperability. For this, OWL is used to facilitate linking of domain concepts to PML through constructs such as subclass relations. Likewise, PML is used for direct representation of provenance concepts (like those defined in its provenance module). This combined PML/domain data can in turn be processed by Semantic Web based tools. Examples include: OWL reasoners, SPARQL engines, and Inference Web based tools (such as Probe-It! [13] for PML visualization, and the OWL Instance Validator [14] for checking validity of PML data).

III. Case Study: SPCDIS

Semantic Provenance Capture in Data Ingest Systems (SPCDIS) [15] is a research project aimed at integrating provenance at data generation/ingest time into a data portal managed by the Mauna Loa Solar Observatory (MLSO). In SPCDIS, provenance annotations are being used to incorporate trust and transparency into generated data products. Figure 2 illustrates the Coronal Helium I Imaging Photometer (CHIP) pipeline - an example collaborative scientific workflow for generating scientific images in a distributed environment. The magnified portion of the workflow shows a fragment of the data flow: the *Instrument Capture* process uses certain configuration data under the *Instrument Configuration* category as input and generates some output image-based data products under the *Image File with Header* category.

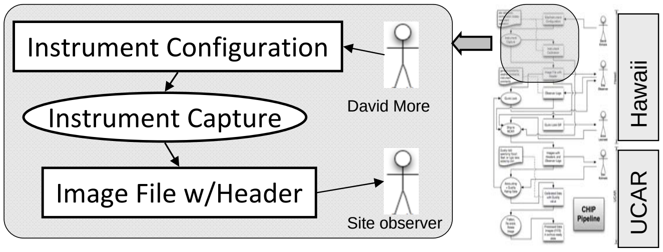


Fig. 2. A Example Fragment of a distributed Workflow from SPCDIS

A. Use of PML

Provenance encoding in SPCDIS faces both usability and interoperability challenges, given the high volumes of data processed by heterogeneous components in diverse locations. We see PML as capable of both of these challenges in the context of this system.

In our introduction, we emphasize that provenance usability depends on its intuitiveness. Here, we consider the idea of intuitiveness from both the perspective of a domain expert and computer scientist – two types of people likely to be collaborating to generate a computer based provenance representation. For instance, consider the query ”Which photometers (or more generally, optical instruments) were used to generate the DataImage at a specific URL?” This is rich in domain knowledge, but may not make sense to a non-expert. Likewise, consider a modified version of the query: ”Which pmlj:InferenceStep instances $X_0 \ldots X_n$ were used to generate the pmlp:Information instance Y?” This would expose more of the representational details than a domain expert needs to see, but captures an abstraction usable by a computer scientist or computer program. By combining domain-dependent concepts with PML, we facilitate its use by individuals with varying degrees of expertise in a target domain.

Likewise, the interoperability challenges faced by SPCDIS stem from its recording of provenance from a series of distinct workflow components with varying terminologies. The issues underlying integrating this heterogeneous provenance are resolved through terminology linking through PML-based concepts.

To carry out the strategy above, we extended the justification and provenance modules of PML through domain-specific concepts from the Virtual Solar Terrestrial Observatory (VSTO) ontology (prefix: vsto) [1]. Figure 3 shows an example of PML provenance data generated for SPCDIS. It conveys the following information: a CSRImage with the name ”MLSO CHIP CSR Image” was generated by the execution of a software agent called ”CSRImageCapture” via the ”CHIP-He-I Continuum Capture” operation (which is a specialized VSTO instrument operation mode) using a sensor (i.e. Photometer) called CHIP. Four different ontologies (namespaces: pmlp, pmlj, vsto, and spcdis) are integrated together in Figure 3: PML contributes domain independent concepts, VSTO contributes a domain ontology and the SPCDIS ontology carries

[1] VSTO ontology: http://vsto.org/forward.htm?forward=ontology

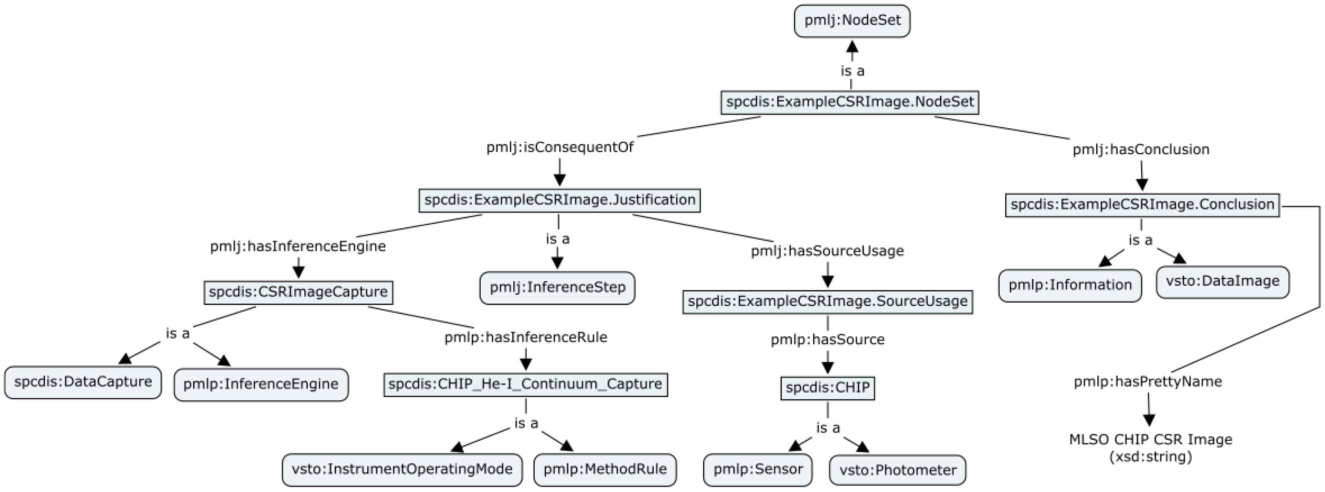


Fig. 3. Generated PML Provenance Metadata. The rounded rectangles denote concepts, and the rectangles represent instances. The edge labels denote properties, with "a" meaning instantiation of concepts (rdf:type), and "are" representing sub-class relations (rdfs:subClassOf).

out the integration of concepts that connect PML and VSTO. For example, the class spcdis:DataImage is a subclass of both the pmlp:Information concept in PML and the vsto:DataImage concept.

Here, PML's integration with domain-specific ontologies is necessary to answer the question from the beginning of this section (specifically, to determine which pmlp:Information instances are also of type vsto:DataImage). Below is a SPARQL query for accomplishing this, which leverages our provenance representation strategy:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX pmlj: <http://inference-web.org/2.0/pml-justification.owl#>
PREFIX pmlp: <http://inference-web.org/2.0/pml-provenance.owl#>
PREFIX vsto: <http://dataportal.ucar.edu/schemas/vsto.owl#>
PREFIX spcdis: <http://example.com/spcdis.owl#>
PREFIX image: <http://iw.vsto.org/data/mlso/chip/raw/>
SELECT ?photometer
WHERE  { ?image pmlp:hasURL "image:2008_09_03_00_04_07.csr"^^xsd:anyURI .
         ?nodeset pmlj:hasConclusion ?image ;
                  pmlj:isConsequentOf ?step .
         ?step pmlj:hasSourceUsage ?usage .
         ?usage pmlp:hasSource ?source .
         ?source a ?spcdis:Photometer . }
```

B. *Lessons learned*

In this case study, we showed how PML could be used, in conjunction with domain-specific ontologies, to address provenance usability and interoperability challenges. This effort hinged on manual ontology mapping, which did require domain expertise. However, most of the mappings were done simply by linking domain-specific concepts with generic provenance terminology from PML (e.g. pmlp:Information, pmlp:Source, pmlp:InferenceEngine and pmlp:InferenceRule). This helped establish value restrictions on provenance concepts, so we could support appropriate qualified searches (such as the one above that needs only particular kinds of images).

IV. Case Study: GILA

The Generalized Integrated Learning Architecture (GILA) [9] is a multi-agent platform for learning how to solve domain-specific problems. Initially, GILA's agents learn from domain expert generated workflow traces, each consisting of an encoded sample problem with an accompanying solution sequence. Following this, these agents collaborate to solve similar problems. During both steps, all agent knowledge and work (solutions) are recorded to a communal blackboard as a way to facilitate inter-agent communication. The GILA system was tested using the Airspace Control Order (ACO) Scheduling Scenario, which consisted of the following parts: (i) a problem state is submitted to a central scheduler (usually a domain expert) - defined as a list of requests for temporal-spatial airspace allocation, each encoded as an Airspace Control Means (ACM), (ii) the scheduler selects and updates the ACMs one at a time to resolve their temporal-spatial conflicts (generating a new problem state each time to reflect the remainder of the problem). This deconfliction process requires domain expertise in, for instance, prioritizing which ACMs should be changed initially. In this scenario, GILA was compared against novice human participants in playing the role of the scheduler.

A. *Use of PML*

GILA's logs, derived from agent submitted information on the communal blackboard, were used to evaluate its performance. These were structured as RDF graphs, with their semantics preserved by a handful of domain ontologies encoded in OWL. These domain ontologies implicitly covered both the provenance annotations for domain entities and derivation relations among data products. However, the derivation relations were represented using complex domain structuring, such that it was hard to see a clear picture of GILA's data flow. To address this, we had to overcome a usability challenge on provenance normalization - that is, to normalize derivation relations to facilitate intuitive querying. One such query, which could not easily be answered by the original log, was to list all the problem states $P_1 \ldots P_n$ generated before a given ACM deconfliction S was generated.

This challenge was approached in a two-step process [16]. Starting with a set of domain ontologies and a GILA log instance, an *analysis phase* would first be conducted. This would return the following: (i) from the domain ontologies, a list of OWL classes and properties corresponding to PML classes (e.g., pmlp:Agent, pmlp:InferenceRule), and relationships (e.g., pmlj:isConsequentOf), and (ii) from the log instance, a set of RDF based structural relations, not captured by the domain ontologies, which correspond to PML relationships. Following the analysis phase, a *mapping phase* would be conducted, in which PML-based information would be inserted into the log instance.

Figure 4 illustrates how provenance normalization could be applied for the example above. First, in the analysis phase, two domain ontologies - gilcore and gilaco - are inspected to identify the hidden provenance information from the original GILA log. The following domain knowledge is uncovered: (i) ACM deconflictions are represented as instances of the class gilcore:Solution, (ii) each solution S_i has a corresponding problem state P_i, defined as an instance of gilcore:Problem, (iii) the property gilcore:hasProblem is used to link S_i to P_i (in the figure, property names are omitted due to limited space), (iii) the property gilaco:hasSolutionListResolveConflict links P_i to a recursively declared list of instances of gilcore:SolutionListResolveConflict where each list item $list_{i,i-1}$ represents an earlier solution S_{i-1}. This list helps define the context of a current problem, but doesnt explicitly define the solution S_{i-1} used to transform $P_i - 1$ to P_i knowledge required for uncovering the solution generation data flow.

In the mapping phase, PML data is built in the following steps: (i) an OWL ontology is defined for linking the domain ontologies to PML, which asserts gilcore:Solution and gilcore:Problem as subclasses of the OWL class pmlp:Information, (ii) through OWL inference, instances of these two classes will be inferred to be of type pmlp:Information, (iii) through JENA [2] (a Java-based RDF data processing API) and SPARQL, PML data is generated from a GILA log instance which normalizes links between problems and solutions (e.g. from P_i to S_{i-1}).

B. Lessons learned

In this case study, we demonstrated how to use provenance normalization to address usability challenges by generating PML data based on both GILA's domain ontologies and logs. Although non-trivial domain expertise was needed to (i) identify the provenance components in the domain ontologies and log data, and (ii) establish mappings from the domain ontologies to PML, such work usually ended up being a one-time job. Subsequent generation of PML data in the mapping phase could then be automated using off-the-shelf tools and easily be performed.

One of our future goals will involve determining ways in which the analysis phase could be (at least partially) automated. However, as part of this, a set of constraints on how log and domain ontology information can be structured will be required.

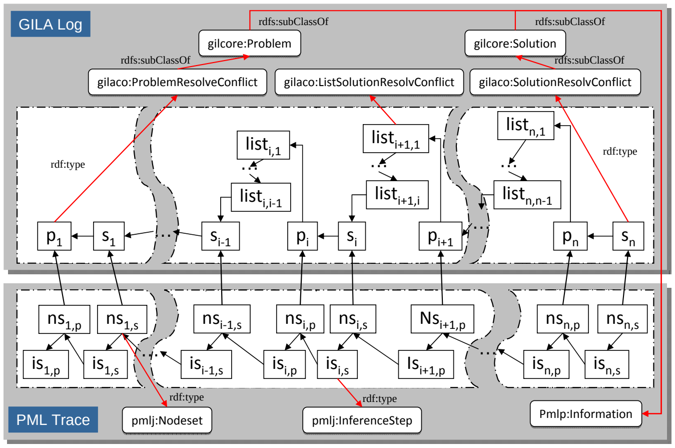


Fig. 4. PML encoding based on GILA log

In general, many complex systems like GILA can record provenance in their workflow logs, but usually use domain-specific terminology and structure. A fair amount of work must be done to individually tailor explanation interfaces for these systems. By normalizing workflow provenance into PML, we can more easily apply general-purpose explainers [13], [5] to various workflows from different domains.

V. Case Study: PC3

In the Third Provenance Challenge (PC3) [3], 15 research groups were asked to use their own approaches to: (i) generate provenance metadata for exposing the execution of a given workflow, (ii) use this metadata to answer a set of provenance-based queries, (iii) export this metadata, and (iv) import metadata from other teams and answer the queries from (ii) using it. A common interchange format, the Open Provenance Model (OPM) [17], was chosen for teams to import and export their provenance metadata. The workflow investigated in this effort was derived from the Pan-STARRS project [4], which processes data on 99% of visible stars in the northern hemisphere, and manages a pipeline for loading domain data in CSV files into a relational database and validating it. Here, control flow was viewed as the sequence of processes executed within the workflow, subject to conditional branching (e.g., if a process fails to complete correctly, halt the workflow otherwise, continue normally). Likewise, dataflow was defined as a sequence of steps by which data would be generated and used by processes.

A. Use of PML

During PC3, two important requirements emerged for us (and many other groups) in encoding provenance capable of answering the queries.

[2] http://jena.sourceforge.net/

[3] http://twiki.ipaw.info/bin/view/Challenge/ThirdProvenanceChallenge

[4] http://pan-starrs.ifa.hawaii.edu/

First, domain-specific provenance (as with GILA and SPCDIS) was needed to answer many of the queries. One such query, known in PC3 as Core Query 1, reads: ”For a given *detection*, which CSV file(s) *contributed* to it?” Here, two domain-specific concepts are referenced: (i) a detection, which is a type of data handled in the workflow, and (ii) a contribution, which references a data loading sequence carried out by the workflow.

Second, many queries required the control flow of the workflow to be tracked in parallel with the dataflow. We viewed this challenge as consisting of two parts: (i) explicitly distinguishing execution of operations from the operations themselves, and (ii) representing the dependencies among executions of operations. An example of a control-flow based query, Optional Query 2, reads: ”Which pairs of procedures in the workflow could be swapped and the same result still be obtained (given the particular data input)?”

The generation of provenance meeting both these requirements that both we and other groups could answer queries over constituted an interoperability challenge in PC3. To address this, we used Semantic Web technologies to manage and query provenance - both recorded from the system workflow and imported from other teams.

Specifically, we explored storing provenance as RDF data structured around a prototype ontology containing both OPM and PML based concepts [5]. In turn, we were able to export provenance in both the OPM and PML formats for use by other groups (although during PC3, only OPM was used by other groups). In both our exported OPM and PML, we were able to handle the provenance specialization requirement mentioned above. However, some interesting issues emerged with both OPM and PML in control flow tracking.

For OPM, these seemed to emerge from ambiguities in its Process concept definition - which could either be viewed as an operation, or the execution of an operation. Such ambiguity was avoided in PML through the concepts pmlj:InferenceRule (the operation) and pmlj:InferenceStep (its execution). While the names InferenceRule and InferenceStep may be used most often in logical theorem provers, they are applied in any setting where some inference (possibly statistical or process) is used to manipulate information thus they are easily applied in a process setting.

Likewise, PML lacked a mechanism for directly tracking dependencies between operations and their executions. However, OPM did provide a way to track dependencies between instances of its Process concept - through the provided wasTriggeredBy relation. Both PML and OPM are evolving to meet community needs and we might expect co-evolution and potentially inclusion or importing of some representational features from one into the other. A comparison of the OPM and PML models can be found at [18].

B. Lessons Learned

Based on our experiences with PC3, and the other case studies, we feel that Semantic Web technologies are well suited for representing workflow provenance (in particular, for facilitating integration of provenance from heterogeneous sources). In addition, while we found some expressivity limitations in PML, these could easily be fixed by adding/referencing other ontology modules (e.g. for control flow concepts).

VI. Related Work

Workflow Provenance Models. There is a diverse literature on workflow provenance models [19]. Although these models differ in certain aspects, they all model some general provenance concepts, including processes, data, and process-data dependencies [1]. Many of them include domain specific concepts required by applications. For instance, Taverna [20] has included bioinformatics ontologies and the VisTrails [21] system adds *workflow description* as a kind of data in tracking user behavior in assembling workflows. PML, as a provenance interlingua, covers these general concepts. It is notable that PML, as an OWL ontology, can be connected to domain concepts (without hard-coding) via ontology mapping (declaring the rdf:type of certain domain data as a subclass of pmlp:Information).

Beyond the basic provenance concepts, some useful concepts like control flow may also be captured by workflow provenance. Furthermore, [22] identified prospective provenance (abstract workflow descriptions) and retrospective provenance (workflow execution logs) in a layered model, and both types are supported by the REDUX [23], Taverna, Pegasus [24], and Karma [25] provenance models. PML core vocabularies only cover the basic provenance concepts in the retrospective provenance because they were designed to only capture generic data derivation processes. However, PML can be extended with workflow specific modules, such as WDO (http://trust.utep.edu/wdo/) and SAWs [26] for capturing prospective provenance.

The Open Provenance Model (OPM) is another general-purposed provenance model. While OPM remains technology agnostic, PML presently provides a family of OWL ontologies with RDF syntax. This brings about a current implementation advantage of PML: it can be seamlessly integrated with domain ontologies and thus support queries involving both domain constraints and generic provenance relations.

Semantic Web Vocabulary for Provenance. There are some existing works on provenance representation in Semantic Web communities. The Dublin Core (DC) ontology (http://dublincore.org/documents/dc-rdf/) offers generic provenance related properties. The Friend of a Friend (FOAF) ontology (http://xmlns.com/foaf/spec/) offers classes and properties for annotating entities involved in provenance, such as people (foaf:person). It is also notable that there are some emerging provenance ontologies [27]. These ontologies have a good overlap with PML, especially its provenance module. However, PML differs from these works based on its justification module which offers support for tracking complex relationships between provenance-based entities.

[5] http://www.cs.rpi.edu/˜michaj6/provenance/PC3OPM.owl

VII. Conclusion

In this paper, we have shown the usage of PML in representing workflow provenance through three case studies. In addressing these case studies, both usability and interoperability challenges emerged in various forms, which required differing strategies to handle. With this, we give some final words on both the challenges of provenance usability and interoperability.

For the usability challenge, many workflow systems (such as SPCDIS and PC3) will rely upon domain-specific concepts that cannot be expressed using a domain independent representation alone. Likewise, others (GILA) will encode provenance data using a domain-specific log that is not intuitive for a general audience. In our case studies, PML proved effective by (i) helping users answer queries involving both domain specific and independent provenance knowledge, and (ii) helping with normalization of domain-specific provenance relationships.

Likewise, to address the interoperability challenge, PML can be (and has been) easily connected to domain ontologies and other provenance models, including OPM, via ontology mappings (as was done with SPCDIS and GILA) and ontology extensions (like with PC3). Here, It should be emphasized that the interoperability challenge requires the establishment of best practices for information exchange (as well as an effectively designed provenance representation like PML).

To establish best practices for provenance interoperability, we stress the adoption of Semantic Web languages (such as OWL) as a common data exchange medium. Through this, explicit mappings can be established between concepts (for instance, by adding "owl:sameAs" assertions). In addition, this would allow for a wider degree of terminology to be used for concept descriptions (such as the PML terms "pmlp:hasName" and "pmlp:hasFormat").

Acknowledgment

This work is partially supported by NSF award #0524481,HRD-0734825, IARPA award #FA8750-07-2-0031, DARPA award #FA8750-07-D-0185,#55-002001,#FA8650-06-C-7605.

References

[1] S. B. Davidson and J. Freire, "Provenance and scientific workflows: challenges and opportunities," in *SIGMOD Conference*, 2008.

[2] Y. Simmhan, B. Plale, and D. Gannon, "A survey of data provenance in e-science," *SIGMOD Record*, vol. 34, no. 3, pp. 31–36, 2005.

[3] S. Miles, P. T. Groth, M. Branco, and L. Moreau, "The requirements of using provenance in e-science experiments," *J. Grid Comput.*, vol. 5, no. 1, pp. 1–25, 2007.

[4] D. L. McGuinness, L. Ding, P. Pinheiro da Silva, and C. Chang, "Pml 2: A modular explanation interlingua," in *ExaCt*, 2007.

[5] D. L. McGuinness and P. Pinheiro da Silva, "Explaining answers from the semantic web: the inference web approach," *Journal of Web Semantics*, vol. 1, no. 4, pp. 397–413, 2004.

[6] P. Pinheiro da Silva, D. L. McGuinness, N. D. Rio, and L. Ding, "Inference web in action: Lightweight use of the proof markup language," in *International Semantic Web Conference*, 2008.

[7] J. W. Murdock, D. L. McGuinness, P. Pinheiro da Silva, C. A. Welty, and D. A. Ferrucci, "Explaining conclusions from diverse knowledge sources," in *International Semantic Web Conference*, 2006.

[8] P. Pinheiro da Silva, P. J. Hayes, D. L. McGuinness, and R. Fikes, "Ppdr: A proof protocol for deductive reasoning," Knowledge Systems, AI Laboratory, Stanford University, Tech. Rep. KSL-04-04, 2004.

[9] X. S. Zhang, S. Yoon, P. DiBona, D. S. Appling, L. Ding, J. R. Doppa, D. Greeny, J. K. Guo, U. Kuter, G. Levine, R. L. MacTavish, D. McFarlane, J. Michaelis, H. Mostafa, S. Ontanon, C. Parker, J. Radhakrishnan, A. Rebgunsy, B. Shrestha, Z. Song, E. B. Trewhitt, H. Zafar, C. Zhang, D. Corkill, G. DeJong, T. G. Dietterich, S. Kambhampati, V. Lesser, D. L. McGuinness, A. Ram, D. Spearsy, P. Tadepalli, E. T. Whitaker, W.-K. Wong, J. A. Hendler, M. O. Hofmann, and K. Whitebread, "An ensemble learning and problem solving architecture for airspace management," in *IAAI'2009*, 2009.

[10] D. L. McGuinness, "Explaining complex systems," in *Semantic e-Science Workshop co-located with the Association for the Advancement of Artificial Intelligence Conference*, 2007.

[11] D. L. McGuinness, A. Glass, M. Wolverton, and P. Pinheiro da Silva, "Explaining task processing in cognitive assistants that learn," in *Proceedings of the 20th International FLAIRS Conference (FLAIRS-20)*, 2007, pp. 284–289.

[12] A. Glass, D. L. McGuinness, and M. Wolverton, "Toward establishing trust in adaptive agents," in *IUI*, 2008.

[13] N. Del Rio and P. Pinheiro da Silva, "Probe-it! visualization support for provenance," in *ISVC (2)*, 2007, pp. 732–741.

[14] J. Tao, L. Ding, and D. L. McGuinness, "Instance data evaluation for semantic web-based knowledge management systems," in *HICSS*, 2009.

[15] D. L. McGuinness, P. Fox, P. Pinheiro da Silva, S. Zednik, N. D. Rio, L. Ding, P. West, and C. Chang, "Annotating and embedding provenance in science data repositories to enable next generation science applications," in *American Geophysical Union, Fall Meeting (AGU2008), Eos Trans. AGU, 89(53), Fall Meet. Suppl., Abstract IN11C-1052*, 2008.

[16] J. R. Michaelis, L. Ding, and D. L. McGuinness, "Towards the explanation of workflows." in *Proceedings of the IJCAI'09 Workshop on Explanation-Aware Computing.*, 2009.

[17] L. Moreau, B. Plale, S. Miles, C. Goble, P. Missier, R. Barga, Y. Simmhan, J. Futrelle, R. McGrath, J. Myers, P. Paulson, S. Bowers, B. Ludaescher, N. Kwasnikowska, J. V. den Bussche, T. Ellkvist, J. Freire, and P. Groth, "The open provenance model (v1.01)," December 2008. [Online]. Available: http://eprints.ecs.soton.ac.uk/16148/

[18] J. R. Michaelis, S. Zednik, L. Ding, and D. L. McGuinness, "A comparison of the opm and pml provenance models," in *Tetherless World Constellation (RPI) Technical Report*, 2009, pp. TW–2009–21.

[19] J. Freire, D. Koop, E. Santos, and C. T. Silva, "Provenance for computational tasks: A survey," *Computing in Science and Engineering*, vol. 10, no. 3, pp. 11–21, 2008.

[20] J. Zhao, C. A. Goble, R. Stevens, and D. Turi, "Mining taverna's semantic web of provenance," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 5, pp. 463–472, 2008.

[21] J. Freire, C. T. Silva, S. P. Callahan, E. Santos, C. E. Scheidegger, and H. T. Vo, "Managing rapidly-evolving scientific workflows," in *IPAW*, 2006, pp. 10–18.

[22] B. Clifford, I. T. Foster, J.-S. Vöckler, M. Wilde, and Y. Zhao, "Tracking provenance in a virtual data grid," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 5, pp. 565–575, 2008.

[23] R. S. Barga and L. A. Digiampietri, "Automatic capture and efficient storage of e-science experiment provenance," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 5, pp. 419–429, 2008.

[24] J. Kim, E. Deelman, Y. Gil, G. Mehta, and V. Ratnakar, "Provenance trails in the wings/pegasus system," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 5, pp. 587–597, 2008.

[25] Y. L. Simmhan, B. Plale, and D. Gannon, "Karma2: Provenance management for data-driven workflows," *Int. J. Web Service Res.*, vol. 5, no. 2, pp. 1–22, 2008.

[26] A. Gates, P. Pinheiro da Silva, L. Salayandia, O. Ochoa, A. Gandara, and N. Del Rio, "Use of abstraction to support geoscientists' understanding and production of scientific artifacts," in *Geoinformatics: Cyberinfrastructure for the Solid Earth Science*. Cambridge University Press, 2009.

[27] O. Hartig, "Provenance information in the web of data," in *In Proceedings of the Linked Data on the Web (LDOW) Workshop at WWW'09*, 2009.

Using Web Data Provenance for Quality Assessment

Olaf Hartig
Databases and Information Systems Research Group
Department of Computer Science
Humboldt-Universität zu Berlin
Email: hartig@informatik.hu-berlin.de

Jun Zhao
Image Bioinformatics Research Group
Department of Zoology
University of Oxford
Email: jun.zhao@zoo.ox.ac.uk

***Abstract*—The Web of Data cannot be a trustworthy data source unless an approach for evaluating the quality of data on the Web is established and integrated as part of the data publication and access process. In this paper, we propose an approach of using provenance information about the data on the Web to assess their quality and trustworthiness. Our contributions include a model for Web data provenance and an assessment method that can be adapted for specific quality criteria. We demonstrate how this method can be used to evaluate the timeliness of data on the Web, to reflect how up-to-date the data is. We also propose a possible solution to deal with missing provenance information by associating certainty values with calculated quality values.**

I. Introduction

With the growth of the open-accessible Web of Data [1] the needs for evaluating the quality of the data in applications are becoming more and more pressing. Information quality research has been successfully applied to evaluate the quality of organizational information and to monitor the improvement of work practice [2]. Quality assessment of data on Web should be a paramount task in order to ensure that the most appropriate and trustworthy data are made available and delivered to the users. Scientific applications built upon the Web of Data will be of little value if scientists are skeptical of the quality of data; financial systems will be untrustworthy and fragile without any policies for quality control and evaluation.

To assess the quality of data, we need to identify the types of information that can be used for evaluation and a method for calculating quality values. In this paper, we present an approach that uses provenance information to assess quality of data on the Web; and we propose a generic assessment procedure that can be adapted for evaluating specific quality-criteria, such as accuracy and timeliness.

As the base of our approach we introduce a provenance model tailored to the needs for tracking and tracing provenance information about data on the Web. This model not only represents the creation of a data item, but also describes provenance information about the entities who made the data accessible on the Web [3]. We call this *Web data provenance*.

Many existing information quality assessment approaches are based on information contributed by users. In this paper, we focus on using a quantitative approach for calculating quality of data. This assessment approach takes three steps: collecting the elements of provenance information needed for quality assessment, then deciding on the influence of these elements on the assessment, and, finally, applying a function to calculate the quality.

To demonstrate how design decisions can be made when developing this method into assessing specific quality criteria we walk through the development using the *timeliness* criterion as an exemplar. Since the provenance information required for quality assessment might be incomplete or fragmentary, the assessment method must be capable to deal with missing information. We introduce a possible approach of associating certainty values with the calculated timeliness value.

This paper is structured as follows. Section II reviews related work and Section III introduces the model for Web data provenance. Section IV describes our assessment method that can be adapted for specific quality criterion, like timeliness, as demonstrated in Section V. We conclude in Section VI.

II. Related Work

In this paper we consider information quality (IQ) as “an aggregated value of multiple IQ-criteria” [4], such as accuracy, completeness, believability, and timeliness. The assessment of information quality can be regarded as “the process of assigning numerical values (IQ-scores) to IQ-criteria” [4]. IQ assessment is known to be hard [4]. Although there are many related work on conceptualizing the problem of IQ and its assessment, there are fewer work proposing concrete methods for quantifying the quality assessment. In the following we first introduce different approaches for IQ assessment in general, and then we focus on provenance-based.

Lee et al. [2] propose a quality assessment methodology that measures IQ from four quadrants: soundness, dependability, usefulness, and usable information. Each dimension includes several IQ-criteria. For example the dependability of information is measured by its timeliness and security. A questionnaire is designed to measure users’ feedback to each IQ-criterion in a scale of 0-10. The assessment value for each quadrant is computed as a mean of the measurements of its constituent IQ-criteria. Similarly, Bobrowski et al. [5] also use questionnaires to assess information quality. Both methods, although being quantitative, are based on subjective, users’ inputs.

In certain circumstances, an automatic assessment of information quality could be feasible with available metadata information and reliable auto-assessment techniques. Depending on the application and users’ needs, this automatic approach could be more desirable than a subjective, manual approach.

Motro and Rakov propose automated assessment methods for evaluating the soundness and completeness of data sources [6]. Gruser et al. present a prediction algorithm to learn and predict response times of Web data sources [7]. Ballou et al. [8] introduce a quantitative assessment method for measuring and calculating the timeliness of data. Their formulas laid the foundation for our work and will be detailed in Section V.

Ballou et al.'s method for assessing the timeliness is partially based on the provenance information of a data item, e.g., the time when the data was obtained. Provenance metadata has been used to evaluate other IQ-criteria, such as the believability and trustworthiness. Wong et al. [9] use information about the types of services or data involved in a data creation process to validate the believability of derived data items. Golbeck and Mannes [10] use provenance of user-contributed annotations to compute trust values and to recommend how much a user should trust others. This method does not compute the trustworthiness of the annotations themselves using provenance.

III. A Model for Web Data Provenance

Our provenance-based IQ assessment method is based on our model for Web data provenance. We give a brief introduction to the model in this section. A detailed discussion of the model can be found in [3].

Traditional provenance research usually addresses the creation of data. While many approaches exist that represent provenance [11], [12], none of these explicitly addresses the characteristics of data that was not only created but also retrieved over the Web. Provenance of data from the Web comprises information about the entities that published the data and that made it accessible on the Web, information not required in the context of self-contained systems such as a DBMS or a workflow management system. Hence, our model for Web data provenance comprises two dimensions: data creation and data access.

Our model identifies types of so called *provenance elements* and the relationships between these types. The provenance elements represent pieces of provenance information; such an element might be the actual creator of a specific data item what makes this element an instance of the 'data creator' type. The types are classified in three categories: actors, executions, and artifacts. An *actor* usually performed the *execution* of an action or a process which – in most cases – yielded an *artifact* such as a specific data item. An execution might have included the use of artifacts which, in turn, might be the result of another execution. Furthermore, direct relationships between artifacts as well as between actors may exist. For instance, a specific company was responsible for its Web server. All other element types are specializations of actors, executions, and artifacts.

The central type in the data creation dimension is the *data creation* execution by which a data item was created. A data creation was performed by a *data creator*. For the creation *source data* and *creation guidelines* could have been used by the data creator.

The data access dimension centers around *data access* executions. *Data accessors* perform data access executions to retrieve data items contained in *documents* from a provider on the Web. To enable a detailed representation of providers the model distinguishes *data providing services* that process data access requests and send the documents over the Web, *data publishers* who use data providing services to publish their data, and *service providers* who operate data providing services. Furthermore, the model represents the execution of *integrity verifications* of artifacts and the results thereof.

Based on the element types and their relationships identified by our provenance model it is possible to represent provenance of data items from the Web by, so called, *provenance graph*s. The nodes in these graphs are the provenance elements; the edges correspond to the relationships between the element types of adjacent elements; edges are labeled with the relationship name. Notice, to allow for a wide range of applications of our model we do not prescribe a specific granularity by which provenance information has to be described in provenance graphs. For instance, a data item could be a whole linked dataset as well as a single RDF statement, depending on the granularity required for the use case at hand. A data item could have been created by the use of creation guidelines and source data which also have provenance. This provenance should be represented by subgraphs in the provenance graph of the created data item.

Formally, we represent a provenance graph as a tuple $(PE, R, type, attr)$ where

- PE denotes the set of provenance elements in the graph,
- $R \subseteq PE \times PE \times RN$ denotes the labeled edges in the graph where RN is the set of relationship names as introduced by our provenance model,
- $type : PE \rightarrow \mathfrak{P}(T)$ is a mapping that associates each provenance element with its types where T is the set of element types as introduced by our provenance model,
- $attr : PE \rightarrow \mathfrak{P}(A \times V)$ is a mapping that associates each provenance element with additional properties represented by attribute-value pairs where A is a set of available attributes and V is a set of values.

We do not specify the sets A and V any further because the available attributes, possible values, and the meaning of these depend on the use case. However, we introduce an abbreviated notation to refer to the target of an edge in a provenance graph: if $(pe_1, pe_2, \mathrm{rn}) \in R$ we write $pe_1 \xrightarrow{\mathrm{rn}} \circ = pe_2$.

IV. Provenance-Based Quality Assessment

The method is based on provenance graphs represented using our provenance model. This approach should be regarded as a blueprint for the development of actual assessment methods that address specific scenarios and focus on specific quality criteria. This section introduces the general method and discusses questions that must be addressed when applying this method for a specific quality criterion.

A. The General Assessment Approach

The main idea of our approach is the automated determination of a quality measure for a data item, from so called *impact value*s, which represent the influence of the elements in a provenance graph on the particular quality of the assessed data item. We divide the assessment procedure into three steps:

1) Generate a provenance graph for the data item;
2) Annotate the provenance graph with impact values;
3) Calculate an IQ-score for the data item from the annotated provenance graph.

In order to use the provenance of a data item for automated quality assessment this provenance has to be represented in the assessment system. We propose to use provenance graphs as introduced in Section III for this purpose. Hence, the first step of an assessment procedure must be the generation of such a graph for the data item that is to be assessed, i.e., the *considered data item*. This step comprises collecting the necessary provenance information about the data item.

Some, if not all, of the provenance elements might have had an influence on certain qualities of the assessed data item. Some of these influences are known to us; others are possible or cannot be ruled out. Both types of influences, known as well as possible influences, have an impact on our assessment of the qualities. We propose to represent this impact by *impact value*s associated with the corresponding provenance elements. For instance, the possibility of manipulating published data by a service provider may affect the believability and the assumed accuracy of the data; an impact value for a service provider could represent the provider's manipulation probability. An example for a known influence is the execution time of a data creation which has an impact on the timeliness assumed for the created data item. Notice, there can be different kinds of impact values for different types of provenance elements.

The second step of our assessment procedure comprises determining these impact values; the system adds annotations to the provenance graph generated from step 1, associating elements in the provenance graph with estimated impact values. Formally, an *annotated provenance graph* is a pair (pg, ann) where $pg = (PE, R, type, attr)$ is a provenance graph and $ann : PE \rightarrow \mathfrak{P}(I)$ is a mapping that associates a provenance element with a set of impact values; each impact value $(\mathsf{n}, v) \in I$ has a name n and the actual value v. For $(\mathsf{n}, v) \in ann(pe)$ we write $\mathsf{n}\big[pe\big] = v$.

In the final step the system executes a function to calculate a value that represents the information quality of the considered data item using the annotated provenance graph from step 2.

B. Designing Actual Assessment Methods

To apply our assessment approach one must first develop the presented method into an actual assessment method that is tailored for the quality criterion of interest. In the following we discuss design decisions that must be considered at each step and we specify the questions that must be addressed.

The most fundamental question that must be answered in the beginning is: *For which quality criterion do we want to apply the method?* This decision influences every aspect of an application of our approach. In the remainder we consecutively focus on the three steps of our assessment method. However, the design decisions for the three steps partly depend on each other. For this reason, designing an actual assessment method should be an iterative process.

Considering step 1 of the assessment method it is necessary to ensure the generation of a provenance graph that is suitable for the assessment. To specify suitability in the given context one has to ask: *What types of provenance elements are necessary to determine the considered information quality* and *what level of detail (i.e. granularity) is necessary to describe the provenance elements in the application scenario?* To answer these questions we propose to study the literature that deals with the considered quality criterion. A good starting point is Pipino et al. [13]. Based on the answers to the above two questions, the procedure for generating provenance graphs can be developed. Defining this procedure requires to address the question: *Where and how do we get the provenance information to generate the provenance graph for a data item?* Basically, there are two complementary options to obtain provenance information: some pieces of provenance information can be recorded by the system; for other pieces the system relies on meta-data provided by third parties. In [3] we discuss these options. Furthermore, we are working on the Provenance Vocabulary[1] to enable the publication of provenance-related metadata in the Web of data.

The fundamental questions that have to be answered for the implementation of step 2 are: *How might each type of provenance element influence the quality of interest* and *what kind of impact values are necessary for the application scenario?* The answers to these questions substantially depend on the considered quality criterion as well as on the assessment function used in step 3. Notice, impact values need not necessarily be numerical; they could also be of a more abstract nature such as the simple weighting "high impact". After specifying the impact values it is necessary to address the question: *How do we determine the impact values* or *where do we get them from?* Some of the impact values might already be part of the provenance information such as the creation time in the aforementioned timeliness example; others might be calculated based on the provenance graph. Certain kinds of impact values could also be determined based on user input. Another possibility is to estimate impact values by taking background knowledge about information consumers or providers into account. For instance, a data creator's credibility which influences believability assessments could be determined based on former experiences as well as on recommendations from other users.

The main questions regarding step 3 of the assessment method are: *How can we represent the considered information quality by a value* and *what function do we use to calculate such a value from the annotated provenance graph?* Again, answering these questions fundamentally depends on the quality criterion. The calculated value could be a single number in a specific interval; but it could also be a vector of numbers or an element of a set of discrete values. In any case, it is important to specify what such a value means. The

[1] http://purl.org/net/provenance/

definition of the applied function depends on the impact values introduced at step 2. For this reason, we recommend to develop the function together with specification of the impact values. For the development of this function it is important to bear in mind that the results of steps 1 and 2 cannot be guaranteed to be complete in many cases; the provenance graph could be fragmentary or some annotations could be missing due to the lack of certain information required during steps 1 and 2. Hence, the function for step 3 must not assume to operate on an ideal annotated provenance graph but it must be able to deal with incompleteness.

V. Provenance-Based Assessment of Timeliness

In this section we exemplarily apply our general assessment approach to assess the timeliness of data from the Web. We first give a brief introduction to timeliness and how it can be calculated; we then illustrate the design and the execution of the three steps of assessment; finally, we propose a way to deal with incomplete provenance information.

A. Representing and Calculating Timeliness

Timeliness is an intrinsic IQ criterion [14] that is often referred to as a task-dependend up-to-dateness of a data item [13], [15]. Ballou et al. represent timeliness by an absolute measure on a continuous scale from 0 to 1 where data with 1 “meet the most strict timeliness standard” [8] and 0 is unacceptable. This timeliness measure can be calculated using the following formula [8]:

$$Timeliness = (max(1 - Currency/Volatility, 0))^s \quad (1)$$

In this formula, *Volatility* is “the length of time the data remains valid” which is analogous to the shelf life of perishable products [8]; *Currency* is “the age of the data when it is delivered to the user” [15] which can be calculated according to [8] by the following formula:

$$Currency = Delivery\ Time - Input\ Time + Age \quad (2)$$

where *Delivery Time* is the time when the data was delivered to the user; *Input Time* is the time when the data was entered in the system; and *Age* is how old the data was at *Input Time*.

The exponent s in (1) is a parameter that controls the sensitivity of *Timeliness* to the *Currency-Volatility* ratio. The ratio should be large (e.g., $s = 2$) for highly volatile data and be small (e.g., $s = 0.5$) for long shelf life data [8].

Note that in Ballou et al.’s paper [8] the timeliness formula is defined in a closed “information manufacturing system”, which processes primitive data units from outside. Hence, the semantics of *Age*, *Input Time*, and *Delivery Time* might be different w.r.t. to an open-world system, like the Web.

On the Web, we do not have primitive data from the *outside*. Instead, we have *unprocessed data* and *derived data*. Unprocessed data are data items for which the creation did not depend on other data items; i.e., no source data was used for their creation. Derived data, in contrast, was derived from other data items.

1) Timeliness of Unprocessed Data: For an unprocessed data item, its *Age* is 0 because it did not exist before; its *Input Time* is the time when its creation was finished; and its *Delivery Time* should be “now”, i.e., the time when the timeliness of the *considered data item* is assessed. This means that the *Currency* values for unprocessed data items differ only by their creation time. To calculate the *Timeliness* of unprocessed data items using formula (1) we also need the *Volatility*. We could speak of volatility exclusively as *shelf life*, as Ballou et al. [8] do. Alternatively, we could speak of *expiry time* and adapt the formula from the Sampaio et al. [15]:

$$Volatility = Expiry\ Time - Input\ Time + Age \quad (3)$$

2) Timeliness of Derived Data: Ballou et al. compute the timeliness of data outputs from a *processing block* as a weighted average value [8]. In our method, for a derived data item, if it is caused by only one source data item, then it has the same timeliness value as the source data item; if it is caused by multiple source data items, then its timeliness value should be a weighted average of the timeliness values of the source data items.

B. Constructing the Provenance Graph

We adopt the calculation approach outlined in the previous section to apply our provenance-based assessment method for the determination of timeliness. The first step is to generate a provenance graph for the considered data item. For this work we assume the availability of all provenance information.

Example 1: We demonstrate the method applied to assessing timeliness of temperature measurements taken by a sensor. These measurements are unprocessed data items. They are taken every 1 hour, and they are stored in a Web-accessible storage device immediately. A system accesses these measurement from the storage device for further processing; in order to process the measures the system evaluates their timeliness.

We represent the provenance of a specific measure by a provenance graph $pe = (PE, R, type, attr)$ which is illustrated in Figure 1. PE contains the measure msr, the sensor $sens$, the data creation $cExc$ that produced msr, the storage device $stor$, the system sys, the data access $aExc$, and the document doc with which msr was retrieved during $aExc$. Given msr was taken at 10:00 and doc was retrieved at 10:13 it holds $attr(cExc) = \{(execTime, 10{:}00)\}$ and $attr(aExc) = \{(execTime, 10{:}13)\}$. □

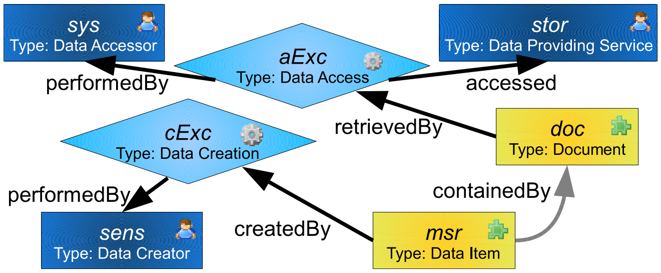


Fig. 1. Provenance graph representing our running example (cf. Example 1).

C. Adding Impact Values

The second step of the assessment method includes the annotation of the provenance graph with impact values. In order to design this step we study the relevance of different pieces of provenance information for the timeliness assessment. In particular, we discuss the relation of the provenance element types introduced by our provenance model to the calculation approach outlined before (cf. Section V-A).

Data creation executions have a direct influence on the timeliness assessment. As discussed before, the creation time of unprocessed data items corresponds to the input time in formula (2). Hence, we annotate each data creation element that is not associated with source data with a *creation time impact value*. It is not necessary to explicitly determine these kind of impact values because they are already represented in the provenance graph as an attribute of the data creation elements.

Data creations that yield a derived data item have an influence on the timeliness of this item if multiple source data items were used (cf. Section V-A2). We reflect this influence by another impact value: each data creation element that takes multiple source data items as inputs is annotated with a *weights impact value*. This impact value represents the weights that can be used to calculate the weighted average of the timeliness values of the source data items. Ballou et al. write: “The weights could reflect the size of the data units that are merged, their importance or some combination of attributes.” [8]. In this paper, we leave the choice of the weights to applications adopting our assessment method, because this choice should be based on actual needs from their information consumers.

Creation guidelines may have an impact on IQ criteria such as accuracy and reliability. However, creation guidelines have no influence on the timeliness of the created data.

Source data may have an impact on the timeliness assessment. According to the calculation approach the timeliness of each derived data item can be ascribed to a combination of the timeliness values of unprocessed data items. Hence, in the ideal case of a complete provenance graph only the unprocessed data items have a direct influence on the timeliness of the considered data item. While their currency can be determined with the aforementioned creation time impact values we also need the volatility to calculate their timeliness using formula (1). To enable the calculation of their volatility using formula (3) we annotate each unprocessed data item with an *expiry time impact value*. We assume these impact values can be determined based on the input from users who configure a default expiry time for data from specific data creators or for data with a specific content.

Data creators have an influence on the volatility of unprocessed data items as discussed before. The previously mentioned strategy for determining the expiry time impact values reflects this influence.

Example 2: We annotate the provenance graph pg from Example 1 with impact values as follows. The data creation $cExc$ is not associated with source data; hence, it has to be annotated with a creation time impact value that refers to its $execTime$ attribute: $ann(cExc) = \{(\mathsf{crtT}, 10{:}00)\}$. Furthermore, pg contains the unprocessed data item msr which has to be annotated with an expiry time impact value. It is possible to determine this value based on the information that $sens$ takes the measures every hour. Hence, it holds: $ann(msr) = \{(\mathsf{expT}, 11{:}00)\}$. The other elements in pg do not have an influence on our timeliness assessment. □

In the ideal case of a complete provenance graph the elements that belong to the data access dimension can be ignored for the timeliness assessment. However, in the likely case that information about the creation of a (source) data item is missing or that it is impossible to determine one of the impact values introduced so far. Hence, we propose to consider the data access related elements as fall-back. For these data items, the *Input Time* in formulas (2) and (3) is the access time associated with the corresponding data access execution. Furthermore, the *Age* for these items is probably larger than 0, assuming that they, or the data from which they were derived, were not created at the time of the access. We annotate each of these data items with a *timeliness impact value* that represents a timeliness value estimated for them. This value could be estimated based on different data access related provenance elements. For instance, knowing when a data publisher updates her data may, in combination with the access time, be an indicator for the *Age*. The *Expiry Time* might be estimated based on information about the update frequency of the data publisher. After all, it must be realized that the timeliness impact values can only be estimates at best.

D. Calculating Timeliness

Based on the impact values in the annotated provenance graph it is possible to calculate timeliness by adopting formulas (1) to (3). The recursive function t in Figure 2 implements this idea: t incorporates (1) to (3) to calculate timeliness at step 3 of our assessment method. For a data item with incomplete provenance information t returns the timeliness impact value that is annotated to this item (cf. first case in the equation in Figure 2). For unprocessed data items t applies the formulas (1) to (3) using the corresponding creation time and expiry time impact values (cf. second case in the equation). For derived data items that were created with a single source data item t returns the timeliness value that is recursively calculated for the source data item (cf. third case). Finally, for other derived data items t uses the weights impact value of the corresponding data creation element to calculate a weighted average of the recursively calculated timeliness values of the source items (cf. fourth case).

Example 3: Based on the annotated provenance graph (pg, ann) from Example 2 it is possible to calculate the timeliness of msr. Since $msr \xrightarrow{\text{createdBy}} \circ \xrightarrow{\text{used}} \circ = \varnothing$ it holds:

$$t(msr) = \left(\max\left(1 - \frac{now - \mathsf{crtT}\left[cExc\right]}{\mathsf{expT}\left[msr\right] - \mathsf{crtT}\left[cExc\right]}, 0\right)\right)^s$$
$$= \left(\max\left(1 - \frac{now - 10{:}00}{11{:}00 - 10{:}00}, 0\right)\right)^s$$

$$t(d) = \begin{cases} \mathsf{timeliness}\left[d\right] & \text{if } d \xrightarrow{\text{createdBy}} \circ \text{ is unknown,} \\ \left(\max\left(1 - \dfrac{now - \mathsf{crtT}\left[d^{\xrightarrow{\text{createdBy}}}\circ\right]}{\mathsf{expT}\left[d\right] - \mathsf{crtT}\left[d^{\xrightarrow{\text{createdBy}}}\circ\right]}, 0\right)\right)^{s} & \text{if } d \xrightarrow{\text{createdBy}} \circ \xrightarrow{\text{used}} \circ = \varnothing, \\ t(d_s) & \text{if } d \xrightarrow{\text{createdBy}} \circ \xrightarrow{\text{used}} \circ = \{d_s\}, \\ \dfrac{\sum_{d_s \in d^{\xrightarrow{\text{createdBy}}}\circ \xrightarrow{\text{used}} \circ} \mathsf{weight}\left[d^{\xrightarrow{\text{createdBy}}}\circ\right]_{d_s} \cdot t(d_s)}{\sum_{d_s \in d^{\xrightarrow{\text{createdBy}}}\circ \xrightarrow{\text{used}} \circ} \mathsf{weight}\left[d^{\xrightarrow{\text{createdBy}}}\circ\right]_{d_s}} & \text{if } \left|d \xrightarrow{\text{createdBy}} \circ \xrightarrow{\text{used}} \circ\right| > 1. \end{cases}$$

Fig. 2. The recursive function that calculates the timeliness of a data item d based on impact values from the annotated provenance graph for d.

Given $s = 1$ and the timeliness assessment happens at 10:15, i.e. $now = 10{:}15$, we get the result:

$$= \max\left(1 - \frac{0.25\text{h}}{1\text{h}}, 0\right) = \underline{\underline{0.75}} \qquad \square$$

E. Dealing with Incomplete Provenance Information

Our timeliness assessment method deals with incomplete information by using alternative impact values; furthermore, certain impact values can only be determined by estimation. Thus, the calculated timeliness value becomes an approximation rather than an exact assessment. To make the degree of inexactness explicit we propose to associate the calculated timeliness value with a certainty value. This additional value represents the certainty of whether the calculated timeliness is correct. We suggest to represent certainty with a value in the interval [0,1] where 1 represents absolute certainty, i.e. no doubt, and 0 represents absolute unvertainty, i.e. the calculated timeliness value is useless. In the following we outline an approach to calculate the certainty value during the execution of our assessment method.

We assume a value, initialized to 1, that is accessible throughout the whole assessment procedure. During the execution of steps 1 and 2 this value is incrementally decreased whenever i) a part of the provenance graph cannot be generated appropriately due to missing provenance information and whenever ii) an impact value is estimated. With each decrease the value should be reduced by a certain percentage where the extent of this percentage may differ for different decreases. Identifying appropriate extents is subject to further research. For instance, in the case of missing provenance information the importance of this information to the assessment may affect the amount of reduction. Decreases due to impact value estimation may differ depending on the reliability of the applied estimation strategy. However, after the completion of step 2 the decreased value represents the reliability of the annotated provenance graph. Since the calculation in step 3 is solely based on this graph the value also represents a certainty regarding the correctness of the calculated timeliness value.

VI. Conclusion

In this paper we propose a provenance model for Web data provenance and an assessment method for evaluating the quality of data on the Web using provenance graphs based on this model. Our provenance model introduces a new dimension of provenance information, i.e. the provenance of data access, to the existing provenance research. We are gathering feedback to our model from different communities and we foresee continuing development of our provenance vocabulary driven by well-defined use cases. In this paper, we demonstrate assessing the timeliness of data on the Web using our method. We plan to implement this method as part of a Web data publication framework in the near future and to apply this method to the assessment of other quality criteria, such as accuracy. Our method should be generic enough to incorporate subjective quality indicators derived from Web data provenance. Existing work on evaluating and filtering subjective quality indicators will be considered and appropriately applied.

VII. Acknowledgement

Part of this work is funded by EPSRC Postdoctoral Fellowship to Dr Jun Zhao (Grant EP/G049327/1).

References

[1] C. Bizer, T. Heath, and T. Berners-Lee, “Linked Data - The Story So Far,” *Int. Journal on Semantic Web and Information Systems, Special Issue on Linked Data*, 2009, in press.

[2] W. L. Yang, M. S. Diane, B. K. Kahn, and Y. W. Richard, “AIMQ: a methodology for information quality assessment,” *Information & Management*, vol. 40, no. 2, 2002.

[3] O. Hartig, “Provenance Information in the Web of Data,” in *Proc. of the Linked Data on the Web Workshop at WWW*, 2009.

[4] F. Naumann, *Quality-driven query answering for integrated information systems*. Springer Verlag, 2002.

[5] M. Bobrowski, M. Marré, and D. Yankelevich., “A homogeneous framework to measure data quality,” in *Proc. of IQ*, 1999.

[6] A. Motro and I. Rakov, “Estimating the quality of databases,” in *Proc. of FQAS*, 1998.

[7] J.-R. Gruser, L. Raschid, V. Zadorozhny, and T. Zhan, “Learning response time for websources using query feedback and application in query optimization,” *VLDB Journal*, vol. 9, no. 1, 2000.

[8] D. Ballou, R. Wang, H. Pazer, and G. K. Tayi, “Modeling Information Manufacturing Systems to Determine Information Product Quality,” *Management Science*, vol. 44, no. 4, 1998.

[9] S. C. Wong, S. Miles, W. Fang, P. Groth, and L. Moreau, “Provenance-based validation of e-science experiments,” in *Proc. of ISWC*, 2005.

[10] J. Golbeck and A. Mannes, “Using Trust and Provenance for Content Filtering on the Semantic Web,” in *Proc. of the Models of Trust for the Web Workshop at WWW*, 2006.

[11] Y. Simmhan, B. Plale, and D. Gannon, “A Survey of Data Provenance in e-Science,” *SIGMOD Record*, vol. 34, no. 3, 2005.

[12] W. C. Tan, “Provenance in Databases: Past, Current, and Future,” *IEEE Data Engineering Bulletin*, vol. 30, no. 4, 2007.

[13] L. L. Pipino, Y. W. Lee, and R. Y. Wang, “Data Quality Assessment,” *Communications of the ACM*, vol. 45, no. 4, 2002.

[14] C. Bizer, *Quality-Driven Information Filtering in the Context of Web-Based Information Systems*. VDM Verlag, 2007.

[15] S. de F. Mendes Sampaio, C. Dong, and P. Sampaio, “Incorporating the Timeliness Quality Dimension in Internet Query Systems,” in *Proc. of WISE*, 2005.

A New Perspective on Semantics of Data Provenance

Sudha Ram, Jun Liu
430J McClelland Hall, Department of MIS, Eller School of Management,
University of Arizona, Tucson, AZ 85721

Abstract: Data Provenance refers to the “origin”, “lineage”, and “source” of data. In this work, we examine provenance from a semantics perspective and present the W7 model, an ontological model of data provenance. In the W7 model, provenance is conceptualized as a combination of seven interconnected elements including “what”, “when”, “where”, “how”, “who”, “which” and “why”. Each of these components may be used to track events that affect data during its lifetime. The W7 model is general and extensible enough to capture provenance semantics for data in different domains. Using the example of the Wikipedia, we illustrate how the W7 model can capture domain or application specific provenance.

1. Introduction

Data provenance is an overloaded term that has been defined differently by different people. A recent survey [1] reviews the various definitions of provenance in literature. Some researchers define provenance as the origin or source of data [2]. As an example, Buneman puts forth two forms of data provenance, i.e., “why” provenance and “where” provenance [3]. Both “why” and “where” provenance deal with tracing the source from which the data came. Others view provenance as metadata recording the process of experimental workflows, annotations and notes about scientific experiments [4]. In research such as [5, 6], the data generating processes in the form of workflows are the primary entities for which provenance are collected. Due to the lack of consensus on the semantics or meaning of provenance, current efforts on capturing data provenance have focused on only one or two aspects of provenance while ignoring others. As a result, the provenance is often incomplete and cannot be shared across applications. In response to this challenge, we attempt to formally define the semantics of provenance that can be agreed upon by people from different domains. To our knowledge, our research is the first of its kind to explore the “semantics” of provenance.

In this research, we define the W7 model, an ontology that clarifies the semantics of data provenance. The W7 model represents data provenance as a combination of seven interconnected elements including, “what”, “when”, “where”, “how”, “who”, “which”, and “why”. The W7 model is general and extensible enough to capture provenance semantics for data in different domains. Using examples in Wikipedia, we illustrate how the W7 model can help define, capture, and use data provenance.

2. Use cases and competency questions

Following the formal methodology for ontology development proposed in [7], we started by collecting use cases from different domains. Given the set of use cases, a set of competency questions were identified. The competency questions are those that our ontology must be “competent” to answer. Our use cases and their corresponding competency questions describe a set of requirements the ontology must satisfy. They helped us understand the intended informal semantics of the concepts and relations to be included in the ontology. We gathered 188 use cases from users in various domains including biology, businesses (such as the manufacturing, defense, and pharmaceutical organizations), and physical sciences. We present some of the use cases as well as the competency questions.

Use Case 1: In a missile manufacturing company, an engineer performs a material test to measure the transverse tension fatigue life of a particular material "S2/8552 glass-epoxy". She then publishes the results and test procedure online. Another engineer discovers the published results years later. Before reusing the results, he verifies whether the results are valid by repeating the test procedure, in the test environment that was described.

Competency Questions: A replication of a material test requires recording provenance that is competent in answering the following questions: 1) *how was the material data created*, and 2) *how was the material test conducted in terms of the test procedure, test environment, sample condition, temperature, etc*.

Use Case 2: To organize the huge amounts of bio images being generated, the bio-computing lab stores bio images on different storage devices based on their "value". For instance, images created by a graduate student doing an internship or images that have not been accessed for 5 five years are deemed less valuable.

Competency Questions: Use case 2 demonstrates the desire for recording "*who created the bio images*" and "*when the bio images have been accessed*".

Use Case 3: An engineer obtains the composite material "Cycom 381/S2 Uni-glass" and performs a test to measure the tensile strength of the composite. Another engineer in a different lab later performs a test on the same material, again provided by the same vendor. She compares the two results and notices significant differences. She needs to assess whether the differences are because of different test methods or different instruments used in the test.

Competency Questions: To determine the quality or reliability of material test results, it is necessary to provide answers to the following two questions: 1*) how were the results generated*, and 2) *which instrument was used in created the data and what were its parameter settings?*

Use Case 4: A genetics researcher records in his lab notebook the reason for using specific data records in an *in silico* experiment, e.g., "I chose this restriction enzyme as it cut only three times within 200 base pairs of the SNP".

Competency Questions: The relevant question is *why certain records data were used*.

Use Case 5: A scientist, S, is interested in rainfall and water levels in neighboring rivers and lakes for a part of the Sierra Nevada mountain range in California. He is trying to acquire sensor signals captured in Southern California.

Competency Questions: Use Case 5 indicates the use of data provenance for data discovery. In this use case, the question the scientist needs to answer is *"where was the data measured",* so that he can locate the appropriate data.

Table 1: Summary of use cases and their competency questions

Competency question	*Number of use cases*
What	188
How	156
Who	145
Which	91
When	131
Where	113
Why	86

Table 1 summarizes the use cases and their competency questions. As an example, the *how* question was necessary to answer in 156 use cases. Our analysis of the use cases and their competency questions indicates that the provenance ontology must contain information regarding *who*, *when*, *where*, *how*, *why* and *which*. Moreover, all of the use cases indicate that the central element of interest is the event that affects each piece of data during its life cycle from birth (creation) to death (deletion or archiving). While many of the use cases point out the need to understand the data creation related provenance, in many cases, other life cycle events are even more useful. For example, *Use case 4* requires us to record the *why* associated the *use* of data. Also, for some domains, the most critical provenance events are *changes in the ownership* of the data and *archiving* of data. As a result, our provenance ontology should be competent to answer the question of "*what"*, i.e., *events* that affect the data. Thus our ontology is anchored around the "what" or the life cycle events affecting the data.

3. Conceptualization of provenance based on Bunge's theory

The use case analysis helped us identify the basic components of data provenance including the 7 Ws (*w*hat, ho*w*, *w*hen, *w*here, *w*ho, *w*hich, and *w*hy). We then adopt Bunge's ontology [8] to define these components and identify the relationships between them.

State, event and history: The elementary notion of Bunge's ontology is a *thing*. The *state* of a thing is the set of property values of the thing at a given time. Bunge's ontology postulates that everything changes, and every change is a change of state of things, that is the change of properties of things. A change of state is termed an *event*. It follows that an event occurs when a thing acquires or loses a property or changes the value of a property. Based on the constructs of *event* and *state*, Bunge defines the concept of history: History of a thing is a sequence of *event*s that happens to the thing.

Action, agent, time and space: These are constructs related to events. An event on a thing occurs when it is *acted* upon by another thing, which is often a human or a software *agent.* An event happens in *time* and *space*.

Data are also "things". Bunge's theory regarding *history* and *events* is a perfect match for defining data provenance and its semantics since data provenance is often referred to as the pedigree or *history* of data. More importantly, our use case analysis indicates that data provenance is really all about various *events* that affect data during its life cycle. Thus, the constructs in Bunge's ontology including *history*, *event*, *action*, etc. lay a theoretical foundation for defining provenance and its components. We define provenance and the 7 Ws and develop connections between them using the constructs in Bunge's ontology.

4. An ontological model of data provenance – the W7 model

We conceptualize data provenance as consisting of seven interconnected elements including what, when, where, who, how, which, and why.

Definition (Provenance). Provenance of some data *D* is a set of n-tuples: *p(D)* = {< *What, When, Where, How, Who, Which, Why* >}. *What* denotes an event that affected data during its lifetime; *When* refers to the time at which the event occurred ; *Where, is* the location of the event; *How*, is the action leading up to the event; *Who*, is agents involved in the event; *Which*, are the programs or instruments used in the event; and *Why*, the reasons for the events. We therefore name our ontological model for provenance the W7 model. A graphical representation of the W7 model is shown in Figure 1. We represent the W7 model as conceptual graphs (CGs) developed

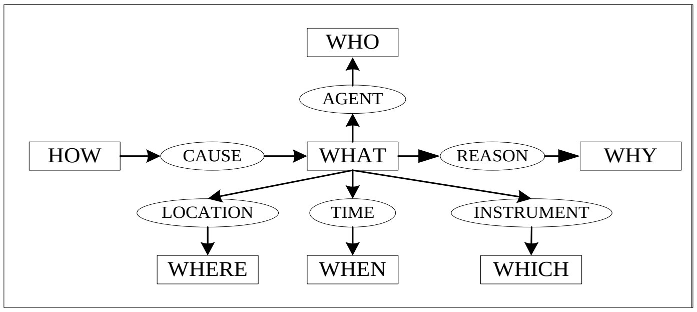


Figure 1. Overview of the W7 model

by Sowa [9], which has been widely as a language for ontology. The boxes in CGs represent concepts and the bubbles are the relationships. As shown in Figure 1, *what*, i.e., events, is the anchor of our model. In essence, data provenance includes events and various information (including who, how, when, where, which and why) associated with and describing the events.

Tables 2 summarizes the definition of each of the 7 Ws and shows the correspondence between the Ws and Bunge's ontology concepts. For interested readers, please refer to our previous research [10] for a more detailed discussion of each of the 7 Ws.

Table 2: Definition of the 7 Ws

Provenance Element	*Construct in Bunge's ontology*	*Definition*
What	Event	An event (i.e. change of state) that happens to data during its life time
How	Action	An action leading to the events. An event may occur, when it is *acted* upon by another thing, which is often a human or a software agent
When	Time	Time or more accurately the duration of an event
Where	Space	Locations associated with an event
Who	Agent and other things	Agents including persons or organizations involved in an event
Which		Instruments or software programs used in the event
Why	-	Reasons that explain why an event occurred

In [11], Simmhan et al argue that due to the diverse needs across disciplines, it is challenging to develop a standard model for capturing provenance. To address this concern, we developed the W7 model as a generic ontology of provenance that captures the semantics of data provenance and can thus be applied to various domains. However, for our model to be of any practical use, it must be easily adaptable to address domain specific provenance needs. We use the "type definition" mechanism developed by Sowa [9] to provide the domain specific extension of the W7 model. The CG formalism enables to explicitly define the semantics of a concept via a type definition. As an example, in the domain of design and manufacturing, *how* often refers to a material test, using which material data is created. The specification of the test and the material sample used in the test are critical provenance that needs to be captured. We thus formally define "material test", as shown in Figure 2.

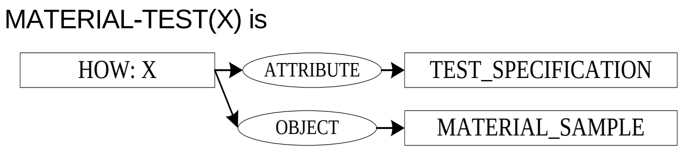


Figure 2. Type definition of the concept "material-test"

The CG in Figure 2 defines MATERIAL-TEST as a subtype of HOW. A material test is carried out upon material samples and it has an attribute "test specification". Type definitions represent the semantics and necessary attributes of a concept that have been agreed upon by people in a domain and therefore can be used to provide domain specific extensions of the W7 model.

5. Application of the W7 model – the Wikipedia example

We use Wikipedia as an example to illustrate the application of the W7 model to harvest and structure data provenance. Table 3 summarizes the application of the W7 model in Wikipedia. *What* or events that affect a Wikipedia page are primarily creation, modification and destruction of the page. Other events may include "quality assessment" (e.g., a page may be designated as a featured page) or "change in access rights" (e.g., a page may be locked to prevent editing by anonymous editors). The "*How" construct* for a page modification event may be sentence insertion/update/deletion, link insertion/update/deletion, reference insertion/update/deletion, and reverts (see Table 3). These are actions made by editors that may lead to the modification of a page. *Who* represents the editors of a Wikipedia page. The Wikipedia distinguishes between three types of users: 1) administrators, 2) registered editors, and 3) anonymous editors. *When* refers to the time an event occurs. In the Wikipedia, a timestamp is automatically recorded in the database whenever an event occurs. *Where* in the Wikipedia represents the IP address from which an editor makes a change. *Which* in Wikipedia refers to bots, i.e., software that automatically edits Wikipedia pages. The Wikipedia allows an editor to input *why*, i.e., justifications for a change, in the "comment" field.

Table 3: Application of the W7 model in Wikipedia

Provenance Element	*Application to a Wikipedia article*
What	Creation, modification, destruction, quality assessment, access rights change
How	Sentence insertion/update/deletion, link insertion /update/deletion, reference insertion/update/ deletion, revert (reverting the article to a previous version)
Who	Administrators, registered editors, and anonymous editors
When	Timestamps of the events
Where	IP address of the editor
Which	Software used in editing the page
Why	User comments

Harvesting data provenance in the Wikipedia requires little human effort. The Mediawiki software used by the Wikipedia is set to automatically capture the *what*, *who*, *when*, *where*, and *which.* The *how* provenance can be derived by comparing two versions of a page using the *diff* function. Only the *why* provenance demands manual input. Applying the W7 model to the Wikipedia enables us to harvest provenance of the Wikipedia pages in a structured and comprehensive way. Data provenance in the Wikipedia has widely been used to automatically assess the quality of Wikipedia pages. As an example, [12] suggests metrics such as "rigor" (total number of changes made for the article) and "diversity" (total number of unique editors for the article) as measures of quality. In our recent study [13], we track every action by an editor that affects the life of a Wikipedia article from its creation to the present time. We classify roles by mining the provenance, i.e., various actions carried out by a contributor on an article. We then further identify collaboration patterns based on provenance in terms of *who* does *what.* The collaboration patterns derived from data provenance have been proved to be correlated with data quality of Wikipedia pages.

6. Conclusion and Future Research

In conclusion, the focus of our research is on investigating the semantics of provenance. We have developed a generic provenance model, i.e., the W7 model, to represent these semantics. We identify various elements of provenance such as "what", "where", "when", "who", "how", "which" and "why" and present the semantics of each of these elements. Our W7 model is inspired by theoretical work such as Bunge's ontology as well as our empirical analysis of provenance use in many application domains. It is a generic model of data provenance and is intended to be easily adaptable to represent domain specific provenance requirements. Using the Wikipedia as an example application, we illustrate the use of the W7 model to harvest and track data provenance. We are continuing to use this model to harvest and track provenance in a variety of other application domains.

References

[1] S. Ram and J. Liu, "A Semiotics Framework for Analyzing Data Provenance Research," *Journal of computing Science and Engineering*, vol. 2, pp. 221-248, 2008.

[2] P. Buneman, S. Khanna, and W. C. Tan, "Data Provenance: Some Basic Issues," Proceedings of FSTTCS, New Delhi, India, 2000.

[3] P. Buneman, S. Khanna, and C. T. Wang, "Why and Where: A Characterization of Data Provenance," in *Lecture Notes in Computer Science*, vol. 1973, pp 316-330, Springer, 2001.

[4] J. Frew and R. Bose, "Earth System Science Workbench: A Data Management Infrastructure for Earth Science Products," Proceedings of 13th International Conference on Scientific and Statistical Database Management, Fairfax, VA, 2001.

[5] R. Bose, "A conceptual framework for composing and managing scientific data lineage," Proceedings of 14th International Conference on Scientific and Statistical Database Management, 2002.

[6] M. Greenwood, C. Goble, R. Stevens, J. Zhao, M. Addis, D. Marvin, L. Moreau, and T. Oinn, "Provenance of e-Science Experiments - experience from Bioinformatics," Proceedings of UK e-Science All Hands Meeting, Nottingham, UK, 2003.

[7] M. Grueninger and M. Fox, "Methodology for the Design and Evaluation of Ontologies," Proceedings of Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Quebec, Canada, 1995.

[8] M. Bunge, *Treatise on Basic Philosophy: Vol. 3: Ontology I: The Furniture of the World*. Boston, MA: Reidel, 1977.

[9] J. Sowa, *Conceptual structures: Information processing in Mind and Machine*. Reading, MA: Addison-Wesley, 1984.

[10] S. Ram and J. Liu, "Understanding the Semantics of Data Provenance to Support Active Conceptual Modeling," in *Lecture Notes in Computer Science*, vol. 4521, pp 17-29, Springer-Verlag, 2007.

[11] Y. Simmhan, B. Plale, and D. Gannon, "A Survey of Data Provenance Techniques," Indiana University, Technical Report IUB-CS-TR618, 2005.

[12] A. Lih, "Wikipedia as Participatory Journalism: Reliable Sources? Metrics for Evaluating Collaborative Media as a News Resource," Proceedings of 5th International Symposium on Online Journalism, 2004.

[13] J. Liu and S. Ram, "Who Does What: Collaboration Patterns in the Wikipedia and Their Impact on Data Quality," Proceedings of nineteenth Annual Workshop on Information Technologies and Systems(WITS 2009), Phoenix, Arizona, USA, December, 2009.

Provenance information in biomedical knowledge repositories – A use case

Olivier Bodenreider
Lister Hill National Center for Biomedical Communications
US National Library of Medicine
Bethesda, Maryland, USA
olivier@nlm.nih.gov

***Abstract*—We present a use case for provenance information in biomedical knowledge repositories designed to support applications including information retrieval and knowledge discovery. We show that information about the knowledge sources from which statements are extracted must be recorded in addition to the statement themselves in order to support these applications. While the storage and processing of statements has been greatly facilitated by the emergence of powerful triple stores and the standardization of query languages (e.g., SPARQL), recording and exploiting provenance information (i.e., statements about statements) remains challenging.**

Keywords-provenance information; use case; biomedical knowledge repository

I. INTRODUCTION

Biomedical knowledge is produced and consumed by biomedical researchers and health care practitioners. The biomedical literature (textbooks and journal articles) represents the main source of unstructured biomedical knowledge. Knowledge bases (e.g., model organism databases annotated to the Gene Ontology) result from the curation of the primary literature in an attempt to make knowledge more accessible and actionable. Finally, ontologies represent the ultimate form of computationable knowledge, but are often limited in scope and tend to focus on definitional, as opposed to assertional knowledge. Rich sets of metadata have been defined and are collected along with the primary data, using standards such as the Dublin Core for the literature [1] and MIAME for gene expression data [2].

Attempts to make knowledge accessible to agents in addition to humans have focused on the extraction of knowledge from unstructured sources, as well as the interoperability of structured knowledge sources. Text mining techniques are used to extract “predications” (i.e., statements) from text, for example in the Semantic Medline project [3]. Metadata are often stored in an *ad hoc* format in order to help associate predications with the articles from which they have been extracted. The Linked Data initiative [4] promotes the use of RDF (the Resource Description Framework) [5] to link biomedical datasets, with a strong emphasis on shared URIs (Uniform Resource Identifiers) in order to relate concepts sharing the same identifiers across datasets. In most cases, however, such repositories of linked data have little metadata, in part because simple RDF representations make it difficult to represent statements about statements. These examples illustrate the difficulty of representing – let along computing with – provenance information in biomedical knowledge repositories.

One such repository is being created as part of a research project at the National Library of Medicine [6]. It includes knowledge extracted from Medline abstracts by text mining tools, structured knowledge derived from existing knowledge bases (e.g., NCBI’s Entrez system [7]) and terminological knowledge from the Unified Medical Language System [8]. In this project, we are also interested in recording and processing information about the statements (e.g., location in the information space and time annotations), in order to support applications including enhanced information retrieval, multi-document summarization, question answering and knowledge discovery.

In this paper, we briefly examine the types of metadata required in the context of our biomedical knowledge repository. In other words, we look at provenance information through the use case of this knowledge repository and discuss some of the issues encountered along the way and challenges ahead.

II. PROVENANCE INFORMATION IN TYPICAL APPLICATIONS

The four applications our repository has been designed to support require various types of provenance information [6]. Common to all applications is the requirement that the origin of any statement be identifiable (e.g., from which knowledge sources was it extracted?, using which extraction techniques, if any?) Because biomedical knowledge evolves over time, it is also indispensable that some time annotation be associated with each statement (e.g., date of publication of the article from which the statement was extracted, date when the statement was curated in a given knowledge base, or date when a given ontology was last revised.) When available, the degree of confidence associated with a given statement should also be recorded. Confidence can be indicated by the tools used for the production of the statements (e.g., text mining tools) or approximated through frequency information. In the following discussion, the association between types of applications and types of provenance information is somewhat arbitrary and presented essentially for illustrative purposes.

A. *Information retrieval*

The enhanced information retrieval envisioned goes beyond keyword or concept searches and supports searches based on relations. For example, finding all the documents in which the statement "IL-13 inhibits COX-2" is found. Like with traditional search engines, there is a need for associating a document identifier with a given statement. The list of all document identifiers associated with a given statement forms the basic index in such a system. Conversely, indexing a document consists in associating this document with all the statements extracted from it by the text mining tool.

B. *Multi-document summarization*

In addition to the basic index required for information retrieval, information is needed for the prioritization of statements (among all relevant statements) in multi-document summarization. Statements below a certain threshold of confidence may be hidden as a way of restricting the amount of information provided in the summary. Low confidence can be indicated by a text mining tool, for example, when ambiguity in natural language cannot be resolved by the system.

C. *Question answering*

In question answering applications, answers must be collected from reputable sources. Here, statements from the biomedical knowledge repository are used as potential answers to input questions (e.g., what genes does IL-13 inhibit?) Not only must the origin of the statement be present as for information retrieval and summarization purposes, but additional metatada associated with the document must also be available (e.g., does this document come from a reputable source, such as an article about randomized clinical in the case of clinical effectiveness statements? Does this statement come from a document published/a knowledge base revised recently?) The distinction here is between metadata directly associated with the statement (e.g., document identifiers), and metadata about the documents themselves, indirectly associated with the statement (reputability of the source, publication date).

D. *Knowledge discovery*

Information retrieval, summarization and question answering can be thought of as exploiting a static repository, mostly through look-ups in the repository, with no (or limited) need for inference. In contrast, knowledge discovery processes aim at inferring new knowledge from patterns of statements in the repository. Inference is one major technique for deriving new knowledge from existing knowledge. Production rules provide a simple mechanism for formalizing inference and rule engines are implemented in many systems that store statements. Knowledge discovery systems require not only production rules and rule engines for the production of entailed statements from rules, but also the production of the metadata associated with the entailed statements (i.e., inferred provenance information). Provenance for both asserted and inferred statements is required so that the universe of statements can be restricted to degree of confidence, specific time periods or sources. For example, can a path be found in a graph, directly (asserted links) or indirectly (inferred links), between two nodes (e.g., between a disease and a drug), when links are restricted to a specific source? The issue here is not only to associate provenance information to asserted statements, but also to compute such information for inferred statements as well.

III. Issues and Challenges

Limitation of naïve implementations. RDF provides a simple mechanism for recording statements about statements through "blank nodes" [5]. A blank node can be used as an identifier for the statement, each component of which – subject, predicate and object – is then linked to it through predicates such as *hasSubject*, *hasPredicate* and *hasObject*. Similarly, provenance information can be linked to the statement identifier (e.g., link to the article from which it is extracted through a *hasSource* predicate). This mechanism, called reification, is inefficient as it increases the number of triples required for implementing a statement (at least one for the relation of the blank node to each of the three components of the original statement). Scalability issues are thus likely with large biomedical repositories (typically several hundred million asserted statements). Moreover, by significantly increasing the complexity of queries, reification also puts an unnecessary cognitive burden on the user.

Lack of support for provenance information in mainstream triple stores, There is currently no support for exploiting provenance information in off-the-shelf triple stores. Support is generally provided for named graphs in so-called "quad stores", but named graphs hardly provide the level of granularity needed for provenance information required in biomedical applications. Beside reification, SPARQL does not offer support for seamless processing of provenance information. There is a need for a standardization of emerging models of provenance (e.g., OPM [9]) and their efficient implementation in triple stores.

Limitations for the applications. The emerging paradigm of linked data and mashups had met tremendous enthusiasm in the biomedical community [10, 11]. At this early stage, the possibility of easily integrating disparate datasets still outweighs the lack of fine control over constraints on these data sources. However, when applications mature beyond answering questions such as "is there a path between this and that?" to restricting graph traversal with constraints specific to properties of the links (statements, not simply predicates), the lack of standard models and implementations for provenance information will appear as a serious limitation.

Acknowledgment

This research was supported in part by the Intramural Research Program of the National Institutes of Health (NIH), National Library of Medicine (NLM).

References

[1] "The Dublin Core® Metadata Initiative," http://www.dublincore.org/.

[2] A. Brazma, "Minimum Information About a Microarray Experiment (MIAME)--successes, failures, challenges," ScientificWorldJournal, vol. 9, 2009, pp. 420-423.

[3] M. Fiszman, et al., “Automatic summarization of MEDLINE citations for evidence-based medical treatment: a topic-oriented evaluation,” J Biomed Inform, vol. 42, no. 5, 2009, pp. 801-813.

[4] “Linked data,” http://linkeddata.org/.

[5] W3C, “Resource Description Framework (RDF),” http://www.w3.org/RDF/.

[6] O. Bodenreider and T.C. Rindflesch, Advanced library services: Developing a biomedical knowledge repository to support advanced information management applications, Technical report, Lister Hill National Center for Biomedical Communications, National Library of Medicine, 2006.

[7] E.W. Sayers, et al., “Database resources of the National Center for Biotechnology Information,” Nucleic Acids Res, vol. 37, no. Database issue, 2009, pp. D5-15.

[8] O. Bodenreider, “The Unified Medical Language System (UMLS): Integrating biomedical terminology,” Nucleic Acids Res, vol. 32 Database issue, 2004, pp. D267-270.

[9] L. Moreau, et al., “The Open Provenance Model: An Overview,” Provenance and Annotation of Data and Processes, Lecture Notes in Computer Science 5272, Springer, 2008, pp. 323-326.

[10] F. Belleau, et al., “Bio2RDF: towards a mashup to build bioinformatics knowledge systems,” J Biomed Inform, vol. 41, no. 5, 2008, pp. 706-716.

[11] K.H. Cheung, et al., “Semantic mashup of biomedical data,” J Biomed Inform, vol. 41, no. 5, 2008, pp. 683-686.

On User Views in Scientific Workflow Systems

Susan Davidson
University of Pennsylvania
Email: susan@cis.upenn.edu

Yi Chen, Peng Sun
Arizona State University
Email: {yi, psun5}@asu.edu

Sarah Cohen-Boulakia
Universite Paris-Sud
Email: cohen@lri.fr

***Abstract*—An increasing number of scientific workflow systems are providing support for the automated tracking and storage of provenance information. However, the amount of provenance information recorded can become very large, even for a single execution of a workflow – [6] estimates a ten-fold blowup of the size of the original input data. There is therefore a need to provide ways of allowing users to focus their attention on meaningful provenance information in provenance queries. We highlight recent work in this area on *user views*, showing how they can be efficiently computed given user input on relevance, or and how pre-existing views can be corrected to provide accurate provenance information. We also discuss how to search a repository of workflow specifications and their views, returning workflows at an appropriate level of complexity with respect to a hierarchy of views.**

I. Introduction

Scientific workflow management systems (*e.g.*, myGrid/Taverna [11], Kepler [5], VisTrails [9], and Chimera [8]) have become increasingly popular as a way of specifying and executing data-intensive analyses. To ensure reproducibility of results and track the large amount of final and intermediate data products that are produced in a workflow execution, many of these systems are beginning to provide support for managing and querying provenance information.

However, the amount of provenance information recorded even for a single execution of a workflow can be extremely large; [6] estimates a ten-fold blowup of the size of the original input data. While databases are adept at storing and efficiently answering queries over large amounts of information, users are not adept at *assimilating* large amounts of information. It is therefore important to develop techniques to minimize the cognitive overload resulting from provenance queries, providing provenance information that is relevant to users.

As an example, consider the workflow specification (a.k.a workflow definition or schema) in Fig. 1 (a)[1], which describes a common analysis in molecular biology: *Phylogenomic inference of protein biological function*. This workflow first takes in a set of entries selected by the user from a database (such as GenBank), and formats these entries to extract a set of sequences, and, possibly, a set of annotations (M1). An alignment is then created (M3), and the result formatted (M4). The user may also be interested in rectifying the alignment (M5). M3 to M5 are repeated until the biologist is satisfied with the result obtained. The user may also inspect the annotations provided by GenBank (M2) and generate a set of curated annotations; new user input is needed for this. The annotations are then formatted (M8) to be taken as input to the phylogenetic tree reconstruction task (M7). Other annotations are also considered: M6 takes in annotations from the user's lab and formats them to be taken as input to M7. From the annotations produced by M8 (and possibly M6) together with the alignment produced by M4, M7 provides a phylogenetic tree labeled with functional annotations. Note that a number of these tasks or *modules* (e.g. M1, M4, M8) involve formatting and are not central to the scientific goal of the experiment, and that edges represent the precedence and potential *dataflow* between modules during an execution.

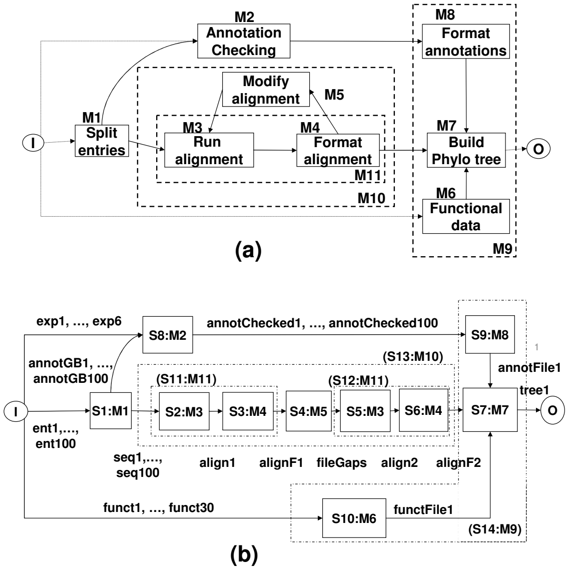


Fig. 1. Phylogenetic workflow specification (a) and run (b)

The result of executing a scientific workflow is called a *run*. As a workflow executes, data flows between module *invocations* (or *steps*). For example, a run of the phylogenomics workflow is shown in Fig. 1(b). Nodes represent steps that are labeled by a unique step identifier and a corresponding module name (*e.g.*, S1:M1). Edges denote the flow of data between steps, and are labeled accordingly (*e.g.*, data objects `ent1,...,ent10` flow from input I to the first step S1). Note that loops in the workflow specification are always unrolled in the run graph, *e.g.*, two steps S3 and S6 of M4 are shown in the run of Fig. 1(b).

[1] The reader should ignore the dotted boxes for now.

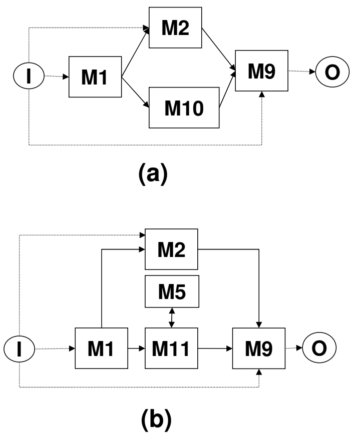


Fig. 2. Joe's (a) and Mary's (b) user views.

Data provenance in workflows is typically captured as a set of dependencies between data objects [7]. Essentially, the graph of Fig. 1(b) becomes one in which the nodes are data; each edge is labeled with the module execution which produced the data at its start and taking as one of its inputs the data at its end. Note that module names are repeated for every input-output pair. Thus, a query of the provenance of the final data product tree1 would return a graph similar to that in Fig. 1(b), which (even for this simple example) is quite large.

In this paper, we discuss a technique called *user views* which uses composite modules, i.e. modules which may themselves contain subworkflows, to hide portions of a workflow run and thus simplify the workflow specification as well as provenance information. Section II shows how users can indicate which modules are relevant within a specification, and have a user view automatically created around those relevant modules. Section III discusses how to refine pre-defined views to ones which correctly portray the provenance relationships between the input and output of composite modules. Section IV shows how a database of specifications and their views can be searched using keyword queries, returning workflows at an appropriate level of complexity with respect to a hierarchy of views.

II. User Views

As illustrated in Fig. 1 (b), a workflow run may comprise many steps and intermediate data objects, and therefore the amount of information provided in response to a provenance query can be overwhelming. A user may therefore wish to indicate which modules in the workflow specification are *relevant*, and have provenance information presented with respect to that user view. To do this, composite modules are used as an abstraction mechanism [3].

As an example, for the workflow in Fig. 1 (a), user Joe might indicate that M2: *Annotation Checking*, M3: *Run Alignment*, and M7: *Build Phylo Tree* are relevant to him. In this case, composite modules M9 and M10 would automatically be constructed (indicated by dotted boxes labeled M9 and M10 in Fig. Fig. 1(a)), and Joe's user view would be {M1, M2, M9, M10} as shown in Fig. 2(a). When answering provenance queries with respect to a user view, only data passed between modules in the user view would be visible. Data and module executions internal to a composite module in the view would be hidden; this corresponds to hiding the module executions and data shown within the dotted boxes M9 and M10 in Fig. 1(b). Thus, the provenance for tree1 presented according to Joe's user view would no longer include annotFile1, functFile1 (both pieces of data are hidden inside M9), or align1, alignF1, fileGaps, align2 (hidden inside M10).

Views are individualized according the user's interests. For example, another user, Mary, may be interested in modules M2, M3 and M7 (like Joe) but additionally interested in M5: *Modify alignment*. Mary's user view would therefore be constructed as {M1, M2, M5, M9, M11} (shown in Fig. 2(b)), and her view for the provenance of tree1 would expose alignF1 and fileGaps (unlike Joe's view) while hiding annotFile1, functFile1, align1, and align2.

More formally, a *user view* is a partition of the workflow modules. It induces a "higher level" workflow in which nodes represent composite modules in the partition (*e.g.*, M9 and M10) and edges are induced by dataflow between modules in different composite modules (*e.g.*, an edge between M10 and M9 is induced by the edge from M4 to M7 in the original workflow). Provenance information is then seen by a user with respect to the flow of data between modules in his view.

In the ZOOM system [3], [2], user views are constructed automatically given input on what modules the user finds relevant such that (1) a composite module contains at most one relevant (atomic) module, thus assuming the "meaning" of that module; and (2) no data dependencies (either direct or indirect) are introduced or removed between relevant modules. In this way, the meaning of the original workflow specification is preserved, and only relevant provenance information is provided to the user.

An interesting theoretical question is whether there are efficient algorithms for constructing a user view which obeys conditions (1) and (2) above and which are as small as possible, i.e. in which the number of composite modules is *minimized*. We call such a view *optimum*. It turns out that whether or not the optimum user view can be constructed depends on the graphical structure of the workflow specification [4]: For specifications that are general graphs, regardless of the number of distinct modules in the input workflow and the structure of interaction between them, the number of composite modules can be exponentially large in the number of relevant modules in an optimum user view for the specification. However, for *series-parallel* workflow graphs [14] there is a simple, linear time algorithm for constructing an optimum user view for a given specification [4]. A study of scientific workflow specifications collected from our collaborators as well as those found at myexperiment.org has shown that over 80% of scientific workflow specifications are series-parallel graphs,

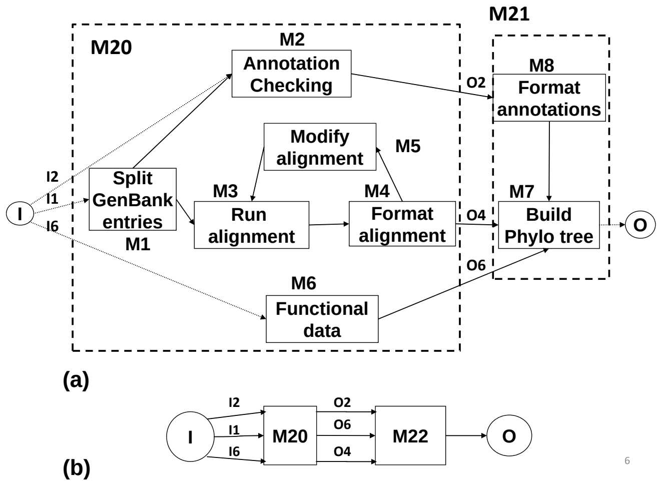


Fig. 3. Unsound User View (a) and Projected View (b)

and those that are not can be easily transformed by adding control points.

III. Correcting Composite Modules for Provenance

The previous section focused on how to create views give user input on what modules were relevant. However, composite modules are frequently used for purposes of modularization, abstraction and reuse when specifying workflows, and therefore workflow views may already exist.

However, unless a view is carefully designed, it may not preserve the dataflow between modules in a workflow, and thus can be misleading and lead to incorrect provenance analysis. For example, consider the view defined in Fig. 3(a), and suppose a user would like to determine the provenance of output O2 of module M20 in the projected view in Fig. 3(b). Based on the abstracted provenance graph, she would believe that inputs I2, I1 and I6 are all involved. However, there is no path between I6 and O2 in the original workflow; only I1 and I2 are in the provenance of O2.

Ideally, a view should preserve all the data dependencies between composite tasks in the workflow, without adding or removing dependencies. We call such a view *sound* with respect to provenance. In our example, the view in Figure 3(b) indicates a data dependency path between I6 and O2, which does not exist in the original workflow in Figure 3(a), and thus unsound. Although it would seem natural to design views which are sound, our survey of workflow designs in a well-curated workflow repository [1] revealed several unsound views. The goal of the WOLVES system [13] is therefore to diagnose and correct unsound views.

We prove that a view is sound if every composite task in the view is sound. Two alternatives can be pursued for correcting an unsound task: Splitting it into multiple smaller tasks, or merging it with other tasks. Note that splitting composite tasks refines the initial view to a lower level and provides more provenance information. In contrast, merging tasks loses information, as tasks that are important to the user may be invisible after the merge. Therefore, in WOLVES we focus on techniques that resolve an unsound view by splitting unsound composite tasks rather than merging them. For example, we could split M20 into three composite tasks: {M1}, {M2}, {M6}, {M3, M4, M5}.

Our goal is to correct an unsound view by splitting its unsound composite tasks to a minimal number of tasks, each of which is sound. However, this problem is NP-hard by reduction from the independent set problem. To efficiently tackle this problem, we propose two optimality criteria: weak local optimality and strong local optimality. A weak local optimal solution is one in which no two tasks in the resulting view can be merged into a sound task, and strong local optimal solution is one in which no set of two or more tasks in the view can be merged. Weak local optimality can be achieved with an $O(n^2)$ algorithm, and strong local optimality with an $O(n^3)$ algorithm, where n is the number of tasks in the workflow. The proposed algorithms are much more efficient than the algorithm which produces an optimal solution. The strongly local optimal algorithm often has comparable processing efficiency to the weakly local optimal algorithm, and produces views that are comparable to the optimal one.

IV. Searching Workflows through Views

An increasing number of workflow specifications and their views are being stored, either as part of a local workflow system or collected to form a community repository (e.g. myexperiment.org [10]). It is therefore important for workflow designers to be able to *search* these repositories and then reuse, include or revise the retrieved workflows to simplify the design of a new workflow.

Techniques for finding workflows of interest are currently limited to keyword searches based on the name of the workflow or tags explicitly associated with the workflow, and the result is a set of workflows shown at an arbitrary level of detail. However, by using a notion of *hierarchical* user views, this rudimentary way of searching for workflows of interest can be significantly improved. In a hierarchical user view, composite modules may themselves contain composite modules, and names can be associated with each atomic or composite module.

For example, suppose that a user would like to make a sauteed dish which uses chicken breast and coconut, and needs a recipe. She would then issue a keyword query Q, "*chicken breast, coconut milk, saute*" on a repository of recipes (workflows) to search for relevant recipes.

Now suppose that Fig. IV is one of the relevant recipes in the repository, where all query keywords have matches. Obviously, returning the entire workflow hierarchy as a query result, i.e. the one in which all the composite modules are exposed, is difficult to understand since too much irrelevant information is exposed to the user. We therefore need to find ways of exposing relevant information in our query results.

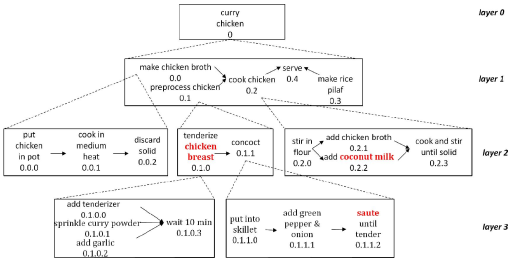
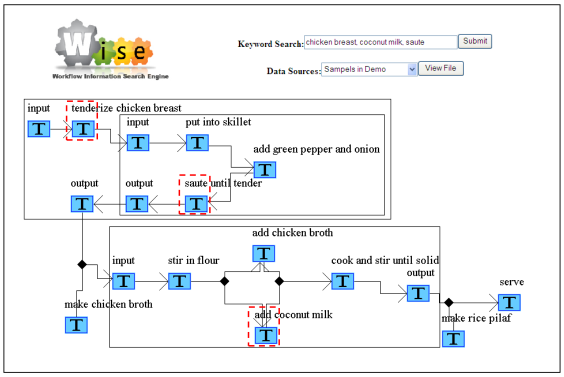


Fig. 4. Recipe Workflow Hierarchy



Keyword Search: chicken breast, coconut milk, saute

Data Sources: Sampels in Demo

Fig. 5. Query Result for Q:{chicken breast, coconut milk, saute}

The immediate question is how to define search results when users issue keyword queries on a repository of workflow hierarchies. Recall that much research has been done on keyword search on graph-structured data (e.g., relational databases) and tree-structured data (e.g., XML), where a result is defined as a smallest data tree that contains the query keywords. Unfortunately, this definition of a query result is not appropriate for workflow search as the result is not guaranteed to capture the dependencies and dataflow among tasks that contain keyword matches.

For our example, a "good" query result is shown in Fig. IV, which is a *query-driven view* that visualizes keyword matches shown in dashed rectangles together with their dataflow. For example, after `saute (the chicken) until tender`, we `stir (it) in flour` and then `add coconut milk`. Note that the dataflow paths among these tasks are not explicitly shown in the workflow hierarchy in Fig. IV, but are derived. Also, expansion edges that represent irrelevant views are avoided.

Supporting keyword search on workflow hierarchies poses new challenges beyond keyword search on relational and XML data. First we need to define meaningful query results. We propose that a query result should be a *minimal query-driven view* of the workflow hierarchy that contains all keyword matches, which is a graph containing all matches and dataflow edges among them. Second, we must design efficient techniques to generate such query results.

To address these challenges, we have developed WISE [12], a Workflow Information Search Engine (available at http://wise.asu.edu/). WISE allows users to search a repository of workflow hierarchies using simple keywords, and returns concise and informative query results. The query results can be efficiently and dynamically synthesized by exploiting indexes and labeling schemes. To the best of our knowledge, this is the first work that supports keyword search on repositories of workflow hierarchies and returns query results capturing the dataflows among tasks matching keywords.

V. Conclusion

User views, in which composite modules are used to hide portions of a workflow specification or execution, are a useful abstraction for simplifying information. We have shown how they can be used to create individualized views of provenance information by having users indicate which modules are relevant, and how existing user views can be corrected to accurately capture provenance information. We have also discussed how they can be used to simplify the result of a keyword search over a repository of workflow specifications.

We are currently pursuing several other interesting applications of user views, including provenance query languages, and secure views of workflows and their executions.

References

[1] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludäscher, and S. Mock. Kepler: An extensible system for design and execution of scientific workflows. In *SSDBM*, pages 423–424, 2004.

[2] O. Biton, S. C. Boulakia, and S. B. Davidson. Zoom*userviews: Querying relevant provenance in workflow systems. In *VLDB*, pages 1366–1369, 2007.

[3] O. Biton, S. C. Boulakia, S. B. Davidson, and C. S. Hara. Querying and managing provenance through user views in scientific workflows. In *ICDE*, pages 1072–1081. IEEE, 2008.

[4] O. Biton, S. B. Davidson, S. Khanna, and S. Roy. Optimizing user views for workflows. In *ICDT '09: Proceedings of the 12th International Conference on Database Theory*, pages 310–323, 2009.

[5] S. Bowers and B. Ludäscher. Actor-oriented design of scientific workflows. In *Int. Conf. on Concept. Modeling*, pages 369–384, 2005.

[6] A. Chapman, H. V. Jagadish, and P. Ramanan. Efficient provenance storage. In *SIGMOD Conference*, pages 993–1006, 2008.

[7] S. B. Davidson, S. C. Boulakia, A. Eyal, B. Ludäscher, T. M. McPhillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4):44–50, 2007.

[8] I. T. Foster, J.-S. Vöckler, M. Wilde, and Y. Zhao. Chimera: Avirtual data system for representing, querying, and automating data derivation. In *SSDBM*, pages 37–46, 2002.

[9] J. Freire, C. T. Silva, S. P. Callahan, E. Santos, C. E. Scheidegger, and H. T. Vo. Managing rapidly-evolving scientific workflows. In *IPAW*, volume 4145 of *LNCS*, pages 10–18. Springer, 2006.

[10] myExperiment. http://www.myexperiment.org/workflows.

[11] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, R. Greenwood, K. Carver, M. G. Pocock, A. Wipat, and P. Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(1):3045–3054, 2003.

[12] Q. Shao, P. Sun, and Y. Chen. Wise: a workflow information search engine. In *ICDE*, 2009.

[13] P. Sun, Z. Liu, S. B. Davidson, S. N., and Y. Chen. WOLVES: Achieving Correct Provenance Analysis by Detecting and Resolving Unsound Workflow Views. In *PVLDB*, 2009.

[14] J. Valdes, R. E. Tarjan, and E. L. Lawler. The recognition of series parallel digraphs. *SIAM J. Comput.*, 11(2):298–313, 1982.