1-1-2018

Scalable And Secure Provenance Querying For Scientific Workflows And Its Application In Autism Study

Fahima Amin Bhuyan
Wayne State University,

SCALABLE AND SECURE PROVENANCE QUERYING FOR SCIENTIFIC WORKFLOWS AND ITS APPLICATION IN AUTISM STUDY

by

FAHIMA AMIN BHUYAN

DISSERTATION

Submitted to the Graduate School

of Wayne State University

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2018

MAJOR: COMPUTER SCIENCE

Approved By:

_____

Advisor Date

_____

_____

_____

_____

DEDICATION

Dedicated to my beloved husband Dr. Mohammad Shafkat Amin,

my children Faaris Shadmehr Amin & Shehreen Farida Amin and

my mother Farida Begum.

ACKNOWLEDGEMENTS

I would first like to thank Almighty Allah for giving me the opportunity to pursue a Ph.D. degree in a fascinating field of Computer Science, for giving me a passion for research, for protecting me, for answering my prayers, and for giving me the strength to persevere. I firmly believe, coming to Wayne State University to work with my advisor Dr. Shiyong Lu was a major decision in my life.

I would also like to express my profound and sincere gratitude to my advisor Dr. Shiyong Lu, for his constant encouragement, guidance, and support throughout my Ph.D. studies. Dr. Lu's vision and suggestion have enabled me to remain focused and to succeed in my studies. I am deeply thankful for Dr. Lu's empathy and kindness. I am also very grateful to all my dissertation committee members: Dr. Robert Reynolds, Dr. Alexander Kotov, and Dr. Jia Zhang, for being on my dissertation committee and for providing their helpful feedback, valuable comments, and constructive suggestions.

I would also like to thank my bright academic colleagues from the Big Data Research Laboratory: Ishtiaq Ahmed, Changxin Bai, Dr. Andrey Kashlev, Dr. Aravind Mohan, Dr. Dong Ruan, Dr. Mahdi Ebrahimi for their academic cooperation and close friendship, as well as alumni Dr. Artem Chebotko, Dr. Cui Lin, Dr. Chunhyeok Lim.

I am especially thankful to my beloved husband, Dr. Mohammad Shafkat Amin. Throughout my study he has always been supportive, helpful, friendly, and has protected me from all the challenges life has given, with a smiling face. It was his faith in me which kept me going through rough times. I am thankful to my children Faaris Shadmehr Amin and Shehreen Farida Amin, who have been incredibly supportive throughout my studies. Without their immense sacrifices, it would not have been possible for me to complete my pro-

gram.

My sincere gratitude goes to my mother Farida Begum and my father Dr. Aminul Haque Bhuyan. I am truly thankful to my two sisters Saima Amin Bhuyan and Shaila Amin Bhuyan, for their unconditional love, encouragement and support since childhood. My special thanks to my niece, Aura Simrah Choudhury.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

Provenance is information about the history, origin, derivation, and context of data. Provenance management has become critical in various data systems such as database, workflow, and web systems [29, 46, 99]. For example, all major scientific workflow systems [122, 133, 114, 48, 44] support provenance. The past few years have witnessed the much progress on provenance standardization, including OPM [100] and PROV [96], and active community engagement in the provenance challenge series [2]. Provenance is useful to interpret analysis of results, to repeat a scientific discovery, and to trace errors in data. Provenance is also useful to answer data lineage queries and decide the trustworthiness of a data product.

With the advent of internet scale data, the complexity of scientific workflows and ensuring provenance management systems have grown significantly. Traditional approaches to provenance queries do not adapt adequately to account for this scale, necessitating scalable provenance query and data management systems. As science becomes more and more interdisciplinary and collaborative, the notion of *collaborative scientific workflows* has been proposed to address the increasing need of collective data analytics using the scientific workflow paradigm [90, 125, 132, 131, 55]. In such collaborative environments, adequate access control policies are necessary to safeguard sensitive information and facilitate privacy-aware sharing of workflows, data products, and provenance information among collaborating parties [36].

It has been well recognized that the general provenance security problem is critical for modern scientific workflow systems [36, 92]. Unauthorized access to provenance might disclose confidential information about the related data products. The code for collecting, querying and mining of provenance can be compromised, forged, or replayed by intruders. The linkages among data products, provenance, and workflow specifications can be severed or forged in a malicious environment. Compromised provenance can lead to misinterpretation of analysis results, unintentional errors, and can compromise the confidentiality

of related datasets. In this research, we focus on the confidentiality of provenance so that provenance is accessible only to authorized users. This is important because provenance often encodes the detailed protocol of a scientific experiment and constitutes the intellectual property of the respective stakeholders. Various access control mechanisms have been proposed for the protection of the confidentiality of scientific workflow provenance [36, 92]. This is significantly underscored in workflows that process sensitive health or financial information.

In contrast to business workflows, which are relatively stable over time, scientific workflows tend to evolve rapidly as scientists frequently generate, explore, and test multiple hypotheses about a scientific problem simultaneously [57]. For example, an existing workflow w_1 might be extended with additional sub-workflows into workflow w_{11} to perform the more advanced scientific analysis. The sub-workflow w_{11} can be further evolved into w_{111} and w_{112} with additional sub-workflows, tasks, and data channels. All these workflows can be used simultaneously to explore different hypotheses or to perform different but related scientific analysis. As a result, it is important to evolve the corresponding access control policies as well. In dealing with such large sets of evolving policies, manually checking the quality of each policy becomes impractical, and automated analysis algorithms for access control policies of scientific workflow provenance are necessary to ensure the correctness and performance of the policy enforcement engine.

One domain, that could benefit from automation, computational power, and data integrity that comes with scientific workflow, is in therapeutic treatment planning in autism spectrum disorders. Scientific workflows could seamlessly integrate disparate data sources, consolidate wisdom of the crowd, and harness the power of machine learning to aid service providers with their therapy and service planning. This, however, is a data-intensive process and requires an underlying provenance system that is flexible and scalable enough to cater to a deluge of data. Heterogeneity in data collections with disparate and sparse data sources make composing the workflow challenging and maintaining the provenance

system complex. In this research, we demonstrate the complexity and scale requirement of next-generation provenance systems by introducing a scientific workflow that consolidates heterogeneous data sources to predict efficacy of therapeutic services of autism spectrum disorder. This big scale data, inherent in such domains, has brought forth a new research direction.

With autism on the rise and a lack of retrospective study in management and alignment of behavior intervention and educational plan and how it affects the manifestation of ASD symptoms, parents, and the community at large are finding it challenging to individualize forward-looking goals and educational plans for kids with ASD. There is a flurry of anecdotal evidence, parent, caregiver and therapist data, that, if mined retrospectively, can lead to better understanding of how appropriate goals, given a child's traits and needs, could potentially mitigate autistic behavior and result in overall improvement. In this work, we aim to delineate a scientific workflow framework that can be employed to apply data-mining techniques to improve predictability in the domain of autism spectrum disorder.

In Big data research, the provenance of big data [60, 59, 63, 33, 7] plays a major role. Big data provenance deals with many research challenges and open issues which need deep investigation like confidentiality of the data provenance process, secure and privacy-preserving big data provenance, flexible big data provenance query tools, etc [45]. One of the platforms used for handling big data provenance is Pig. Pig is a platform for analyzing large datasets on top of Hadoop, with a rich, multi-valued and nested data model. Pig's language, Pig Latin is a comprehensive imperative query language that lets us express data transformations such as filtering datasets, merge them and apply functions to a groups of records. It is simple to understand data flow language, yet a fast iterative language with strong MapReduce compilation engine. Pig Latin gives a level of abstraction from MapReduce procedural model by providing join and filter like relational style operators which do not have out-of-the-box analogs in MapReduce framework.

The remaining chapters of the dissertation are organized as follows: In Chapter 1,

we present the core concepts of provenance models along with scalable query and secure access control policies. In Chapter 2, we review research background on reasoning and analysis of secure and scalable workflow provenance system. In Chapter 3, we propose our query language and create the constructs to facilitate complex queries in a visual workflow style. In Chapter 4, we propose secure access control policies and validate it with policy quality requirements. In Chapter 5, we explore the use of data mining approaches in the DATAVIEW workflow system to understand the initial feasibility of our approach. Finally, in Chapter 6, we draw concluding remarks of the dissertation and provide the directions for future work.

CHAPTER 1 PROBLEM FORMULATION

Provenance refers to the information about the derivation history of a data product [46, 35]. It is important for evaluating the quality and trustworthiness of a data product and ensuring the reproducibility of scientific discoveries [64, 41]. Much research has been done on storing and querying scientific workflow provenance - a provenance that is produced in the execution of data-centric scientific workflows [35, 14]. Since scientific workflow provenance are essentially directed acyclic graphs, a leading trend of querying models is the graph-based querying models, represented by two provenance graph query languages: OPQL [85] and QLP [14]. While QLP provides query constructs for querying both structure and lineage information in provenance graphs, OPQL, in addition, supports the Open Provenance Model [100], a community-driven data model, which captures main aspects of the workflow provenance and does not enforce a particular physical representation of the provenance data.

1.1 The OPM Provenance Model

The Open Provenance Model (OPM) is the first community-based provenance model that supports the digital representation and inference of provenance [91]. In this model, provenance data is modeled as directed acyclic graphs in which there are three types of nodes, *Artifact*, *Process*, and *Agent*, and five types of edges, *Used*, *WasGeneratedBy*, *WasControlledBy*, *WasTriggeredBy*, and *WasDerivedFrom*. In the domain of big data workflows, for example, *artifacts* are mapped to data products, *processes* are mapped to workflow tasks, and *agents* are mapped to data scientists who perform the execution of a workflow task. Meanwhile, *Used* links a workflow task to an input data product; *WasGeneratedBy* links an output data product to a workflow task; *WasControlledBy* links a workflow task to the data scientist who performs its execution; *WasTriggeredBy* links a downstream workflow task to an immediate upstream workflow task, and *WasDerivedFrom* links an output data product of a workflow task to an input data product of the same workflow task. The OPM model is illustrated in Figure 1. The OPM model specification does not include querying languages

for provenance, which is the motivation for efforts including OPQL and QLP. Formally, a provenance graph PG = (N, E) consists of:

- a set of nodes $N = A \bigcup P \bigcup AG$, where A is a set of artifacts, P is a set of processes, and AG is a set of agents;

- a set of directed edges E = $E_u \bigcup E_g \bigcup E_d \bigcup E_t \bigcup E_c$

  where i) $E_u \subseteq$ P $\times$ A and (p, a) $\in E_u$ states that process p *used* artifact a.

  ii) $E_g \subseteq$ A $\times$ P and (a, p) $\in E_g$ states that artifact a *was generated by* process p.

  iii) $E_d \subseteq$ A $\times$ A and (a_1, a_2) $\in E_d$ states that artifact a_1 *was derived from* artifact a_2.

  iv) $E_t \subseteq$ P $\times$ P and (p_1, p_2) $\in E_t$ states that process p_1 *was triggered by* process p_2.

  v) $E_c \subseteq$ P $\times$ AG and (p, ag) $\in E_c$ states that process p *was controlled by* agent ag.

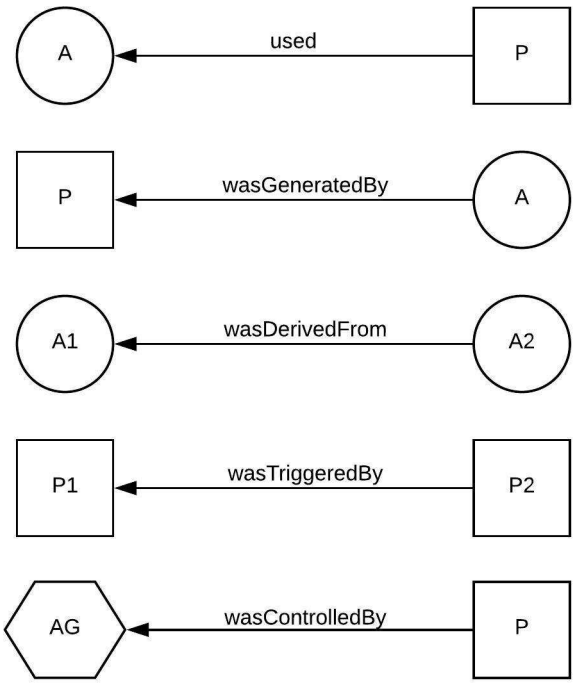


Figure 1: OPM Provenance Model.

1.2 The PROV-DM Provenance Model

Another provenance data model was introduced in 2013, named PROV-DM. Fig. 2 represents PROV-DM provenance model. The PROV provenance model is more generic and domain-agnostic. The PROV set of specifications is designed to promote easily exploited nodes and relations for modeling specific domains. This provenance model offers interoperability across diverse provenance management systems and accommodates data generation from a diverse data sources.

PROV provenance model has flexibility when it deals with attributes. Most of the provenance statements are annotated with optional attributes. This provenance model has a mechanism for asserting provenance of provenance specified as 'bundles'. To avoid to make the model unnecessary complex, PROV-DM does not model uncertainty. In Table. 1, we can see the types and relations in PROV-DM.

The PROV-DM provenance model is the conceptual data model that forms a basis for the W3C provenance (PROV) family of specifications [61], which currently contains four recommendations and eight notes. PROV-DM is one of the four recommendations, besides PROV-O, the PROV ontology, an OWL2 ontology allowing the mapping of the PROV data model to RDF; PROV-NA, a notation for provenance aimed at human consumption; PROV-CONSTRAINTS, a set of constraints applying to the PROV-DM data model. Like OPM, PROV-DM also models provenance as a directed acyclic graph, in which there are three types of nodes, *Entity*, *Activity*, and *Agent*, and seven types of edges, *Used*, *WasGeneratedBy*, *WasAssociatedWith*, *WasInformedBy*, *WasDerivedFrom*, *ActedOnBehalfOf*, and *WasAttributedTo*. The number of types of nodes in PROV-DM is the same as in OPM, but their names are different. In PROV-DM, *Artifact* becomes *Entity*, *Process* becomes *Activity*, and *Agent* remains the same. Moreover, two edge types are introduced: *ActedOnBehalfOf*, to model delegation relationships between agents, and *WasAttributedTo*, to model attribution of entities to agents; and two edge types are renamed: *WasTriggeredBy* was renamed to *WasInformedBy*, and *WasControlledBy* to *WasAssociatedWith*. In the domain of big data

workflows, for example, *entities* are mapped to data products, *activities* are mapped to workflow tasks, and *agents* are mapped to data scientists who perform the execution of a workflow task. Meanwhile, *Used* links a workflow task to an input data product; *WasGeneratedBy* links an output data product to a workflow task; *WasAssociatedWith* links a workflow task to the data scientist who performs its execution; *WasInformedBy* links a downstream workflow task to an immediate upstream workflow task, *WasDerivedFrom* links an output data product of a workflow task to an input data product of the same workflow task, *ActedOnBehalfOf* links a data scientist to another, and *WasAttributedTo* links a data product to a data scientist. The PROV-DM model is illustrated in Figure 2. Like OPM, the PROV-DM model specification does not include querying languages for provenance either. Formally, a provenance graph PG = (N, E) in PROV-DM consists of:

- a set of nodes $N = EN \bigcup AC \bigcup AG$, where EN is a set of entities, AC is a set of activities, and AG is a set of agents, based on the PROV-DM model.

- a set of directed edges E = $E_u \bigcup E_g \bigcup E_d \bigcup E_i \bigcup E_a \bigcup E_{ab} \bigcup E_{at}$

  where i) $E_u \subseteq$ AC $\times$ EN and (ac, en) $\in E_u$ means that activity ac *used* entity en.

  ii) $E_g \subseteq$ EN $\times$ AC and (en, ac) $\in E_g$ means that entity en *was generated by* activity ac.

  iii) $E_d \subseteq$ EN $\times$ EN and (en_1, en_2) $\in E_d$ means that entity en_1 *was derived from* entity en_2.

  iv) $E_i \subseteq$ AC $\times$ AC and (ac_1, ac_2) $\in E_i$ means that activity ac_1 *was informed by* activity ac_2.

  v) $E_a \subseteq$ AC $\times$ AG and (ac, ag) $\in E_a$ means that activity ac *was associated with* agent ag.

  vi) $E_{ab} \subseteq$ AG $\times$ AG and (ag_1, ag_2) $\in E_{ab}$ means that agent ag_1 *acted on behalf of* agent ag_2.

  vii) $E_{at} \subseteq$ EN $\times$ AG and (en, ag) $\in E_{at}$ means that entity en *was attributed to* agent ag.

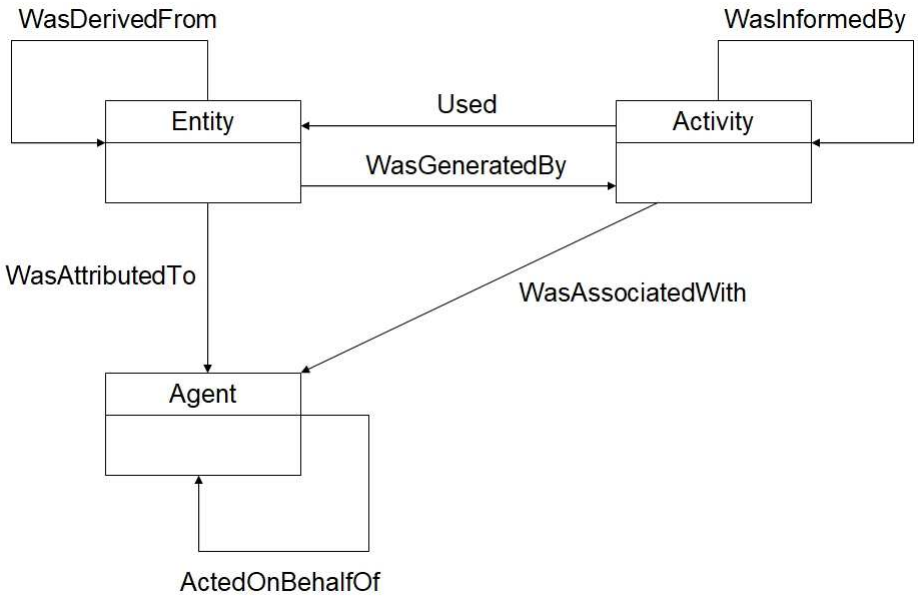


Figure 2: PROV-DM Provenance Model.

Table 1: PROV-DM Core Concepts Mapping to Types and Relations [4].

PROV concepts	***PROV-DM types or relations***	***Name***
Entity Activity Agent	PROV-DM Types	Entity Activity Agent
Generation Usage Communication Derivation Attribution Association Delegation	PROV-DM Relations	WasGeneratedBy Used WasInformedBy WasDerivedFrom WasAttributedTo WasAssociatedWith ActedOnBehalfOf

1.3 Provenance Query Language: OPQL

OPQL (Open Provenance Query Language) is a provenance query language that enables the querying of provenance directly at the graph level. One advantage of OPQL is that, OPQL queries are tightly coupled to the underlying provenance storage strategies. As a result, OPQL can be implemented on top of various storage or database systems. Another advantage of OPQL is its practical expressiveness: a provenance query formulated in OPQL is more concise than its counterparts in other query languages such as SQL, SPARQL, and XQuery, which often need recursive formulation for lineage queries [85]. OPQL includes

six types of graph patterns, a provenance graph algebra, and has clear syntax and semantics to support querying provenance at the graph level. The initial implementation of OPQL shows the feasibility and efficiency of OPQL [85, 84]. A basic OPQL query can be defined in the following ways:

- Single node construct A, P, and AG.
- Single-step-edge-forward constructs USD, WGB, WCB, WDF, and WTB.
- Single-step-edge-backward constructs $USD^{\wedge}$, $WGB^{\wedge}$, $WCB^{\wedge}$, $WDF^{\wedge}$, and $WTB^{\wedge}$.
- Multi-step-edge constructs $USD*$, $WGB*$, $WDF*$, $WTB*$, $WDF^{\wedge}*$ and $WTB^{\wedge}*$.

In addition, composite queries can be composed from basic queries by connectives UNION, INTERSECT, or MINUS. OPQL is based on the OPM provenance model. This paper extends OPQL to OPQL 2.0 to support the PROV-DM provenance model.

1.3.1 Apache Pig

Apache Pig is a fairly competitive framework for processing and continuous optimization, enhanced with new features and maintained by Yahoo! Researchers [109]. It is a platform for analyzing large data sets. Pig's language, Pig Latin, is a simple to understand data flow language. Its query algebra expresses data transformations such as merging data sets, filtering them and applying functions to records or groups of records. This supports rich, multi-valued and nested operations on large datasets. Pig Latin scripts describe Directed Acyclic Graph where edges are data flows, and the nodes are operators that process the data [1]. Pig is extensible to a user-defined function written in Java and other languages. Pig scripts provide a high-level language to create the map-reduce jobs needed to process data in Hadoop cluster [1]. In Fig. 3, the overall framework of Apache pig is presented graphically.

The advantage of using Apache Pig is that it is independent of Hadoop framework changes and can be benefited by all the optimization techniques offered by Pig developer

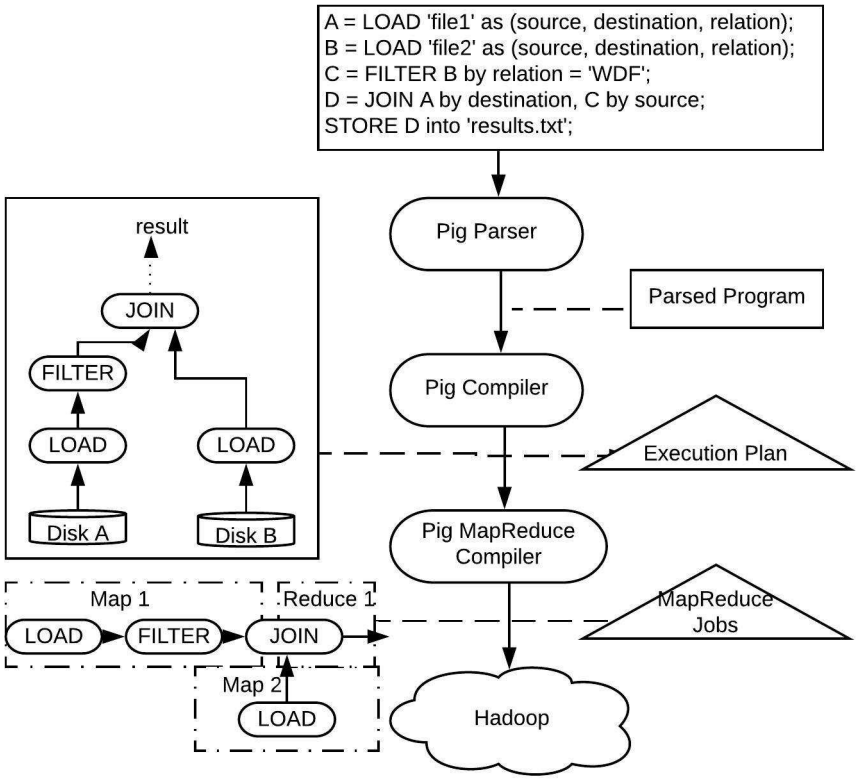


Figure 3: The Framework of Apache Pig.

community. Moreover, as Pig is backward compatible, further developments or optimizations will not affect any single line of code. During runtime, the resulting Pig Latin script automatically maps into a sequence of MapReduce iterations. Hence, complicated deployment, configuration or installation do not require. We use Pig Latin operator in our translation. One benefit of choosing Apache Pig to implement OPQL is that, further optimization and development in both the MapReduce community and the Apache Pig community can immediately benefit OPQL users, which do not need to rewrite a single OPQL query.

We explain each instruction of pig performed in our case in the later section. The more detail description of each operation of Apache Pig is in Pig Latin Manual [1].

1.4 Provenance Security

Recently, the notion of collaborative scientific workflows has been proposed to address the increasing need for collaborative data analytics using the scientific workflow paradigm. In such collaborative environments, adequate access control policies are necessary for controlling the sharing of workflows, data products, and provenance information among collaborating parties. In particular, the protection of workflow provenance is critical as such provenance often encodes the detailed protocol of a scientific experiment and constitutes the intellectual property of the respective stakeholders. Meanwhile, scientific workflows feature quick evolution; therefore, corresponding access control policies for workflow provenance need to evolve as well, and it is important to ensure that the evolution of workflow provenance access control policies meets certain qualities to guarantee the correctness and performance of the policy enforcement engine.

1.4.1 Role Based Access Control (RBAC)

Role Based Access Control (RBAC) consists of the following components: U, R, P, S. Here U stands for a set of users in a system, R stands for a set of rules, P for permission and S for the session. In a dynamic system, User (U) might change, where a user might join or leave; Role (R) might change, an administrator might add more roles in the system or delete some roles from the system, and Permission (P) also might change.

The policy access control rights are associated with roles, where access rights are given to specified roles for accessing certain resources. Users are assigned to designated roles and be part of access control policies. Users can authenticate themselves through activation of one or more roles for themselves.

Some benefits of using Role-based access control are:

- Policy doesn't require to be updated when a certain user with a role leaves the system.
- When a new user assigned to an active role, all the required resources automatically allocated to the user for that role.

- Revisiting least privilege, which means a user in one role has access to a subset of resources, and when they switch roles they gain access to other resources.

In RBAC, permission can be set based on objects with their actions. The action represents the operation performed on the object which could be read or write operation or in perspective of database it could be select, update, delete operations, etc. The sign for permission is either positive or negative. Positive permission means positive authorization where a particular action is allowed on this object and negative permission means authorization of action on that object is not permitted.

There is permission assignment function which assigns permission to roles. For each role, this function assigns a set of permissions where an input is a role and output is a set of permission. This is a subset of the cross product of $R \times P$. For particular function, different permission can be assigned for a role in a different time frame. The assigned permission for a role can be changed over time to different permission.

In RBAC, there is also a user assignment function. One of the typical examples of user assignment function is one student who designated as a GRA now can be later appointed as GTA. In user function, each session assigns to a single user. The session is not shared between users, and each session only allows one user. But role can be different in a specific session. For example, for log-in a particular terminal in Unix, where a user can be a regular user or pseudo-user.

The administrator's job is to specify access control policy first regarding the access rights. In policy enforcement phase, policies are enforced when users perform actions. The action rights are determined by access control policies. In other words, if access control policies are positive then the task can be executed; otherwise, the task cannot be accomplished. Concerning the databases, access control policies are consulted first before the update a table or delete a row or update a column, etc. Similarly, to open a file, close a file, delete a file, update a file, append a file in the operating system, we can also use access control policies to control the access to files. Policy analysis observes all the

actions performed in a particular system either in the database system, workflow system or operating system. Policy analysis evaluates the quality of the policy and then determine how to improve the quality of policy.

CHAPTER 2 RELATED WORK

2.1 Related Work of Provenance

In general, the provenance management system deals with efficiency and effectiveness of capturing, storing, representing, querying, and visualizing provenance data. In this chapter, we first discuss the provenance capture in perspective of different existing scientific workflow provenance management systems. We then discuss related work on provenance models and applications as well as their analysis and visualization. Lastly the recent trends in provenance query and big data provenance state-of-the-art frameworks, methods and mechanisms are presented.

2.1.1 Provenance Query and Big Data Provenance

Provenance problems become prohibitive and very hard to solve when applied to big data repositories. There are many avenues of research challenges and open issues in big data provenance research [52, 53]. Several relevant and advanced concepts and challenges in big data provenance research can arise. They include accessing big data, analyzing big data, scalability issues, information sharing, query optimization issues, data modeling support for provenance, flexible provenance query tools, etc. [119]. Reduce and Map provenance (RAMP) [72, 103] is one of such big data provenance systems that proposes a wrapper based method as an extension of Hadoop by deploying on top of Hadoop. Another extension of Hadoop for implementing provenance detection in MapReduce jobs is called Hadoop-Prov [119, 9]. The proposed system minimizes overheads and computes provenance by providing flexible tools for querying the big data provenance graph.

There is a hybrid big data provenance system named Pig Lipstick [13], that combines the management of fine-grained dependencies, with the management of workflow-grained dependencies [119, 13]. Fine-grained dependencies are typical of database-oriented provenance systems, and workflow-grained dependencies are typical of workflow-oriented provenance systems. Another type of big data provenance supports the functionalities of layer-based architecture by focusing on provenance collection, querying and visualization of

provenance in the context of specialized scientific applications [7]. In the cloud environment, there is a proposed framework for integration, modeling and monitoring data provenance called CloudProv [120]. For real-time applications, this framework continuously acquires and monitors all of the collected provenance information. Another proposed big data provenance framework [59] is based on fine-grained provenance through several operator instrumentations of a query. An innovative privacy-preserving public auditing mechanism in untrusted Cloud environments is Oruta [127]. This mechanism supports data sharing.

2.1.2 Provenance Capture and query in Scientific workflow systems

There are several scientific workflow provenance management system working towards data collection and capturing provenance.

In Kepler [25, 26] the provenance framework is called Collection Oriented Modeling and Design (COMAD) for nested data collections and captures explicit data dependencies. Here provenance information is automatically stored in the relational database with all immediate and transitive closure dependencies derived from provenance reasoning for each node [123, 15]. The COMAD-Kepler provenance system supports the querying of the imported provenance metadata through a high-level query language and an external reasoning engine.

Taverna [97, 95] keeps a logbook plugin based on a provenance ontology to capture provenance information from the workflow. The provenance collection plugin captures both internal provenances locally generated by workflow, and external provenance gathered from third party data providers. Taverna supports the use of a fine-grained data lineage model to perform query collection-based workflow provenance effectively [98]. To store, manage and query provenance, Taverna uses RDF store mechanism and Semantic Web technologies for representing provenance metadata.

Karma [113] captures both process and data provenance from a user-driven workflow system by storing provenance meta-data in a two-layer information model. Karma's

provenance model layers are registry level and execution level. Registry level records the metadata of services and data that may be used in an execution sequence. An execution level models instances of the registry level and records the execution-related information of data products generated by each invocation [31]. To store and query provenance metadata Karma uses XML and relational database technologies [31, 126, 112].

In VisTrails [111, 57], the provenance management system captures provenance information by using change-based provenance mechanism for data products. Also captures all the generated data products in the process of evolution of workflow. This uses relational database and XML to store and query provenance metadata. The provenance management system queries are supported by XQuery for querying exported XML specification and implementing recursive functions to query the transitive closure dependencies. VisTrails can visualize query results by matching query conditions with the VisTrails query language (vtPQL).

Another scientific workflow management system is Swift [134]. It uses provenance for on-demand data generation, tracking the data derivation history, and data product validation. Swift also uses the relational database in order to manage and query provenance metadata.

VIEW [35, 84, 124, 107] supports provenance store and query for both prospective and retrospective provenance collection. Two independent systems RDFProv and OPM-Prov [37, 85] has developed under the umbrella of the VIEW system. The noWorkflow system [101] captures provenance information from scripts and YesWorkflow [94] provides a script with an annotation mechanism to describe prospective provenance.

Panda [73] is a general purpose open-source system that supports provenance capture, storage, operator, and queries. This system seamlessly merges both data-based and process-based provenance. It also develops a model and system from fine-grained to coarse-grained provenance, defines useful operators for taking advantage of captured provenance, defines general purpose language for querying, and analyzes provenance and

combining provenance with relevant data.

2.1.3 Provenance Analysis and Visualization

Provenance visualization can be classified into two ways: workflow management systems that have built-in provenance visualization and standalone provenance visualization tools [79]. The built-in provenance visualization tools [11, 30, 70] allows the easy integration of provenance collection and analysis. However, they do not support the visualization of data by other workflow management systems and different standalone provenance gathering tools. These built-in provenance visualizations in workflow management systems also have shortcoming for viewing graph manipulation features for provenance graphs.

On the other hand, there are standalone provenance visualization tools. Provenance explorer [40] dynamically generate customized views of provenance trail by taking RDF-based provenance outputs from capture system. This provenance model is based on the ABC ontology model and lacking the support for data processing activities in the digital domain. Hence, they support only one expansion level instead of multiple levels of detail. The ZOOM prototype allows users to dynamically update the provenance graph for the new view by hiding irrelevant information. It supports users with an interface to query provenance information through SQL queries which generated by a workflow system.

There is a web-based visualization tool based on the PROV-DM model called PROV-O-Viz [68]. This tool uses Sankey Diagrams for visualization to visualize flow magnitude between nodes in a network. Provenance Explorer is limited to specific domains and ZOOM, PROV-O-Viz requires additional knowledge, they are not compatible with provenance data from other tools like Kepler, VisTrails, Taverna. The stand-alone tools provide some interesting features such as interactive graphs, summary nodes, level of details, filters, merges, but do not implement them in an integrated way.

There is another prototype system called Provenance Difference Viewer (PDiffView) that enables users to compare the differences between two runs which is two provenance graphs of the same workflow specification. Based on [135], the PROV Translator tool

provides graph visualizations.

2.1.4 Provenance Models and Applications

In a provenance management system, one of the main issues is to make the system interoperability among different scientific workflow management systems. IPAW workshop first initiated this idea in 2006 through Provenance Challenge series and as a result came up with standardized provenance model called Open Provenance Model (OPM). The OPM model [3, 6, 104] efficiently allows provenance information exchange between systems regardless of compatibility layer on shared provenance model. This model defines provenance in a precise and technology-agnostic manner, facilitate developers to build and share tools to operate on any such provenance systems, defines a set of core rules to identify the valid inferences that can make on provenance representation. OPM was originally crafted in 2007 and released to the community with version OPM $v1.00$. Later revised in 2008 with version OPM $v1.01$ and again in 2009 with OPM $v1.1$. OPM uses directed graph to express the dependencies.

Recently, the well-accepted provenance model is PROV-DM. PROV-DM was adopted here because of its core structures to distinct extended structures toward a step forward to interoperability. Core structures provide domain-agnostic vocabularies by serving essence of provenance information, whereas extended structures are more towards enhancing and refining more significant capabilities for more advanced uses of provenance.

Only a few of scientific workflow system have adopted the PROV extension. Taverna is one of them [15]. DataONE scientific workflow and provenance working group specified D-PROV provenance model [96]. Few PROV applications use and extend PROV, the name of those projects are UrbanMatch [118, 32], which extends PROV model by using Human computation ontology and CollabMap [105], to record provenance information that logs citizens actions [96].

2.2 Related Work on Provenance Security

For business workflows, the importance and requirements of security are well understood [117, 49, 24, 8, 67, 23, 17]. In perspective of the workflow system, the requirements for security can be managed by either the workflow system itself [121] or by an outside engine [42]. Most of the security work has been done in authentication [93], authorization [128, 18, 69, 108, 115, 76], data privacy and secure workflow models [71, 75]. The security issues of provenance have recently been identified by some researchers [27, 116].

The authors of [38] formalize a model for provenance with security properties like disclosure and obfuscation on workflow provenance graph, database queries, and automata. They explain the most general form of provenance for the system through traces. Their framework defines primarily the static provenance situation, not dynamic provenance in an existing system.

In [28, 66], the authors address a number of research questions on provenance security and develop a mechanism for securing provenance by using appropriate encryption and digital signature. They allow auditors to check the integrity of provenance without necessary access to underlying data and vice versa [38]. In [66], the authors maintain the integrity of provenance records in a stateful system and prevent forgery.

Based on the work of Cheney et. al.[39], Chong [43] formulated a syntactic model of traces and proposed semantic definitions of provenance security policies. Chong [43] formalized two properties, "provenance security" and "data security". In provenance security, the provenance of a workflow run is not inferred from data, whereas, in data security, high-security workflow data are not inferred from its provenance.

In [47], Davidson et al. proposed a formal definition of privacy and confidentiality policies for workflow provenance and formalize the notion of privacy and focus on a mathematical model for solving privacy-preserving view as a result of the query by an auditor. Their approach is theoretic and does not provide a framework for provenance models for addressing security.

In [16], the authors investigated the problem of securing data provenance in the cloud and propose the schema that supports encrypted search while protecting the confidentiality of data provenance stored in the cloud. Their main contribution of the proposed approach is that neither an adversary nor a cloud service provider learns about the data provenance or the query [16].

The Secure Provenance (SPROV) scheme in [66, 65], provides security assurances of confidentiality and integrity of the data provenance and automatically collects data provenance at the application layer. They ensure confidentiality by employing state-of-the-art encryption techniques where integrity is preserved by using the digital signature of the user who takes actions. SPROV scheme has some limitations too. It does not provide confidentiality to the source data whose data provenance is being recorded, and it does not provide any mechanisms to query data provenance [16].

PSecOn scheme in [74] proposes a cyber laboratory to collaborate and share scientific resources for provenance security from the origin. The integrity of the scientific results and corresponding data provenance can be ensured through PSecOn scheme in an e-science grid. This scheme encrypts the source data. The limitation of PSecOn is its strong assumption of relying on a trusted infrastructure, restricting the possibility of managing data provenance in the cloud [16].

Lu et al. [89], introduce a scheme to manage data provenance in the cloud and provided users access to the online data where data is shared among multiple users. Confidentiality and integrity are guaranteed through user encryption. This work signs the data where a cloud service provider receives and verifies the signature before storing that data. The main drawback of this approach is that it only traces the user while it does not provide any details about how the data provenance is managed by the cloud service provider [16].

Aldeco et al. [10] provide concrete cryptographic constructs to ensure the integrity of data provenance. They describe four stages: recording provenance, storing provenance, querying provenance and analyzing provenance graph for answering questions regarding

the execution of the entities in the system. When data provenance is recorded and stored, integrity is ensured. The limitation is lack of details about how to provide confidentiality to data provenance.

In [28], data provenance is considered as a causality graph with annotations. They focus on security model of data provenance at the abstract level. They mention the security of data provenance is different from the source data. They describe access controls but do not address how to define and enforce these access controls.

Security issues related to a Service Oriented Architecture (SOA) based provenance system is discussed in [116]. Here they suggest to restrain auditors by limiting the access to the results of a query using cryptographic techniques, but they have no concrete solution.

In [36], the authors provide a secure view of workflow provenance where they define security specification based on only inheritance. They did not provide any provenance access control policies or any quality analysis for their proposed protocol.

2.3 Related Work on Autism, Scientific Workflow and Machine Learning

Little research has been conducted on the amalgamation of machine learning, data mining, and autism data analysis. Aforementioned is a comparatively new area where a behavioral and neurological problem can be deciphered in perspective of machine learning and data mining techniques. The research in this area mostly gained attention since 2009. Currently, there is no uniform platform for addressing all the issues. We have discussed some of the related work done in the area of ASD based on scientific workflow system, machine learning, and data mining.

An automatic alert system, Autistic Child Sensor and Assistant System (ACSA), has been developed for autistic children and their families for protecting the child from overstimulating environments, incidents, and injuries, using wireless sensor network [12]. This system is for detecting and processing autistic movement based on machine learning algorithms. The ACSA works on three different components: ACSA wearable sensory device is

worn by the autistic child on the appropriate location of their body determined by physician or family; ACSA parent application is on parents' smart devices, and machine learning algorithms are used for actively recognizing and accurately detecting child's gesture and motion.

The presence of discriminative eye movement patterns in ASD individuals, a prediction system has been developed for discovering the latent patterns based on eye movement from the sequentially recorded image [88]. There are other studies that show how the machine learning process is used to optimize the diagnosis process by tracking eye movements of ASD children [22, 80, 50]. When compared to typically developing individual, ASD children and adults show reduced visual attention to faces [54]. Work in [129, 130] shows the evidence of different eye movement by ASD individuals when scanning faces.

There are Autism and Emotion sub-challenges introduced by INTERSPEECH 2013 computational Paralinguistics Challenge [83]. They provide the result based on an integration of multiple well-known machine learning techniques, like Support vector machine, Deep neural network, Weighted discrete K-nearest neighbor; and ASM (Acoustic segment model) approach.

A human-robot interaction technique is a possible future direction for modeling the behavior of children with autism [110]. There are case studies and anecdotal evidence that shows children with autism exhibit improved social behavior with robots. They have used two machine learning techniques Conditional Random Fields(CRF) and Decision Trees. CRF is used to classify the segment in time series data generated by experiment and decision tree $C4.5$ algorithm is in order to predict vocalization. There are some research groups who have used robots for children with autism in [56, 81, 106].

The approach in [62] provides a better understanding of Autism Spectrum Disorder by selecting features for identifying subtypes of autism to determine the effectiveness of clustering for further classification purposes.

Also research has been done on the mining ASD data based on twitter information

[19]. This is the first paper focusing on using Twitter for data-mining information related to ASD. Their investigation gives more concerns, practices and generally more topics in conversation with people interested in ASD and motivating further work towards that direction.

Another data mining technology has been used for investigating feature selection in gene expression [82]. They used classification method for selecting genes and gene sequences between ASD and healthy cases.

AMP (Autism Management Platform) [87] is a mobile application and intelligent web interface for capturing, analyzing and managing data which is associated with diagnosis and treatment of ASD. Though continuous data management is a challenge, this analytic platform aggregates and mines patient data in real-time and gives relevant feedback based on automatically learned data by filtering preferences over time.

Although some research work has been done in the Autism community with machine learning and data mining, the scientific workflow community has not began to explore this area. There is no state-of-the-art collaboration of scientific workflow and autism.

CHAPTER 3 SCALABLE PROVENANCE QUERY FRAMEWORK

Recently, big data workflows have emerged as the next generation of data-centric workflow technologies to address the five “V" challenges of big data: volume, variety, velocity, veracity, and value [122, 78]. While scientific workflows focused on data flow and automation management [86], big data workflows focus on large-scale data processing and analytics with a “scale-out" architecture and a “moving-computation-to-data" processing paradigm [51]. As both data and workflow increase in their scale, the scale of provenance naturally increases as well, calling for new scalable storage and querying infrastructure for big data workflow provenance.

To this end, we propose to leverage Pig Latin [102], a high-level platform for creating programs that run on Apache Hadoop, and OPQL [85], the most famous graph-level provenance query language, to build scalable provenance storage and querying system. Apache Pig is a platform for analyzing large datasets on top of Hadoop with a rich, multi-valued, and nested data model. Pig’s language, Pig Latin, is a comprehensive imperative query language that let users express data transformation such as filtering datasets, merging them and applying functions to groups of records or records. It is a simple data flow language, yet a fast iterative language with an efficient MapReduce compilation engine. Pig Latin gives a higher level of abstraction than the MapReduce procedural model by providing join, a filter like relational style operators, which are not feasible in MapReduce out of the box.

Meanwhile, OPQL is a graph-level provenance query language that includes graph patterns. It is based on a rigid provenance graph algebra and explicit syntax and semantics. As OPQL queries are not tightly coupled to the underlying provenance storage strategies, a OPQL user does not need to be aware of the underlying schema design. Moreover, OPQL is technology-independent, and therefore can be integrated with any big data workflow system. To the best of our knowledge, this is the first effort to ensure both scalability (leveraging a scalable platform) and usability (leveraging a graph-level provenance query

language) of provenance querying in the area of big data workflows.

Our main contributions are: i) we extend OPQL 1.0 (for the OPM model) to OPQL 2.0 in order to support the W3C PROV-DM standard provenance model, the current de facto standard provenance model. This model extends OPM with additional features to capture provenance and accommodate the Web; ii) we propose framework and efficient algorithms to translate OPQL constructs to equivalent Pig Latin programs; iii) we develop and evaluate our system on provenance datasets from the UTPB benchmark; and (iv) we create some visual OPQL constructs in the DATAVIEW big data workflow system to facilitate the easy creation of complex OPQL queries in a visual workflow style. Our preliminary experimental study shows the feasibility of our framework for big data workflow provenance storage and querying.

The rest of the chapter is organized as follows: In Section 3.1, we propose a new framework called $OPQL^{Pig}$ to translate OPQL constructs to equivalent Pig Latin programs. Section 3.2 proposes our extension of OPQL to support the PROV-DM provenance model. The algorithm for translating OPQL to Pig Latin is presented in Section 3.3. In Section 3.4, we explain our proposed framework and algorithm with a sample query case study. Finally, in Section 3.5, we discuss data preparation and an experimental section with some visual $OPQL^{Pig}$ constructs in the DATAVIEW, a big data workflow system to facilitate the easy creation of complex $OPQL^{Pig}$ queries in a visual workflow style.

3.1 $OPQL^{Pig}$ Querying Framework

$OPQL^{Pig}$ is an integrated system of Storage and Query engines and its underlying MapReduce and Apache Hadoop framework. Though there are several concurrent projects like DryadLINQ, Hive, Jaql, Sawzall, Scope, etc., blending Pig with its underlying Hadoop execution engine shows an impressive benefit of scalability and fault tolerance. $OPQL^{Pig}$ system takes datasets from a OPQL client; processing the data in the storage engine; translate a OPQL query into Pig Latin program; parse and compile into Pig and one or more MapReduce Jobs; and execute those jobs in Hadoop cluster. We will discuss each of the

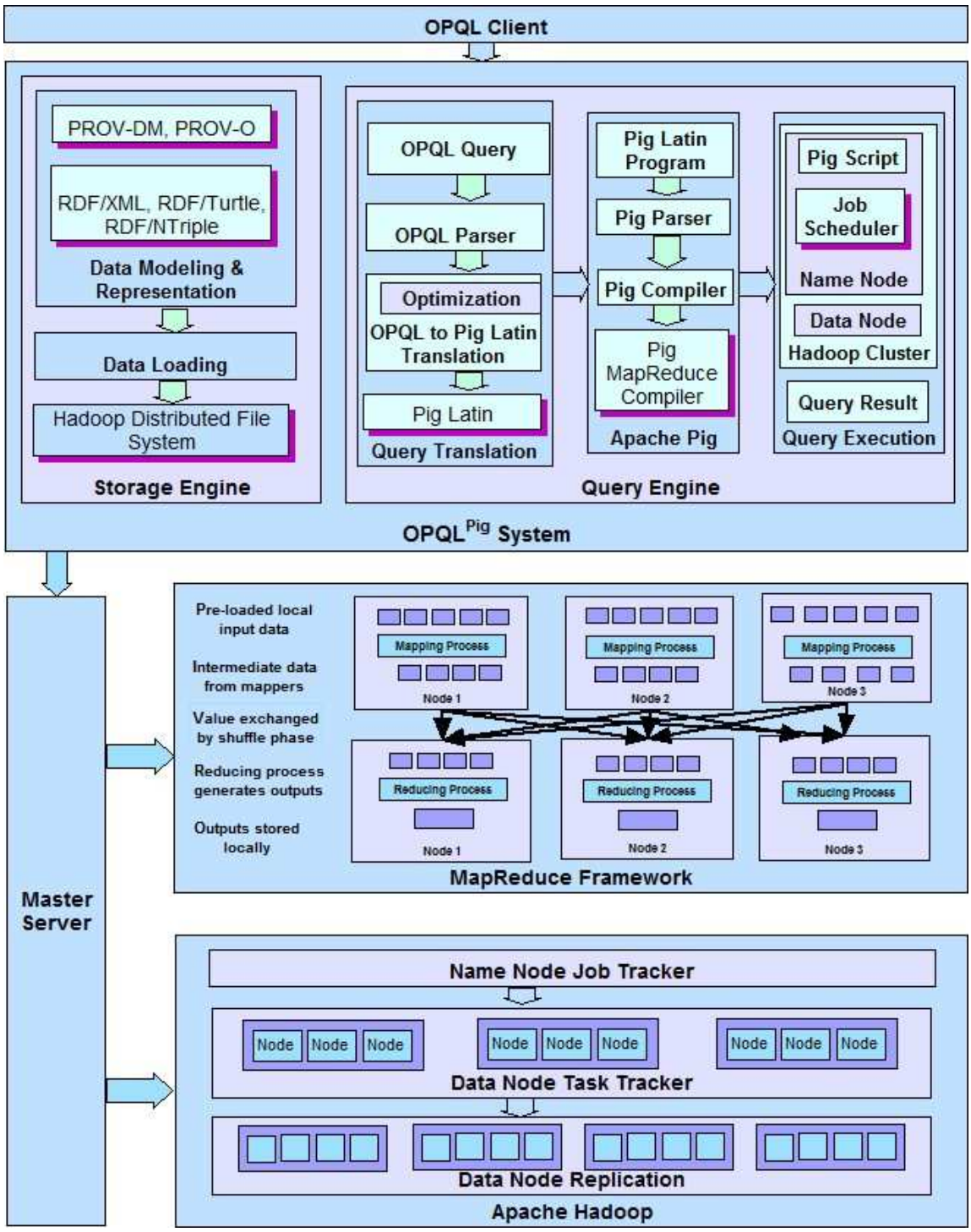


Figure 4: $OPQL^{Pig}$ Architecture.

parts of the system in this section.

3.1.1 Storage Engine

Data Modeling and Representation:

The provenance can be captured in any form of modeling and representation like PROV-DM, PROV-O and RDF/XML, RDF/Turtle or RDF/NTriple. For our $OPQL^{Pig}$ system we have used UTPB benchmark for capturing provenance data from templates presented in [34].

Data Loading:

After capturing provenance data, we perform the data loading phase for creating the dataset in our feasible format and store that in Hadoop Distributed File system for querying. For storing the dataset the feasible format we have used is: source_node, source_node_type, destination_node, destination_node_type, relation_construct.

3.1.2 Query Engine

The proposed Query engine has three major functional units with three unique responsibilities: Query Translation, Apache Pig, and Query execution.

Query Translation:

First, all $OPQL^{Pig}$ queries are passed through the $OPQL^{Pig}$ parser in order to verify their syntactic correctness. Each of the constructs of $OPQL^{Pig}$ is transformed into its corresponding Pig program through a dedicated shell script for each relation in PROV-DM model. The description of each translation from PROV-DM relations to its correspondence pig latin program is explained in section 3.3.

Apache Pig:

After translating the $OPQL^{Pig}$ query into a Pig Latin program which consists of a sequence of instructions, each instruction performs a single data transformation. During the phase of Pig parser schema inference, type checking and all referenced variables are defined. The output of the parser is arranged as a Directed Graph which is a logical plan of one-to-one correspondence between pig latin statements and logical operators [58].

Query Execution:

The MapReduce jobs in the graph are topologically sorted and submitted to the Job Scheduler's Name node. Finally, jobs are executed in that order inside Hadoop cluster's data node. $OPQL^{Pig}$ Querying Framework is represented in Fig. 4.

3.2 Extending OPQL to support PROV-DM

3.2.1 Provenance Constructs

We have modified the constructs based on the PROV-DM standard for all Single-node constructs, Single-step construct (edge-forward and edge-backward) and Multi-step-edge construct.

Single-node Construct:

We have formulated our single-node construct in the following formulation. As a single node construct we have Entities, Activities and Agents. Here en_n, ac_n and ag_n are single node identifiers and X_n is a node expression which can be denoted by either entity node expression X_{en}, an activity node expression X_{ac} or an agent node expression X_{ag}.

$$EN(X_{en}) = \{en_n \mid en_n \in X_{en}\}$$

$$AC(X_{ac}) = \{ac_n \mid ac_n \in X_{ac}\}$$

$$AG(X_{ag}) = \{ag_n \mid ag_n \in X_{ag}\}$$

Single-step Construct:

In a Single-step construct, there are Single-step-edge-forward construct and Single-step-edge-backward construct. We formulate all the relations in both directions. For each relation in the PROV data model we develop single step constructs. For example; for "Used relation" we develop "USD construct"; for "wasGeneratedBy relation" we develop "WGB construct"; for "wasAssociatedWith relation" we develop "WAW construct"; for "wasDerivedFrom relation" we develop "WDF construct"; and for "wasInformedBy relation" we develop "WIB construct"; both for edge forward and edge backward direction. These constructs support tracking the ancestor nodes associated with the relations. For every causal dependency between two nodes, if we provide a node expression X_n for effect nodes, we

receive cause nodes in return. In any single-step-edge-forward constructs when the source node is given, it returns the cause node which is the destination node or nodes. For any single-step-edge-backward constructs, for a given cause node, it returns the effect node which the source of the arc.

$$USD(X_{ac}) = \{en_n \mid ac_n \in X_{ac} \quad and \quad (ac_n, en_n) \in E_u\}$$

$$WGB(X_{en}) = \{ac_n \mid en_n \in X_{en} \quad and \quad (en_n, ac_n) \in E_g\}$$

$$WAW(X_{ac}) = \{ag_n \mid ac_n \in X_{ac} \quad and \quad (ac_n, ag_n) \in E_a\}$$

$$WDF(X_{en_{n1}}) = \{en_{n2} \mid en_{n1} \in X_{en} \quad and \quad (en_{n1}, en_{n2}) \in E_d\}$$

$$WIB(X_{ac_{n1}}) = \{ac_{n2} \mid ac_{n1} \in X_{ac} \quad and \quad (ac_{n1}, ac_{n2}) \in E_i\}$$

$$ACO(X_{ag_{n1}}) = \{ag_{n2} \mid ag_{n1} \in X_{ag} \quad and \quad (ag_{n1}, ag_{n2}) \in E_{ab}\}$$

$$WAT(X_{en}) = \{ag_n \mid en_n \in X_{en} \quad and \quad (en_n, ag_n) \in E_{at}\}$$

$$USD^{\wedge}(X_{en}) = \{ac_n \mid en_n \in X_{en} \quad and \quad (ac_n, en_n) \in E_u\}$$

$$WGB^{\wedge}(X_{ac}) = \{en_n \mid ac_n \in X_{ac} \quad and \quad (en_n, ac_n) \in E_g\}$$

$$WAW^{\wedge}(X_{ag}) = \{ac_n \mid ag_n \in X_{ag} \quad and \quad (ac_n, ag_n) \in E_a\}$$

$$WDF^{\wedge}(X_{en_{n2}}) = \{en_{n1} \mid en_{n2} \in X_{en} \quad and \quad (en_{n1}, en_{n2}) \in E_d\}$$

$$WIB^{\wedge}(X_{ac_{n2}}) = \{ac_{n1} \mid ac_{n2} \in X_{ac} \quad and \quad (ac_{n1}, ac_{n2}) \in E_i\}$$

$$ACO^{\wedge}(X_{ag_{n2}}) = \{ag_{n1} \mid ag_{n2} \in X_{ag} \quad and \quad (ag_{n1}, ag_{n2}) \in E_{ab}\}$$

$$WAT^{\wedge}(X_{ag}) = \{en_n \mid en_n \in X_{en} \quad and \quad (en_n, ag_n) \in E_{at}\}$$

Multi-step Construct:

Multi-step Constructs are a little bit more complicated, it uses both directions of the single-step constructs in a repetitive way. In this multi-step construct, all nodes are returned with either direct or transitive dependencies in the relations. These constructs give us the flexibility and feasibility to track ancestor nodes without formulating any recursive queries. For example, in $WDF^{*}(X_{en})$ construct, returns all the entities that contributed to derive entity from the given entity. The same rule applies for $WIB^{*}(X_{ac})$ construct. It returns

all the activities that contributed to derive activity from the given activity. However, for multi-step constructs, such as $WGB^*(X_{en})$ and $USD^*(X_{ac})$, this scenario is little different. For construct $WGB^*(X_{en})$, a given source entity first derives contributed activities and then based on the set of returned activities it loops through WIB^* construct, and finds all of the contributing activities; and then perform the join operation to find all transitive relations for given entity. The same building rule apply for $USD^*(X_{ac})$. For the $USD^*(X_{ac})$ construct, we can derive all of the contributed entities for a given activity. To do this $USD(X_{ac})$ first returns the set of entities associated with that and then execute WDF^* construct to find all the rest of the entities contributed to that activity.

$$WDF^*(X_{en}) = \{en_n \mid \bigcup_{en_n \in WDF(X_{en})} WDF^*(en_n) \ \cup \ WDF(X_{en})\}$$

$$WIB^*(X_{ac}) = \{ac_n \mid \bigcup_{ac_n \in WIB(X_{ac})} WIB^*(ac_n) \ \cup \ WIB(X_{ac})\}$$

$$WGB^*(X_{en}) = \{ac_n \mid \bigcup_{en_n \in WGB(X_{en})} WIB^*(ac_n) \ \cup \ WGB(X_{en})\}$$

$$USD^*(X_{ac}) = \{en_n \mid \bigcup_{ac_n \in USD(X_{ac})} WDF^*(en_n) \ \cup \ USD(X_{ac})\}$$

3.3 Translating $OPQL$ to Pig Latin

In this section, we explain the translation of $OPQL$ to $OPQL^{Pig}$. We translate each of the constructs to Pig Latin script. The implementation of each of the constructs is handled through separate shell scripts, because Pig Latin does not support loops and conditionals. As each of the constructs is designed to handle inference queries, it needs to traverse the whole provenance graph to provide query answer. In order to handle such a scenario, a shell script is used to run multi-step-edge-constructs by provisioning join operations in one pig file, going through a loop. In addition, the loop break condition is handled by a shell script. The join operation generates a set of records which provide transitive relation and the loop condition will break when there will be no new transitive relations. The loop break

condition measures the file size, and if there are no new relations, which means file size zero, then loop break will happen. The last step of each construct is to run another Pig file which will retrieve all the destination nodes along the path computed by join operations. The pseudo code of translating from $OPQL$ to $OPQL^{Pig}$ for single-step and multi-step constructs are summarized in algorithms in Figs. 5, 6 and 7.

Function 1: ShellScriptForMultistepConstruct

```
1  #!/bin/bash
2  # call me as ./ USD_star.sh prov_data.txt data_product
3  fileName=$1
4  nodeName=$2
5  pig -param input=$nodeName usd.pig

6  echo $fileName
7  index=0
8  hadoop fs -cp $fileName $fileName$index
9  for i in {1..5000}
10 do
11 echo "Looping $i times"
12 pig -param input=$fileName -param inx=$index -param outx=$i
   wdf_star.pig
13 fileSize=$(hadoop fs -dus $fileName$i/| cut  -f2)
14 echo $fileSize
15 if [ $fileSize=="0" ];
16 then
                i.  break
17 fi
18 index=$i
19 done
20 cp -r $fileName MyResultUSD/
21 pig -param name=$nodeName unionPathUSD.pig
```

Figure 5: Driver Function of Multi-step USD* Construct.

In Figs. 5, 6 and 7 we have given the algorithm just for one example construct USD^*. When USD^* Construct is called it conducted three pig files call steps: i) call single-step USD Construct, ii) Based on returned entities of USD construct call WDF^* construct, and iii) Make a union of destination nodes along the path.

The function 1 in Fig. 5 delineates the entire process. Function 1 primarily acts as

the driver program that iteratively invokes the constituent functions until the completion condition is met. It can be noted that function 1 can be written as a driver program in the Map-Reduce framework as well and is preferred, we present it as a shell script to intuitively represent the flow. We assume that the input file has the following format: source_label, source_type, destination_label, destination_type, relation_type. We present the algorithm in terms of USD^* construct; however, the approach is a generic one and can be applied to any other construct type. In line 5, we invoke usd.pig, which essentially selects rows that have the relation USD and have the source that we are interested in making a query against. This part of the algorithm essentially persists destinations that are reachable via one hop from the source of interest. Subsequently, in line 8, we prepare for the iterative self-join phase by cloning the input file. Line number 9 shows the termination condition for the iteration. Line 12 consists of invoking USD_star.pig with variable inputs. In each phase of the iteration, we join the original input with the recently computed output, as shown by the parameters of the pig file. The output of one phase of USD_star.pig is used as one of the inputs of the next phase of USD_star.pig. Finally in UnionPathUSD.pig, we combine outputs from usd.pig and USD_star.pig and project distinct destinations that are reachable. We persist them on $HDFS$ for subsequent usage in the workflow.

Function 2: SingleStepEdgeConstructPigCode

```
1  --USD.pig
2  prov = load '$fileName' using PigStorage(',') as (s:chararray,
   st:chararray, d:chararray, dt:chararray, r:chararray);
3  x = filter prov by s == '$input' AND r == 'USD';
4  y = foreach x generate d;
5  rmf USDoutx;
6  store y into 'USDoutx/' using PigStorage(',');
```

Figure 6: Algorithm for Single-step USD Construct.

The function 2 in Fig. 6 calls the single-step USD construct. Here, the “load" command is used in Pig to load the CSV file of input data. In line 2, “using PigStorage(’,’)" is a syntax

explaining how data is stored in the file. In this case, it is comma separated. The advantage of using Pig Latin is that we do not have to store schema at storage time, instead store data and define schema at loading time. Here “$fileName" is the argument for a name of the file we provided. In line 3, we use FILTER operator in Pig for selecting records based on the given predicates. If the predicates are true only then it will proceed through the pipeline. In this case, we use the filter operation to collect only those entities where starting nodes match to our given input and check whether their relation is USD or not. In line 4, FOREACH expression is used to go through a set of expressions and work like projection operator to send down the new records through the pipeline to next operator. In this case, we run our FOREACH operator on x, the set of records we retrieved using FILTER, to get only the destination of the schema. In line 5 and 6, STORE command is used for storing the data after finishing the data processing. Like LOAD, “using PigStorage(’ ’)" syntax is used for specifying storage structure of data. In this case, we stored our data into ‘USDoutx/’ folder.

The function 3 in Fig. 7, returns entities of the USD construct by calling WDF^* construct. In line 3, here load the data into x and define its schema at load time. In line 4, filter out only those records where relation is WDF. In line 5, load another set of data in y for the purpose of Join operation. Here the argument ‘$input$inx’ is controlled by the shell and each time ‘$inx’ is incremented. In line 6, we filter out only those relations which are WDF. The reason for doing the filtering first is to try to optimize our query. In line 7, now, Join both x1 and y1, based on x1’s destination and y1’s source. In line 8, the result of Join is a new set of transitivity relations, and it is the same schema as original schema out of those transitive relations. In line 9 and 10, we store our result. One noticeable point is that all results are stored in one folder’s different files. The reasons are two fold; one is - it will not expand our big data file even bigger, another one is - this will make error handling easier.

The function 4 in Fig. 7, make union of destination nodes along the path. In line

Function 3: MultiStepEdgeConstructPigCode

```
1  -- USD_star.pig
2  -- Pig file to Join based on condition
3  x = load '$input' using PigStorage(',') as (s:chararray, st:chararray,
   d:chararray, dt:chararray, r:chararray);
4  x1 = filter x by r == 'WDF';
5  y = load '$input$inx' using PigStorage(',') as (s:chararray, st:chararray,
   d:chararray, dt:chararray, r:chararray);
6  y1 = filter y by r == 'WDF';
7  z = join x1 by d, y1 by s;
8  result = foreach z generate x1::s, x1::st, y1::d, y1::dt, y1::r;
9  rmf MyResultUSD/$input$outx;
10 store result into 'MyResultUSD/$input$outx' using PigStorage(',');
```

Function 4: UnionOfConstructPigCode

```
1  ---UnionPathUSD.pig
2  x = load 'MyResultUSD/' using PigStorage(',') as (s:chararray,
   st:chararray, d:chararray, dt:chararray, r:chararray);
3  p = load 'USDoutx' as (value:chararray);
4  z = join x by s, p by value;
5  z = foreach z generate d;
6  t = filter x by s == '$name';
7  t = foreach t generate d;
8  result = union z, t;
9  result = distinct result;
10 dump result;
```

Figure 7: Algorithms for Multi-step USD^* Construct.

2, a specific directory is provided for loading all the related files from that directory, the benefits of which are stated before. In line 3, also loads the file from step 1 and where the value is the destinations. In line 4, this join operation will be performed based on retrieved nodes in single-step USD construct. Line 5 retrieves all the destination nodes. Line 6 filter operation retrieves the output nodes of single-step USD construct. In line 7, we only get destination values here. The union of both values provide complete inferred nodes after traversing the whole graph is in line 8. In line 9 and 10, the distinct operator has the same functionality as other distinct operators; and dump operator is responsible for showing the results.

3.4 $OPQL^{Pig}$: A Case Study

We now explain our proposed framework and algorithm in terms of a sample query. For capturing provenance data, we have used $UTPB$ (University of Texas Provenance Benchmark). Fig. 8 shows a sample provenance graph from the $UTPB$ benchmark that captured provenance data in PROV-DM format. In this provenance graph, each node is categorized as entity or activity or agent. The node identifier of this sample provenance graph is presented in Table. 2. Here, we have 14 Entities, 7 Activities and 1 Agent. We label each node with its corresponding identifier. Each of these nodes, when connected together, holds causal dependencies based on PROV-DM model. Each of causal dependencies, such as entity-entity, entity-activity, entity-agent, activity-agent - has a specific name of these relations. In Table. 3 we present those relations for Fig. 8 provenance graph.

Table 2: Provenance Graph Nodes and Corresponding Node Identifiers.

Node Type	***Node Identifier***	***Node Name***
Entity	en_1	Create Table SQL Statements
	en_2	Create Index SQL Statements
	en_3	Create Trigger SQL Statements
	en_4	Schema
	en_5	Time T_1
	en_6	Dataset
	en_7	Instance
	en_8	Time T_2
	en_9	SQL Query
	en_{10}	Evaluation Plan
	en_{11}	Time T_3
	en_{12}	Result
	en_{13}	Performance Graph
	en_{14}	Log
Activity	ac_1	Create Database Schema
	ac_2	Load Data
	ac_3	Record Log
	ac_4	Optimize Query
	ac_5	Execute Plan
	ac_6	Execute Query
	ac_7	Visualize Performance
Agent	ag_1	Query Optimizer

$OPQL^{Pig}$ can answer any queries presented in UTPB benchmark. Some of the sample queries are shown in Table. 4.

Based on our sample provenance graph in Fig. 8, the query we choose is:

Find all the Entities ever used along the way for deriving the activity "Optimize

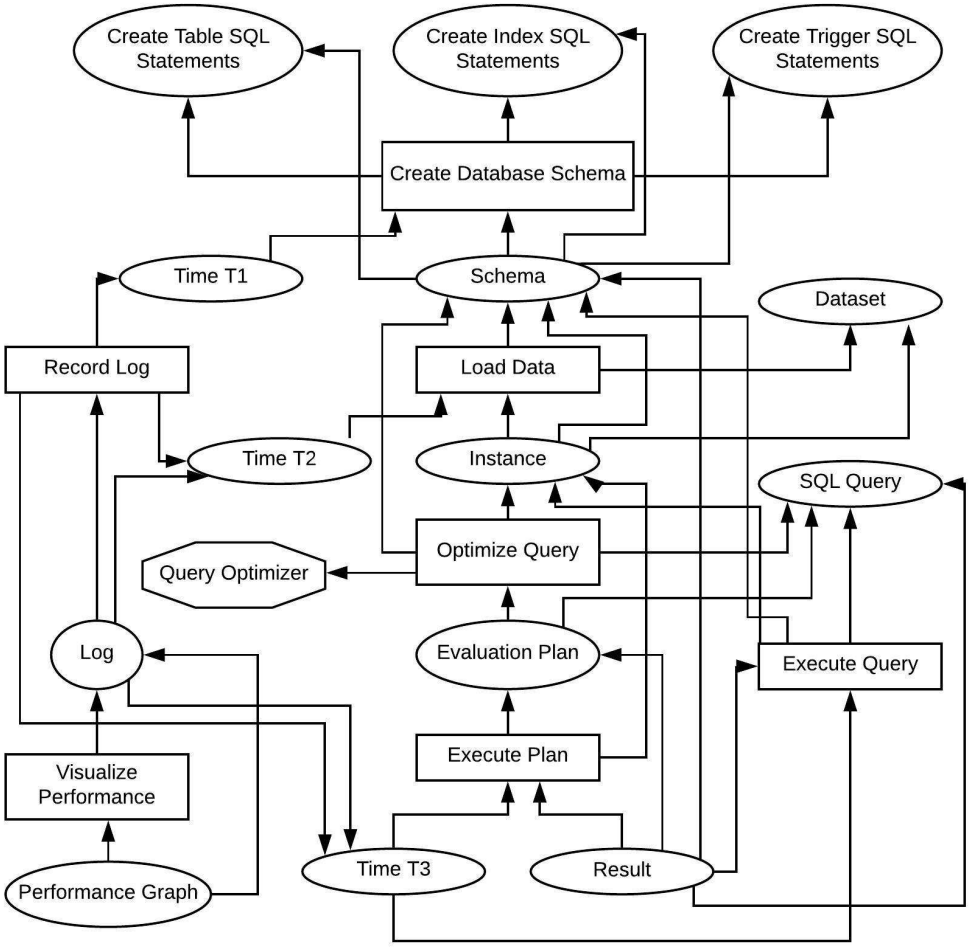


Figure 8: A Sample Provenance Graph for Provenance Data Capturing [34].

Query"

In order to execute the query and get the result, we have our provenance dataset in PROV-DM format. To run this query we provide input node ac_4 = “Optimize Query".

We execute the driver function which is function 1 in Fig. 5 where $prov\_data.txt$ is our file name and ac_4 is our $nodeName$. To get the query result we use USD* construct. USD* construct essentially calls three different pig files from Figs. 6 and 7, Function 2, Function 3 and Function 4.

- Call single-step USD Construct and store all the entities associated with given input

Table 3: Representing Provenance Graph Relations of Fig. 8.

USD	*WDF*	*WGB*	*WAW*
$ac_1 \rightarrow en_1$	$en_4 \rightarrow en_1$	$en_5 \rightarrow ac_1$	$ac_4 \rightarrow ag_1$
$ac_1 \rightarrow en_2$	$en_4 \rightarrow en_2$	$en_4 \rightarrow ac_1$	
$ac_1 \rightarrow en_3$	$en_4 \rightarrow en_3$	$en_8 \rightarrow ac_2$	
$ac_3 \rightarrow en_5$	$en_{14} \rightarrow en_{11}$	$en_7 \rightarrow ac_2$	
$ac_4 \rightarrow en_4$	$en_7 \rightarrow en_4$	$en_{14} \rightarrow ac_3$	
$ac_2 \rightarrow en_4$	$en_{12} \rightarrow en_4$	$en_{10} \rightarrow ac_4$	
$ac_6 \rightarrow en_4$	$en_7 \rightarrow en_6$	$en_{13} \rightarrow ac_7$	
$ac_2 \rightarrow en_6$	$en_{14} \rightarrow en_8$	$en_{11} \rightarrow ac_5$	
$ac_3 \rightarrow en_8$	$en_{12} \rightarrow en_{10}$	$en_{12} \rightarrow ac_5$	
$ac_4 \rightarrow en_7$	$en_{10} \rightarrow en_9$	$en_{12} \rightarrow ac_6$	
$ac_6 \rightarrow en_7$	$en_{13} \rightarrow en_{14}$	$en_{11} \rightarrow ac_6$	
$ac_5 \rightarrow en_7$	$en_{12} \rightarrow en_9$		
$ac_4 \rightarrow en_9$			
$ac_6 \rightarrow en_9$			
$ac_7 \rightarrow en_3$			
$ac_3 \rightarrow en_{11}$			
$ac_5 \rightarrow en_{10}$			

Table 4: Some Sample Queries from UTPB Benchmark.

Category	*Query*
Dependencies	Find all Entities derivation dependencies in a particular provenance graph. Find all Activities Informed-by dependencies for a specific provenance graph. Find all Entity use dependencies for a specific provenance graph. Find all Activity generation dependencies in a particular provenance graph. Find all Associated-with dependencies in a particular provenance graph.
Entities	Find all Entities and their values, if any, in a particular provenance graph. Given one Entity and find all Activities that served along the way
Activities	Find all Activities and their persistent names, if any, in a particular provenance graph. Given one Activity and find all Entities that served along the way

nodeName.

- Based on the given provenance graph, store all the entities in the graph which has WDF relation.
- Conduct the join operation on the returned graph.
- Finally join all the retrieved result along the way.

Now we explain our step by step process and their graphical presentation through Fig. 9.

3.4.1 Step 1

In our first step, while executing the driver function, we call the single-step USD Construct and retrieve all of the entities associated with given input which is ac_4. This execu-

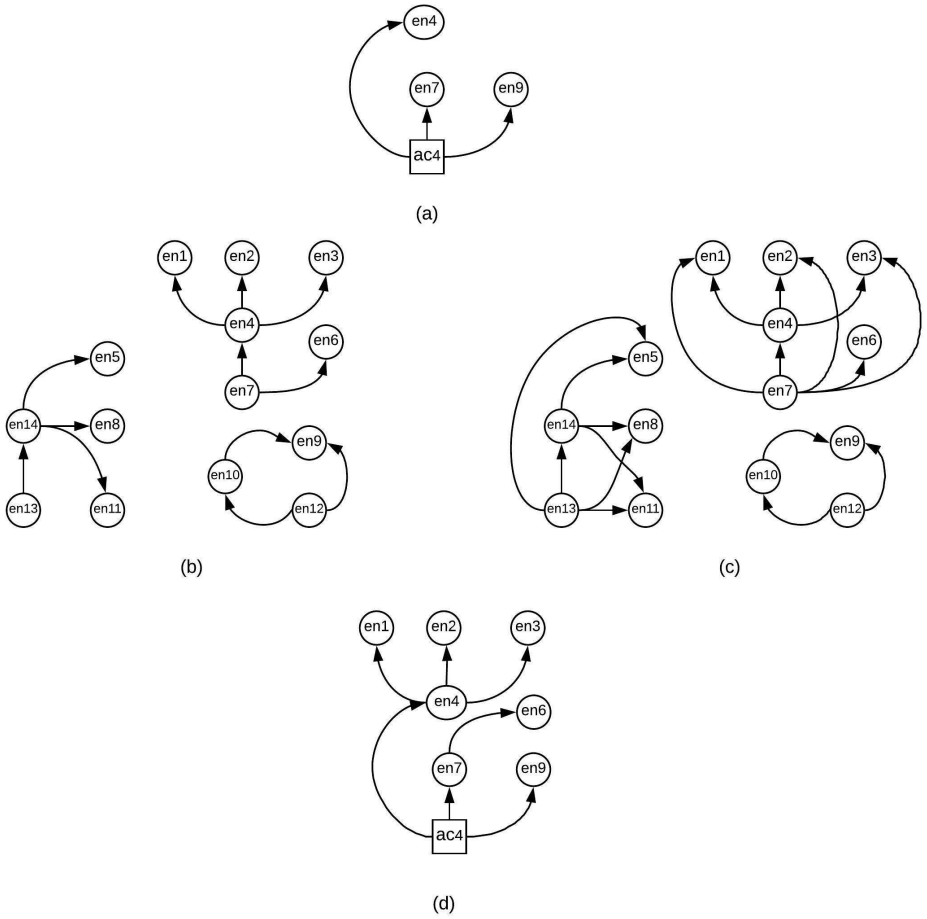


Figure 9: Graphical Representation of Query Processing Steps.

tion only retrieve the single step nodes with relation USD. Based on our provenance graph in Fig. 8, we retrieve three relations: $ac_4 \rightarrow en_4$, $ac_4 \rightarrow en_7$ and $ac_4 \rightarrow en_9$. We store all of the destination nodes en_4, en_7 and en_9 in one designated folder for future reference.

$$\boxed{ac_4 \rightarrow en_4 \quad ac_4 \rightarrow en_7 \quad ac_4 \rightarrow en_9}$$

3.4.2 Step 2

Step 2 is the first iteration in the loop of driver function. In this step, we filter out only those records where relation is WDF. The reason is that, to find out all the entities from given activities, we need to know all the entity-entity relations in the provenance graph

too so that later we can do join and find out all the transitive relations in the graph. From Fig. 8, we have 12 relations with WDF.

$en_4 \rightarrow en_1$	$en_4 \rightarrow en_2$
$en_4 \rightarrow en_3$	$en_7 \rightarrow en_4$
$en_7 \rightarrow en_6$	$en_{10} \rightarrow en_9$
$en_{12} \rightarrow en_{10}$	$en_{12} \rightarrow en_9$
$en_14 \rightarrow en_5$	$en_{14} \rightarrow en_8$
$en_14 \rightarrow en_{11}$	$en_{13} \rightarrow en_{14}$

3.4.3 Step 3

In step 3, we load the data into variable x and define its schema at load time and load another set of data in variable y for the purpose of Join operation. The result we found after Join is a new set of transitivity relations and we make the same schema like original schema out of those transitive relations. This step iterates until there are no new transitive relations. If no new transitive relations are generated through iteration then the file size will not change, and that breaks the loop. The total number of relations are graphically presented in Fig. 9(c).

$en_4 \rightarrow en_1$	$en_4 \rightarrow en_2$
$en_4 \rightarrow en_2$	$en_4 \rightarrow en_3$
$en_7 \rightarrow en_4$	$en_7 \rightarrow en_1$
$en_7 \rightarrow en_2$	$en_7 \rightarrow en_3$
$en_7 \rightarrow en_6$	$en_{10} \rightarrow en_9$
$en_{12} \rightarrow en_{10}$	$en_{12} \rightarrow en_9$
$en_{14} \rightarrow en_5$	$en_{14} \rightarrow en_8$
$en_{14} \rightarrow en_{11}$	$en_{13} \rightarrow en_{14}$
$en_{13} \rightarrow en_5$	$en_{13} \rightarrow en_8$
$en_{13} \rightarrow en_{11}$	

3.4.4 Step 4

Now in step 4, we load all of the transitive relations from step 3 and also load our initial relations with USD. After the join operation, we have all of the entities ever used along the way for deriving the activity ac_4. The union of all values provides complete inferred nodes after traversing the whole graph. Here we have 7 relations with 7 entities: en_4, en_7, en_9, en_6, en_1, en_2 and en_3, which used for deriving activity ac_4.

$ac_4 \rightarrow en_4$	$ac_4 \rightarrow en_7$
$ac_4 \rightarrow en_9$	$en_7 \rightarrow en_6$
$en_4 \rightarrow en_1$	$en_4 \rightarrow en_2$
$en_4 \rightarrow en_3$	

3.5 Experiments

A collection of experiments were conducted on a machine with Intel core $i7-3612QM$ CPU $@2.10GHz$ x 8 processor and 7.7 GB memory running on Ubuntu 12.10 (quantal) 64 bit. The experiments were designed on Apache Pig with version $0.8.1-cdh3u6$ and the $Hadoop$ framework with version $hadoop0.20.2-cdh3u6$. We have captured provenance using $UTPB$ template. Even though there are 27 different provenance templates representing provenance capture from three different workflows, we have chosen just one particular presentation capturing data from one specific workflow. $UTPB$ was selected as the benchmark template because it automatically generates datasets with varying sizes.

3.5.1 Data Preparation with benchmark

The original manually created template of PROV-DM contains around 66 triples. It always makes at least 3 copies of the original template to reach around 200 triples, so it is easy to get 200, 400 .... $1,000$, $10,000$, etc. triples depending on the needs. However, it is only the template generator. The data generator takes the "labels" statements and can change the values to generate different kinds of data, either fixed or randomized. The template generator adds a line to connect templates, $dataset_n$ was derived from $results_{(n-1)}$, but this union is very particular to the original template.

There are three main components based on data preparation with the benchmark:

- Original template: This is a provenance graph for one database experiment. It was created manually [34].

- Template generator: It takes the original template and creates a larger template automatically. The bigger template is the result of cloning the original template multiple times and connecting clones together into a single graph. Template generator allows

one to choose any times one would clone her original template and how one would connect the clones, i.e., sequentially or grid-shape. The file output.prov is generated by cloning the original template three times. Connections between clones are:

utpb:$dataset_1$ prov:wasDerivedFrom utpb:$result_0$. and

utpb:$dataset_2$ prov:wasDerivedFrom utpb:$result_1$.

- Data generator: It takes the original or generated template and generates as many clones or template instances of the template as specified. This time, each clone represents a provenance graph for one workflow execution. The data generator ensures that there are no ID conflicts between template instances.

3.5.2 Performance Study

During our experimental study, we focused on the performance and functionality of $OPQL^{Pig}$. The experimental data we have gathered were based on the size of big data that $OPQL^{Pig}$ can query and provide the query results. We prepare the data starting with 1000 instances and eventually going to 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000 and 10000 instances for generating big data, therefore testing the feasibility of processing queries by $OPQL^{Pig}$. Fig. 10 shows the correlation between the size of the instances correspond to the size of the dataset. As the number of instances increased the size of dataset increases too. The dataset size of $10,000$ instances is ten times larger than the data size of 1000 instances.

When we run $OPQL^{Pig}$ on big dataset of instances then we have to handle large number of nodes too. The total number of nodes per number of instances are shown in Fig. 11. As the number of instances increases the number of nodes also increases linearly, which shows the feasibility of our scalable query framework. Also increasing number of nodes demonstrate increasing number of relationship between those nodes. Fig. 11 depicts the correspondence number of nodes with number of relations in specified number of instances.

We also calculate the average query time of each of $OPQL^{Pig}$ construct. Here we

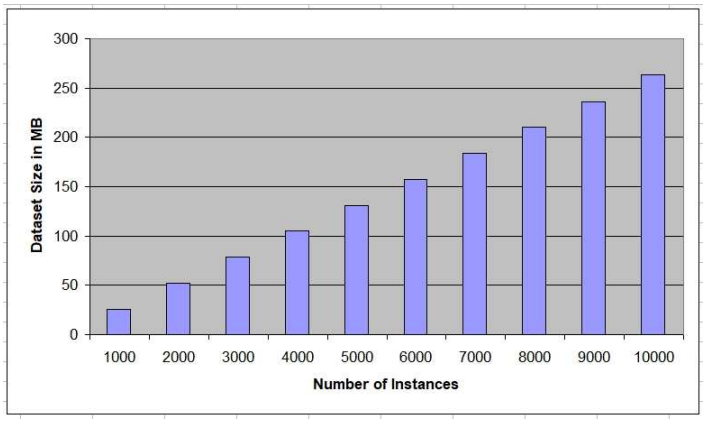


Figure 10: Dataset Size Vs. Number of Instances.

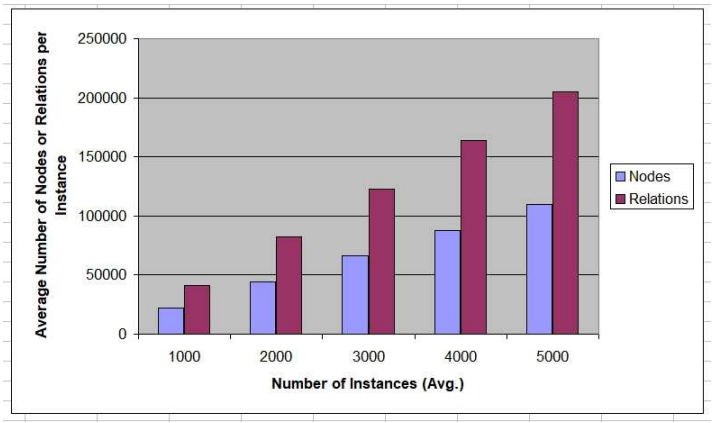


Figure 11: The Number of Nodes and Relations per Instance in Dataset.

choose multi-step constructs, such as USD* and WGB* construct to see the total query time required to find the query result. We plot it based on increasing number of nodes and total query time. We observe from our calculation that the query time doesn't double up as the number of nodes increases. This feature shows the feasibility of the $OPQL^{Pig}$ scalable query framework. The average query time for USD* construct is shown in Fig. 12 and average query time for WGB* construct is shown in Fig. 13.

Deployment of a job in Hadoop cluster has many advantages including increased scalability. In other words, using Pig makes it possible to execute queries on large datasets which are inherently not feasible in single machine solutions. However, deploying a job in

Hadoop cluster has some overhead including allocating containers for the tasks, deploying executables, sending data back and forth across machines. Which essentially means that batch jobs (like pig jobs) on Hadoop and single machine solutions are not comparable within the same parameter constraints. However, for the sake of completeness, we compare it with OPQL which is single machine solution. We observe that for smaller graphs it takes relatively longer to process using $OPQL^{Pig}$ due to the overhead involved in deploying batch jobs on the cluster. However, with increased data size we do not observe a significant increase in execution time. This is due to the fact that we are making effective use of the framework. This scale, however, unattainable in OPQL.

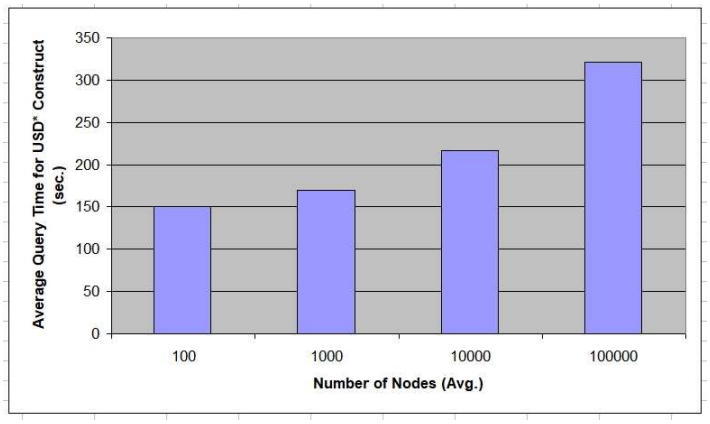


Figure 12: Average Query Time for USD* Construct.

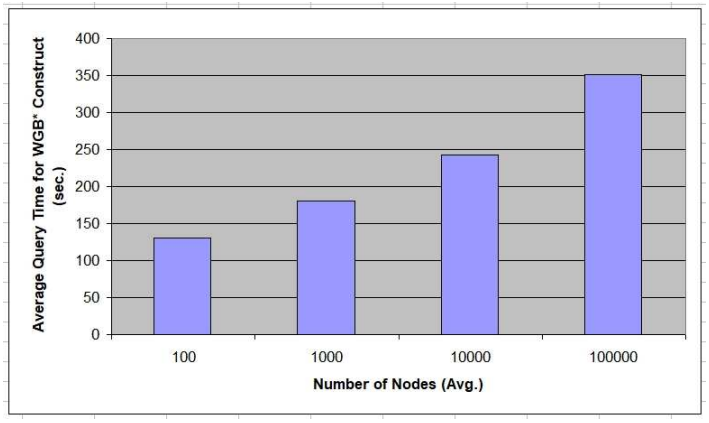


Figure 13: Average Query Time for WGB* Construct.

3.5.3 Primitive or Built-in Query Constructs in DATAVIEW

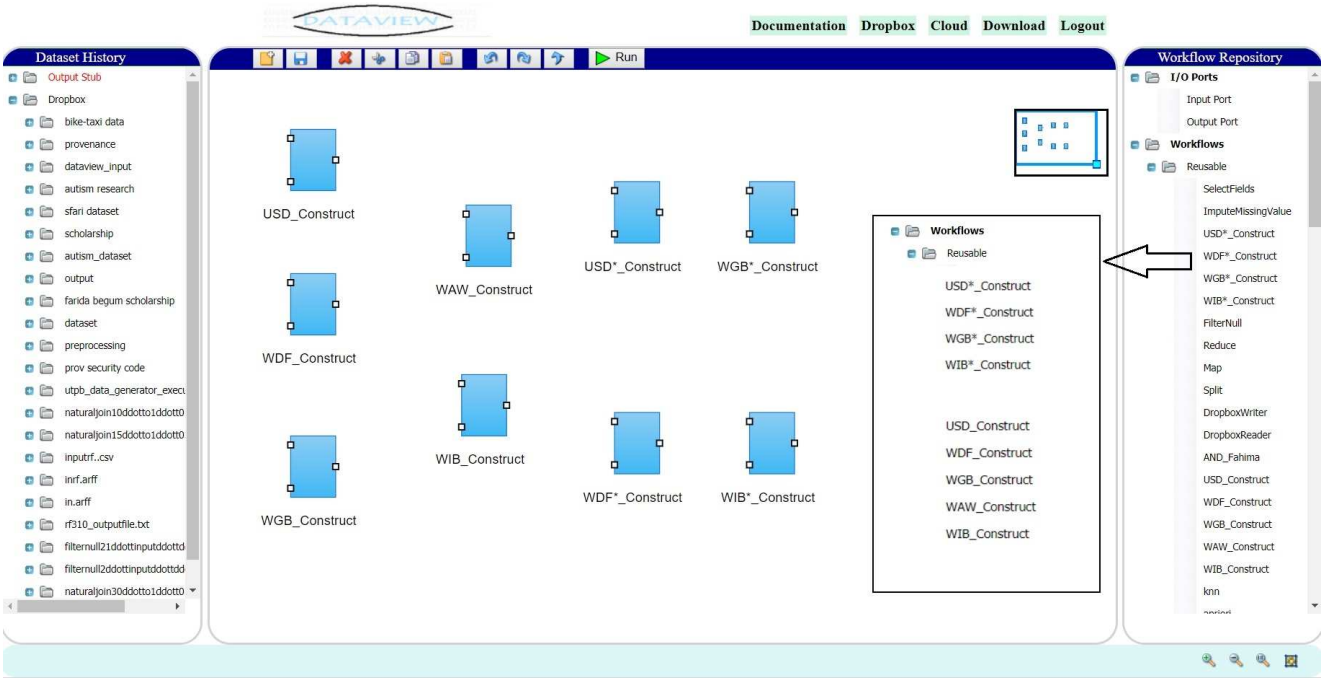


Figure 14: DATAVIEW with Primitive Query Construct Workflows.

We used DATAVIEW, a workflow design and configuration system which provides an intuitive GUI for users to design and configure workflow, and for experimenting our queries [20, 21]. For each provenance construct in PROV-DM model, we created one primitive, also known as a built-in query construct in DATAVIEW for querying big data. Fig. 14 shows each of the built-in constructs in there and zoom-in reusable query constructs as a built-in construct. To summarize the reason for using DATAVIEW is below:

- We support the built-in constructs for the PROV-DM model in DATAVIEW, which essentially provides API for users to take advantage of without dealing with low-level complexities. These constructs are powerful, yet easy to use.
- The constructs we built in DATAVIEW are reusable and compostable, which provide users the flexibility to use them alone for simple queries or compose them together to support more advanced queries.

- DATAVIEW itself is a scientific workflow system, these provenance captures, and query constructs also enable the capability of our system.

3.5.4 Executable Query Constructs in DATAVIEW

When the developers develop the primitive construct, it is comparatively effortless and convenient to built executable query construct. For each built-in or primitive query construct used, needed only the required number of inputs and output. For provenance, based on primitive construct, any user can create and run executable construct to see the query results. In Table. 5, we present each of the queries in the UTPB benchmark in DATAVIEW workflow system by showing the easy creation of executable constructs. In each example executable construct, the built-in constructs are used. A 2 input data product is shown. One is our big data file, and another one is text data file which gives the query input. The result will be saved in an output file. To provide more flexibility to the users, the current DATAVIEW has integrated Dropbox feature so that users can provide any big data file and a simple query in the text file; and drag-drop the construct and finally can get the query result in an output file. The output file is stored in Dropbox and can be accessed in a Dropbox folder. In this way, end users do not have to deal with the underlying complexity and can easily obtain query results.

Now we present some composite queries which requires combination of constructs and also union, intersect, and setdifference operators to get the query results.

- **Composite Query 1**: *Find all the Entities that have common activities and are originated from entity values X and Y.*

  The first phase of this query entails unveiling all the tasks that process entities X and Y. Hence we employ a pair of $USD^{\wedge}$ construct associated with X and Y respectively to obtain the list of downstream tasks. In order to find common activities we input them into “Intersect" construct. Finally, to obtain the final data products, we pipe the output of the intersect construct through $WGB^{\wedge}$ construct.

Fig. 15 shows executable construct for getting query result of composite query 1.

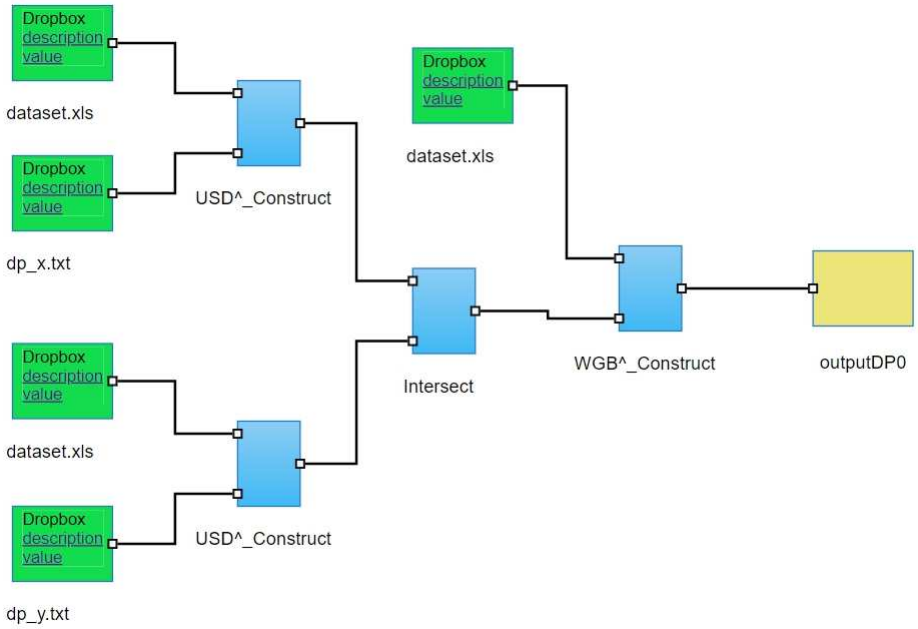


Figure 15: DATAVIEW Query Execution for Composite Query 1.

- **Composite Query 2**: *Find all the Entities except entity X and Y such that they are generated by entity Z.*

  The query identifies source entities for entity Z, we start by extracting all the activities that generates entity Z. In order to obtain the activities, we employ WGB^* construct which pipes into USD^* construct, which results in set of source entities that get transformed into Z. However, since we are interested in filtering out X and Y entities, we first do “Union" operation of both X and Y, then make use of “SetDifference" construct to filter out X and Y entities from entities we get from USD^* construct. The output of “SetDifference" construct gives us the final set of entities.

  Fig. 16 shows executable construct for getting query result of composite query 2.

- **Composite Query 3**: *Find union of all the entities that were generated via a set of common activities either via entities X and Y or via entities P or Q.*

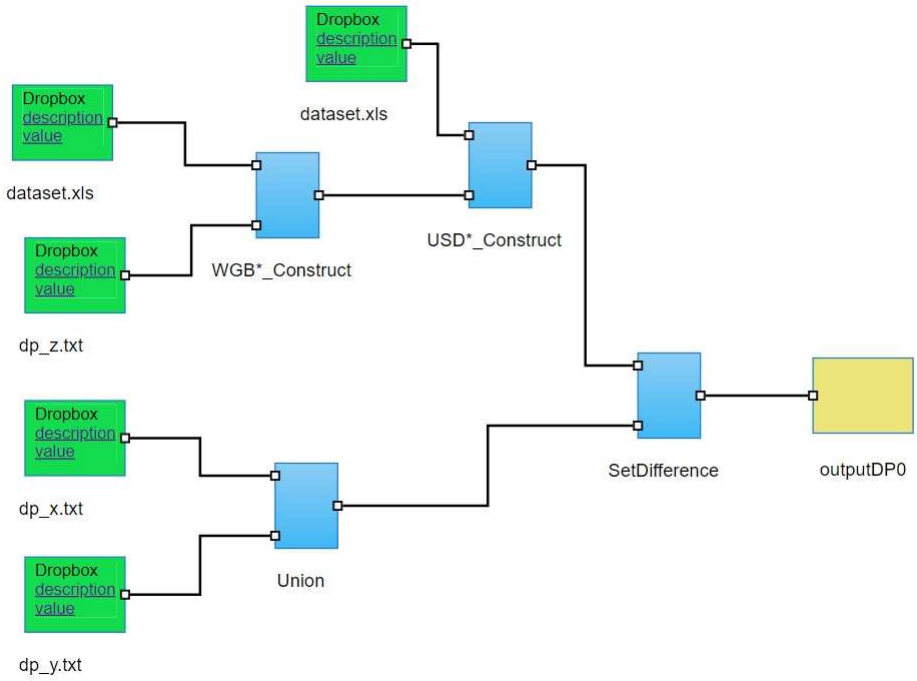


Figure 16: DATAVIEW Query Execution for Composite Query 2.

This query is an extension of query 1. Here we replicate query 1 twice to obtain all the entities that have a common activity. In first phase we employ a pair of $USD^{\wedge}$ construct associated with X and Y respectively to obtain the list of activities, then to find commom activities we input them to “Intersect" construct. We pipe the output of this construct to $WGB^{\wedge}$ construct to obtain the entities. We perform the same set of operations by providing the initial entities P and Q, instead of X and Y. Finally, we pipe both output through “Union" construct for our final result.

Fig. 17 shows executable construct for getting query result of composite query 3.

- **Composite Query 4**: *Find union of all source entities that generated all entities attributed to scientists X and Y.*

In the first phase of this query we identify the data products that are attributed

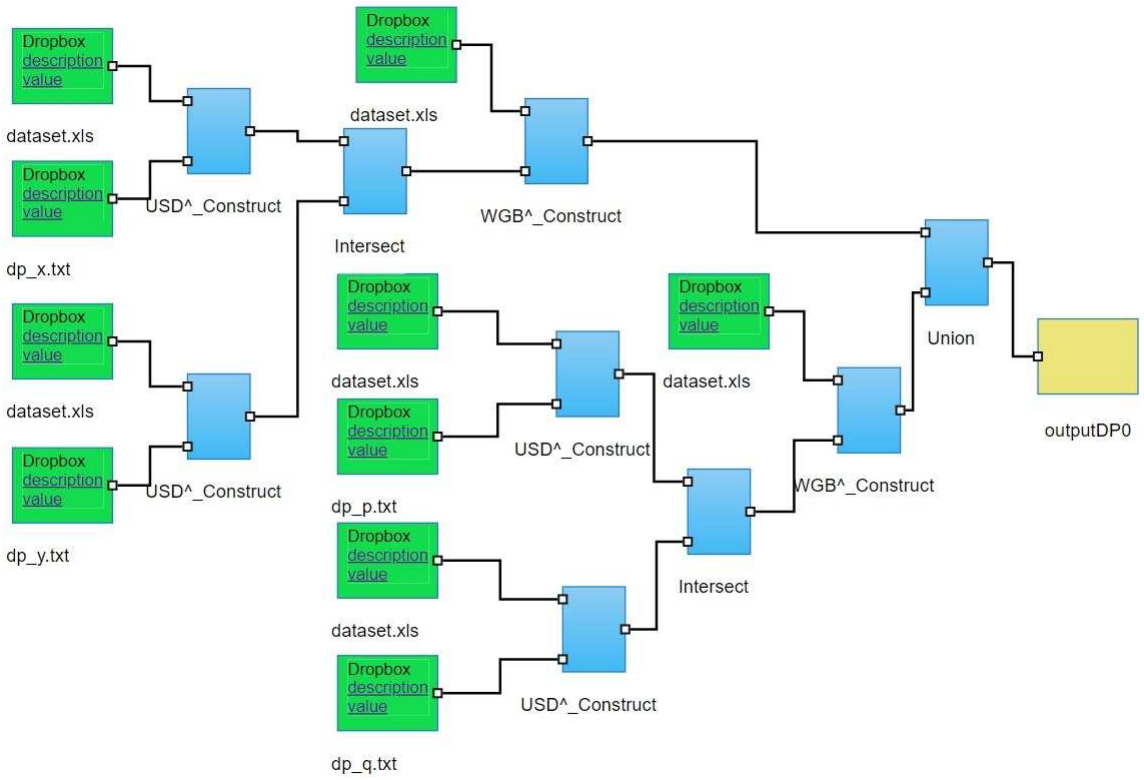


Figure 17: DATAVIEW Query Execution for Composite Query 3.

to scientists X and Y, and we employ 2 $WAT^{\wedge}$ construct for that purpose, one for scientist X and another one is for scientist Y. The output of both $WAT^{\wedge}$ construct then pipe via WGB^{*} to identify the upstream activities. Subsequently, the output of this phase of the query is piped into USD^{*} to extract the input data products. Lastly, a “Union" construct is instrumented to combine input entities for scientists X and Y.

Fig. 18 shows executable construct for getting query result of composite query 4.

- **Composite Query 5**: *Find all common source entities that generated all entities attributed by scientists X and Y.*

To find all common source entities, we first identify the data products that are attributed to scientists X and Y, and employ $WAT^{\wedge}$ for that purpose. Then we identify the upstream activities via WGB^{*} and output of this phase is piped into USD^{*} to

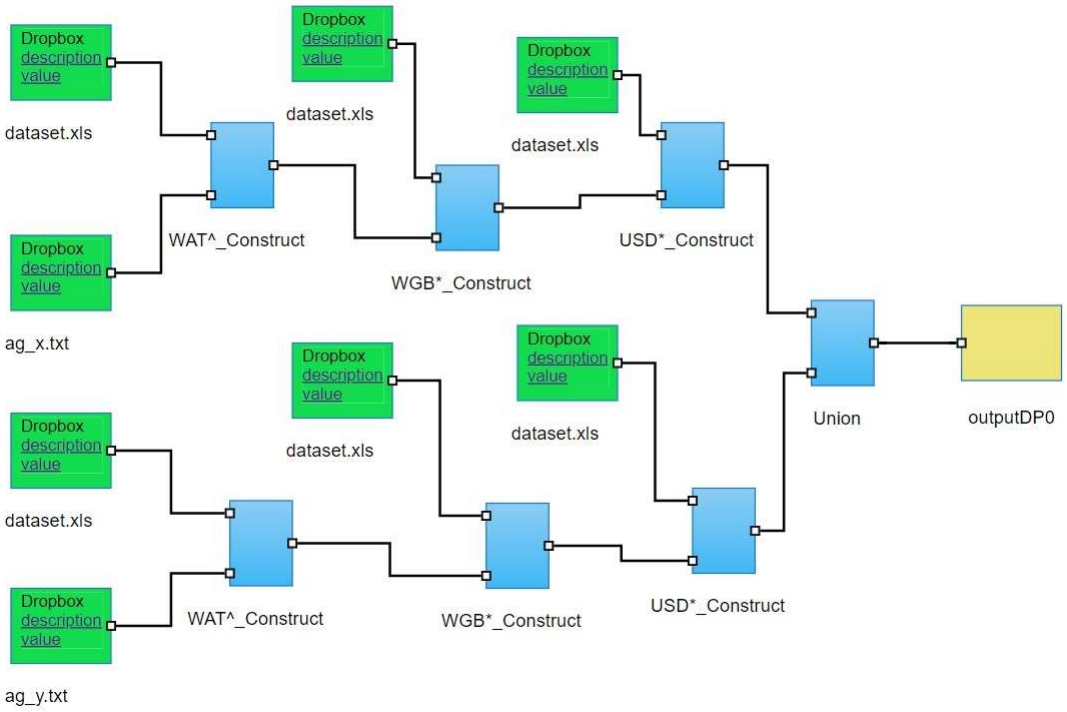


Figure 18: DATAVIEW Query Execution for Composite Query 4.

extract the input entities. Finally, “Intersect" construct finds out only common source entities for both scientists X and Y.

Fig. 19 shows executable construct for getting query result of composite query 5.

As shown in Fig. 20, we executed composite queries on our cluster to benchmark their performance. We noticed that for query 1, since it contains a chain of single step queries, gets executed relatively faster than other queries. This is comparable to query 3, which is also a chain of single step query. Queries 2, 4 and 5, however, chain together, multiple multi-step queries, which in turn spawn multiple map-reduce jobs. Hence these queries are more expensive. In hadoop ecosystem, mappers and reducers, which are full fledged programs are spawned on the system and data are shuffled across nodes. Hence, bootstrapping a job in the cluster takes time. This bootstrap time overhead, may seem significant when the data size is relatively small. However, the bigger the data size becomes, more negligible the overhead becomes and full potential of the cluster becomes noticeable.

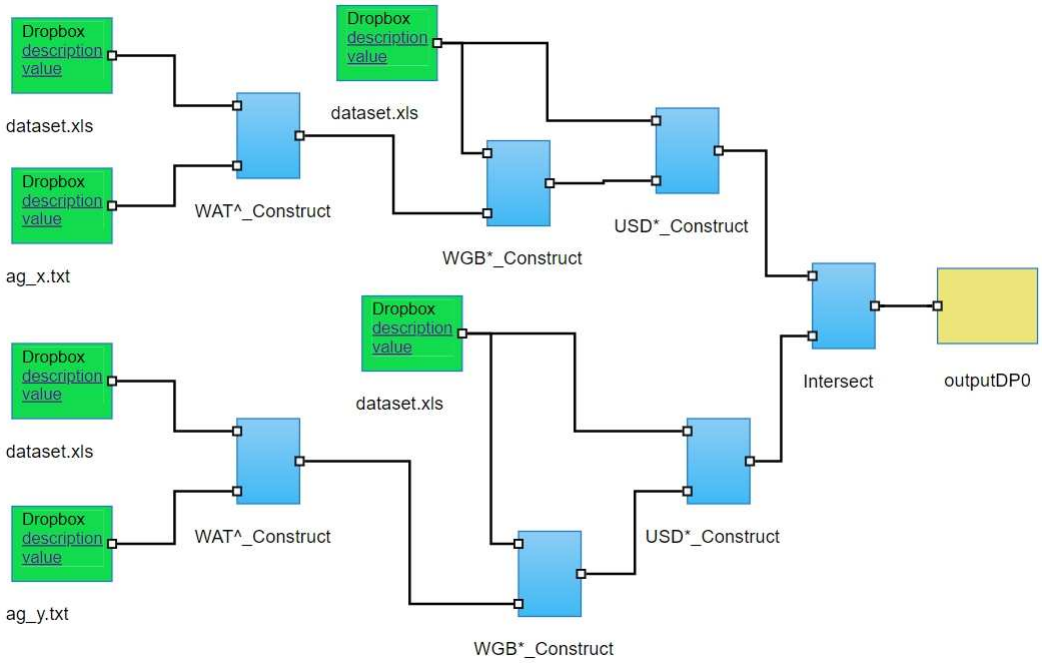


Figure 19: DATAVIEW Query Execution for Composite Query 5.

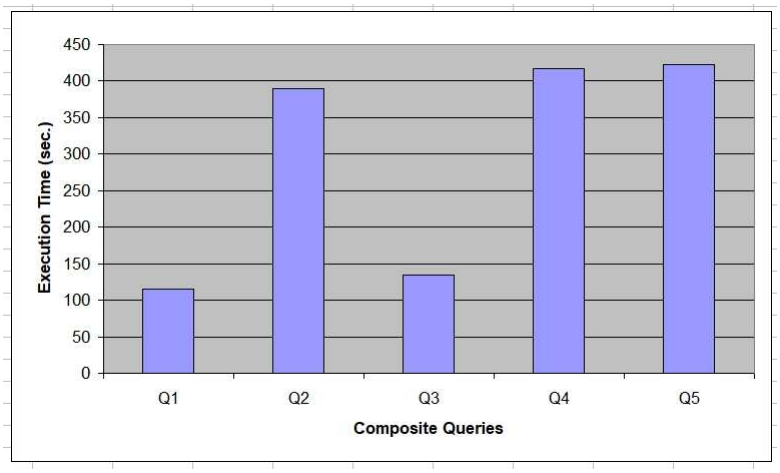


Figure 20: DATAVIEW execution time for Composite Queries.

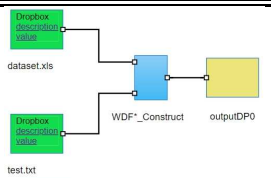
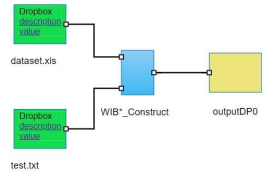
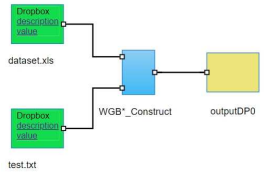
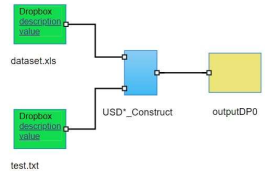
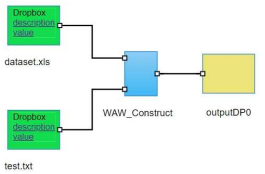
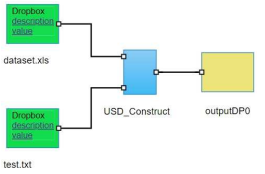
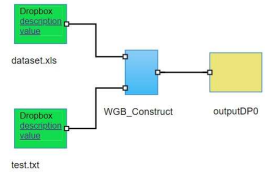
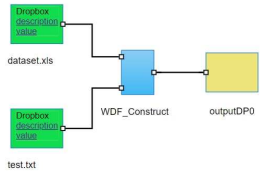
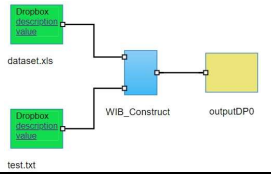
3.6 Conclusions and Future Work

In this chapter, we reported the $OPQL^{Pig}$ query language, extending OPQL for large datasets by implementing the translation operation of OPQL to Pig Latin data flow language with elongated features to make the query language scalable, robust, reliable and

parallel for working on top of Hadoop Distributed File System. This query language relishes the MapReduce model without reinventing common functionality such as join, filter and so on. Based on the graph pattern, provenance graph algebra and syntax-semantics of single-step-edge-forward and Multi-step-edge constructs, the $OPQL^{Pig}$ translation covers every scenario.

In the future, we would like to expand our research in three major directions. First, we plan to extend our query language, so that it can read input from and write output to sources other than HDFS, for example from NoSQL Databases like HBase or Cassandra. Second, we plan to focus on optimization issues in $OPQL^{Pig}$ query language and conduct the experimental study based on current system and NoSQL database techniques. Finally, we will make $OPQL^{Pig}$ applicable to other queries such as sub-graph isomorphism, pattern matching, and shortest path.

Table 5: The Result of each UTPB Query in DATAVIEW.

Query	*Graphical Query Execution*
Find all Entities of a Provenance Graph	
Find all Activities of a Provenance Graph	
Given one Entity to Find all Activities of a Provenance Graph	
Given one Activity to Find all Entities of a Provenance Graph	
Find Association-with dependencies for a Activity	
Find the Entity Used dependencies for a Activity	
Find the Activity Generation dependencies for a Entity	
Find the Entity Derivation dependencies for a Entity	
Find the Activity Informed-by dependencies for a Activity	

CHAPTER 4 SECURITY MANAGEMENT IN PROVENANCE

In Scientific workflow Provenance system, a successful collaboration of information and resources are required. For secure and flexible adaptation of different environments, adequate access control policies are fundamentally imperative. Data products and derivation history are essential for recomputing scientific results, and effective access control mechanism are indispensable for all the sensitive data and processes. In this chapter, we do the following 1) Propose a role-based access control model for scientific workflow provenance; 2) Define three quality requirements for scientific workflow provenance access control policies - consistency, completeness, and conciseness, 3) Provide a mapping from specifications over workflows to their counterparts on provenance that preserves the quality properties, and 4) Provide a case study on a scientific workflow for autism behavioral data analysis to show the feasibility of our proposed analysis algorithms.

4.1 Introduction

Security in provenance is an important research topic in a scientific workflow system. Provenance systems must adhere to the same security and access control protocol that the workflow system supports and maintains, otherwise sensitive data and access pattern might be susceptible to vulnerability. There has been a great deal of research on capturing, managing and using workflow provenance information, but for shared public and scientific data little progress has been made on defining provenance security.

Sensitive data, lineage and traces inherently necessitate access control and access privilege agnostic, globally accessible view of provenance makes scientific workflows vulnarable to security breach. Thus, to comply with the inherent workflow security protocol, provenance systems need to maintain a different view of information for different roles based on the privilege associated with it.

In the combination of security and provenance, we have to consider the following scenarios. Cases where security and access control is not of concern in the workflow, provenance security need not be any more restrictive than the parent workflow. However, a

less rigorous access control in the provenance specification makes the security protocol for the workflow less efficient. Hence, provenance specification must provide a variable view as it pertains to the workflow to the user based on the security protocol. While decoupling the security specification for provenance may seem more flexible, it introduces additional complexity in the system. In this chapter, we delineate how provenance specifications can inherit security protocols of the parent workflow and adhere to the same level of guarantees.

Scientific workflows and their applications have been monumental in many industries, especially in health informatics. Health informatics, however, have a unique data privacy requirement, access to data has to be strictly enforced and monitored. With the implementation of HIPPA, it has become imperative that scientific workflows and their provenance management systems adhere to strict role-based access control protocol. If we model a scientific workflow and its provenance as two distinct DAGs, each of the nodes and edges need to be augmented with access control requirement. Moreover, all of these constituent components should maintain a user and role-specific access control ensure compliance.

Traditionally analysis of access control policies have limitations that they are not able to incorporate the dynamic execution of workflow information into account.

4.1.1 Security in Workflow vs. security in Provenance

Since scientific workflow captures the intellectual property of scientific experiments and composition of various computational services into the workflow, security in workflow protects the access to those workflow tasks and data to provide access control to those crucial scientific results. There can be different scenarios in perspective of providing access control policies in the workflow, for example, based on scientist's preferences one can only publish source data and final scientific results, but not the scientific workflow altogether. Whereas, for other scientist's they can publish source data, scientific results and all the workflow used there, but keep the parameter setting as a secret for the workflow.

The security in provenance is an important aspect in the scientific workflow. As prove-

nance captures all the derivation history including original data sources, intermediary data products and all the steps involved to produce those data products. Imposing security means implementing access control policies on those data products(source, intermediary, final) and the dependencies among them. Provenance access control policies can be applied and can release provenance information of source data, scientific results, and parameter setting. They still can hide an intellectual property of certain provenance information.

Access control policies can be applied to composite tasks or sub-workflows of provenance or at different abstraction levels, where users are only allowed to access provenance information based on their requirements and preferences. In provenance security, there are no foundational models yet, to define and relate security goals such as availability, confidentiality, and privacy. For making meaningful progress on these issues, a foundational model will be outlined and developed.

4.1.2 Examples for Importance of Provenance Security

- Without proper provenance or in circumstances of provenance failure, information could be misinterpreted. An old news article can bring misinterpretation when the date of information is not stored and can tie up with sudden economic loss[38].

- For the scientist, any lack of information makes it difficult for reviewers to evaluate contributions of the authors. Keeping provenance of those scientific discoveries aim to help to keep transparency and repeatability [38].

- Unintentional provenance information can violate privacy and confidentiality too. One example from [38] is: The government documents published in word version, has a tremendous risk of privacy and confidentiality.

- At the end of the process of peer-review, the content of the reviews are delivered to the authors, but the identity of the reviewers are not delivered. Here the reviews (data) are public, but who wrote the review (provenance) is confidential.

- In the letter of recommendation, the subject of the letter is not allowed to know

the content, but allowed to know the author. Here the content of letter (data) is confidential, but the author of the letter (provenance) is public.

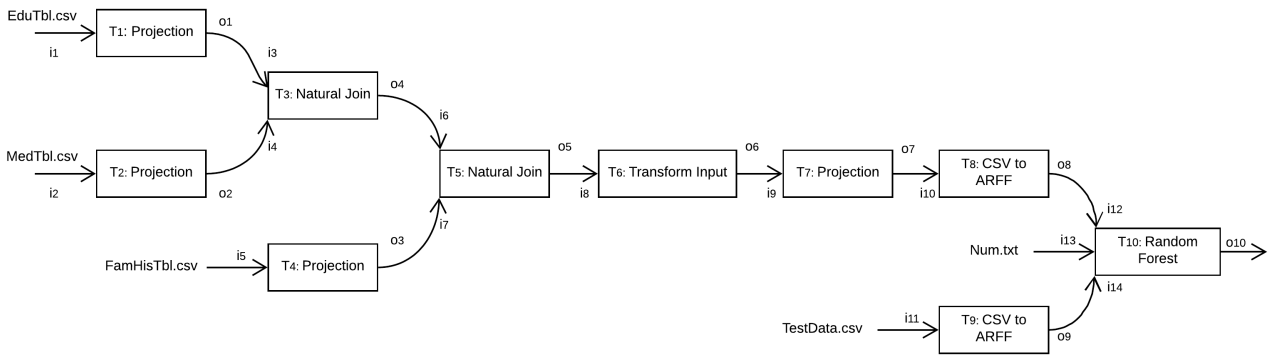


Figure 21: Autism Workflow.

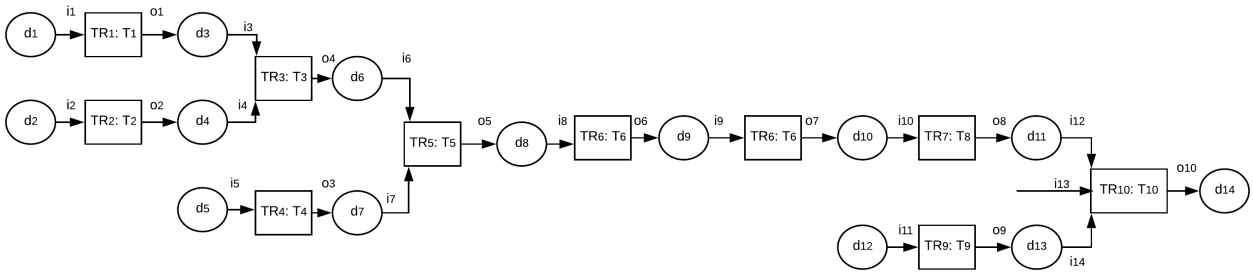


Figure 22: Provenance of Autism Workflow.

The rest of this chapter is organized as follows: Section 4.2 defines basic terminologies of the security framework. Section 4.3 of access control policies for workflow provenance system. Section 4.4 presents a formal security scientific workflow specification for task, port and data channel with proposed algorithms of access control policies. Section 4.5 formalizes a mapping between workflow to security view and presents security view for provenance. The analysis of those policies in perspective of policy quality requirements to find out these evolving policies are consistent, complete, and concise is presented in

Section 4.6. This section provides proof of the maintenance of policy quality requirements for provenance. Section 4.7 modifies the policies based on quality requirements. Section 4.8 presents prototype and provides an example from the Autism community to show the validity of our quality of access control policies for provenance.

4.2 Provenance security framework

For a provenance security framework, formal and precise security properties like confidentiality, privacy, and availability are needed for enforcing suitable and desired security policy are desirable.

In the era of big data, scientific workflows have become essential to automate scientific experiments and guarantee repeatability. Increasingly in many scientific domains such as health and medication, personalization in information processing has become key to success. Hence, access control protocols in scientific workflows have become a prerequisite. Workflow provenance systems, while makes managing data and process lineage possible, also need to adhere to the access control protocol inherent in the scientific workflows. Here, we propose a security scientific workflow specification with role-based access control policy and demonstrate how the policy is inherited by the workflow provenance system. We characterize the desirable properties of role-based access control protocol in scientific workflows and delineate how the properties are maintained in the workflow provenance systems as well.

The concept is illustrated with an example from health informatics. In such an application secure communication in scientific workflow plays an imperative part for Autism Spectrum Disorder. In [20], an autism workflow system has been created for analyzing, predicting, classifying and mining big pool of autism data. From the security perspective, accessing and analyzing these sensitive data should be handled based on a particular set of users for a particular role. For this reason, we need a provenance security framework to allow permission for specific task and data products for specific roles. Ideally, in the Autism community, parents can have full access to all the diagnosis data which includes

medical, therapeutic, school and other information. But, for the school district, teachers by default do not have privileges to see child's medical details unless explicitly granted by the parents. Similarly, therapists could have access to certain sensitive part of the workflow, but not all. For implementing a secure communication of workflow in autism community a security framework is needed.

Below we define the basic PROV-DM provenance graph and access control policy.

Definition 4.1 (Provenance Graph). *A provenance graph PG = (N, E) consists of:*

- *a set of nodes* $N = EN \bigcup AC \bigcup AG$*, where EN is a set of entities, AC is a set of activities, and AG is a set of agents, based on the PROV-DM model.*

- *a set of directed edges E =* $E_u \bigcup E_g \bigcup E_d \bigcup E_i \bigcup E_a \bigcup E_{ab} \bigcup E_{at}$
  where i) $E_u \subseteq$ *AC* $\times$ *EN and (ac, en)* $\in E_u$ *means that activity ac* used *entity en.*
  ii) $E_g \subseteq$ *EN* $\times$ *AC and (en, ac)* $\in E_g$ *means that entity en* was generated by *activity ac.*
  iii) $E_d \subseteq$ *EN* $\times$ *EN and (*en_1*,* en_2*)* $\in E_d$ *means that entity* en_1 was derived from *entity* en_2*.*
  iv) $E_i \subseteq$ *AC* $\times$ *AC and (*ac_1*,* ac_2*)* $\in E_i$ *means that activity* ac_1 was informed by *activity* ac_2*.*
  v) $E_a \subseteq$ *AC* $\times$ *AG and (ac, ag)* $\in E_a$ *means that activity ac* was associated with *agent ag.*
  vi) $E_{ab} \subseteq$ *AG* $\times$ *AG and (*ag_1*,* ag_2*)* $\in E_{ab}$ *means that agent* ag_1 acted on behalf of *agent* ag_2*.*
  vii) $E_{at} \subseteq$ *EN* $\times$ *AG and (en, ag)* $\in E_{at}$ *means that entity en* was attributed to *agent ag.*

Definition 4.2 (Role Based Access Control). *Let Role-Based Access control* $\hat{R}$ *for provenance security be defined as a tuple (U, R,* A*, W, E,* ϕ*,* μ *), where*

- *U is a set of users;*

- *R is a set of roles;*
- *A is a set of actions;*
- *W is a workflow;*
- *E is the set of elements in workflow W including all the tasks, ports, and data channels.*
- *ϕ: $R \times E \times A \rightarrow \{0, 1\}$ is a function that maps permissions for roles, elements, and actions to 0 or 1. Here, 0 denotes restricted access and 1 denotes full access.*
- *μ: $U \rightarrow R$ is a function that maps users to their roles.*

The function ϕ is further defined as:

$$\phi(e, r, \alpha) = \begin{cases} \Gamma(e, r, \alpha), & \text{if } e \text{ is a task} \quad (4.1a) \\ \rho(e, r, \alpha), & \text{if } e \text{ is a port} \quad (4.1b) \\ \delta(p_1, p_2, r, \alpha), & \text{if } (p_1, p_2) \text{ is a data channel} \quad (4.1c) \end{cases}$$

For the function ϕ the element could be either task, port or data channel. For task we define the function Γ, for the port we define the function ρ and for the data channel, we define the function δ. The functions Γ, ρ and δ are defined in the following sections.

4.3 Provenance Security Policy Life Span

The provenance security policy life cycle is composed of four iterative stages: i) Security policy specification, ii) Security policy enforcement, iii) Security policy analysis, and iv) Security policy evaluation. The administrator of access control policies coordinates with the system users and determines the policies to be enforced in either or all task, port and data channel level. In security policy enforcement stage, based on system users access on protected elements, the policies are applied to either grant or restrict access. In correspondence to context or environment of the application, the policies evolve to adopt correlated changes. In policy analysis, policy quality requirements are analyzed. This phase analyzes the policy qualities like consistency, completeness, conciseness to make sure the proposed

policies adhere to all those qualities. Finally, in policy evaluation, we evaluate quality requirements and identify any quality discrepancy and modifies those policies based on the identified discrepancy in policies. Fig. 23 shows a graphical representation of provenance security policy lifespan.

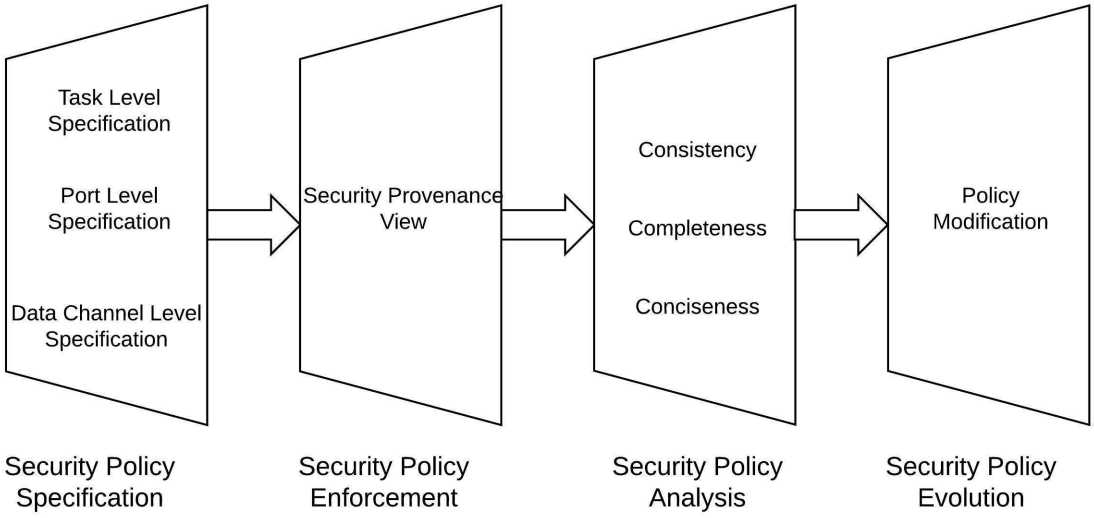


Figure 23: Provenance Security Policy Life Span.

4.4 Security Policy Specification

4.4.1 Task Level Specification

Definition 4.3 (Task Annotation)**.** *A task level specification is denoted by* Γ*:* $T \times R \times A \rightarrow \{0, 1\}$ *that maps specific user and tasks to the permission level and is defined by:*

$$\Gamma(t, r, \alpha) = \begin{cases} Invalid, & \text{if } \Pi(t, r, \alpha) = 1 \text{ and } \Gamma(t_p, r, \alpha) = 0 \quad (4.2a) \\ \Pi(t, r, \alpha), & \text{if } \Pi(t, r, \alpha) \neq -1 \quad (4.2b) \\ \Gamma(t_p, r, \alpha), & t_p \text{ is not null and } \Pi(t, r, \alpha) \neq -1 \quad (4.2c) \\ Invalid, & t_p \text{ is null and } \Pi(t, r, \alpha) \neq -1 \quad (4.2d) \end{cases}$$

In task specification, the access permission can be annotated by 0 or 1. Here we define

a function Π: $R \times E \times A \rightarrow \{0, 1, -1\}$, that returns permission of role, element, and action triplet. If it returns -1, it means there is no explicit specification for (t, r, α); otherwise, it return the explicit annotation for the triple (t, r, α).

If the permission is not explicitly specified in RBAC then child task t can inherit permission from task t_p, here t_p denotes parent of t, $\alpha \in A$, $r \in R$. In other words, the task level security specification, if explicitly stated, is validated against consistency requirement of the protocol. In this case, if the parent task does not have security access, the child task inherits the restriction, and this restriction cannot be overridden by explicit specification. One important feature of the task is that when it is annotated as 1 then all other task, ports or data channels contained in task T should be accessible otherwise a 0 annotation is explicitly specified or derived from them.

Our definition captures the inconsistency specification between a task and any of its ancestors while [36] only captures the inconsistency specification between a task and its parent task; parent(t) the task that immediately contains task t.

Here we have 4 cases:

- Case a: If the parent task differs with the child task in question in terms of access control permission such that the parent task does not have access yet, the child task has explicit specification to have secure access, this will result in inconsistency in access control protocol.

- Case b: If the task in question has access control protocol explicitly specified then this will override ancestral access control protocols.

- Case c: If the current task does not have explicit specification but has a valid parent then it will inherit it's parent's access control privileges.

- Case d: Lastly, if the current task does not have a valid parent and valid specification, an exception will be thrown.

The permission specification can be calculated using function “FindTaskSpec" in Algorithm.

Algorithm 1: Algorithm for calculating security specification in Task

Input: Task t, Role r, Action α.
Output: Task security annotation $< t, a \in \{0, 1\} >$.

```
1 If ∃ a = Π(t, r, α)
2         a_p = FindTaskSpec(t_p, r, α), where t_p ∈ Parent (t)
3         if (a & ! a_p)
4                 return Invalid
5         return < t, a >
6 else
7         return < t, FindTaskSpec(t_p, r, α) >
```

Figure 24: Task Level Security Specification.

4.4.2 Port Level Specification

Definition 4.4 (Port Annotation)**.** *A port level specification is denoted by* ρ*:* $P \times R \times A \rightarrow \{0, 1\}$ *that maps specific role and ports to the permission level and is defined by:*

$$\rho(p, r, \alpha) = \begin{cases} Invalid, & \text{if } \Pi(p, r, \alpha) = 1 \text{ and } \Gamma(t_p, r, \alpha) = 0 \quad (4.3a) \\ \Pi(p, r, \alpha), & \text{if } \Pi(p, r, \alpha) \neq -1 \quad (4.3b) \\ \Gamma(t_p, r, \alpha), & \text{otherwise} \quad (4.3c) \end{cases}$$

Ports can be specified with 0 or 1. In the Port level specification, when any port has no specified security specification, then that inherits either access or denied permission from owning task. The administrator can explicitly specify all or some ports access permissions. For all workflow run, the port annotation 1 or 0 specified for any given task, demonstrate the accessibility of data product.

Here we have 3 cases:

- Case a: If the parent task does not have access permission, but the port contained in that task has explicit specification to have secure access, then this will result in invalid access control protocol.

- Case b: If the port in question has access control protocol explicitly specified then this will override ancestral access control protocols.

- Case c: If the port does not have explicit specification but it's containing task has access control specified then it will inherit the task's access control privileges.

Here, t_p denote the owner task of port p.

In appearance, our port-level security specification is the same as [36], but it improves the inconsistency specification check due to the improvement of task-level security specification, which affects the result of port-level specification inconsistency check.

Our port-level specification is greatly simplified from our previous definition as we do not allow the accessibility of a data channel when its respective ports are not accessible.

The annotation of port is calculated by function "FindPortSpec" in Algorithm.

Algorithm 2: Algorithm for calculating security specification on Port

Input: Port p, Role r, Action α.
Output: Port security annotation $< p, a \in \{0, 1\} >$.

```
1 If ∃ a = Π(p, r, α)
2         a_p = FindTaskSpec(t_p, r, α), where t_p is the
  task that contains p
3         if (a & ! a_p)
4                 return Invalid
5         return < p, a >
6 else
7         return < p, FindPortSpec(t_p, r, α) >
```

Figure 25: Port Level Security Specification.

4.4.3 Data Channel Level Specification

Definition 4.5 (Data Channel Annotation)**.** *A data channel level specification is denoted by* δ*:* $P \times R \times A \rightarrow \{0, 1\}$ *that maps specific roles and ports to the permission level and is defined by:*

$$\delta(p_1, p_2, r, \alpha) = \begin{cases} \rho(p_1, r, \alpha), & \text{if } \rho(p_1, r, \alpha) = \rho(p_2, r, \alpha) \quad (4.4a) \\ Invalid & \text{Otherwise} \quad (4.4b) \end{cases}$$

The Data Channel specification is quite straight-forward. When both ports have access permission, then data channel must have access permission. When both ports permission is denied, the data channel's permission is denied too.

Our definition greatly simplified the specification effort at a small cost of not allowing the specification of data dependency without the accessibility of respective ports, which has a very rare use case in practice.

The permission specification can be calculated using function "FindDataChannelSpec" in Algorithm.

Algorithm 3: Algorithm for calculating security specification on Data Channel

Input: p_1, p_2, Role r, Action α
Output: Port security annotation
$< (p_1, p_2), a \in \{0, 1\} >$.

```
If (FindPortSpec(p1, r, α) = FindPortSpec(p2, r, α))
    return < (p1, p2), FindPortSpec(p1, r, α) >
else
    return Invalid
```

Figure 26: Data Channel Level Security Specification.

4.5 Security Policy Enforcement

In security policy enforcement, provenance systems can maintain a different view of information for different roles and enforce associated privileges.

We define security provenance view as a restricted view of a provenance only consisting of that information that users are authorized to access. To illustrate this view in PROV-DM model we graphically represent the provenance model relation “Used" in Fig. 27 and “wasGeneratedBy" in Fig. 28 and corresponding mapping from workflow to provenance. Table. 6 shows the specification mapping from workflow to provenance.

Let E be the elements in a workflow consisting of tasks, ports and data channel and let Ψ be a mapping function $\Psi : E \rightarrow N$ that maps elements in the workflow to their corresponding nodes in the provenance graph. The inverse function $\Psi^{-1} : N \rightarrow E$ returns the reverse mapping.

We also introduce the following two notations, Let $\Im : E \rightarrow E$ be a function defined as follows:

$$\Im(e) = \begin{cases} e, & \text{if } e \text{ is task} \quad (4.5a) \\ t_p, & \text{if } e \text{ is port, } t_p \text{ is container task.} \quad (4.5b) \end{cases}$$

Let $\wp : E \rightarrow E$ be a function defined as follows:

$$\wp(e) = \begin{cases} e, & \text{if } e \text{ is port} \quad (4.6a) \\ \{p_e\}, & \text{if } e \text{ is task, } \{p_e\} \text{ are ports of } e. \quad (4.6b) \end{cases}$$

Definition 4.6 (Security Provenance View of Used Relation)**.**

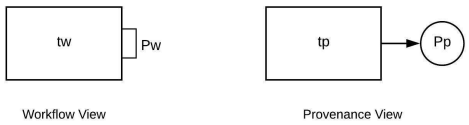


Figure 27: Provenance Security in USED Relation.

- $\Gamma(\Psi(t_w)$,r,view) = $\Gamma(t_w$,r,view)
- $\Delta(\Psi(P_w)$,r,view) = $\rho(P_w$,r,view)

- ζ(edge ($\Psi(t_w)$, $\Psi(P_w)$), r, view) = $\Gamma(t_w$,r,view)

Definition 4.7 (Security Provenance View of wasGeneratedBy Relation)**.**

Figure 28: Provenance Security in wasGeneratedBy Relation.

- $\Gamma(\Psi(t_w)$,r,view) = $\Gamma(t_w$,r,view)
- $\Delta(\Psi(P_w)$,r,view) = $\rho(P_w$,r,view)
- ζ(edge ($\Psi(P_w)$, $\Psi(t_w)$), r, view) = $\Gamma(t_w$,r,view)

Table 6: RBAC Security Specification for “Used" and “wasGeneratedBy" Relation.

Workflow RBAC			***Provenance RBAC***	
Task	***Port***	***Task***	***Data Product***	***Relation***
+	-	+	-	+
+	+	+	+	+
-	-	-	-	-
-	+		INVALID	

We illustrate security policy requirements based on Autism provenance system in 22 and defines those access control policies in Table 7.

4.6 Security Policy Quality Requirements and Analysis

We define and illustrate our security policy quality requirements below:

4.6.1 Consistency

acp_i and acp_j are consistent if and only if

Table 7: Role Based Access Control Policy for Provenance System.

Access Control Policy	***Role***	***Permission***		
		Element	***Action***	***Sign***
acp_1	Parents	T_1	Read	+
acp_2		i_1	Read	+
acp_3		T_2	Read	+
acp_4		i_2	Read	+
acp_5		T_4	Read	+
acp_6		i_5	Read	+
acp_7		T_9	Read	+
acp_8		O_{10}	Read	+
acp_9	Teachers	i_1	Read	+
acp_{10}		T_2	Read	+
acp_{11}		i_2	Read	−
acp_{12}		O_1	Read	+
acp_{13}		O_2	Read	+
acp_{14}		O_6	Read	−
acp_{15}		i_9	Read	+
acp_{16}		O_{10}	Read	+
acp_{17}	Therapist	T_1	Read	+
acp_{18}		i_1	Read	+
acp_{19}		T_2	Read	+
acp_{20}		i_2	Read	+
acp_{21}		T_4	Read	+
acp_{22}		T_5	Read	+
acp_{23}		T_9	Read	+
acp_{24}		T_{10}	Read	+
acp_{25}		O_{10}	Read	+

acp_i.u = acp_j.u, $\wedge$ $\mu(acp_i.u) = \mu(acp_j.u)$ $\wedge$ acp_i.e = acp_j.e, $\wedge$ acp_i.a = acp_j.a $\implies$ $\phi(\mu(acp_i.u),e,a) = \phi(\mu(acp_j.u),e,a)$,

$\forall$ u $\in$ U, $\forall$ e $\in$ E, $\forall$ a $\in$ A

Here we refer consistency between two policies acp_i and acp_j where for the same user with the same role, same element, and activity, both policies should have the same access rights. If one policy allows access implies another policy allows access too. If there is any inconsistency in policy, that requires conflict resolution which can be minimized with consistent policies.

Example 1**:** As shown in Table. 7, for teachers role, acp_{14} and acp_{15} are not consistent. Based on our specification an access control policy, both policies need to have the same access rights when they have the same role, user, element, and activity. Here acp_{14} and acp_{15} do not meet that criteria. They are inconsistent because one port is specified as

negative access whereas at another end of data channel another port specified as positive access. In 7, for a single data channel the output port O_6 specified negative and the input port i_9 specified positive. From our Port level specification algorithm, both ports should have same permission. In this case, the output and the input port of single data channel have different permissions. Therefore, this is inconsistency in policy. We can correct this inconsistency in policy evolution phase.

4.6.2 Completeness

Any access control policy acp_i is complete if and only if

$\forall$ i, $\mu(acp_i.\text{u})$ is defined $\wedge$ $\phi(\mu(acp_i.\text{u}),\text{e},\alpha)$ is defined;

where $\exists$ u $\in$ U, $\exists$ e $\in$ E, $\exists$ $\alpha \in$ A

Completeness of an access control policy is where for any roles access control policy is defined. A complete access control policy has both role defined and access policy defined. An incomplete policy has either role undefined or access policy for task/port undefined.

Example 2**:** In Table. 7, there is no access control policy for teachers role for allowing or denying access to Family History table dataset of Task T_4. Without setting up the access control policy for input i_5 or task T_4 the policy defined accessing or denying the information of family history is incomplete.

4.6.3 Conciseness

An access control policy $acp_i \in \hat{R}$ is concise if and only if;

$\exists\ acp_j \in \hat{R} \wedge \mu(acp_i.\text{u}) = \mu(acp_j.\text{u})$,

$\wedge\ acp_i.\text{e} = acp_j.\text{e}, \wedge\ acp_i.\text{a} = acp_j.\text{a}$,

$\wedge\ \phi(\mu(acp_i.\text{u}),\text{e},\text{a}) = \phi(\mu(acp_j.\text{u}),\text{e},\text{a}) \implies \text{i} = \text{j}$;

$\forall$ u $\in$ U, $\forall$ e $\in$ E, $\forall$ a $\in$ A.

The conciseness of access control policy means that for any policy if the role are the same, element same, the actions are the same, permissions are the same that means those implies to the same policy. If there are two access control policies acp_i and acp_j, where both policies have the same role, same user, same element and same activity, but defined

as two different policies then we refer these two policies are not concise or redundant.

Example 3**:** Based on access control policies in Table. 7, acp_{23} and acp_{24} are not concise. From task specification, we know that when the parent task's accessibility is positive then child task's accessibility is positive too unless otherwise stated. We do not have to specify both cases here.

Theorem 1. *If RBAC is in WF_{RBAC} is consistent, then RBAC in Provenance $Prov_{RBAC}$ is consistent as well.*

Proof. Lets assume that WF_{RBAC} is consistent and $Prov_{RBAC}$ is not consistent.
From the definition we know WF_{RBAC} consistent if and only if $i \neq j \wedge acp_i$.r $= acp_j$.r $\wedge$ acp_i.e $= acp_j$.e $\wedge$ acp_i.a $= acp_j$.a Implies ϕ(acp_i.r, acp_i.e, acp_i.a) = ϕ(acp_j.r, acp_j.e, acp_j.a).
If $Prov_{RBAC}$ is inconsistent then one or more of the following is true:
$\Gamma(\Psi(\Im(acp_i.e)), acp_i.r, acp_i.a) \neq \Gamma(\Psi(\Im(acp_j.e)), acp_j.r, acp_j.a)$ or
$\rho(\Psi(\wp(acp_i.e)), acp_i.r, acp_i.a) \neq \rho(\Psi(\wp(acp_j.e)), acp_j.r, acp_j.a)$ or
$\zeta(edge(\Psi(\Im(acp_i.e)), \Psi(\wp(acp_i.e))), acp_i.r, acp_i.a) \neq \zeta(edge(\Psi(\Im(acp_j.e)), \Psi(\wp(acp_j.e))), acp_j.r, acp_j.a)$

However, $\Gamma(\Psi(\Im(acp_i.e)), acp_i.r, acp_i.a) = \Gamma(\Im(acp_i.e), acp_i.r, acp_i.a)$ and
$\Gamma(\Psi(\Im(acp_j.e)), acp_j.r, acp_j.a) = \Gamma(\Im(acp_j.e), acp_j.r, acp_j.a)$.

Again since,
ϕ(acp_i.r, $\Im(acp_i$.e), acp_i.a) = ϕ(acp_j.r, $\Im(acp_j$.e), acp_j.a),

We can conclude,
$\Gamma(\Im(acp_i.e), acp_i.r, acp_i.a) = \Gamma(\Im(acp_j.e), acp_j.r, acp_j.a)$.
Hence,
$\Gamma(\Psi(\Im(acp_i.e)), acp_i.r, acp_i.a) = \Gamma(\Psi(\Im(acp_j.e)), acp_j.r, acp_j.a)$.

Similarly, we can show that,

$\rho(\Psi(\wp(acp_i.e)), acp_i.r, acp_i.a) = \rho(\Psi(\wp(acp_j.e)), acp_j.r, acp_j.a)$.

Lastly, since,

$\zeta(edge(\Psi(\Im(acp_i.e)), \Psi(\wp(acp_i.e))), acp_i.r, acp_i.a) = \Gamma(\Im(acp_i.e), acp_i.r, acp_i.a)$

and $\zeta(edge(\Psi(\Im(acp_j.e), \Psi(\wp(acp_j.e)), acp_j.r, acp_j.a) = \Gamma(\Im(acp_j.e), acp_j.r, acp_j.a)$ and

$\Gamma(\Im(acp_i.e), acp_i.r, acp_i.a) = \Gamma(\Im(acp_j.e), acp_j.r, acp_j.a)$,

We can conclude that

$\zeta(edge(\Psi(\Im(acp_i.e)), \Psi(\wp(acp_i.e)), acp_i.r, acp_i.a) = \zeta(edge(\Psi(\Im(acp_j.e)), \Psi(\wp(acp_j.e))), acp_j.r, acp_j.a)$.

So, $Prov_{RBAC}$ cannot be inconsistent.

□

Theorem 2. *If RBAC is in WF_{RBAC} is complete, then RBAC in Provenance $Prov_{RBAC}$ is complete as well.*

Proof. An access control policy acp_i is complete if and only if $\mu(acp_i.\text{u})$ is defined $\wedge$ $\phi(\mu(acp_i.\text{u}), acp_i.e, \alpha)$ is defined $\forall$ u $\in$ U, $\forall$ e $\in$ E, $\forall$ $\alpha \in$ A.

Again, since we are assuming that RBAC in $Prov_{RBAC}$ is incomplete:

- $\Gamma(\Psi(\Im(acp_i.e))$,r,view) is undefined)
  $\vee$ $\Delta(\Psi(\wp(acp_i.e))$,r,view) is undefined
  $\vee$ ζ(edge ($\Psi(\Im(acp_i.e))$, $\Psi(\wp(acp_i.e))$), r, view) is undefined.

However, since,

- $\Gamma(\Psi(\Im(acp_i.e))$,r,view) = $\Gamma(\Im(acp_i.e)$,r,view)
- $\Delta(\Psi(\wp(acp_i.e))$,r,view) = $\rho(\wp(acp_i.e)$,r,view)
- ζ(edge ($\Psi(\Im(acp_i.e))$, $\Psi(\wp(acp_i.e))$), r, view) = $\Gamma(acp_i.e$,r,view)

and $\Gamma(\Im(acp_i.e)$,r,view), $\rho(\wp(acp_i.e)$,r,view) and $\Gamma(acp_i.e$,r,view) are defined.

Hence $Prov_(RBAC)$ cannot be incomplete. □

Theorem 3. *If RBAC is in WF_{RBAC} is concise, then RBAC in Provenance $Prov_{RBAC}$ is concise as well.*

Proof. Since, RBAC in WF_{RBAC} is concise, we get if $\exists\ acp_i, acp_j \in \hat{R}$ such that:
$\mu(acp_i.\text{u}) = \mu(acp_j.\text{u}), \wedge\ acp_i.\text{e} = acp_j.\text{e}, \wedge\ acp_i.\text{a} = acp_j.\text{a}, \wedge\ \phi(acp_i.\text{r}, acp_i.\text{e}, acp_i.\text{a}) = \phi(acp_j.\text{r}, acp_j.\text{e}, acp_j.\text{a}) \wedge \text{i} = \text{j}$; where $\forall$ u $\in$ U, $\forall$ e $\in$ E, $\forall$ a $\in$ A.
Since we are assuming that RBAC in $Prov_{RBAC}$ is redundant, it implies:

- $\Gamma(\Psi(\Im(acp_i.e))$,r,view) = $\Gamma(\Psi(\Im(acp_j.e))$,r,view) and
- $\Delta(\Psi(\wp(acp_i.e))$,r,view) = $\Delta(\Psi(\wp(acp_j.e))$,r,view) and
- ζ(edge ($\Psi(\Im(acp_i.e))$, $\Psi(\wp(acp_i.e))$), r, view) = ζ(edge ($\Psi(\Im(acp_j.e))$, $\Psi(\wp(acp_j.e))$), r, view) and
- $i \neq j$

  However, from the definition we know:

- $\Gamma(\Psi(\Im(acp_i.e))$,r,view) = $\Gamma(\Im(acp_i.e)$,r,view)
- $\Delta(\Psi(\wp(acp_i.e)$,r,view) = $\rho(\wp(acp_i.e)$,r,view)

  And

- $\Gamma(\Psi(\Im(acp_j.e))$,r,view) = $\Gamma(\Im(acp_j.e)$,r,view)
- $\Delta(\Psi(\wp(acp_j.e)$,r,view) = $\rho(\wp(acp_j.e)$,r,view)

And since $\Gamma(\Im(acp_i.e)$,r,view) = $\Gamma(\Im(acp_j.e)$,r,view) and
$\rho(\wp(acp_i.e)$,r,view) = $\rho(\wp(acp_j.e)$,r,view),
it implies that i = j.

Hence, RBAC in $Prov_{RBAC}$ should be concise as well.

□

4.7 Security Policy Evolution

Security policy Evolution phase is needed for the modification of policies based on quality analysis phase after finding all the inconsistent, incomplete and redundant policies. The administrator retains the right to do the modification after finding those incorrect policies. For instance, inconsistent policies in Table. 7, for the role of teachers policy acp_{14} and acp_{15}, are inconsistent because the ports in data channel are specified with two different permission. For a single data channel, the output port O_6 specified negative and the input port i_9 specified positive. From our Port level and data channel level specification algorithms, both ports should have same permission. In this case, the output and the input port of single data channel have different permission, in evolution phase, the administrator will do the modification and specify explicitly both ports O_6 and i_9 are negative. For incomplete policies like the one in the example, when no access control policy for teachers role is specified for allowing or denying access to Family History table dataset of Task T_4, a policy evolution is needed. Without setting up the access control policy for input i_5 or task T_4 the policy defined accessing or denying the information of family history is incomplete. For that, the administrator modifies the policies by adding an access right for Task T_4 or input i_5. For redundant policies like acp_{23} and acp_{24}, the administrator can remove the policy acp_{24} because when the parent task's accessibility is positive, the child task's accessibility is positive as well unless otherwise stated.

4.8 ProvSec Prototype and Services

We developed a ProvSec prototype to validate the effectiveness of our protocol, with workflow view and mapped provenance views, in DATAVIEW. We specified our security in workflow and mapped that security to provenance, based on the role of the user. The security view of provenance does not have to be a connected graph. The reason is that security is imposed based on corresponding roles. Therefore the dependencies between

the subgraphs are hidden. In DATAVIEW system, *Provenance Manager* is the key to manage scientific workflow provenance. ProvSec prototype is managed by the provenance manager.

We illustrate our workflow provenance security mechanism with a real-life example by collecting data from SFARI project about Autism Spectrum Disorder(ASD). The autism workflow created in DATAVIEW [21, 20] system is used here. This running workflow system has ten tasks. The workflow in Fig. 21 explores all of the unique attributes of each child's Family history, education history, and medical history and identify predictive features pertaining to each individual child. This workflow implements data mining techniques for predicting the outcome based on the features availability. Both tasks T_1 and T_2 perform Projection p-workflow, which projects the predominant attributes out of a pool of attributes. Based on the SFARI id the task T_3 then performs another p-workflow task, the Natural Join operation. Task T_4 performs Projection on SFARI's follow-up Family history dataset. The retrieved result of both task T_3 and T_4, T_5, natural join operation is performed. Task T_6 checks to see if there are any missing or null values in a retrieved data set. Then Task T_7 performs another Projection operation. The output of this task works as an input of task T_8 which then converts CSV files to ARFF file format. The final result predictive dataset retrieved by executing data mining task T_{10}. For data mining and predicting, a test dataset is required, and that test dataset is provided to task T_9 for converting to ARFF format. After cumulating train set, test test and sample number of tree parameter we get the final prediction result. After executing this workflow in Fig. 22 we illustrated most detailed workflow run provenance information. In Fig. 22, circles represent data products, and rectangles represent workflow task run. The edge between data products and tasks are relations. For example, an edge from data product to task is called “wasGeneratedBy" relation, and an edge from task to data product is call “used" relation.

We use ProvSec prototype for autism workflow with the defined and modified policies. Based on each role we can see a security view of provenance by imposing defined policies.

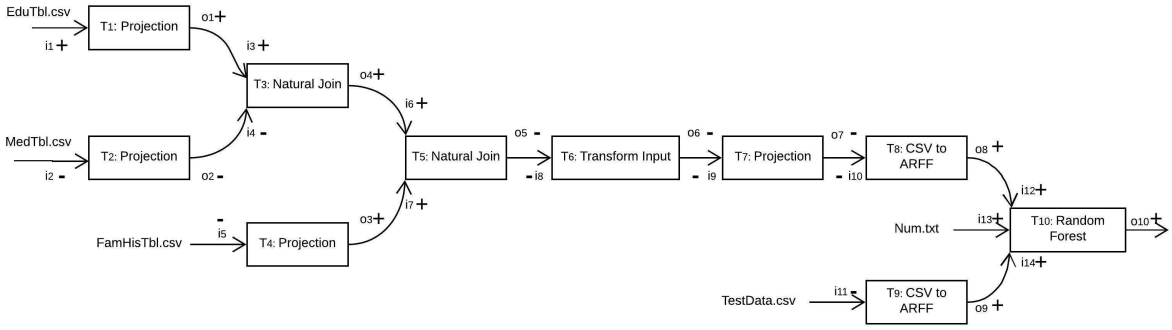


Figure 29: Workflow Permission for Teachers in Autism Provenance System.

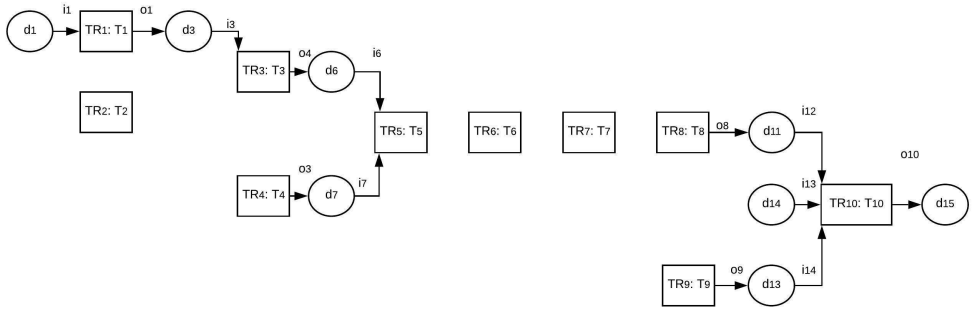


Figure 30: Security View of Teachers in Autism Provenance System.

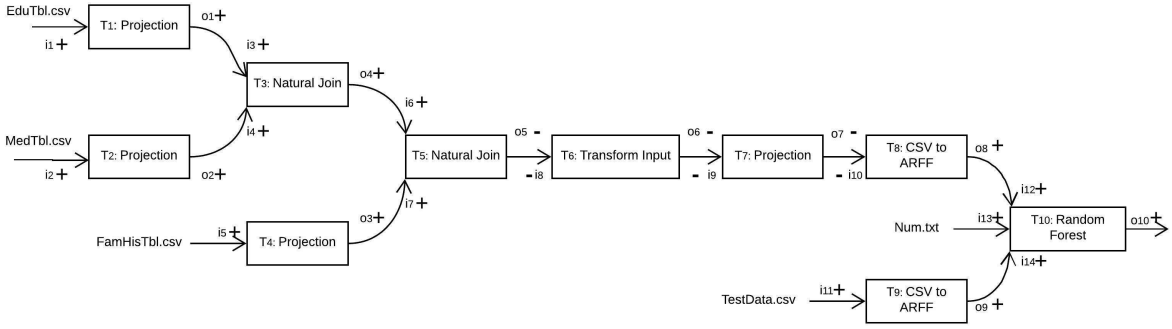


Figure 31: Workflow Permission for Therapists in Autism Provenance System.

Because of the sensitive nature of an autism workflow, we propose the restriction on data product and their provenance information for different roles. In ProvSec we defined

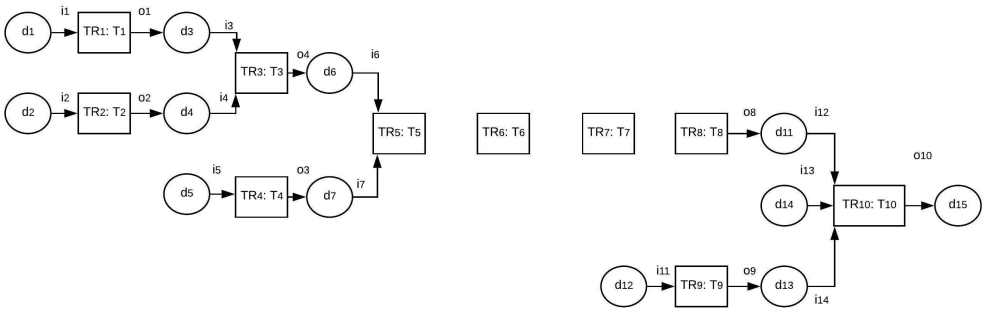


Figure 32: Security View of Therapists in Autism Provenance System.

3 types of the role for autism workflow.

- Parent's access permission specification and representing provenance security view
- Teacher's access permission specification and representing provenance security view
- Therapist's access permission specification and representing provenance security view

The parents have access permission to all the tasks, ports and data channels. For the parent role, in the provenance security view, parents can see all the sensitive data products and their corresponding relations. In addition to input and output data product, they can have access to all of the intermediary data products and can provide test set of data for projecting output.

For the teacher role, teacher or educators can have access to everything except Medical input data product i_2, the projected output O_2 of the data product, Family history input data i_5. When any data channel in a workflow is specified as negative then the data product generated for the provenance is not allowed to be seen by users. Any negative annotation on ports implies merely that the generated data product should not be visible to users of that particular role. Fig. 29 shows the workflow permission for teachers and Fig. 30 shows the security provenance view for teachers.

For the Therapist role, all therapist or clinician can have access to initial raw data to know about ASD children and prototyping appropriate program. This role doesn't require

to access intermediate data products or relations. However, they have permission to view predicted output for provided input parameters.

Fig. 31 shows the workflow permission for therapist and Fig. 32 shows the security provenance view for them, after implementing all the modified policies.

4.8.1 Performance Study

A collection of experiments were conducted on a machine with Intel core $i7-3612QM$ CPU $@2.10GHz$ x 8 processor and 7.7 GB memory.

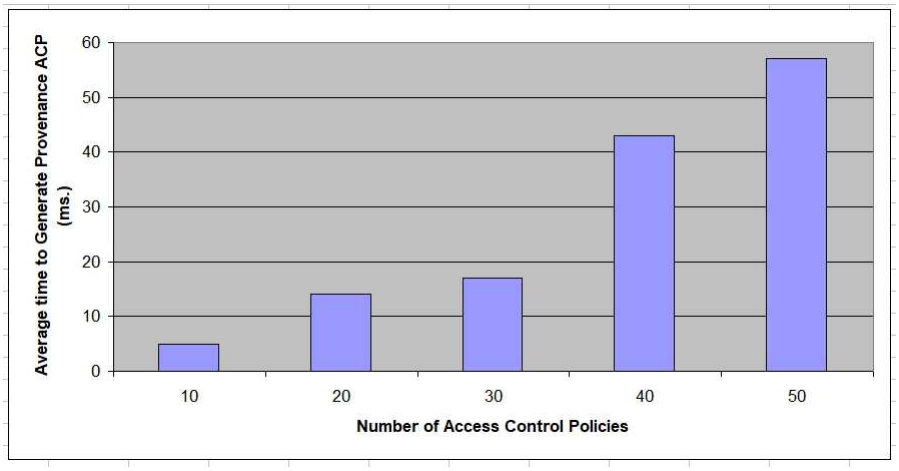


Figure 33: The Average Time to Generate Provenance Access Control Policies.

In Figure 33, we plot the time taken to inherit workflow specific access control protocol by the provenance system. We can observe that the inheritance process is not time intensive and can be computed very fast. We also observe a linear relationship between the number of access control protocols in the scientific workflow system and the time it takes to execute translation process. For example for a scientific workflow with 10, 20, 30, 40, 50, etc. access control protocols specified, it takes 5, 19, 17, 43, 57 milliseconds respectively.

4.9 Conclusion and Future Work

In this work, we studied access control policies for data products and their derivation history for protecting sensitive data and processes. First, we formalized a security scientific

workflow specification for task, port and data channel with proposed algorithms of access control policies. Second, we analyze those policies in terms of the policy quality requirements. Third, we formalized a security view of provenance based upon a mapping between workflow and provenance. Forth, we provide proof of holding policy quality requirements for provenance. Lastly, we evaluated an example in Autism community in order to show the validity of our quality of access control policies for provenance.

In the future, we will consider conducting security case studies with more complex data patterns and integrate our access control policies to deal with a different granularities of data. We will study cases of relative to their usability of the system.

CHAPTER 5 PREDICTING ONSET OF AUTISM USING SCIENTIFIC WORKFLOWS

Internet scale data and collaborative research initiatives have opened up new possibilities in health and medical domain. However, this domain intrinsically deals with sensitive data that needs to be carefully managed. Scientific workflows have been a big boon to tackling complex research questions while fostering collaboration across multiple geographic locations. In this chapter, we present one such framework that requires processing of sensitive data, namely therapeutic data in autism spectrum disorder, while providing repeatability guarantees. This framework, on one hand delineates how scientific workflow systems like DATAVIEW can be employed to answer complex research questions with repeatability guarantees, on the other hand, sets up the motivation to augmenting workflows and provenance systems with privacy and security protocols.

Early intervention in autism, although deemed as essential, has high variance in the outcomes attained. The variability in outcome is partly due to a complex interaction between a multitude of factors and variables involved and lack of principled study to untangle their influence in the outcome. Therefore, preparing a set of interventions for an individual child to cater for their need has been uniquely challenging. From the perspective of parents, unknown factors emanate from their unfamiliarity with what interventions are out there and why; analogously, for caregivers understanding unique attributes of the individual child develops with time. There is a scarcity of research that explores the interactions between attributes specific to a child, family characteristics, and therapeutic, medical and educational services.

In this chapter, we outline a scientific workflow framework that can be employed to bridge the gap. We show that using DATAVIEW, data mining techniques can be used to predict manifestation of autism. We used data collected by SFARI [5] dataset.

We frame the problem of understanding phenotypes in autism as a scientific workflow problem. With that end in view, we propose a workflow that analyzes SFARI dataset for

understanding phenotypes. We then outline how time-agnostic and windowed-temporal prediction models can be integrated to sift through features that can explain the data better.

5.1 Introduction

Recently, we have seen a sudden spike in Autism Spectrum Disorder (ASD) among the US population. Heterogeneity in data collection and interpretation in treating autism spectrum disorder is one of the most fundamental challenges in treating autism and planning and pragmatic intervention plan. Since hundreds of genes [82] cause manifestation of autism, several neuro-cognitive mechanisms, variance in phenotypical features also lie in a broad spectrum. Different areas in this phenotypic spectrum warrant uniquely catered intervention plan.

Analyzing collective data suggests that early intervention in treating autism spectrum disorder is highly effective in improving social, adaptive and communication skills. However, drilling down into the data suggests that individual responses to early intervention is highly variable with some children responding with substantial improvement, while others with marginal or no improvement at all. Hence, a blanket statement about the efficacy of the early intervention on an aggregated level, while true, does little to warrant modification to improve its effectiveness on a more individual basis.

The Autism spectrum of disorders, from the standpoint of parents and caregivers, can sometimes be construed as an enigma. There are potentially numerous reasons why ASD is deemed somewhat perplexing: lack of data, no clear treatment plan other than Adaptive Behavior Analysis, lack of fine-grained studies are a few of the salient reasons. A fundamental question that parents keep asking experts is how they can add support systems, e.g., speech therapy, ABA, and occupational therapy hours and create an appropriate goal/plan that can help their children make progress. Since the goals that are designed for a child, not only depend on the behavior the child exhibits, but also varies because of the quality of the therapy provided, resources available, it has become imperative to aid families come up

with appropriate set of goals generated from a data-driven standpoint and correcting for all individual biases. For example, knowing that providing more mainstreamed education might help a child make progress, irrespective of whether mainstreaming opportunities are available at the institution, might help parents make an informed decision about their child's placement.

We aim to demonstrate the example workflow using information pertaining to phenotype data and medical, developmental and educational history data. We also claim that these data are sensitive in nature and researchers and caregivers should have selective access to this data. We claim that research that entails processing sensitive data would flourish more and see better collaboration across industry and geographical location if scientific workflow platforms inherently provide secure access.

In this study we aim to answer the following questions:

- Provide a scientific workflow framework to facilitate the analysis of autism data and showcase the workflow as one that processes sensitive data and hence requires data security protcol.
- Compose a workflow that, based on individual and anonymized data unveils predictive traits of autism.
- Propose a scientific workflow to automate the modeling process and rely on DATAVIEW to guarantee computational reproducibility and data fidelity.

5.2 Background

Autism, a recent epidemic of a medical condition requires attention from medical, research and behavioral community for analyzing cause and trend and also predicting future outcomes. It is a neuro-developmental disorder that impairs natural development, causes challenges in emotional interaction, social communication, sensory processing, etc. It has a wide range of symptoms that is why this is referred to as Autism Spectrum Disorder (ASD). In the early 1940's, the condition was named as "Autism" and "Asperger Syndrome"

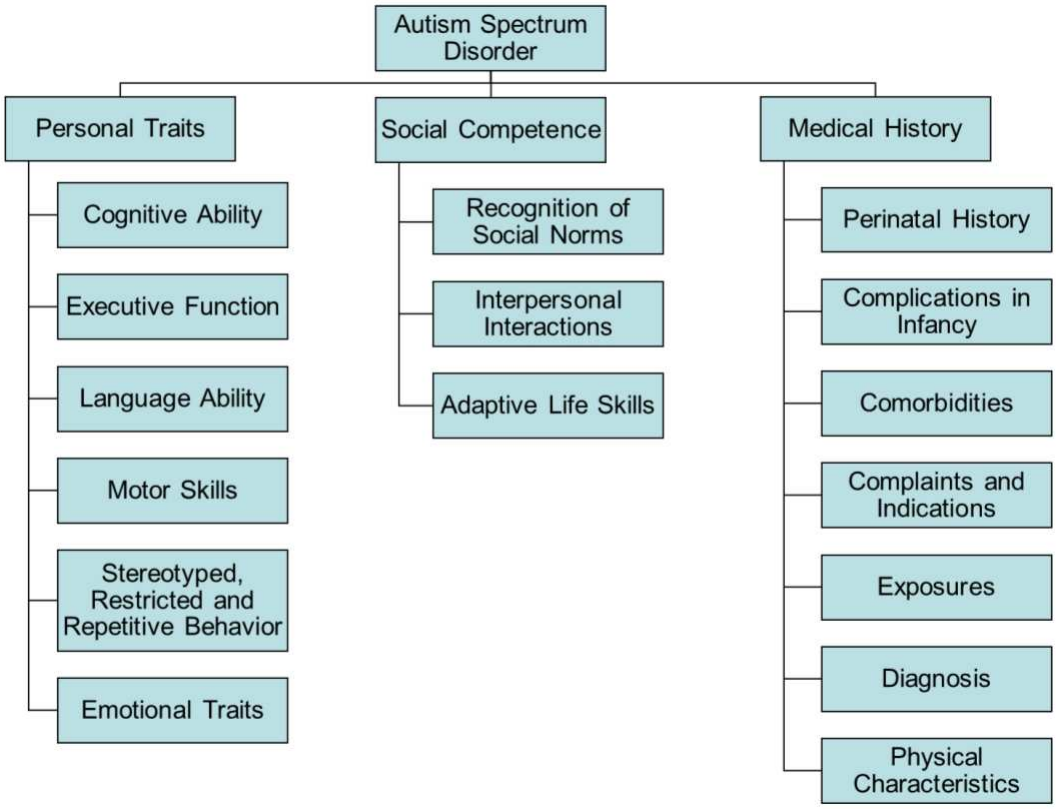


Figure 34: Overall View of Autism Spectrum Disorder.

by Leo Kanner and Hans Asperger, respectively [62]. Based on the data on National Institute of Mental Health (NIMH), in the 1970's the ASD rate was 1 in 10,000. In late 1995 the rate increased up to 1 in 1,000. In 1999 the rate became 1 in 500. In 2001, the rate was 1 in 250. In 2005, the rate was 1 in 166. By 2007 the rate was revised to 1 in 150. In 2009, the rate rose to 1 in 90 [62]. ASD affects approximately 0.5-0.6% of the population [19].

This rapid growth in the ASD rate warrants a reason for thinking about environmental factors and data-driven analysis of causes and symptoms. Due to a proliferation of this condition, now we have a large pool of data to start analyzing using data mining techniques. Moreover, storage and mining of big data leveraged to handle difficult problems like this can help understand ASD better.

In the process of promoting research and collecting data, Dr. Bernard Rimland [62]

founded the Autism Research Institute (ARI) in San Diego, CA in 1967. In their mission, they try to identify the cause of autism and evaluate treatment efficacy. They have collected survey data from over 40,000 parents of children with ASD throughout the world [62].

To better understand this epidemic medical condition there are different organization collecting confidential data from family's with ASD children. Each year more families are coming forward to store their data either for the sake of predicting the chances of siblings having the same condition or with the hope of knowing the cause, as it is still unknown. The National Database of Autism Research[1] has archived the Phenotype data collected from families and professionals. Based on the available concepts from NDAR, we have generated a graphical representation for the depiction of the wide range of features associated with Autism Spectrum Disorder (ASD) in Appendix B. Fig. 34 shows very brief overall graphical representation of the concepts.

5.3 Problem statement

In this research problem, we propose a scientific workflow to analyze phenotype data and find attributes and relations in these features. We investigate the predictive traits for autism and showcase the workflow as one that processes sensitive information and hence requires data access protocol to foster privacy-aware research.

For ASD children, one of the useful intervention methods is ABA (Applied Behavioral Analysis). Evidence shows that ABA works better than all other behavioral therapy. ABA data are collected in the form of A-B-C - Antecedent, Behavior, and Consequences. As most of ASD kids have limited language, A-B-C data gives a good understanding of their behavior, both problematic and good.

- A. Antecedent Data: It gives a good insight of antecedent of any problem behavior that triggered the behavior.
- B. Behavior Data: Behavior that is presented by the child.

[1] https://ndar.nih.gov/

- C. Consequences: A protocol of good consequence is by which the behavior can be shaped.

There are several ways to analyze child's data. One way could be, in our workflow management system, we can examine childś behavioral data to investigate what kind of educational setting he/she is more compliant to, which mostly predicts the effective environment for the child for learning purpose.

5.4 Proposed Work

5.4.1 Predictors of Improvement in Treatment Response

Traits in individual children and fine-grained intervention plans can be viewed as one of many temporal instances of a bipartite graph, where some of the interactions between these two groups result in positive outcomes while others result in regression or no progress. The temporal aspect of the assignment means these interactions and their characteristics alter in terms of efficacy and effectiveness with time. Having a data-driven model that answers these questions while planning early and subsequent intervention would improve the expected outcome. Many times interventions chosen for a specific child are decided by the availability of services, anecdotal evidence instead of a data-driven decision, resulting in sub-optimal outcome from the response. Having children enroll in less than ideal interventions cause the financial burden on the families with little or no noticeable gain. In this work, we aim to establish a theoretical framework to mine relevant information from data to guide intervention plan catered with individuality in mind. Historically, data collected during interventions can be broken down into several groups:

- Aberrant Behavior Checklist (ABC)
- Adult Behavior Checklist for Ages 19 to 59 (ABCL)
- Autism Diagnostic Interview-Revised (ADI-R)
- Autism Diagnostic Observation Schedule (ADOS)

- Broad Autism Phenotype Questionnaire (BAPQ)
- Child Behavior Checklist for ages 6 to 18 years (CBCL)
- Repetitive Behavior Scale-Revised (RBS-R)
- Social Communication Questionnaire (SCQ-L) - Parent report
- Social Communication Questionnaire (SCQ-C) - Teacher report
- Social Responsiveness Scale (SRS)
- Vineland Adaptive Behavior Scale-II (VABS-II)

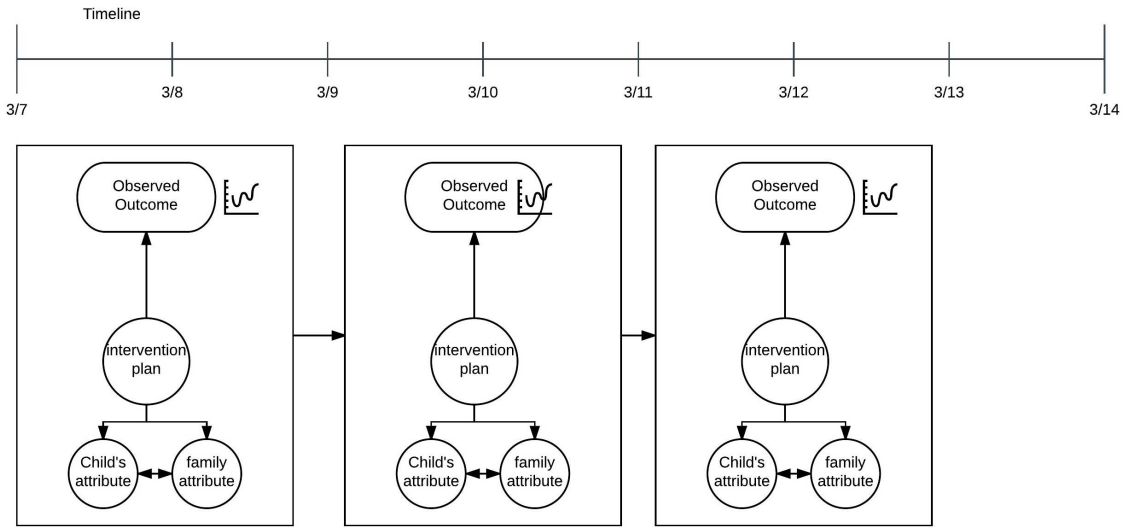


Figure 35: Treatment Improvement Predictor.

5.4.2 Methods

We can model the problem as a classification problem, where we learn the functional relationship of the behavioral, educational and medical features presented, to the onset of autism.

So as to understand the influence of time agnostic variables, we can frame the problem without considering temporal variables in order to identify expected improvement

in behavior based on features observed in a specific window of time independent of other timestamps. A by-product of the analysis is an identification of attributes (i.e., intervention plans and interaction between the trifecta as stated above) that lead to a better response to a specific intervention plan. Many classification algorithms can be employed, i.e., logistic regression. With proper regularization, logistic regression analysis can also provide significance of individual features and hence can guide what interventions to pay attention to. We can also employ other non-linear classification techniques like random forest or support vector machines, i.e., SVM. The Random forest operates on an ensemble of decision trees at training time and at inference time predicts the class that is the mode of the classes of all the constituent trees. The random forest can counter overfitting problem as observed in decision trees. On the other hand, SVM learns a hyperplane in a high dimensional space that separates the classes in question. Among a set of feasible hyperplanes, the one that is chosen has the largest distance from the support vectors. Since, non-linear classification are more expressive in nature, in this chapter, we chose to explore decision trees and SVMs that support those relation. However, the notion of dependence in the time window is not explicitly modeled in this case and we model the problem by collapsing multiple windows into a single input frame.

DATAVIEW supports non-linear classifiers e.g. Random forest and Support vector machine. We have used DATAVIEW to employ these non-linear classification techniques.

5.4.3 Modeling Scientific Workflow

We propose to structure the entire learning problem as a scientific workflow and introduce a strategy to guarantee reproducibility and data fidelity. Hence, the workflow can be deemed as portable and can be retrained and reused in isolation, for example, for different age groups.

After the data are collected, the entire process is modeled as a scientific workflow. Validation of the method is done based on standard techniques, i.e., label-wise precision-recall.

Table 8: Factors Involved for Predicting Treatment Outcomes.

ASD child predictable features c1	*ASD child given features c2*	*Family Features f*	*Intervention Plan i*
Denial Tolerance/time unit	Age	Father's education level	Hours of ABA/week
Elopement/time unit	Weight	Mother's education level	Speech/week
Listener Response/ time unit	IQ	Father's IQ level	Behavior Intervention Plan
Mand/time unit	Food Habit	Mother's IQ level	Nutrition supplements/ week
MLU/time unit	Geographic location	Family History of ASD from Father's side	
Throwing/time unit		Family History of ASD from Mother's side	
Hitting/time unit		Family stress level	
Dropping/time unit		Family's positive involvement	
Self InjuriousBehavior/time unit		Social support	
Property Destruction/time unit		Family's expectation about treatment	

Table 9: Feature Prediction Based on Timestamp.

Timestamp 1	*Timestamp 2*	*Timestamp 3*	*Timestamp 4*
c1 c2	c1 c2	c1 c2	? c2
F	F	F	F
I	I	I	I

Modeling it as a time-agnostic classification problem requires deciding on predefined window size, in the example, it is 4. The features of the window are fed into the algorithm and the outcome modeled as a function of the features.

5.5 Implementation and Experiments

5.5.1 DATAVIEW: A Big Data Workflow Management System

DATAVIEW is a big data workflow management system [77], that shows the feasibility of learning computational thinking in perspective of scientific workflow. We have used this workflow management system to implement the data mining techniques for predicting the outcome of the intervention technique based on the features available. The main reason of using DATAVIEW is to give flexibility to the researcher of Autism Community and also parents and caregiver not to deal with any underlying complexity of computation and can predict or correlate between features based on given train dataset.

This also gives us a platform for working on big data. As more researchers, with their heterogeneous data sources, collaborate in this domain, the data size is likely to

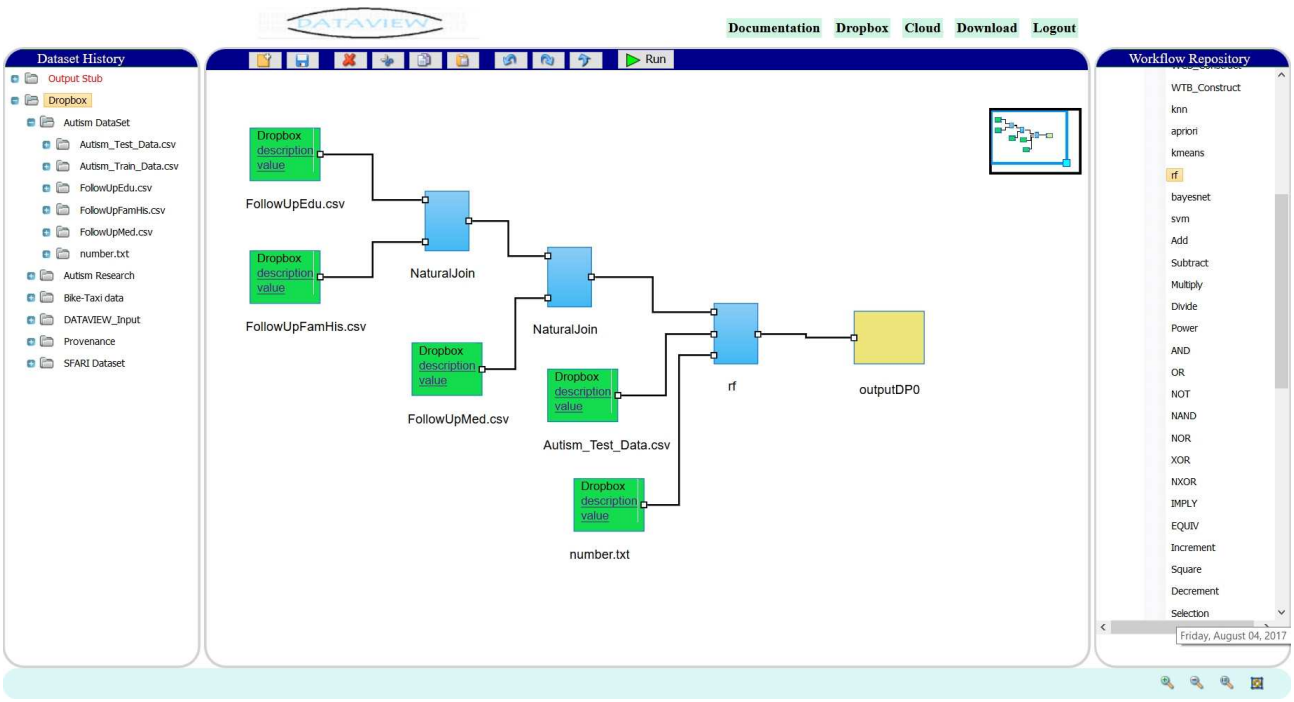


Figure 36: Running Workflow Predicting Classes on Data Mining Technique.

grow. Hence, for scalability and collaboration purposes, portable scientific workflow like DATAVIEW is an obvious choice.

The DATAVIEW has many primitive workflows, called built-in workflows. After the developers establish primitive construct, it is comparatively effortless and convenient to built executable workflows. For each built-in or primitive workflows used, needed only the required number of inputs and output.

Moreover, for providing more flexibility to the users, the current DATAVIEW has integrated Dropbox feature, so that users can offer any big data file and computation tools to analyze those data and drag-drop the built-in construct provided by DATAVIEW team and saved the result in an output which also can be accessed in a Dropbox folder. In this way, end users do not have to deal with the underlying complexity and can quickly obtain results.

In Fig. 39, shows one example of our executable workflow for predicting classes based on Random Forest data mining technique.

5.5.2 SFARI Dataset

We collaborated with SFARI (Simons Foundation Autism Research Initiative) [5] research and gathered a wide range of data which are collected over a period. Here SFARI gathered raw data of 440 families where a number of individual with diagnosis is 1050. The total size of the gathered dataset is 95MB. For our scientific workflow system, we explore “SSC_Version_15_Phenotype_Data_Set" and “SSCIAN_Follow_Up_Study_Dataset" dataset from SFARI.

In this follow-up study participants have completed a variety of measures which includes updated medical, educational histories and developmental updates based on standardized questionnaires.

Based upon the raw data collected, we have chosen 3 sets of Phenotype and their follow up data. Those are:

- FollowUpMedTbl: uses SFARI dataset “ssc_follow_up_medications",
- FollowUpEduTbl: uses SFARI dataset “ssc_follow_up_eduhx_child" and
- FollowUpFamHisTbl: uses SFARI dataset “ssc_follow_up_family_history".

In Table. 10, we can see the attribute set selected for each table from three different contexts. These are all Proband data which means ASD diagnosed person.

In FollowUpMedTble, some of the salient attributes are *sscmedcodevalue*, *symptomstatus*, *type* were based on the medications given, we can see the symptom status of the proband, i.e., symptoms worsened, No change in symptoms, symptoms have improved for past, current or current other.

In FollowUpEduTbl, we can see the following: which school type they are in, i.e., Special Ed or General ed or combination of both; what is their grade level; what kind of services they are getting; how is the classroom setting; do they have any personal aide; information about the siblings; and are siblings also receiving intervention services or special ed services.

Table 10: Selected Features Based on Each Dataset.

FollowUpMedTbl	*FollowUpEduTbl*	*FollowUpFamHisTbl*
sfari_id	sfari_id	sfari_id
multiplex	multiplex	multiplex
lost_diagnosis	lost_diagnosis	lost_diagnosis
sex	sex	sex
role	role	role
Fcode	Fcode	Fcode
sscmedFcodedvalue	school_type	relationship
Ftype	grade_level	yearofbirth
symptomstatus	special_ed_services	gender
age_at_eval	special_ed_Fcode	ASD
measure_Fcode	classroom_setting	scd
measure_Ftype_revision	personal_aide	language
study	siblings_intervention_services	phonological
	siblings_special_ed_services	developmental
	age_at_eval	learning
	measure_Fcode	intellectual
	measure_Ftype_revision	epilepsy
	study	ADHD
		OCD
		anxiety
		depression
		bipolar
		scizophrenia
		other
		age_at_eval
		measure_Fcode
		measure_Ftype_revision
		study

In FollowUpFamHisTbl, we have all the information about their age, what kind of challenges they are facing. ASD is a combination of challenges. When we look into those attributes we can see the challenges can be any one of the following: *language*, *phonological*, *developmental*, *learning*, *intellectual*, *epilepsy*, *adhd*, *ocd*, *anxiety*, *depression*, *bipolar*, *schizophrenia*, or other.

After getting all the information from 3 different context and joining all the information, we have a complete set of finding for one individual. We analyze this complete set of data using 2 data mining algorithms and run that in DATAVIEW.

5.5.3 Running Random Forest Algorithm in DATAVIEW

We used Random Forest as our prediction method. Random Forest, an ensemble learning method for classification, regression, and other tasks by composes and combines a multitude of decision trees at training time and uses a majority vote to infer the output at prediction time. For Random Forest, we didn't restrict the maximum depth parameter and

set bag size percent to be 80%.

To run Random Forest algorithm in DATAVIEW, we use DATAVIEWś primitive workflows, “FilterNull", “SelectFields", “ImputeMissingValue" and “NaturalJoin". We filter out all the rows with null values using “FilterNull" p-workflow. Then use “SelectFields" to select only those fields where maximum string size is less than 50. Then we impute missing value with 0 in “ImputeMissingValue" p-workflow. We join Education History and Medical History first, then use another “NaturalJoin" primitive workflow to join individual’s Family history too.

Now we use our primitive Random Forest workflow named “rf" which run the data mining algorithm, Random Forest. It has 3 input port and 1 output port. Input ports are training set, test set and the number of trees we would like to generate. The final join output works as an input for the training set. For test set, we can label any attribute we want to predict. The output port will return the correct label of the test set.

In Fig. 37 shows the example of our executable workflow for predicting classes based on Random Forest data mining technique.

For this experiment we predict, the symptom status of proband based on the training data. We show our prediction accuracy based on PR curve in Fig. 38. Here the precision-recall curve is plotted for the class “Symptoms improved". We observe similar performance for the other classes as well.

For validation purposes, we split the data 70%-30% and used the smaller data pool as a validation set.

5.5.4 Running Support Vector Machine Algorithm in DATAVIEW

We run the same experiment using Support Vector Machine. For SVM, we use the polynomial kernel with an exponent of 1.

We have used same 3 datasets and our primitive workflows, “FilterNull", “SelectFields", “ImputeMissingValue" and “NaturalJoin". After doing preprocessing on datasets we join them, FollowUpEduTbl, FollowUpFamHisTbl, and FollowUpMedTble. We use the final

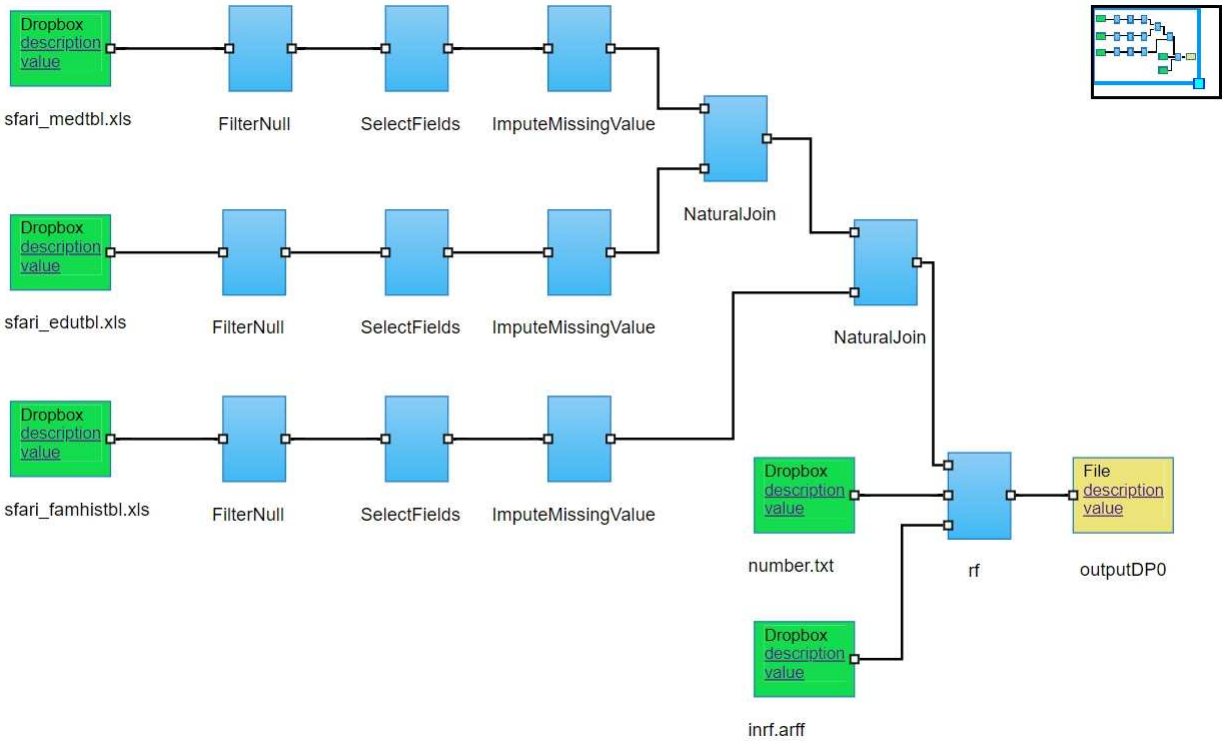


Figure 37: RF Workflow in DATAVIEW.

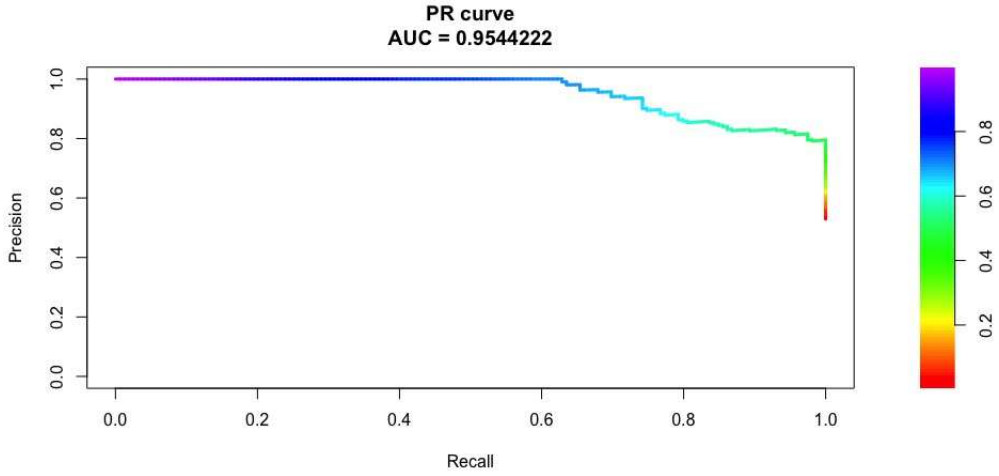


Figure 38: PR Curve Based on Random Forest.

table as our training dataset for SVM algorithm. Here also we predict proband's symptom status based on the training set.

Our PR curve for "Symptoms improved" class depicts the prediction accuracy of our

algorithm.

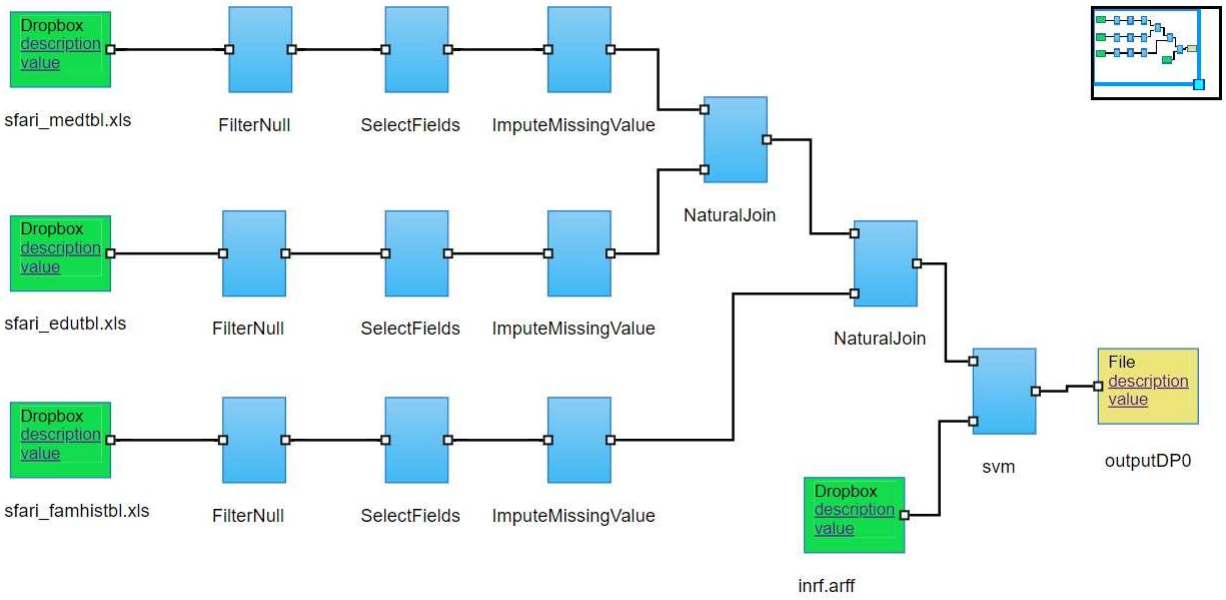


Figure 39: SVM Workflow in DATAVIEW.

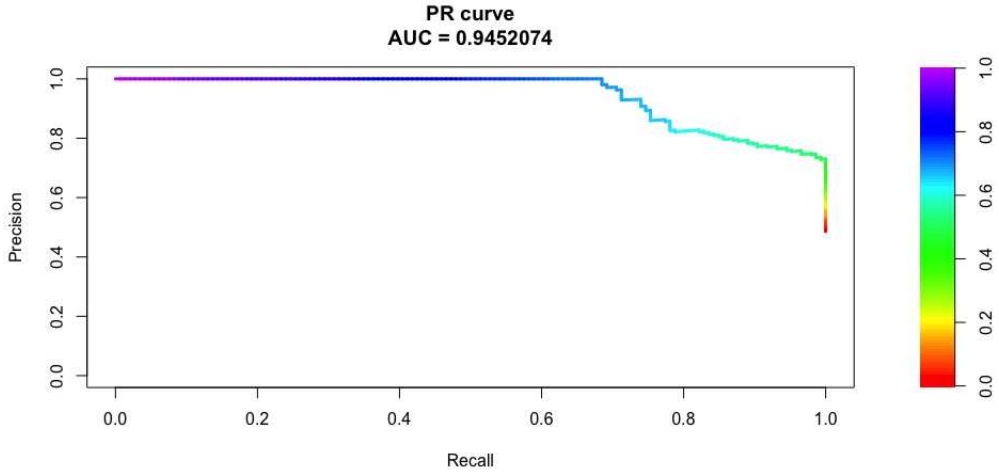


Figure 40: PR Curve Based for Support Vector Machine.

In a comparison of both Random Forest and Support Vector Machine, Random Forest inherently models the nonlinearity of the data and is relatively straightforward to counter for overfitting. In SVM, the appropriate kernel needs to be chosen to model non-linearity

in the data. Because of the nature of the data, we did not see a significant advantage of using one over the other, in this case.

5.6 Sensitivity of Data

This workflow demonstrates how scientific workflows can be used to process sensitive data to answer research questions. However, it is evident that these datasets require special handling and managing on access privileges. In autism research community physicians, scientist, teachers, psychologists can be involved. However, all the contributors should not be granted unrestricted access to all the datasets. The data needs to be partitioned, and access permission to specific dataset should be made to the responsible contributor. If the underlying scientific workflow and the provenance system does not provide support for such access control protocol, research collaboration will fail to flourish as data access compliance as mandated by HIPPA, for example, will have been violated.

5.7 Conclusion and Future Work

In this chapter, we demonstrated amalgamation between Autism Health informatics community and Workflow community. This research is motivated by augmenting health informatics into scientific workflows to guarantee data reproducibility.

This can be extended in the following major research directions.

- To identify variables that are involved and the most likely set of variables to have triggered the incident, for each individual episode of manifestation of problem behavior, based on ASD data.
- To recommend the next set of goals that are appropriate and beneficial, based on the trend of data.
- To develop tools of built-in construct for facilitating analysis of big data in DATAVIEW platform.

CHAPTER 6 CONCLUSIONS AND FUTURE WORK

Our research focuses on secure data-intensive computation and scalable querying and analysis. Our main contributions are:

- In this dissertation research, we propose $OPQL^{Pig}$, a parallel, robust, reliable and scalable translation of OPQL to Pig Latin programs for supporting the W3C PROV-DM standard provenance model. We propose algorithms to translate OPQL constructs to equivalent Pig Latin programs and develop and evaluate our $OPQL^{Pig}$ solution on provenance datasets from the UTPB benchmark. We then create some visual OPQL constructs in the DATAVIEW big data workflow system to facilitate the simple creation of complex OPQL queries in a visual workflow style.

- Next, for secure analysis and demonstration of health informatics data, we propose a secure scientific workflow specification with role-based access control policy and demonstrate how the workflow provenance system inherits those policies. We characterize the desirable properties of role-based access control protocol in scientific workflows and delineate how the properties are maintained in the workflow provenance systems as well. We proposed formal secure scientific workflow specification and algorithms and access control policies, analyze those policies in perspective of policy quality requirements to find out these evolving policies are up-to-date, complete, relevant and free of inconsistencies. We validated the quality of access control policies for provenance.

- Lastly, for automated computation-intensive and data-intensive analysis of big data, we have chosen Autism domain. Analyzing and mining big data, we use DATAVIEW to provide a platform for identifying the factors for success, antecedents for behavior and positive outcome based on collected data in the domain of autism. Analyzing collective data suggests that early intervention in treating autism spectrum disorder is highly effective in improving social, adaptive and communication skills.

As a future direction I plan to investigate the following research issues:

- To extend our $OPQL^{Pig}$ query language and conduct the experimental study based on current system and NoSQL Database like HBase or Cassandra; and applicable for other queries such as sub-graph isomorphism, pattern matching, and shortest path.
- To conduct security case studies with more complex data patterns and integrate our access control policies to deal with a different granularity of data and study cases of usability of the system.
- To recommend next set of goals that are appropriate and beneficial for ASD children, based on the trend of data and develop built-in analysis tools in DATAVIEW platform.

APPENDIX A

DATAVIEW is a big data workflow management system. DATAVIEW consists of 4 major parts consists of different modules in each part: Presentation Layer, Workflow Management Layer, Task Management Layer and Infrastructure Layer. The presentation layer is responsible for the design of scientific workflows, the presentation of data product and data provenance information, as well as the system status. The workflow design and configuration module provide intuitive GUI for users to design and configure workflows. The workflow engine is the central module that controls the execution of workflows. This module allows the users to drag and drop existing workflows and compose them with workflow constructs to formalize new composite workflows. The presentation layer automatically translates the graphical composition into a workflow definition file, and then send it to the workflow engine to register the new workflow. The workflow monitor module keeps track of the status of individual components, i.e.,"initialized", "executing", "finished", and "error". This layer allows users to create workflows and to browse, search, manage, execute, and reuse existing workflows. The data product manager module stores all data products used in workflows. This module allows users to create data products, and to browse, search, manage, and use existing data products. The provenance manager module is responsible for storing, browsing, and querying workflow provenance. The task manager module enables the execution of heterogeneous atomic tasks such as Web services and scripts.

The infrastructure layer plays a key role in provisioning, cataloging, configuring, and terminating virtual resources in clouds and data centers. Using DATAVIEW, a user not only easily share data and workflows with peer collaborators, but also design and run big data scientific workflows in the cloud, which includes commercial Amazon EC2 and academic

FutureSystems.

The workflow design and configuration module interact with Provenance Collector, which gathers provenance data. In particular, every time when a workflow design specification updated or saved, Provenance Collector translates the specification into a prospective provenance and stores that data into a provenance store which is in provenance manager. Provenance manager provides the functionality of provenance visualization via user-friendly GUIs, data insertion and provenance querying.

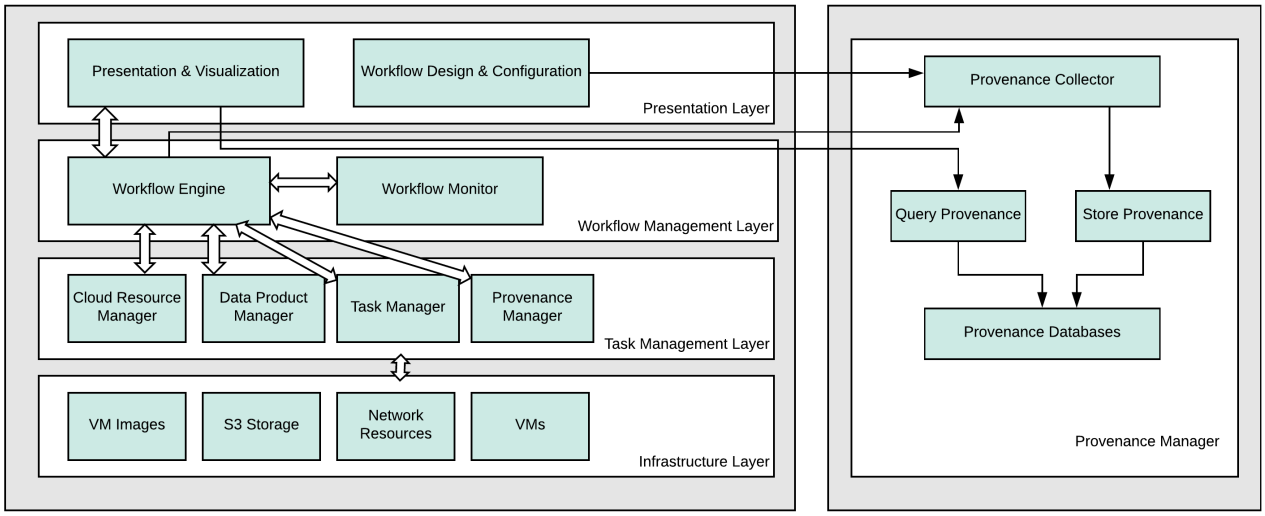


Figure 41: DATAVIEW: A big data scientific workflow management tool.

APPENDIX B

Figure 42: Autism Spectrum Disorder Personal Traits

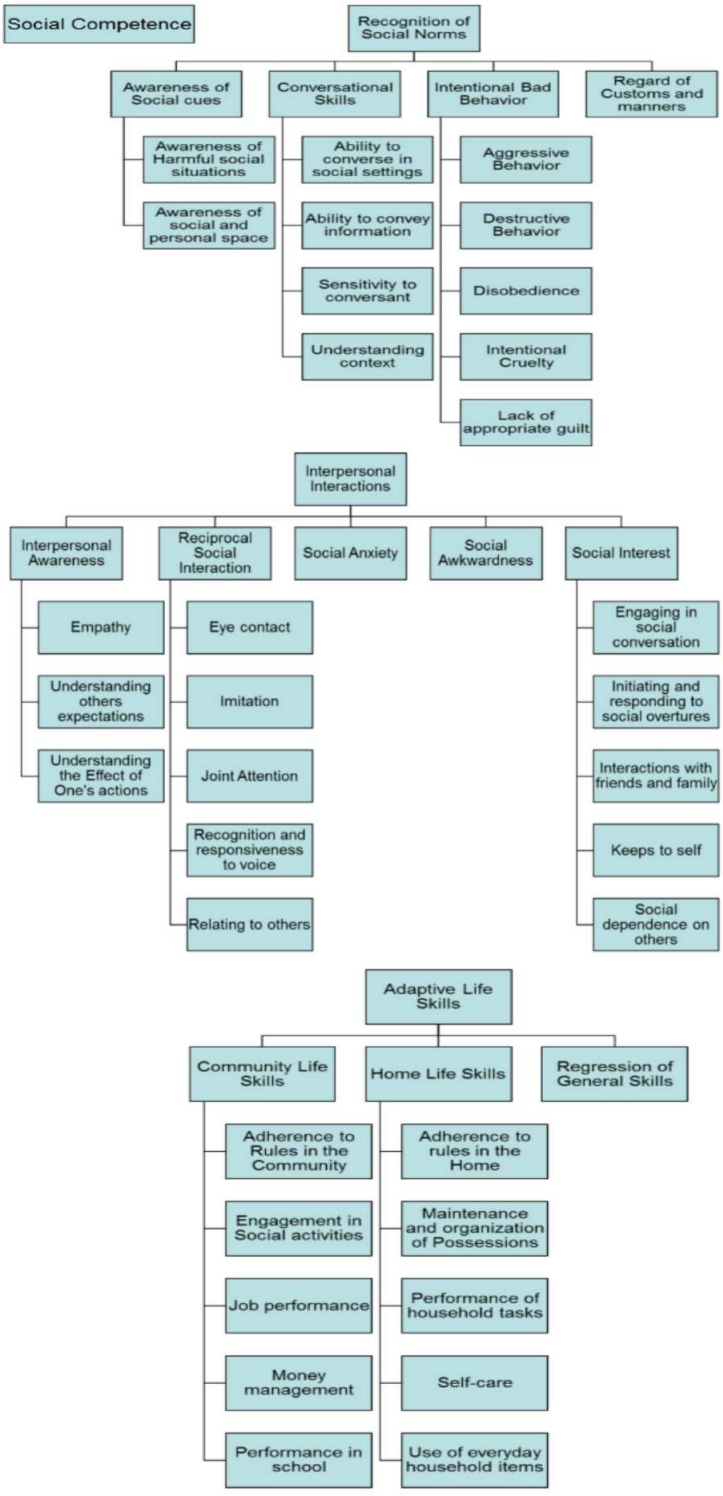


Figure 43: Autism Spectrum Disorder Social Competence

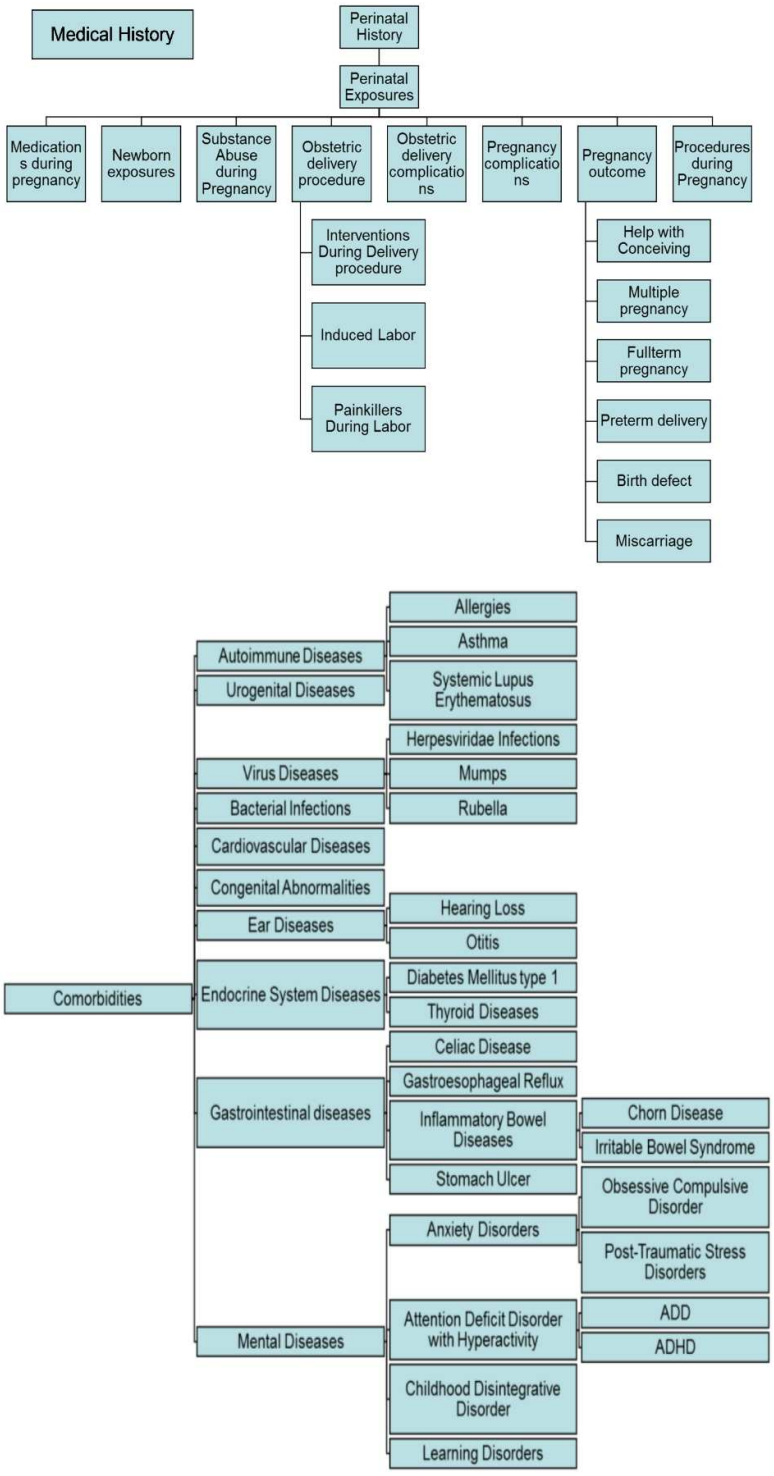


Figure 44: Autism Spectrum Disorder Medical History

REFERENCES

[1] *Apache Pig*, https://pig.apache.org/.

[2] *Provenance Challenge Series*, http://twiki.ipaw.info/bin/view/Challenge/FourthProvenanceChallenge.

[3] *Provenance Challenge Wiki, 2006*, http://twiki.ipaw.info/bin/view/Challenge/WebHome.

[4] *Provenance Model PROV-DM*, https://www.w3.org/TR/prov-dm/.

[5] *Simons Foundation Autism Research Initiative (SFARI)*, https://www.sfari.org/.

[6] *The Third Provenance Challenge*, http://twiki.ipaw.info/bin/view/Challenge/ThirdProvenanceChallenge.

[7] Rajeev Agrawal, Ashiq Imran, Cameron Seay, and Jessie J. Walker, *A Layer Based Architecture for Provenance in Big Data*, In Proc. of IEEE International Conference on Big Data, 2014, pp. 1–7.

[8] G. Ahn, R. Sandhu, M. Kang, and J. Park, *Injecting RBAC to Secure a Web-based Workflow System*, In Proc. of the fifth ACM Workshop on RBAC, 2000, pp. 1–10.

[9] Sherif Akoush, Ripduman Sohan, and Andy Hopper, *HadoopProv: Towards Provenance as a First Class Citizen in MapReduce*, Presented as part of the 5th USENIX Workshop on the Theory and Practice of Provenance, 2013.

[10] R. Aldeco-Pérez and L. Moreau, *Securing Provenance-Based Audits*, McGuinness IPAW, LNCS **6378** (2010), 148–164.

[11] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock, *Kepler: An Extensible System for Design and Execution of Scientific Workflows*, In Proc. of the 16th International Conference on Scientific and Statistical Database Management, 2004, pp. 423–424.

[12] Sami S. Alwakeel, Bassem Alhalabi, Hadi M. Aggoune, and Mohammad Alwakeel, *A Machine Learning Based WSN System for Autism Activity Recognition*, In Proc. of the IEEE International Conference on Machine Learning and Applications, ICMLA, 2015, pp. 771–776.

[13] Yael Amsterdamer, Susan B. Davidson, Daniel Deutch, Tova Milo, Julia Stoyanovich, and Val Tannen, *Putting Lipstick on Pig: Enabling Database-style Workflow Provenance*, The Proceedings of the Very Large Database Endowment (PVLDB) **5** (2011), no. 4, 346–357.

[14] Manish Kumar Anand, Shawn Bowers, and Bertram Ludäscher, *Techniques for Efficiently Querying Scientific Workflow Provenance Graphs*, International Conference on Extending Database Technology (EDBT) **10** (2010), 287–298.

[15] Manish Kumar Anand, Shawn Bowers, Timothy Mcphillips, and Bertram Ludäscher, *Efficient Provenance Storage over Nested Data Collections*, In Proc. of the 12th International Conference on Extending Database Technology: Advances in Database Technology, 2009, pp. 958–969.

[16] Muhammad Rizwan Asghar, Mihaela Ion, Giovanni Russello, and Bruno Crispo, *Securing Data Provenance in the Cloud*, In Proc. of the International Federation for Information Processing IFIP, 2012, pp. 145–160.

[17] V. Atluri and W. Huang, *An Authorization Model for Workflows*, In Proc. of the fourth European Symposium on Research in Computer Security, 1996, pp. 44–64.

[18] E. Bertino, E. Ferrari, and V. Atluri, *The Specification and Enforcement of Authorization Constraints in Workflow Management Systems*, ACM Transactions on Information and System Security (TISSEC) - Special issue on role-based access control **2** (1999), no. 1, 65–104.

[19] Adham Beykikhoshk, Ognjen Arandjelovic, Dinh Q. Phung, Svetha Venkatesh, and

Terry Caelli, *Data-mining Twitter and the Autism Spectrum Disorder: A Pilot Study*, In Proc. of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM, 2014, pp. 349–356.

[20] Fahima Bhuyan, Shiyong Lu, Ishtiaq Ahmed, and Jia Zhang, *Predicting Efficacy of Therapeutic Services for Autism Spectrum Disorder using Scientific Workflows*, In Proc. of the IEEE International Conference on Big Data, 2017.

[21] Fahima Bhuyan, Shiyong Lu, Dong Ruan, and Jia Zhang, *Scalable Provenance Storage and Querying Using Pig Latin for Big Data Workflows*, In Proc. of the IEEE Conference on Services Computing, 2017, pp. 459–466.

[22] Daniel Bone, Matthew S. Goodwin, Matthew P. Black, Chi-Chun Lee, Kartik Audhkhasi, and Shrikanth Narayanan, *Applying Machine Learning to Facilitate Autism Diagnostics: Pitfalls and Promises*, Journal of autism and developmental disorders **45** (2014), no. 5, 1121–1136.

[23] R. Botha and J. Eloff, *A Security Interpretation of the Workflow Reference Model*, In Proc. of the Information Security - from Small System to Management of Secure Infrastructure, 1998, pp. 43–51.

[24] R. A. Botha and J. H. P. Eloff, *Separation of Duties for Access Control Enforcement in Workflow Environments*, End-to-end Security **40** (2001), no. 3.

[25] Shawn Bowers, Timothy Mcphillips, Sean Riddle, Manish An, and Bertram Ludäscher, *Kepler/pPOD: Scientific Workflow and Provenance Support for Assembling the Tree of Life*, In Proc. of International Provenance and Annotation Workshop (IPAW), 2008.

[26] Shawn Bowers, Timothy M. McPhillips, and Bertram Ludäscher, *Provenance in Collection-oriented Scientific Workflows*, Concurrency and Computation: Practice and Experience (CONCURRENCY) **20** (2008), no. 5, 519–529.

[27] U. Braun and A. Shinna, *A Security Model for Provenance*, Technical Report TR-04-06 (2006).

[28] Uri Braun, Avraham Shinnar, and Margo Seltzer, *Securing Provenance*, In Proc. of the 3rd USENIX Workshop on Hot Topics in Security, HotSec, 2008.

[29] Peter Buneman and Wang Chiew Tan, *Provenance in Databases*, In Proc. of the ACM SIGMOD International Conference on Management of Data, 2007, pp. 1171–1173.

[30] Steven P. Callahan, Juliana Freire, Emanuele Santos, Carlos E. Scheidegger, Cláudio T. Silva, and Huy T. Vo, *VisTrails: Visualization Meets Data Management*, In Proc. of the 2006 ACM SIGMOD International Conference on Management of Data, 2006, pp. 745–747.

[31] Bin Cao, Beth Plale, Girish Subramanian, Ed Robertson, , and Yogesh L. Simmhan, *Provenance Information Model of Karma Version 3.*, IEEE Congress on SERVICES I, 2009, pp. 348–351.

[32] Irene Celino, Simone Contessa, Marta Corubolo, Daniele Dell'Aglio, Emanuele Della Valle, Stefano Fumeo, and Thorsten Krüger, *Linking Smart Cities Datasets with Human Computation - The Case of UrbanMatch*, In Proc. of the 11th International Semantic Web Conference ISWC, 2012, pp. 34–49.

[33] You-Wei Cheah, Shane Richard Canon, Beth Plale, and Lavanya Ramakrishnan, *Milieu: Lightweight and Configurable Big Data Provenance for Science*, In Proc. of IEEE International Congress on BigData Congress, 2013, pp. 46–53.

[34] A. Chebotko, E. De Hoyos, C. Gomez, A. Kashlev, X. Lian, and C. Reilly, *UTPB: A Benchmark for Scientific Workflow Provenance Storage and Querying Systems*, In Proc. of IEEE Eighth World Congress on Services, 2012, pp. 17–24.

[35] Artem Chebotko, Xubo Fei, Cui Lin, Shiyong Lu, and Farshad Fotouhi, *Storing and Querying Scientific Workflow Provenance Metadata using an RDBMS*, In Proc. of the

IEEE International Conference on e-Science and Grid Computing, 2007, pp. 611–618.

[36] Artem Chebotko, Shiyong Lu, Seunghan Chang, Farshad Fotouhi, and Ping Yang, *Secure Abstraction Views for Scientific Workflow Provenance Querying*, IEEE Transactions on Services Computing **3** (2010), no. 4, 322–337.

[37] Artem Chebotko, Shiyong Lu, Xubo Fei, and Farshad Fotouhi, *RDFProv: A Relational RDF Store for Querying and Managing Scientific Workflow Provenance*, Data & Knowledge Engineering **69** (2010), no. 8, 836–865.

[38] James Chency, *A Formal Framework for Provenance Security*, In Proc. of the 24th Computer Security Foundations Symposium, 2011, pp. 281–293.

[39] James Chency, Umut A. Acar, and Amal Ahmed, *Provenance Traces*, In Proc. of the CoRR Extended report, 2008.

[40] Kwok Cheung and Jane Hunter, *Provenance Explorer – Customized Provenance Views Using Semantic Inferencing*, In Proc. of the 5th International Semantic Web Conference, ISWC, 2006, pp. 215–227.

[41] Fernando Seabra Chirigati, Dennis Shasha, and Juliana Freire, *ReproZip: Using Provenance to Support Computational Reproducibility*, In Proc. of the Theory and Practice of Provenance (TaPP), 2009.

[42] H. Chivers and J. McDermid, *Refactoring Service-based Systems: How to Avoid Trusting a Workflow Service*, Concurrency and Computation : Practice and Experience **18** (2006), no. 10, 1255–1275.

[43] Stephen Chong, *Towards Semantics for Provenance Security*, In Proc. of the First Workshop on the Theory and Practice of Provenance, TaPP, 2009.

[44] Daniel Crawl, Alok Singh, and Ilkay Altintas, *Kepler WebView: A Lightweight, Portable Framework for Constructing Real-time Web Interfaces of Scientific Workflows*,

In Proc. of the International Conference on Computational Science, ICCS, 2016, pp. 673–679.

[45] Alfredo Cuzzocrea, *Big Data Provenance: State-Of-The-Art Analysis and Emerging Research Challenges*, In Proc. of the Workshops of the EDBT/ICDT Joint Conference, 2016.

[46] Susan B. Davidson and Juliana Freire, *Provenance and Scientific Workflows: Challenges and Opportunities*, In Proc. of the ACM SIGMOD international conference on Management of data, 2008, pp. 1345–1350.

[47] Susan B. Davidson, Sanjeev Khanna, Sudeepa Roy, Julia Stoyanovich, Val Tannen, and Yi Chen, *On Provenance and Privacy*, In Proc. of the 14th International Conference Database Theory ICDT, 2011, pp. 3–10.

[48] Ewa Deelman, Karan Vahi, Mats Rynge, Gideon Juve, Rajiv Mayani, and Rafael Ferreira da Silva, *Pegasus in the Cloud: Science Automation through Workflow Technologies*, IEEE Internet Computing **20** (2016), no. 1, 70–76.

[49] D. Domingos, A. Silva, and P. Veiga, *Workflow Access Control from a Business Perspective*, In Proc. of the International Conferance on Enterprise Information Systems, 2004, pp. 18–25.

[50] M. Duda, J. Kosmicki, and D. Wall, *Testing the Accuracy of an Observation-based Classifier for Rapid Detection of Autism Risk*, Translational psychiatry **4** (2014), no. 8, 424.

[51] Mahdi Ebrahimi, Aravind Mohan, Andrey Kashlev, and Shiyong Lu, *BDAP: A Big Data Placement Strategy for Cloud-Based Scientific Workflows*, In Proc. of the IEEE First International Conference on Big Data Computing Service and Applications, 2015, pp. 105–114.

[52] Mahdi Ebrahimi, Aravind Mohan, Andrey Kashlev, Shiyong Lu, and Robert G.

Reynolds, *Task And Data Allocation Strategies for Big Data Workflows*, International Journal of Big Data (IJBD) **2** (2015), no. 2, 28–42.

[53] Mahdi Ebrahimi, Aravind Mohan, Shiyong Lu, and Robert Reynolds, *TPS: A Task Placement Strategy for Big Data Workflows*, In Proc. of the IEEE International Conference on Big Data, 2015.

[54] T. Falck-Ytter and C. von Hofsten, *How Special is Social Looking in ASD: A Review*, Progress in brain research **189** (2011), 209–222.

[55] Xubo Fei, Shiyong Lu, and Jia Zhang, *A Granular Concurrency Control for Collaborative Scientific Workflow Composition*, In Proc. of the IEEE International Conference on Services Computing, 2011, pp. 410–417.

[56] David Feil-seifer and Maja J Mataric, *Toward Socially Assistive Robotics for Augmenting Interventions for Children with Autism Spectrum Disorders*, The Eleventh International Symposium on Experimental Robotics, ISER (2008).

[57] J. Freire, C. T. Silva, Emanuele Santos S. P. Callahan, Carlos E. Scheidegger, and Huy T. Vo, *Managing Rapidly-Evolving Scientific Workflows*, In Proc. of the International Provenance and Annotation Workshop IPAW, 2006, pp. 10–18.

[58] Alan F. Gates, Olga Natkovich, Shubham Chopra, Pradeep Kamath, Shravan M. Narayanamurthy, Christopher Olston, Benjamin Reed, Santhosh Srinivasan, and Utkarsh Srivastava, *Building a High-level Dataflow System on Top of Map-Reduce: The Pig Experience*, The Proceedings of the Very Large Database Endowment (PVLDB) **2** (2009), no. 2, 1414–1425.

[59] Boris Glavic, Kyumars Sheykh Esmaili, Peter M. Fischer, and Nesime Tatbul, *Efficient Stream Provenance via Operator Instrumentation*, ACM Transactions on Internet Technology **14** (2014), no. 1, 7:1–7:26.

[60] Boris Glavic, Kyumars Sheykh Esmaili, Peter Michael Fischer, and Nesime Tatbul,

Ariadne: Managing Fine-grained Provenance on Data Streams, In Proc. of the 7th ACM International Conference on Distributed Event-Based Systems, DEBS, 2013, pp. 39–50.

[61] Paul Groth and Luc Moreau, *An Overview of the PROV Family of Documents*, W3C Working Group Note (2013).

[62] Rocio Guillén, Curtis Jensen, and Stephen Edelson, *A Machine Learning Approach for Identifying Subtypes of Autism*, In Proc. of the ACM International Health Informatics Symposium, IHI, 2010, pp. 620–628.

[63] Rafat Hammad and Ching-Seh Wu, *Provenance as a Service: A Data-centric Approach for Real-Time Monitoring*, In Proc. of the IEEE International Congress on Big Data, 2014, pp. 258–265.

[64] Olaf Hartig and Jun Zhao, *Using Web Data Provenance for Quality Assessment*, In Proc. of the First International Conference on Semantic Web in Provenance Management-Volume 526, 2009, pp. 29–34.

[65] Ragib Hasan and Rasib Khan, *Unified Authentication Factors and Fuzzy Service Access using Interaction Provenance*, Computers & Security **67** (2017), 211–231.

[66] Ragib Hasan, Radu Sion, and Marianne Winslett, *Preventing History Forgery with Secure Provenance*, Journal of Intelligent Information Systems **5** (2009), no. 4, 12:1–12:43.

[67] G. Herrmann and G. Pernul, *Toward Security Semantics in Workflow Management*, In Proc. of the Thirty-First Annual Hawaii International Conference on System Sciences, 1998.

[68] Rinke Hoekstra and Paul Groth, *PROV-O-Viz - Understanding the Role of Activities in Provenance*, In Proc. of the 5th International Provenance and Annotation Workshop, IPAW, 2015, pp. 215–220.

[69] W. Huang and V. Atluri, *Analysing the Safety of Workflow Authorization Models*, In Proc. of the Twelfth International Working Conference on Database Security, 1999, pp. 43–57.

[70] D Hull, K Wolstencroft, R Stevens, C Goble, M R Pocock, P Li, and T. Oinn, *Taverna: A Tool for Building and Running Workflows of Services*, In Proc. of the Nucleic Acids Research, 2006, pp. 729–732.

[71] P. Hung and K. Karlapalem, *A Secure Workflow Model*, In Proc. of the Australasian Information Security Workshop Conference on ACSW Frontiers, 2003.

[72] Robert Ikeda, Hyunjung Park, and Jennifer Widom, *Provenance for Generalized Map and Reduce Workflows*, In Proc. of Fifth Biennial Conference on Innovative Data Systems Research (CIDR), 2011, pp. 273–283.

[73] Robert Ikeda and Jennifer Widom, *Panda: A System for Provenance and Data*, In Proc. of the 2nd Conference on Theory and Practice of Provenance TAPP, 2010, pp. 5–5.

[74] I.Y. Jung and H.Y. Yeom, *Provenance Security Guarantee from Origin up to Now in the e-Science Environment*, Journal of Systems Architecture (2010).

[75] S. Kandala and R. Sandhu, *Secure Role-based Workflow Models*, In Proc. of the Fifteenth Annual Working Conference on Database and Application Security, 2001, pp. 45–58.

[76] M. Kang, J. Park, and J. Froscher, *Access Control Mechanisms for Inter-organizational Workflow*, In Proc. of the sixth ACM Symposium on Access Control Models and Technologies, 2001, pp. 66–74.

[77] Andrey Kashlev and Shiyong Lu, *A System Architecture for Running Big Data Workflows in the Cloud*, In Proc. of the IEEE International Conference on Services Computing, 2014, pp. 51–58.

[78] Andrey Kashlev, Shiyong Lu, and Aravind Mohan, *Big Data Workflows: A Reference Architecture and The Dataview System*, Services Transactions on Big Data (STBD) **4** (2017), no. 1, 1–19.

[79] Troy Kohwalter, Thiago Oliveira, Juliana Freire, Esteban Clua, and Leonardo Murta, *Prov Viewer: A Graph-Based Visualization Tool for Interactive Exploration of Provenance Data*, In Proc. of the 6th International Provenance and Annotation Workshop, IPAW, 2016, pp. 71–82.

[80] J. Kosmicki, V. Sochat, M. Duda, and D. Wall, *Searching for a Minimal Set of Behaviors for Autism Detection through Feature Selection-based Machine Learning*, Translational psychiatry **5** (2015), no. 2, 514.

[81] H. Kozima, C. Nakagawa, and Y. Yasuda, *Interactive Robots for Communication-care: A Case-study in Autism Therapy*, In Proc. of the IEEE International Workshop on Robot and Human Interactive Communication, 2005, pp. 341–346.

[82] Tomasz Latkowski and Stanislaw Osowski, *Data Mining for Feature Selection in Gene Expression Autism Data*, Expert Systems with Applications **42** (2015), no. 2, 864–872.

[83] Hung-yi Lee, Ting-yao Hu, How Jing, Yun-Fan Chang, Yu Tsao, Yu-Cheng Kao, and Tsang-Long Pao, *Ensemble of Machine Learning and Acoustic Segment Model Techniques for Speech Emotion and Autism Spectrum Disorders Recognition*, In Proc. of the 14th Annual Conference of the International Speech Communication Association INTERSPEECH, 2013, pp. 215–219.

[84] Chunhyeok Lim, Shiyong Lu, Artem Chebotko, and Farshad Fotouhi, *OPQL: A First OPM-Level Query Language for Scientific Workflow Provenance*, In Proc. of IEEE International Conference on Services Computing, 2011, pp. 136–143.

[85] Chunhyeok Lim, Shiyong Lu, Artem Chebotko, Farshad Fotouhi, and Andrey Kash-

lev, *OPQL: Querying Scientific Workflow Provenance at the Graph Level*, Data & Knowledge Engineering **88** (2013), 37–59.

[86] Cui Lin, Shiyong Lu, Xubo Fei, Artem Chebotko, Darshan Pai, Zhaoqiang Lai, Farshad Fotouhi, and Jing Hua, *A Reference Architecture for Scientific Workflow Management Systems and the VIEW SOA Solution*, IEEE Transactions on Services Computing (TSC) **2** (2009), no. 1, 79–92.

[87] Erik Linstead, Ryan Burns, Duy Nguyen, and David Tyler, *AMP: A Platform for Managing and Mining Data in the Treatment of Autism Spectrum Disorder*, In Proc. of the 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC, 2016, pp. 2545–2549.

[88] Wenbo Liu, Li Yi, Zhiding Yu, Xiaobing Zou, Bhiksha Raj, and Ming Li, *Efficient Autism Spectrum Disorder Prediction with Eye Movement: A Machine Learning Framework*, In Proc. of the International Conference on Affective Computing and Intelligent Interaction, ACII, 2015, pp. 649–655.

[89] R. Lu, X. Lin, X. Liang, and X. Shen, *Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing*, In Proc. of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS, 2010, pp. 282–292.

[90] Shiyong Lu and Jia Zhang, *Collaborative Scientific Workflows*, In Proc. of the IEEE International Conference on Web Services, ICWS, 2009, pp. 527–534.

[91] Moreau Luc, Juliana Freire, Joe Futrelle, Robert E. McGrath, Jim Myers, and Patrick Paulson., *The Open Provenance Model: An overview*, In Proc. of the International Provenance and Annotation Workshop, 2008, pp. 323–326.

[92] Ruiqi Luo, Ping Yang, Shiyong Lu, , and Mikhail I. Gofman, *Analysis of Scientific Workflow Provenance Access Control Policies*, In Proc. of the IEEE Ninth International

Conference on Services Computing, 2012, pp. 266–273.

[93] R. Martinho, D. Domingos, and A. Rito-Silvas, *Supporting Authentication Requirements in Workflows*, In Proc. of the Eighth International Conference on Enterprise Information System: Databases and Information Systems Integration, 2006, pp. 181–188.

[94] Timothy McPhillips, Tianhong Song, Tyler Kolisnik, Steve Aulenbach, Khalid Belhajjame, R Kyle Bocinsky, Yang Cao, James Cheney, Fernando Chirigati, Saumen Dey, Juliana Freire, Christopher Jones, James Hanken, Keith W. Kintigh, Timothy A. Kohler, David Koop, James A. Macklin, Paolo Missier, Mark Schildhauer, Christopher Schwalm, Yaxing Wei, Mark Bieda, and Bertram Ludäscher, *YesWorkflow: A User-Oriented, Language-Independent Tool for Recovering Workflow Information from Scripts*, International Journal of Digital Curation **10** (2015), 298–313.

[95] P. Missier, P. Alper, O. Corcho, I. Dunlop, and C. Goble, *Requirements and Services for Metadata Management*, IEEE Internet Computing **11** (2007), no. 5, 17–25.

[96] Paolo Missier, Khalid Belhajjame, and James Cheney, *The W3C PROV Family of Specifications for Modelling Provenance Metadatag*, In Proc. of the Joint EDBT/ICDT Conferences, 2013, pp. 773–776.

[97] Paolo Missier, Khalid Belhajjame, Jun Zhao, Marco Roos, and Carole Goble, *Data Lineage Model for Taverna Workflows with Lightweight Annotation Requirements*, In Proc. of the Second International Provenance and Annotation Workshop (IPAW), 2008, pp. 17–30.

[98] Paolo Missier, Norman W. Paton, and Khalid Belhajjame, *Fine-grained and Efficient Lineage Querying of Collection-based Workflow Provenance*, In Proc. of the 13th International Conference on Extending Database Technology, 2010, pp. 299–310.

[99] Luc Moreau, *The Foundations for Provenance on the Web*, Foundations and Trends in

Web Science **2** (2010), no. 2-3, 99–241.

[100] Luc Moreau, Ben Clifford, Juliana Freire, Joe Futrelle, Yolanda Gil, Paul Groth, Natalia Kwasnikowska, Simon Miles, Paolo Missier, Jim Myers, et al., *The Open Provenance Model Core Specification (v1. 1)*, Future generation computer systems **27** (2011), no. 6, 743–756.

[101] Leonardo Murta, Vanessa Braganholo, Fernando Chirigati, David Koop, and Juliana Freire, *noWorkflow: Capturing and Analyzing Provenance of Scripts*, In Proc. of the 5th International Provenance and Annotation Workshop IPAW, 2015, pp. 71–83.

[102] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins, *Pig Latin: A Not-so-foreign Language for Data Processing*, In Proc. of the ACM SIGMOD International Conference on Management of Data, 2008, pp. 1099–1110.

[103] Hyunjung Park, Robert Ikeda, and Jennifer Widom, *RAMP: A System for Capturing and Tracing Provenance in MapReduce Workflows*, In Proc. of the Very Large Database Endowment (PVLDB) **4** (2011), no. 12, 1351–1354.

[104] B. Plale, C. Goble S. Miles, P. Missier, R. Barga, Y. Simmhan, J. Futrelle, R. E. McGrath, J. Myers, P. Paulson, S. Bowers, B. Ludäscher, N. Kwasnikowska, J. V. Bussche, T. Ellkvist, J. Freire‘, and P. Groth, *The Open Provenance Model (v1.01)*, Technical report, University of Southampton (2008).

[105] Sarvapali D. Ramchurn, Trung Dong Huynh, Matteo Venanzi, and Bing Shi, *Collabmap: Crowdsourcing Maps for Emergency Planning*, In Proc. of the Web Science, 2013, pp. 326–335.

[106] Ben Robins, Kerstin Dautenhahn, and Paul Dickerson, *From Isolation to Communication: A Case Study Evaluation of Robot Assisted Play for Children with Autism with a Minimally Expressive Humanoid Robot*, In Proc. of the Second International

Conferences on Advances in Computer-Human Interactions, 2009, pp. 205–211.

[107] Dong Ruan, Shiyong Lu, Aravind Mohan, Xubo Fei, and Jia Zhang, *A User-Defined Exception Handling Framework in the VIEW Scientific Workflow Management System*, In Proc. of the IEEE International Conference on Services Computing, 2012, pp. 274–281.

[108] R. Sandhu, *Transaction Control Expressions for Separation of Duties*, In Proc. of the Fourth Computer Security Applications Conference, 1988, pp. 282–286.

[109] Alexander Schätzle, Martin Przyjaciel-Zablocki, Thomas Hornung, and Georg Lausen, *PigSPARQL: A SPARQL Query Processing Baseline for Big Data*, In Proc. of International Semantic Web Conference (Posters and Demos), 2013, pp. 241–244.

[110] Elaine Short, David Feil-Seifer, and Maja J. Mataric, *A Comparison of Machine Learning Techniques for Modeling Human-robot Interaction with Children with Autism*, In Proc. of the 6th International Conference on Human Robot Interaction, HRI, 2011, pp. 251–252.

[111] C. T. Silva, J. Freire, and S. P. Callahan, *Provenance for Visualizations: Reproducibility and Beyond*, Computing in Science Engineering **9** (2007), no. 5, 82–89.

[112] Y. L. Simmhan, B. Plale, and D. Gannon, *A Survey of Data Provenance in e-Science*, Special Interest Group on Management of Data SIGMOD Rec. **34** (2005), no. 3, 31–36.

[113] Yogesh L. Simmhan, Beth Plale, and Dennis Gannon, *Query Capabilities of the Karma Provenance Framework*, Concurrency and Computation: Practice and Experience **20** (2008), no. 5, 441–451.

[114] Jacek Sroka, Jan Hidders, Paolo Missier, and Carole A. Goble, *A Formal Semantics for the Taverna 2 Workflow Model*, Journal of Computer and System Sciences **76** (2010), no. 6, 490–508.

[115] A. Sheth S.Wu, J. Miller, and Z. Luo, *Authorization and Access Control of Application Data in Workflow Systems*, Journal of Intelligent Information Systems **18** (2002), no. 1, 71–94.

[116] V. Tan, P. Groth, S. Miles, S. Jiang, S. Munroe, S. Tsasakou, and L. Moreau, *Security Issues in a SOA-based Provenance System*, In Proc. of the third International Provenance and Annotation Workshop (IPAW), 2006.

[117] V. Atluri and W. Huang, *Security for Workflow Systems*, Handbook of Database Security Applications and Trends, 2007, pp. 213–230.

[118] Irene Celino, Simone Contessa, Marta Corubolo, Daniele Dell'Aglio, Emanuele Della Valle, Stefano Fumeo, and Thorsten Krüger, *UrbanMatch - Linking and Improving Smart Cities Data*, In Proc. of the Workshop on Linked Data on the Web, 2012.

[119] Alfredo Cuzzocrea, *Big Data Provenance: State-Of-The-Art Analysis and Emerging Research Challenges*, In Proc. of the Workshops of the EDBT/ICDT Joint Conference, 2016.

[120] Rafat Hammad and Ching-Seh Wu, *Provenance as a Service: A Data-centric Approach for Real-Time Monitoring*, In Proc. of the IEEE International Congress on Big Data, 2014, pp. 258–265.

[121] W. Huang and V. Atluri, *SecureFlow: A Secure Web Enabled Workflow Management System*, In Proc. of the fourth ACM Workshop on Role-based Access Control, 1999, pp. 83–94.

[122] Andrey Kashlev and Shiyong Lu, *A System Architecture for Running Big Data Workflows in the Cloud*, In Proc. of 2014 IEEE International Conference on Services Computing, 2014, pp. 51–58.

[123] Manish Kumar Anand, Shawn Bowers, Timothy Mcphillips, and Bertram Ludäscher, *Exploring Scientific Workflow Provenance Using Hybrid Queries over Nested Data and*

Lineage Graphs, In Proc. of the 21st International Conference on Scientific and Statistical Database Management, 2009, pp. 237–254.

[124] Chunhyeok Lim, Shiyong Lu, Artem Chebotko, and Farshad Fotouhi, *Storing, Reasoning, and Querying OPM-compliant Scientific Workflow Provenance using Relational Databases*, Future Generation Computer Systems **27** (2011), no. 6, 781–789.

[125] Shiyong Lu and Jia Zhang, *Collaborative Scientific Workflows Supporting Collaborative Science*, International Journal of Business Process Integration and Management IJBPIM **5** (2011), no. 2, 185–199.

[126] Y. L. Simmhan, B. Plale, and D. Gannon, *A Framework for Collecting Provenance in Data-Centric Scientific Workflows*, In Proc. of the IEEE International Conference on Web Services ICWS, 2006, pp. 427–436.

[127] Boyang Wang, Baochun Li, and Hui Li, *Oruta: Privacy-Preserving Public Auditingfor Shared Data in the Cloud*, IEEE Transactions on Cloud Computing **2** (2014), no. 1, 43–56.

[128] J. Warner and V. Atluri, *Inter-instance Authorization Constraints for Secure Workflow Management*, In Proc. of the eleventh ACM Symposium on Access Control Models and Technologies, 2006, pp. 190–199.

[129] Li Yi, Yuebo Fan, Paul C. Quinn, Cong Feng, Dan Huang, Jiao Li, Guoquan Mao, and Kang Lee, *Abnormality in Face Scanning by Children with Autism Spectrum Disorder is Limited to the Eye Region: Evidence from Multi-method Analyses of Eye Tracking Data*, Journal of Vision **13** (2013), no. 10.

[130] Li Yi, Cong Feng, Paul C. Quinn, Haiyan Ding, Jiao Li, Yubing Liu, and Kang Lee, *Do Individuals with and without Autism Spectrum Disorder Scan Faces Differently? A New Multi-Method Look at an Existing Controversy*, Autism Research **7** (2013), no. 1, 72–83.

[131] Jia Zhang, Qihao Bao, Xiaoyi Duan, Shiyong Lu, Lijun Xue, Runyu Shi, and Pingbo Tang, *Collaborative Workflow Composition as a Service - An Infrastructure Supporting Collaborative Data Analytics Workflow Design and Management*, In Proc. of the International Conference on Collaboration and Internet Computing (CIC 2016), 2016.

[132] Jia Zhang, Daniel Kuc, and Shiyong Lu, *Confucius: A Tool Supporting Collaborative Scientific Workflow Composition*, IEEE Transactions on Services Computing **7** (2014), no. 1, 2–17.

[133] Jia Zhang, Petr Votava, Tsengdar J. Lee, Owen Chu, Clyde Li, David Liu, Kate Liu, Norman Xin, and Ramakrishna R. Nemani, *Bridging VisTrails Scientific Workflow Management System to High Performance Computing*, In Proc. of the IEEE Ninth World Congress on Services, SERVICES, 2013, pp. 29–36.

[134] Y. Zhao, M. Hategan, B. Clifford, I. Foster, G. von Laszewski, V. Nefedova, I. Raicu, T. Stef-Praun, and M. Wilde, *Swift: Fast, Reliable, Loosely Coupled Parallel Computation*, In Proc. of the IEEE Congress on Services, 2007, pp. 199–206.

[135] Bao Zhuowei, Sarah Cohen-Boulakia, Susan B. Davidson, and Pierrick Girard, *PDiffView: Viewing the Difference in Provenance of Workflow Results*, VLDB- Very Large DataBase systems **2** (2009), 1638–1641.

ABSTRACT

SCALABLE AND SECURE PROVENANCE QUERYING FOR SCIENTIFIC WORKFLOWS AND ITS APPLICATION IN AUTISM STUDY

by

FAHIMA AMIN BHUYAN

August 2018

Advisor: Dr. Shiyong Lu

Major: Computer Science

Degree: Doctor of Philosophy

In the era of big data, scientific workflows have become essential to automate scientific experiments and guarantee repeatability. As both data and workflow increase in their scale, requirements for having a data lineage management system commensurate with the complexity of the workflow also become necessary, calling for new scalable storage, query, and analytics infrastructure. This system that manages and preserves the derivation history and morphosis of data, known as provenance system, is essential for maintaining quality and trustworthiness of data products and ensuring reproducibility of scientific discoveries. With a flurry of research and increased adoption of scientific workflows in processing sensitive data, i.e., health and medication domain, securing information flow and instrumenting access privileges in the system have become a fundamental precursor to deploying large-scale scientific workflows. That has become more important now since today team of scientists around the world can collaborate on experiments using globally distributed sensitive data sources. Hence, it has become imperative to augment scientific workflow systems as well as the underlying provenance management systems with data security protocols. Provenance systems, void of data security protocol, are susceptible to vulnerability. In this dissertation research, we delineate how scientific workflows can improve therapeutic practices in autism spectrum disorders. Showcasing scientific exploration in the domain of autism spectrum disorder demonstrates the need for privacy-aware

scientific workflows and provenance systems. We also aim to underscore the significance of data driven analysis of therapeutic studies for children on the spectrum. The data-intensive computation inherent in these workflows and sensitive nature of the data, necessitate support for scalable, parallel and robust provenance queries and secured view of data. With that in perspective, we propose $OPQL^{Pig}$, a parallel, robust, reliable and scalable provenance query language and introduce the concept of access privilege inheritance in the provenance systems. We characterize desirable properties of role-based access control protocol in scientific workflows and demonstrate how the qualities are integrated into the workflow provenance systems as well. Finally, we describe how these concepts fit within the DATAVIEW workflow management system.

AUTOBIOGRAPHICAL STATEMENT

EDUCATION

- Doctor of Philosophy (Computer Science), April 2018
  Wayne State University, Detroit, Michigan, United States
- Master of Science (Computer Science), August 2010
  Wayne State University, Detroit, Michigan, United States

PUBLICATIONS

- **Fahima Amin Bhuyan**, Shiyong Lu, Robert Reynolds, Ishtiaq Ahmed, and Jia Zhang. Quality Analysis for Scientific Workflow Provenance Access Control Policies. *In Proc. of the IEEE Conference on Services Computing SCC*, 2018.
- Ishtiaq Ahmed, Shiyong Lu, Changxin Bai, and **Fahima Amin Bhuyan**. Diagnosis Recommendation Using Machine Learning Scientific Workflows. *In Proc. of the IEEE Big Data Congress*, 2018.
- **Fahima Amin Bhuyan**, Shiyong Lu, Ishtiaq Ahmed, and Jia Zhang. Predicting Efficacy of Therapeutic Services for Autism Spectrum Disorder using Scientific Workflows. *In Proc. of the IEEE International Conference on Big Data*, pages 3847-3856, 2017.
- **Fahima Amin Bhuyan**, Shiyong Lu, Dong Ruan, and Jia Zhang. Scalable Provenance Storage and Querying Using Pig Latin for Big Data Workflows. *In Proc. of the IEEE Conference on Services Computing SCC*, pages 459-466, 2017.
- **Fahima Amin Bhuyan** and Shiyong Lu., $OPQL^{Pig}$: A Seamless Synergy between Pig and Provenance Query for Big Data. Grace Hopper Women in Computing Conference, 2016.
- Zijing Yang, Shiyong Lu, Ping Yang, **Fahima Amin Bhuyan**. Model Checking Approach to Secure Host Access Enforcement of Mobile Tasks in Scientific Workflows. *Special Issue on Scientific Workflows, Provenance and Their Applications of International Journal of Computers and Their Applications (IJCA)*, 18(3), 2011.
- **Fahima Amin Bhuyan**, Shiyong Lu, and Jia Zhang. Storing and Querying Big Data Workflow Provenance as Graphs in Apache Pig. *IEEE Transactions on Services Computing (TSC)*, 2018. (submitted).
- **Fahima Amin Bhuyan**, Shiyong Lu, Robert Reynolds, Ishtiaq Ahmed, and Jia Zhang. A Security Framework for Provenance Access Control Policies. *IEEE Transactions on Services Computing (TSC)*, 2018. (submitted).