Temporal Representation for Scientific Data Provenance

Preprint Version
Forthcoming in IEEE eScience 2012

Peng Chen
School of Informatics and Computing
Indiana University
chenpeng@cs.indiana.edu

Beth Plale
School of Informatics and Computing
Indiana University
plale@cs.indiana.edu

Mehmet S. Aktas
Information Technologies Institute
TUBITAK
mehmet.aktas@tubitak.gov.tr

***Abstract*—Provenance of digital scientific data is an important piece of the metadata of a data object. It can however grow voluminous quickly because the granularity level of capture can be high. It can also be quite feature rich. We propose a representation of the provenance data based on logical time that reduces the feature space. Creating time and frequency domain representations of the provenance, we apply clustering, classification and association rule mining to the abstract representations to determine the usefulness of the temporal representation. We evaluate the temporal representation using an existing 10 GB database of provenance captured from a range of scientific workflows.**

I. Introduction

The provenance of a scientific data product or collection is a record of the factors contributing to the product as it exists today. That is, it identifies the what, where, when, how, and who of an object. What type of actions were applied that yielded a particular result? How and where were those actions applied? And by whom? To the extent that a data product results from raw data that itself has simple lineage, the lineage record of a data product is the latest set of activities (or "workflow") applied.

Provenance of digital scientific data is an important piece of the metadata of a data object. It can be used to determine attribution, to identify relationships between objects [3], to trace back differences in similar results, and in a more far reaching goal, to aid a researcher who is trying to determine whether or not an acquired data set can be reused in his or her work, by providing lineage information to support their trust in the quality of the data set. However, provenance can be highly voluminous, as capture can be carried out at a high level of granularity. This can occur for instance with a workflow system that encourages fine grained nodes (i. e., at the level of a mathematical operation) instead of coarse-grained (i.e., at the level of a large parallel computing job.) The sheer volume of data has been dealt with in different ways, by developing views on the provenance [26], or by caching select content [10]. Visualization techniques are effective in making sense of large data [22]. One could throttle provenance capture to control the volume [5] of provenance generated at the source.

We take a different approach to dealing with the large volumes of provenance, and that is to assume volumes will be large, then selectively reduce the feature space while simultaneously preserving interesting features so that data mining on the reduced space yields provenance-useful information. The mining tasks include generating patterns that describe and distinguish the general properties of the datasets in provenance repositories (by training classifier and mining association rule set), detecting faulty provenance data (by checking cluster centroids in the case where correct and faulty provenance are naturally separated into different clusters) and finding more descriptive knowledge of provenance clusters (by mining association rules that reflects workflow variants).

Provenance can be represented as a directed graph of entities related by causal dependencies. An accepted model for representing provenance entities and relationships is the Open Provenance Model (OPM) [19]. OPM defines a historical record of dependencies between entities, hence OPM compliant graphs have implicit temporal ordering which we exploit in our proposed representation.

In this paper, we propose the temporal provenance representation as an efficient and useful statistical feature representation of provenance. In order to establish the usefulness of the temporal representation, we apply classification and clustering algorithms to the representation. We also derive data mining association rules for each cluster of the provenance graphs. The goal of this study is to evaluate our proposed temporal provenance representation for temporal data mining kinds of tasks. The contributions of the paper are Logical Clock-P, an algorithm that partitions provenance graphs, and an assessment of the representation using temporal data mining techniques on the generated temporal representation. Evaluation is carried out against a large 10 GB database [6] of provenance traces generated from six real-life workflows.

The remainder of the paper is organized as follows: Section II reviews related work. Section III introduces the causal graph partitioning approach, while Section IV describes the temporal representation. The experimental evaluation against a large database of provenance is presented in Section V. Section VI concludes the paper and discusses future work.

II. Related Work

The value that provenance brings to e-Science applications is first suggested in a 2005 survey of provenance [23]. Davidson and Freire [9] provide an additional survey view of provenance. Davidson et al. [8] first introduce the problem of mining and extracting knowledge from provenance.

Margo and Smogor [18] use data mining and machine learning techniques to extract semantic information from I/O provenance gathered through the file system interface of a computer. The mining step reduces the large, singular provenance graph to a small number of per-file features. Our research is complementary in that we examine a collection of provenance graphs and treat a whole provenance graph as an entity. Like Margo's work, we also reduce the size and dimensionality of provenance by partitioning the graph and applying statistical post-processing. Phala [17] uses provenance information as a new experience-based knowledge source, and utilizes the information to suggest possible completion scenarios to workflow graphs. It does not, however, provide descriptive knowledge for a large provenance dataset.

Clustering techniques have been applied to workflow graphs. A workflow script or graph is either an abstract or implementation plan of execution. A provenance graph, on the other hand, is a record of execution. A provenance record may or may not have the benefit of an accompanying workflow script, so a workflow graph is in some cases a coarse approximation of provenance graph. Santos et al. [21] apply clustering techniques to organize large collections of workflow graphs. They propose two different representations: the labeled workflow graph and the multidimensional vector. However, their representation using labeled workflow graphs becomes too large if the workflow is big, and the structural information is completely lost if using a multidimensional vector.

Jung and Bae [13] propose the cluster process model represented as a weighted complete dependency graph. Similarities among graph vectors are measured based on relative frequency of each activity and transition. It has the same issue as Santos et al. Our work addresses the problem of mining and discovering knowledge from provenance graphs, while overcoming the scalability issue by reducing the large provenance graph to a small temporal representation sequence, and retaining structural information together with attribute information.

How to treat data with temporal dependencies is another problem in the discovery process of hidden information. The ultimate goal of temporal data mining is to discover hidden relations between sequences and subsequences of events [2]. Provenance information stored in a form amenable to representation as a graph has an implicit temporal ordering, which can be exploited for data clustering and relationship discovery. To our best knowledge, there is no previous study on discovering the hidden relations in provenance.

III. Provenance graph partitioning

A directed, annotated provenance graph is not ideally suited to data mining for two reasons: 1) provenance graphs can have thousands of nodes and attributes. Clustering in such a high dimensional space presents tremendous difficulty [4], and 2) it is difficult to place both structural and non-structural information in a single uniform attribute space. Hence, we propose a graph partitioning algorithm that uses Lamport's logical clocks [16] as the basis for an abstract representation of provenance. Our approach has the assumption that the provenance graphs to which the representation is applied are compliant with the Open Provenance Model [19].

A. Partial ordering

Lamport determines a total ordering of events in a distributed computer system based on logical time order. Since the OPM reference specification [19] defines edges as causal relationships, we define the "happened before" relation in a provenance graph based on its causal relationships.

Definition The "happened before" relation, denoted by "$\rightarrow$", on the set of nodes in a provenance graph is the smallest relation satisfying the following two conditions:

1) If a and b are nodes that have an edge between them, and a is the cause, then $a \rightarrow b$
2) If $a \rightarrow b$ and $b \rightarrow c$ then $a \rightarrow c$

We assume that $a \not\rightarrow a$ for any node, which implies that $\rightarrow$ is an irreflexible partial ordering on the set of all nodes in the provenance graph. We define node a and b to be concurrent nodes if $a \not\rightarrow b$ and $b \not\rightarrow a$. For example, in Figure1(c): Node "6" $\rightarrow$ Node "multiplier", and Node "multiplier' $\rightarrow$ Node "54" so that Node "6" $\rightarrow$ Node "54"; Node "6" and Node "9" are concurrent nodes.

While the current OPM reference document forbids cycles, a new definition [15] allows the presence of derived-from cycles (simple cycles composed of derived-from edges) after a merge operation. However, an OPM graph resulting from a typical experimental provenance collection procedure, which is the target of this study, does not contain such cycles. In addition, new definitions (e.g., [15]) avoid using the term *causal relationship*, but the constraints in their temporal theory are more similar to Lamport's ordering, and they are still using "cause" to represent an edge source and "effect" to represent an edge destination. Thus our definition above also works in this case.

B. Logical Clock-P

We propose the Logical Clock-P, a function C that takes a node as input and produces an integer as output. This function maps an integer to each node of a given provenance graph. The correct logical clocks must satisfy the condition that if a node a occurs before another node b, then a should happen at an earlier time than b. We state this condition more formally as follows.

Definition *Clock Condition:* The Clock condition satisfies the following condition: For any node a and b, if $a \rightarrow b$ then $C(a) < C(b)$.

C. Strict totally ordered partition

With Logical Clock-P defined, we define a strict totally ordered partition that divides a provenance graph into a list of non-empty subsets. A typical provenance graph has three kinds of nodes: artifacts, processes, and agents. A partitioning of a provenance graph is a set of non-overlapping and non-empty subsets of nodes based on the logical clocks. More precisely, a partition of provenance graph $G = (V, E)$, where V denotes the set of all nodes and E denotes the set of all edges, is defined as follows:

Definition For a provenance graph $G = (V, E)$, partition V into k subsets $V_1, V_2, \ldots, V_k$ such that:

1) $\mathrm{V_1}, \mathrm{V_2}, \ldots, \mathrm{V_k} \in \mathrm{V}$ and $\bigcup_{1}^{k} \mathrm{V_i} = \mathrm{U}$
2) $\forall\ i \neq j$ and $1 \leq i\ , j \leq k$, $\mathrm{V_i} \cap \mathrm{V}_j = \phi$
3) $\forall\ a\ , b \in \mathrm{V}_i$, we must have $C(a) = C(b)$, and the node type of a is the same as the node type of b

Furthermore, to place all the subsets $\{V_1, V_2, \ldots, V_k\}$ into an ordered list, we define a "appears before" relation to give the total order on the set of all these subsets. Naturally, node with smaller Logical Clock-P comes before node with larger Logical Clock-P. Furthermore, for nodes with the same Logical Clock-P, we put agent before process and process before artifact. This is because of the implicit order in node definition [19]: agent is defined as an entity enabling process execution, process is defined as action resulting an artifact, and artifact is defined as a state in a physical object. Though this implicit order can be different from the real time order, it is still meaningful for us when putting concurrent nodes into a sequential representation.

Definition The "appears before" relation "$\Rightarrow$" on the set $\{V_1, V_2, \ldots, V_k\}$ in a provenance graph needs to satisfy the following condition that:

1. $\forall a \in V_{\mathrm{i}}\ , \forall b \in V_{\mathrm{j}}$, if $C\,(a) < C\,(b)$, then $V_{\mathrm{i}} \Rightarrow V_{\mathrm{j}}$
2. $\forall a \in V_{\mathrm{i}}\ , \forall b \in V_{\mathrm{j}}\ ,$ if $C\,(a) = C\,(b)$, and node type of a is agent, and node type of b is process, then $V_{\mathrm{i}} \Rightarrow V_{\mathrm{j}}$
3. $\forall a \in V_{\mathrm{i}}\ , \forall b \in V_{\mathrm{j}}$, if $C\,(a) = C\,(b)$, and node type of a is agent, and node type of b is artifact, then $V_{\mathrm{i}} \Rightarrow V_{\mathrm{j}}$
4. $\forall a \in V_{\mathrm{i}}\ , \forall b \in V_{\mathrm{j}}\ ,$ if $C\,(a) = C\,(b)$, and node type of a is process, and node type of b is artifact, then $V_{\mathrm{i}} \Rightarrow V_{\mathrm{j}}$

A partition of a provenance graph with the "appears before" relation on the set $\{V_1, V_2, \ldots, V_k\}$ is asymmetric, transitive and also totally ordered, but not unique. We show an example partitioning generated by our Logical-P algorithm in Figure 1, where the subset with the smaller number (e.g., Subset 1) "appears before" the subset with larger number (e.g., Subset 2, Subset 3).

D. Provenance graph partitioning algorithm (Logical-P algorithm)

Given any provenance graph (we are using the XML representation [11]), we generate an unique strict totally ordered partition with the following algorithm:

1: $S \leftarrow$ Set of all nodes with no incoming edges

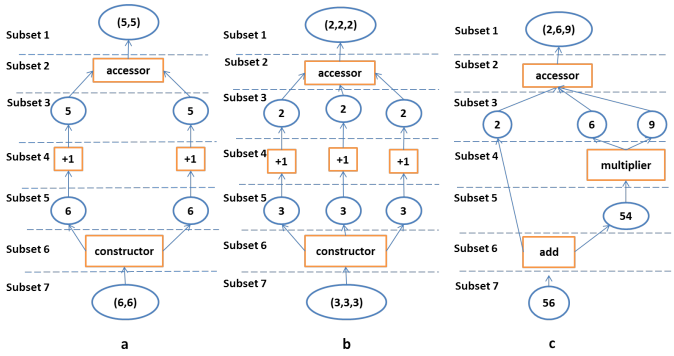

Fig. 1. Temporal partition: (a) is example provenance graph from [19]; (b) is from same experiment as (a) with different input data; (c) has similar graph structure to (a) and (b) but with different nodes.

2: **for all** nodes k in S **do**
3: assign 0 to C(k)
4: **end for**
5: **while** S is non-empty **do**
6: remove node n from S
7: **for all** node m with edge e from n to m **do**
8: remove edge e from graph
9: **if** C(n) + 1 > C(m) **then**
10: assign C(n) + 1 to C(m)
11: **end if**
12: **if** m has no other incoming edges **then**
13: insert m into S
14: **end if**
15: **end for**
16: **end while**
17: Group nodes with same Logical Clock-P value and node type into one subset
18: Sort subsets according to "appears before"

Steps 1–16 are derived from the topological sorting algorithms of Kahn [14], which has linear time in the number of nodes plus the number of edges $O(|V| + |E|)$. The time complexity of step 18 depends on the sorting algorithm that is used. For heapsort, the complexity is $O(klogk)$, where k is the number of subsets in the partition. Note that steps 9–11 give nodes that have multiple causes the maximum possible Logical Clock-P value.

IV. Temporal Provenance Representation

With a provenance graph partitioned into an ordered list of subsets (subgraphs), the next step is to organize the representations of each subset into a sequence to form the representation of the whole graph. However, a typical provenance graph is a fully-labeled graph with annotations (both nodes and edges have labels and annotations), so direct representations such as feature vector space will result in a high dimensional dataset which is not suitable for large scale mining tasks. We address this issue by using attribute transformation [4] such as roll-ups (sums or average over time intervals), and we define a new statistical feature space.

A. Statistical Feature Space

We first give the definition of a feature space of node subset, and then extend this definition to a statistical feature space by introducing a statistical feature function.

Definition For a *feature vector subset* $N = (V, F, D)$, $V = \{v_1, ..., v_n\}$ denotes the node subset, the function $F : \mathrm{V} \rightarrow \mathrm{D}_1 \times \mathrm{D}_2 \times ... \times \mathrm{D}_d$ is a *feature function* that assigns a feature vector to any node $v \in V$, and the set $\mathrm{D} = \{\mathrm{D}_1, \mathrm{D}_2, \mathrm{D}_3, \ldots, \mathrm{D}_d\}$ is called the *feature space* of N.

Definition For a *statistical feature vector subset* $N' = (V, F, G, D, S)$, a *statistical function* $G : \mathrm{D}_i \times \mathrm{D}_i \times ... \times \mathrm{D}_i \rightarrow \mathrm{S}_i$ applies statistical operators such as max, min, avg, std.dev, std.err, sum and variance to feature $D_i \in D$ of all nodes in V, and the set $\mathrm{S} = \{\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3, \ldots, \mathrm{S}_d\}$ is called the *statistical feature space* of N.

The features of a provenance graph node include its attribute feature such as its labels and annotations, and its structural feature such as the attributes of its incoming/outgoing edges.

For example, a simple node attribute feature can be the number of characters in node label, and a simple node structural feature can be the number of in-degree or out-degree. So the feature space for subset 2 in Figure 1(a) can be $D = \{$ number of characters in node label, number of in-degree, number of out-degree $\} = \{(1, 1), (1, 1), (1, 1)\}$, and its statistical feature space can be $S = \{$ average number of characters in node label, average number of in-degree, average number of out-degree $\} = \{1, 1, 1\}$.

B. Feature Selection from Statistical Feature Space

The selection of an optimal feature set depends upon both the mining targets and the nature of the provenance, which is beyond our current research. However, since one of our targets in unsupervised clustering is to group provenance instances based on their original experiment, we want to select a feature set that can discriminate between provenance instances of different experiments. In other words, the distance between two representations of provenance derived from the same experiment should be smaller than the distance between two representations of provenance derived from different experiments.

We assume provenance graphs that have similar structure and similar attribute information are from related experiments (Figure 1(a) and Figure 1(b)); while provenance graphs from different experiments are either different in attribute information (Figure 1(a) and Figure 1(c)), or different in structure information (Figure 1(a) and Figure 5(b)). While using either feature set, Figure 1(a) and Figure 1(b) should be clustered together.

Based on this assumption, we create a simple *attribute feature set* that includes "average number of characters in label" to discriminate between Figure 1(a) and Figure 1(c), and a simple *structural feature set* that includes "average number of in-degree/out-degree" to discriminate between Figure 1(a) and Figure 5(b).

TABLE I
EUCLIDEAN DISTANCE FOR ATTRIBUTE AND STRUCTURAL FEATURE SETS

Figure	Distance in time domain	Distance in frequency domain
	Attribute Feature Set	
1(a) - 1(b)	3.7417	0.2678
1(a) - 1(c)	12.0	0.6183
1(b) - 1(c)	12.7279	0.6764
	Structural Feature Set	
1(a) - 1(b)	2.2361	0.1755
1(a) - 5(b)	10.1281	0.7096
1(b) - 5(b)	10.1113	0.5835

Specifically, for the *attribute feature set* we capture: <*Type of nodes in subset, num nodes in subset, Avg num characters in node name*> which for Figure 1(c), gives:

$(< 2, 1, 7 >, < 1, 1, 8 >, < 2, 3, 1 >, < 1, 1, 10 >, < 2, 1, 2 >, < 1, 1, 3 >, < 2, 1, 2 >)$

For *structural feature* set we capture the following features: <*Type of nodes in subset, Number nodes in subset, Avg number of in-degree of nodes in subset, Avg number of out-degree of nodes in subset* > from each subset V_i. We map the type of nodes from their textual values "Agent", "Process", "Artifact" into numerical values 0, 1, 2. The resulting provenance partition of Figure 1(c) is represented as:

$(< 2, 1, 1, 0 >, < 1, 1, 3, 1 >, < 2, 3, 1, 1 >, < 1, 1, 1, 2 >, < 2, 1, 1, 1 >, < 1, 1, 1, 2 >, < 2, 1, 0, 1 >)$

Furthermore, we apply Discrete Fourier Transform (DFT) [25] to transform the above sequence from the time domain to a point in the frequency domain, by choosing the k first (we use k=3) frequencies, and representing each sequence as a point in the k-dimensional space. The same example of statistical feature in frequency domain yields the following where each value pair represents a frequency in the form of <real part, imaginary part>:

(<1.125, 0>, <-0.1706, 0.1635>, <-0.1547, 0.1077>)

Table I gives the Euclidean distance for the attribute feature set and the structural feature set. As discussed earlier, graph 1(a) from Figure 1 is very similar to 1(b). Their distance is close for both attribute and structural feature sets. 1(a) – 1(c) are different from an attribute perspective but similar structurally. The attribute difference is illustrated in the forth and fifth rows of Table I. Finally, the graph in Figure 5(b) is distinct structurally and this is evident in its Euclidean distance from 1(a) and 1(b). Note that the distinction in frequency domain is as obvious as that in time domain.

V. TEMPORAL DATA MINING EVALUATION

We evaluate the temporal representation by applying data mining to the temporal representation for selected feature sets, and assess the efficacy of the mining in revealing the kinds of information in which we are interested. The information in which we are interested includes "Given a new provenance graph that is either complete or incomplete, can we determine the type of workflow that generated it?" and "Can we detect failed workflows?" The experiment we conduct is multifaceted. We apply the Logical-P algorithm to a 10 GB provenance database discussed below using the structural

feature set discussed in Section IV. We maintain a time domain representation but also, through application of a Fourier transform, transform the representation into frequency domain. To the time domain representation we apply *unsupervised clustering* and *association rules* mining, of which we include results in this paper. To the frequency domain representation we apply *unsupervised clustering* and *sequence classification*.

Using the similarity measure between sequences from Section IV, we cluster the temporal sequences to discover a number of clusters, say K, to represent the different sequences. To prove the sufficiency of our provenance representation for clustering tasks, we apply the simple *K-means clustering* (Weka [12]), and evaluate its performance with *within-cluster sum of squares (WCSS)* and Purity [27].

The discovery of relevant association rules is one of the most important methods used to perform data mining on transactional databases [2]. An effective algorithm to discover association rules is the apriori algorithm [1]. Adapting this method to better deal with temporal information is beyond our current research; instead we apply the apriori method (Weka) on the clusters to get more descriptive knowledge of that cluster.

We use the Karma provenance tool [24] to store the 10GB provenance dataset and to export it in the form of OPM graphs. From these provenance graphs, we first create partitions based on the Logical Clock-P algorithm, and then generate provenance representations in both time and frequency domain.

The features we extract are the same as those discussed earlier, namely, the structural feature set. We choose this simple structural feature set over the attribute feature set or other more complicated feature sets for the purpose of a strong evaluation, in which we do not want to take advantage of obvious difference in node attributes. The disadvantage of this feature set is that if we have two provenance graphs with the same structure but with different node information, then it would be impossible to distinguish the two through graph structure alone. However, results on the 10GB provenance dataset show that even though there is only structural information captured, it is still sufficient for classification and unsupervised clustering.

A. 10GB Provenance Database

We posit that a provenance representation based on graph partitioning can support scalable analysis techniques and further that the solution is also resilient to errors in provenance data. To test this, we apply the reduction technique to a 10GB provenance database that has been generated using the WORKEM emulator [20] and has known failure patterns [6].

The 10GB database is populated with the provenance of approximately 48,000 workflow execution instances, the latter of which are modeled on the six real workflows as shown in Table II. Some of the workflows are small, having a few nodes and edges, while others like the Motif workflow have a few hundred nodes and edges. Each workflow type has approximately 2000 instances per failure mode, with failure modes including random dropped messages and workflows that fail. Table II shows the number of temporal subsets for a successful run.

TABLE II
WORKFLOW TYPE AND NUMBER OF TEMPORAL SUBSETS FOR A SUCCESSFUL EXECUTION INSTANCE

Workflow Type	**Temporal length of a complete run**
LEAD North American Mesoscale (weather) forecast	10
SCOOP ADCIRC (coastline)	5
NCFS (ocean)	10
Gene2Life (bio)	10
Animation (CS)	8
MotifNetwork (bio)	10

To generate a temporal representation for the 10GB provenance dataset, we apply partitioning to each provenance graph using the Logical Clock-P algorithm. We then extract structural features from each vertex subset to create time-domain provenance representations. The size of the original database is 10GB; the size of the temporal representation in time domain is 10.01 MB, a decrease by several orders of magnitude. The size of the temporal representation in frequency domain is 2.3MB, a further reduction by 25%.

B. Unsupervised clustering, time-domain

Assume we know nothing except the structural information in our representation of the 10GB provenance dataset. We want to create a high level view of the dataset by clustering workflow instances, so that we are able to tell the incorrect workflow instances by checking either the temporal length or the cluster centroid. To do this, we apply the simple *K-means clustering* on the provenance representations, and evaluate the performance of clustering using WCSS and Purity. We first evaluate the clustering on time-domain provenance representations. Using Euclidean distance as the similarity measurement limits the application of the simple *K-means clustering* to representation sequences of same length. Thus we first group together the provenance representations by their lengths and then apply the simple *K-means clustering* algorithm within each group.

This first order breakdown by temporal subset length is shown in Figure 2. 46% of the provenance representations have the largest number (10) of subsets, while only a small portion (2%) have very small number of subsets (2, 3); the latter result of workflows subject to early failures and dropped notifications.

For clustering within a grouping, we apply the SimpleK-Means clustering algorithm with euclidean distance measurement on the representation sequences inside each group. To choose the number of clusters, k, we plot the within-cluster sum of squares (WCSS) for each subset and look for the "elbow point". Figure 3 plots WCSS for workflow instances having 2 subsets (a) and 4 subsets (b). For the former, k is chosen as $k = 2$, for latter, we choose $k = 3$ because WCSS decreases slowly after k reaches 3. We use the same procedure to choose k for the remaining groups. Finally k-means is applied for each group creating an overview shown in Figure 4.

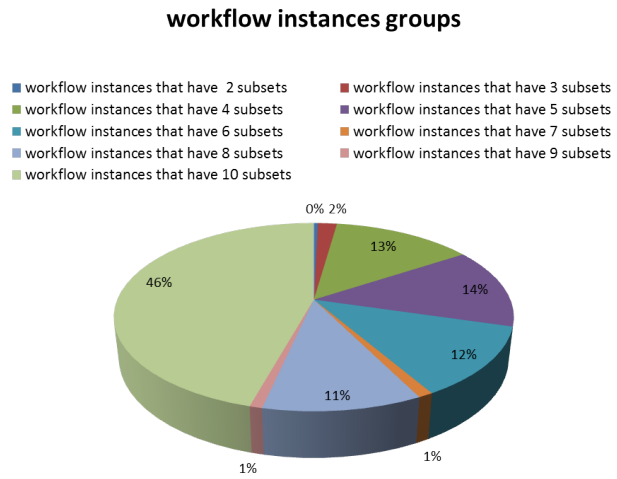

Fig. 2. Grouping result based on temporal subset length

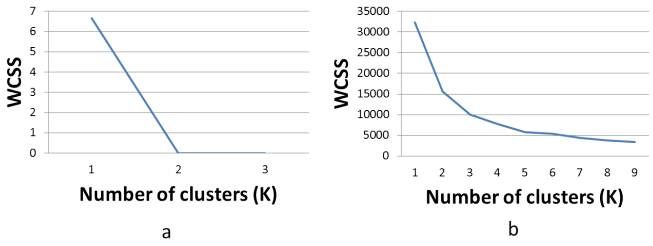

Fig. 3. WCSS as function of number of clusters for different groups of representation sequences: (a) WCCS for workflow instances having 2 subsets, and (b) WCCS for workflow instances having 4 subsets.

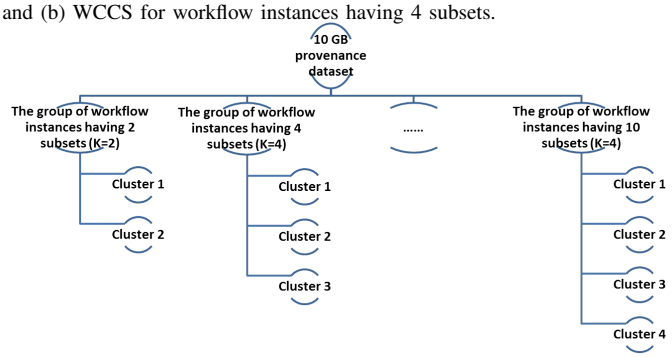

Fig. 4. High level view of 10GB provenance dataset created from its structural information only

Clustering graphs of the same temporal representation length requires different values of k, as shown in Figure 4. We found that the number k determined this way is slightly smaller than the number of actual classes within each group. However, it still generates major clusters and has good clustering quality (to be evaluated later). In fact, there is a trade-off between the number k and the value WCSS, since larger k always results in smaller WCSS but also has the potential to split the natural cluster into smaller clusters.

To help understand how to identify clusters of incorrect workflow instances, Figure 5 shows the provenance graphs of several centroids. We deliberately choose provenance graphs from a weather forecast workflow,because it best illustrates failures in provenance capture. It turns out that the NAM provenance graph with 10 subsets is a complete graph, while difficult to discern, this is evidenced by an artifact (circle) at bottom of graph. The NAM provenance graphs with less than 10 subsets partition the graph, all versions of which are incomplete and caused by dropped notifications. The NAM provenance graph with 2 subsets consists of some units of a complete provenance graph, which is very likely the result of failures.

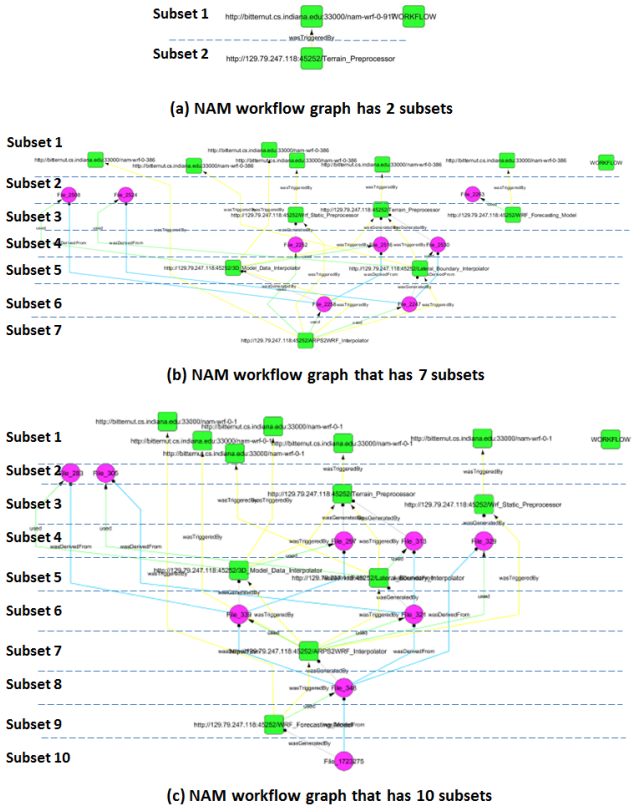

Fig. 5. Provenance graphs of several centroids. Square nodes represent processes, and circles represent artifacts. The graph is read top to bottom, with earlier activity at the top.

We evaluate the quality of resulting clusters by computing the purity as an external evaluation criterion by counting the number of correctly assigned workflow instances and dividing by total number of workflow instances – N. Formally:

$$purity\,(\Omega) = \frac{1}{N}\sum_{k}\max_{j}|\omega_k \cap c_j|$$

in which $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{\mathrm{K}}\}$ is the set of clusters and $C = \{\mathrm{c}_1, \mathrm{c}_2, \ldots, \mathrm{c}_{\mathrm{J}}\}$ is the set of classes(Here we use the workflow type as the class).

Figure 6 shows that the purity is not very high when we have a small number of subsets in the workflow representation. The reason is that most of the workflow instances that have smaller sizes of graph are incomplete and are generated by failures or dropped notifications (as shown in Figure 5), so they are difficult to accurately cluster using only their structural information. But the purity increases as the number

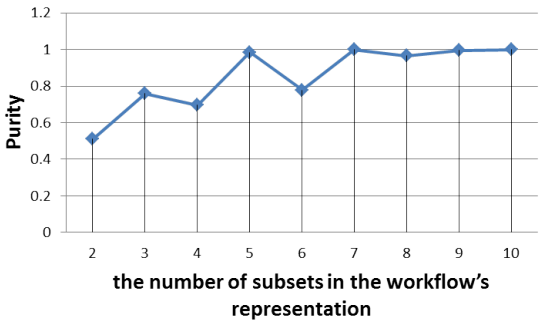

Fig. 6. Purity as external evaluation criterion for cluster quality by workflow instance group

of subsets in the provenance representation increases, and the workflow provenance that has most provenance information (with number of subsets > 4) can still support clustering well. This demonstrates that our representation of workflow provenance provides high level of clustering efficiency and is also robust in dealing with incomplete provenance.

C. Unsupervised clustering, frequency-domain

Compared with clustering time-domain provenance representations, frequency-domain provenance representations do not need to pre-cluster the provenance representations into groups of the same length. We evaluate SimpleKMeans clustering algorithm on a frequency domain representation by plotting the within-cluster sum of squares (WCSS) and computing the purity. WCSS decreases substantially with the increase in number of clusters in k-means algorithm. After the number K reaches 20, the WCSS becomes small enough and very stable as K increases. Purity increases as the K increases. After the number K reaches 20, the purity is high enough (0.88) and it also becomes stable afterwards. Compared with the 42 clusters we created from time-domain provenance representations, we generate only 20 clusters from frequency-domain provenance representations, with a slightly lower Purity. This demonstrates that our provenance representation in frequency domain can also support efficient unsupervised clustering.

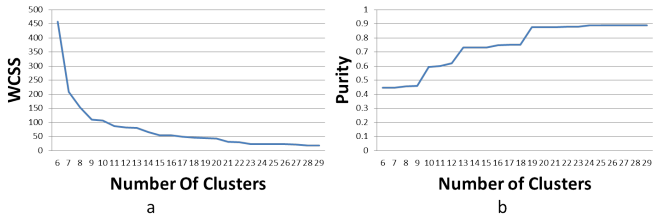

Fig. 7. WCSS (a) and Purity (b) as function of number of clusters in k-means

D. Workflow type classification

To categorize the type of a new workflow instance based on its representation in frequency domain, we train a classifier for workflow type from the 10GB dataset. We utilize the Bayes Network Classifier implemented in Weka (see [7] for full details), and its 10-fold-cross-validation shows that 96.6461% instances are correctly classified. This demonstrates that our provenance representation in frequency domain is sufficient for classification tasks at a high level of accuracy.

E. Association rules mining

We utilize Weka's apriori algorithm to discover the association rules on the resulting clusters generated from unsupervised clustering in the time domain. We are interested in association rules that can expose variants and be used in distinguishing different types of workflows. However, we found two issues before mining interesting association rules from 10GB provenance dataset. The first issue is: after a careful study of the dataset, we found that despite failed workflows and dropped messages, there are few variants amongst the workflow instances in that dataset. So we manually introduce two variants of NAM weather forecast workflow. The first introduces two intermediate data products generated for the last processing step, which leads to two final outputs; the second has one of the two pre-requisite files missing, which leads to the intermediate processes unable to continue, resulting in a failure execution. We generate time domain representation sequences for these two provenance graphs and adds them to the provenance representation of normal (complete) NAM workflow instances. The second issue comes with the apriori algorithm itself: it is less efficient when dealing with long sequences (there are 4 features selected for each subset, which leads to 40 attributes in a provenance representation for a workflow instance having 10 subsets). So we only select the attribute *Number of nodes in the subset* from each subset, forming a new representation sequence of length 10, and feed it into apriori algorithm. This new short sequence is sufficient to expose the variants we introduced, so it is good enough to be used in testing apriori algorithm.

After applying apriori algorithm (the representation sequences need to be discretized first), we look to the resulting association rules for rules related to the two variants.Table III shows the Scheme of the Weka method we applied and the resulting association rules that can reflect the variants we introduced. Rule 1 says that if the number of nodes in subset 8 (which are the data inputs for the last processing step) is between 0.8 and 1 (including 1), then number of nodes in subset 10 (which are the final data outputs) will be between 0.8 and 1 (including 1). Rule 2 says that if the number of nodes in subset 8 is larger than 1.8, then number of nodes in subset 10 will be larger than 1.8. Because the number of nodes can only be integer, rule 1 and rule 2 mean one intermediate data input for the final processing step will lead to one final data output, while more data inputs lead to more final data outputs, which reveals exactly the first variant we introduced. For the same reason, rule 3 reveals the second variant of failure execution. This single example shows that the time domain provenance representation with reduced number of features supports the apriori algorithm well: the association rules can show variants during execution; it describes the cluster well so that they can be further used to distinguish different clusters. See [7] for

TABLE III
SAMPLING OF ASSOCIATION RULES MINED BY APRIORI METHOD

Weka Scheme	Sample of association rules found
weka. associations. Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 0.4 -M 0.1 -S -1.0 -c -1	$1.\text{numberOfNodes\_8} =' (0.8 - 1]' ==>$ $\text{numberOfNodes\_10} =' (0.8 - 1]'$ $2.\text{numberOfNodes\_8} =' (1.8 - \text{inf})' ==>$ $\text{numberOfNodes\_10} =' (1.8 - \text{inf})'$ $3.\text{numberOfNodes\_2} =' (-\text{inf} - 1.1]' ==>$ $\text{numberOfNodes\_8} =' (-\text{inf} - 0.2]'$

TABLE IV
SUMMARY OF TEMPORAL MINING

Approach	Evaluation
Unsupervised clustering, time-domain	**Demonstrates that:** Representation leads to detect failed workflow instances **Disadv:** Need representations grouped based on length; creates small clusters inside representation group.
Unsupervised clustering, freq-domain	**Disadv:** Representations do not maintain meaningful information for mining association rules.
Classification, freq-domain	**Demonstrates that:** Can predict workflow type of new workflow instances.
Association rules mining, time-domain	**Demonstrates that:** Causal relationships captured between subsets; Some association rule sets can be used to distinguish different clusters. Can describe/distinguish different clusters. **Disadv:** Association rules built on time-domain representation reflects patterns on statistical features only; Apriori algorithm favors small representation length (less number of features).

full details.

F. Summary

The results of the evaluation are summarized in Table IV.

VI. CONCLUSION AND FUTURE WORK

In this paper we define a temporal representation for provenance graphs and apply it to produce partitions that preserve temporal orders between node subsets. The temporal representations generated by our method are three orders of magnitude smaller than the original provenance. Size can further be reduced by transformation into frequency domain. We show that the temporal representation is suited to creating a high level overview of an unknown provenance dataset. The representation leads to detection of failed workflow instances through unsupervised clustering. The representation also leads to prediction for the type of new workflow instances using the model trained from frequency-domain representations. The association rules we mined can show variants and can describe/distinguish clusters from one another. Though there is information loss when selecting features from the statistical feature space, the provenance representation we propose is well suited to temporal data mining tasks such as unsupervised clustering, classification and mining association rules, which are previously impossible for large scale provenance database like the 10GB database.

The open questions remaining with this work are several. The applicability of the representation has been shown for the 10GB database of synthetic provenance. How well does it work for a less well controlled provenance data set? How does the approach would extend to other provenance-specific questions, such as data lineage? Furthermore, we will investigate how to improve the scalability of representation process using MapReduce.

ACKNOWLEDGMENT

This work funded in part by NASA under grant number NNX10AM03G.

REFERENCES

[1] Agrawal, R. et al.: Fast algorithms for mining association rules. 20th Int. Conf. Very Large Data Bases, VLDB. 1215, 487-499,1949.
[2] Antunes, C.M. and Oliveira, A.L.: Temporal data mining: An overview. KDD Workshop on Temporal Data Mining. 1–13,2001.
[3] Bechhofer, S. et al.: Research objects: Towards exchange and reuse of digital knowledge. The Future of the Web for Collaborative Science,2010.
[4] Berkhin, P.: Survey of clustering data mining techniques. Grouping Multidimensional Data: Recent Advances in Clustering. 25–71,2006.
[5] Braun, U. et al.: Issues in automatic provenance collection. Provenance and annotation of data. 171–183 ,2006.
[6] Cheah, Y. et al.: A Noisy 10GB Provenance Database. 2nd Int'l Workshop on Traceability and Compliance of Semi-Structured Processes ((TC4SP), co-located with Business Process Management (BPM), 2011.
[7] Chen, P. and Plale, B. and Aktas, M.: Temporal Data Mining of Scientific Data Provenance. Indiana University Computer Science Technique Report. TR701, 2012.
[8] Davidson, S.B. and Freire, J.: Provenance and scientific workflows: challenges and opportunities. SIGMOD Conf. 1345–1350, 2008.
[9] Davidson, S. et al.: Provenance in scientific workflow systems. IEEE Data Eng. Bull. 30, 44–50, 2007.
[10] Gehani, A. and Kim, M. and Malik, T.: Efficient querying of distributed provenance stores. 19th ACM Int'l Symp on High Performance Distributed Computing. 613–621, 2010.
[11] Groth, P. and Moreau, L.: the Open Provenance Model XML Schema. http://openprovenance.org/model/opmx-20101012.xsd, 2010.
[12] Hall, M. et al.: The WEKA data mining software: an update. ACM SIGKDD Explorations Newsletter. 11, 10–18, 2009.
[13] Jung, J.Y. and Bae, J.: Workflow clustering method based on process similarity. Computational Science and Its Applications-ICCSA. 379–389, 2006.
[14] Kahn, A. B., Topological sorting of large networks", Communications of the ACM 5 (11): 558562, 1962.
[15] Kwasnikowska, N. and Moreau, L. and Van den Bussche, J.: A formal account of the open provenance model. Submitted for publication, 2010. http://eprints.ecs.soton.ac.uk/21819/
[16] Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Communications of ACM. 21, 558–565, 1978.
[17] Leake, D. and Kendall-Morwick, J.: Towards case-based support for e-science workflow generation by mining provenance. Advances in Case-Based Reasoning. 269–283, 2008.
[18] Margo, D. and Smogor, R.: Using provenance to extract semantic file attributes. USENIX 2nd Conf on Theory and practice of provenance, 7, 2010.
[19] Moreau, L. and et al.: The open provenance model core specification (v1. 1). Future Generation Computer Systems. 27, 743–756, 2011.
[20] Ramakrishnan, L. and Gannon, D. and Plale, B.: WORKEM: Representing and Emulating Distributed Scientific Workflow Execution State. 10th IEEE/ACM Int'l Conf on Cluster, Cloud and Grid Computing. 283–292, 2010.
[21] Santos, E. and Lins, L. and Ahrens, J.P. and Freire, J. and Silva, C.T.: A first study on clustering collections of workflow graphs. IPAW, 2008.
[22] Silva, C.T. and Freire, J. and Callahan, S.P.: Provenance for visualizations: Reproducibility and beyond. IEEE Computing in Science & Engineering. 82–89, 2007.
[23] Simmhan, Y.L. and Plale, B. and Gannon, D.: A survey of data provenance in e-science. ACM SIGMOD Record. 34, 31–36, 2005.
[24] Simmhan, Y.L. and Plale, B. and Gannon, D.: Karma2: Provenance Management for Data-Driven Workflows. Int'l Journal of Web Services Research. 5, 1–22, 2008.

[25] Smith, Steven W. ”Chapter 8: The Discrete Fourier Transform”. The Scientist and Engineer’s Guide to Digital Signal Processing (Second ed.). San Diego, Calif.: California Technical Publishing, 1999. ISBN 0-9660176-3-3.
[26] Zhao, J. et a.: Using semantic web technologies for representing e-science provenance. Semantic Web–ISWC 2004. 92–106, 2004.
[27] Zhao, Y. and Karypis, G.: Criterion functions for document clustering: Experiments and analysis. Machine Learning, 2002.