# Temporal Data Mining of Scientific Data Provenance

## Indiana University Technical Report TR701

Peng Chen[1], Beth Plale[1], and Mehmet Aktas[2]

[1] School of Informatics and Computing, Indiana University
{chenpeng,plale}@cs.indiana.edu
[2] 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.

## 1 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 MPI 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[9]. Visualization techniques are effective in making sense

of large data[21]. 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)[18]. 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 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 partitioning 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 2 reviews related work. Section 3 introduces the causal graph partitioning approach, while Section 4 describes the temporal representation. The experimental evaluation against a large database of provenance is presented in Section 6.2. Section 7 concludes the paper and discusses future work.

## 2 Related Work

The value that provenance brings to e-Science applications is first suggested in a 2005 survey of provenance[22]. Davidson, S.B. and Freire[8] provide an additional survey view of provenance. Davidson et al.[7] first introduce the problem of mining and extracting knowledge from provenance.

Margo and Smogor[17] 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[16] 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.

Simmhan and Plale [23] uses a decision tree inductive machine learning technique to classify discrete and continuous valued attributes into quality scores, thus to automatically determine the quality of provenance. However, the quality attributes that are identified are too specific for general purpose machine learning and data mining.

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 provenance graph is in some cases a coarse approximation of provenance graph. Santos et al.[20] 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[12] 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.

## 3 Provenance graph partitioning

The Open Provenance Model yields historically based directed graphs enriched with annotations. Nodes represent provenance of objects (whether digital or not) and the edges represent causal dependencies, between its source node, denoting the effect, and its destination, denoting the cause. OPM defines three kinds of nodes (artifact, process, and agent) and five base causal dependencies (used, wasGeneratedBy, wasControlledBy, wasTriggeredBy, wasDerivedFrom). Figure 1 shows an example provenance graph reproduced from the OPM core specification

v1.1 [18]. Ovals represent artifacts, and rectangles represent processes. The two kinds of edges shown, that is, Used and Was Generated By, are that a process used an artifact and that artifact was generated by a process respectively.

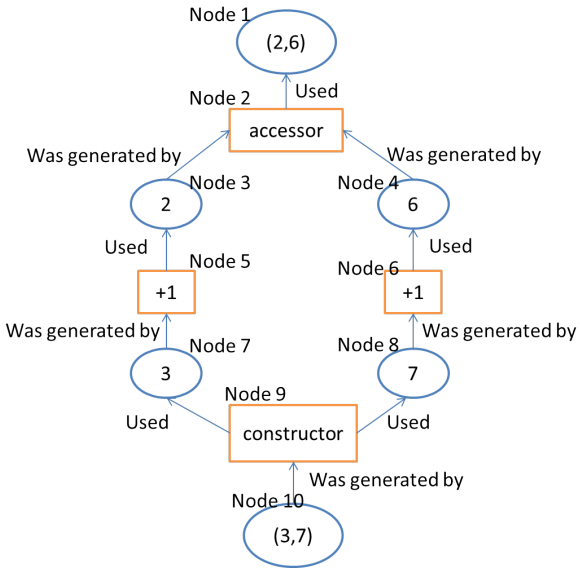


**Fig. 1.** Example of provenance graph from the OPM core specification v1.1

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[15] 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[18].

### 3.1 Partial ordering

Lamport determines a total ordering of events in a distributed computer system based on logical time order. Since the OPM reference specification[18] defines edges as causal relationships, we define the "happened before" relation in a provenance graph based on its causal relationships.

**Definition 1.** *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 Figure2(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[14] 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., [14]) 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.

### 3.2 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 happened at an earlier time than $b$. We state this condition more formally as follows.

**Definition 2.** *Clock Condition: The Clock condition satisfies the following condition: For any node $a$ and $b$, if $a \rightarrow b$ then $C(a) < C(b)$.*

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

1. $\mathrm{V}_1, \mathrm{V}_2, \ldots, \mathrm{V}_\mathrm{k} \in \mathrm{V}$ *and* $\bigcup_{1}^{k} \mathrm{V}_\mathrm{i} = \mathrm{U}$
2. $\forall\ i \neq j$ *and* $1 \leq i\ ,\ j \leq k$, $\mathrm{V}_\mathrm{i} \cap \mathrm{V}_j = \phi$
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[18]: agent is defined as an entity enabling process execution, process is defined as action resulting an articaft, 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 4.** *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 2, where the subset with the smaller number (e.g., Subset 1) "appears before" the subset with larger number (e.g., Subset 2, Subset 3).

### 3.4 Provenance graph partitioning algorithm (Logical-P algorithm)

Given any provenance graph (we are using the XML representation[10]), we generate an unique strict totally ordered partition with the following algorithm:

```
1: S ← Set of all nodes with no incoming edges
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
```

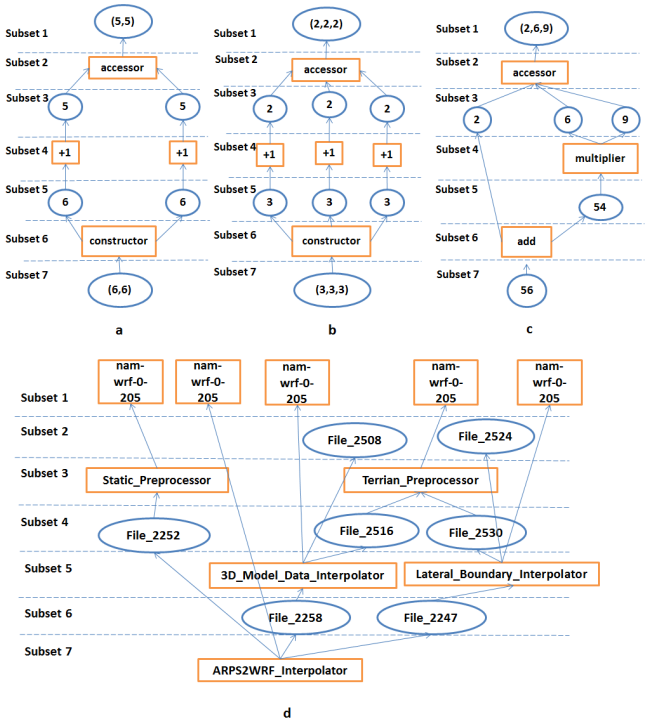


**Fig. 2.** Temporal partition: (a) is example provenance graph from[18]; (b) is from same experiment as (a) with different input data; (c) has similar graph structure to (a) and (b) but with different nodes; (c) has different graph structure to (a) and (b).

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[13], 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 $O(n + klogk)$, where $k$ is the number of subsets in the partition. Note that steps 9–11 gives nodes with multiple causes the maximum possible Logical Clock-P value.

## 4 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.

### 4.1 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 5.** *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 6.** *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 2(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 $D = \{$ average number of characters in node label, average number of in-degree, average number of out-degree $\} = \{1,1,1\}$.

### 4.2 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 have similar structure and similar attribute information are from related experiments (Figure 2(a) and Figure 2(b)); while provenance graphs from different experiments are either different in attribute information (Figure 2(a) and Figure 2(c)), or different in structure information (Figure 2(a) and Figure 2(d)).

Based on this assumption, we create a simple *attribute feature set* that includes "average number of characters in label" to discriminate between Figure 2(a) and Figure 2(c), and a simple *structural feature set* that includes "average number of in-degree/out-degree" to discriminate between Figure 2(a) and Figure 2(d). While using either feature set, Figure 2(a) and Figure 2(b) should be clustered together.

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 2(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 2(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 $< realpart, imaginarypart >$:

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

Table 1 gives the Euclidean distance for the attribute feature set and the structural feature set. As discussed earlier,graph 1(a) from Figure 2 is very similar to 1(b). Their distance is close for both attribute and structural feature sets as shown in Table 1. 1(a) – 1(c) is different from an attribute perspective but similar structurally. The attribute difference is illustrated in the second and third rows of Table 1. Finally, the graph in Figure 7(b) is distinct structurally and this is quite evident in its Euclidean distance from 1(a) and 1(b). Besides, though there is information loss while transforming representations from time domain to frequency domain, the distinction in frequency domain is comparable to that in time domain.

**Table 1.** Euclidean distance for attribute and structural feature sets

| Figure | Distance in time domain | Distance in frequency domain |
|---|---|---|
| | Attribute Feature Set | |
| 2(a) - 2(b) | 3.7417 | 0.2678 |
| 2(a) - 2(c) | 12.0 | 0.6183 |
| 2(b) - 2(c) | 12.7279 | 0.6764 |
| | Structural Feature Set | |
| 2(a) - 2(b) | 2.2361 | 0.1755 |
| 2(a) - 2(d) | 10.1281 | 0.7096 |
| 2(b) - 2(d) | 10.1113 | 0.5835 |

## 5 Temporal data mining methods

In sequence classification, each sequence (provenance representation) presented to the system is assumed to fall into one of finitely many (predefined) categories (workflow type) and the goal is to automatically determine the corresponding category for the given input sequence. To demonstrate that our provenance representation is sufficient for classification tasks, we test the performance of the Bayes Network Classifier implemented in Weka [11] with 10-fold-cross-validation on our temporal representation of a 10 GB provenance dataset.

Having defined an effective simple structural feature set, we can 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 will apply the simple k-means cluster provided by Weka on a 10GB dataset, and evaluate its performance with within-cluster sum of squares (WCSS) and Purity [27] in Section 6.4 and Section 6.5.

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 research; we will only apply the original apriori method provided by Weka on our clusters of representation sequences to get some descriptive knowledge of that cluster. We will demonstrate the power of the association rules we mined from one cluster in distinguishing itself with other clusters in Section 6.7.

## 6 Experimental Evaluation

### 6.1 10GB Provenance Database Dataset

To prove that the provenance representations using the graph partitioning approach can support scalable analysis techniques that are also resilient to errors in provenance data, we conduct our experiment using a 10GB provenance database with known failure patterns [6]. This 10GB database of provenance is populated from a workload of roughly 48,000 workflow instances that are modeled based on six real workflows as shown in Table 2

**Table 2.** Workflow type and the number of temporal subsets for its successful run

| **Workflow Type** | **Number of temporal subsets** |
|---|---|
| LEAD North American Mesoscale (NAM) initialized weather forecast workflow | 10 |
| SCOOP ADCIRC Workflow | 5 |
| NCFS Workflow | 10 |
| Gene2Life Workflow | 10 |
| Animation Workflow | 8 |
| MotifNetwork Workflow | 10 |

The LEAD NAM, SCOOP and NCFS are weather and ocean modeling workflows, Gene2Life and MOTIF are bioinformatics and biomedical workflows, and the Animation workflow carries out computer animation rendering. Some of the workflows are small, having few nodes and edges, while others like Motif have a few hundred nodes and edges. The second column in Table 2 gives the number of temporal subsets for a successful run of this type of workflow.

In the 10GB database, each of the six workflow types has 2000 instances per failure mode, with the failure modes as following:

1. No failures and dropped notifications (success case)
2. 1% failure rate
3. 1% dropped notification rate
4. 1% failure rate and 1% dropped notification rate

### 6.2 Experiment steps

Experimental evaluations are carried out on the 10GB provenance dataset discussed in Section 6.1, and you can see all the steps in Figure 3. We use Karma provenance system [24] to store the 10GB provenance dataset and to export the provenance in the form of OPM graphs. From these provenance graphs, we first create their partitions based on the Logical Clocks-P partitioning technique, and then generate provenance representations in both time and frequency domain. We then evaluate the performance of the classification task on the frequency-domain representation, the performance of unsupervised clustering on both frequency-domain and time-domain provenance representations, and the power of association rules mined on the clusters generated from time-domain provenance representations.

### 6.3 Generating provenance representations

To generate a temporal representation for the 10GB provenance dataset, we first apply partitioning to each provenance graph using the Logical Clock-P algorithm. We then select a set of features to be extracted from each vertex subset

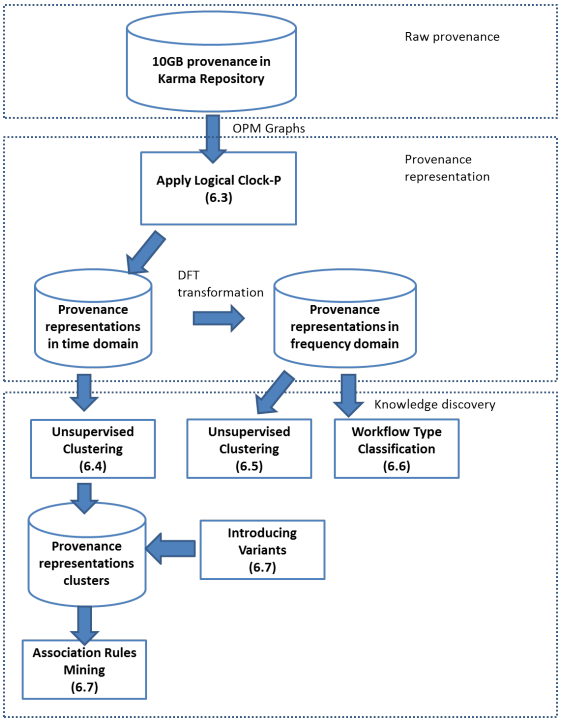


**Fig. 3.** Experiment Steps

to create time-domain provenance representations. The features we extract are the same as the simple structural feature set discussed in Section 4.2, namely,

*<Type of nodes in subset, Number of nodes in subset, Average number of in-degree of nodes in subset, Average number of out-degree of nodes in subset >*

This feature set only captures the structural information of each provenance graph, which has the advantage of generating smaller representations from graphs. The disadvantage is that if we have two provenance graphs with the same structure but with different node information, then it would be impossible for us to distinguish the two only by the graph structure. However, results show that even though there is only structural information captured, it is still sufficient for classification and unsupervised clustering.

For the 10GB provenance dataset, we create 47,914 time domain representation sequences having total size 10.01 MB, and store them as a CSV file. This gives a compression ratio approximately 1014 : 1.

From the time-domain provenance representations, we further transfer them into frequency-domain with first 3 frequencies. The resulting 47,914 representation sequences have a total size of 2.3MB, whose compression ratio is approximately 4348 : 1, and they are stored in a CSV file as well.

### 6.4 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. To do this, we apply k-means clustering algorithm on their provenance representations, and we evaluate the performance of clustering using WCSS and Purity.

We first evaluate the clustering on time-domain provenance representations. These time-domain sequences consist of meaningful attributes that are then used to generate association rules.

Using Euclidean distance as the similarity measurement limits the application of k-means clustering to representation sequences of same length. Thus we must first group together the provenance representations with the same length and then apply k-means clustering algorithm within each group. This first order breakdown by temporal subset length is shown in Figure 4. 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.

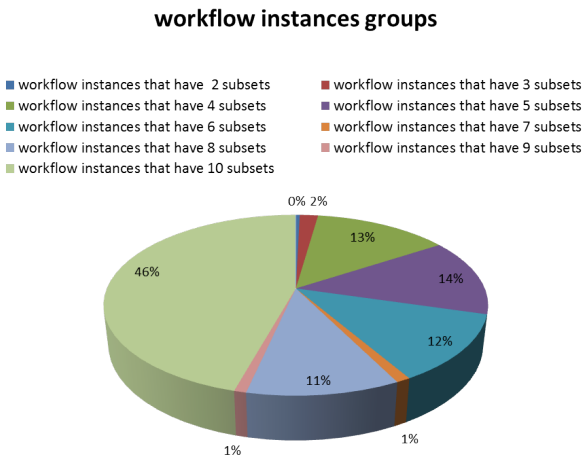


**Fig. 4.** Grouping result based on temporal subset length

For clustering within a grouping, we apply the SimpleKMeans clustering algorithm with euclidean distance measurement on the representation sequences inside each group. While using k-means algorithm, it is very important to choose the correct number k, and we address this problem by plotting the within-cluster sum of squares (WCSS) and looking for the "elbow point". Figure 5 shows the plotting of WCSS as a function of the number of clusters (K). We choose K=2 for

the group of workflow instances having 2 subsets, because the WCSS becomes stable after K reaches 2; we choose K=3 for the group of workflow instances having 4 subsets, because WCSS decreases slowly after K reaches 3. We use the same procedure to choose K for the rest groups, and finally we apply k-means algorithm on each group creating an overview shown in Figure 6.

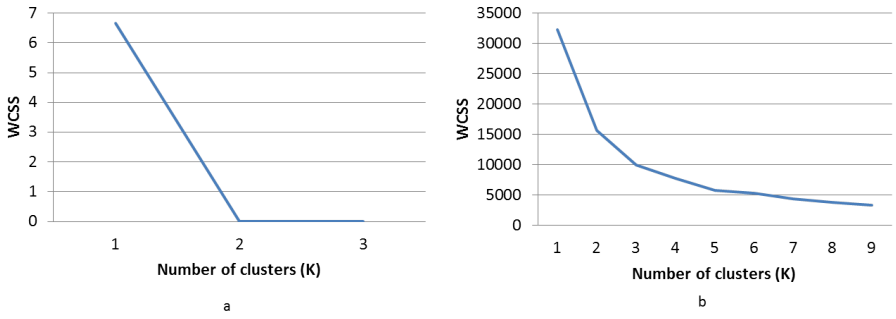


**Fig. 5.** WCSS as a function of number of clusters for different group of representation sequences. Figure 5(a) corresponds to the group of workflow instances having 2 subsets, and Figure 5(b) corresponds to the group of workflow instances having 4 subsets.

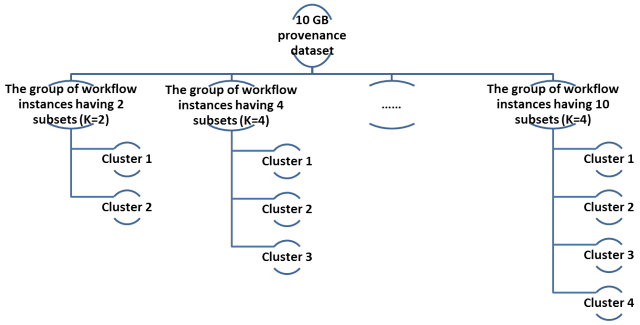


**Fig. 6.** A high level overview of 10GB provenance dataset created from its structural information only.

Clustering with graphs of the same temporal representation length requires different values of K, as shown in Figure 6. We found that the number K decided in 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 tradeoff 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 7 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.

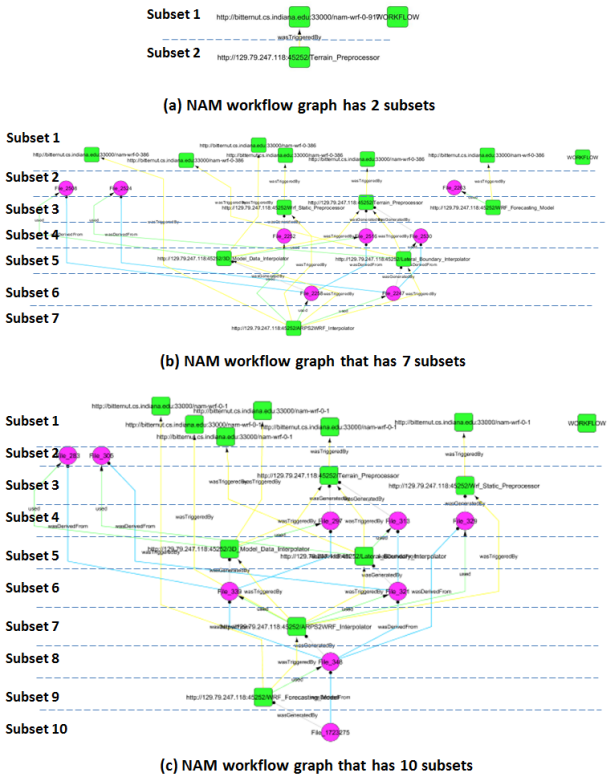


**Fig. 7.** Provenance graphs of several centroids. Square nodes represent processes, and circles represent artifacts. The graph is read top to bottom, with earlier activity at the top. Visualizations were done using a Karma plugin to Cytoscape visualization tool.

So far, we have created a high level view of the 10GB provenance dataset, and are able to tell the failures instances (outliers), but we still need to evaluate the quality of resulting clusters. To do this, we compute 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 $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of clusters and $C = \{c_1, c_2, \ldots, c_J\}$ is the set of classes(Here we use the workflow type as the class).

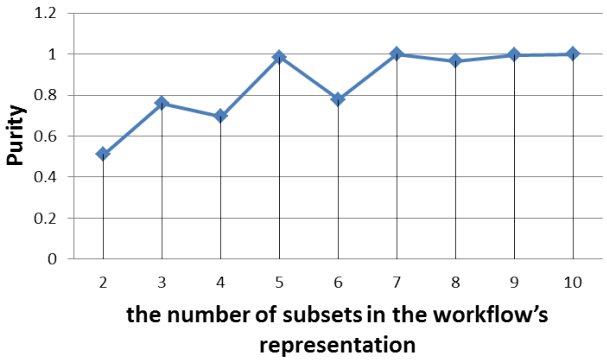


**Fig. 8.** Purity as external evaluation criterion for cluster quality by workflow instance group

Figure 8 shows that the purity is not very high when we have 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 7), so that they are hard to be clustered only by their structural information. But the purity increases as the number 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.

### 6.5 Unsupervised clustering, frequency-domain

The second step is to evaluate clustering on frequency-domain provenance representation. Compared with clustering time-domain provenance representations,

the advantage of clustering frequency-domain provenance representations is that we do not need to pre-cluster the provenance representations into groups of the same length.

We evaluate the application of SimpleKMeans clustering algorithm by plotting the within-cluster sum of squares (WCSS) and computing the purity. Figure 9(a) shows that the WCSS decreases a lot with the increase in number of clusters in k-means algorithm. After the number K reaches 20, the WCSS becomes small enough and become very stable as K increases. Figure 9(b) shows that the Purity increases a lot as the K increases. After the number K reaches 20, the purity is high enough (0.88) and it also becomes stable afterwards. There is also an interesting correlation between WCSS and Purity: the Purity increases whenever the WCSS decreases, and if the WCSS becomes stable, so does Purity. Compared with the 42 clusters we created from time-domain provenance

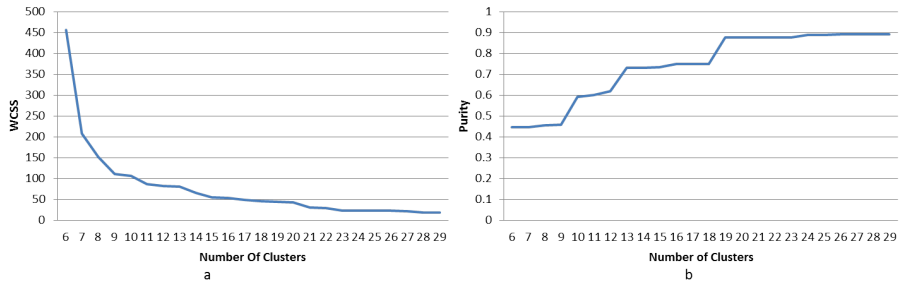


**Fig. 9.** WCSS (a) and Purity (b) as a function of number of clusters in k-means algorithm

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.

### 6.6 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, and provide the summary of its 10-fold-cross validation in Table 3. As shown in Table 3, 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.

**Table 3.** 10-fold-cross-validation summary for Bayes Network Classifier

| Weka Scheme | 10-fold-cross-validation summary |
|---|---|
| weka. classifiers. bayes. BayesNet -D -Q weka. classifiers. bayes. net. search. local.K2 – -P 1 -S BAYES -E weka. classifiers. bayes. net. estimate. SimpleEstimator – -A 0.5 | 1. Correctly Classified Instances 46307 96.6461 %<br>2. Incorrectly Classified Instances 1607 3.3539 %<br>3. Kappa statistic 0.9598 Mean absolute error 0.0113<br>4. Root mean squared error 0.089 Relative absolute error 4.053 %<br>5. Root relative squared error 23.872 %<br>6. Coverage of cases (0.95 level) 98.7811 %<br>7. Mean rel. region size (0.95 level) 17.4591 %<br>8. Total Number of Instances 47914 |

### 6.7 Association rules mining

As a final step, we utilize the apriori algorithm implemented in Weka to discover the association rules on the resulting clusters of Section 6.2.4. We want to find interesting association rules that are capable for runtime prediction, and we also want to use the association rules as part of the descriptive knowledge of each cluster.

The first issue with mining interesting association rules from 10GB provenance dataset is: after a careful study of the dataset, we found that there are no variants during the execution of all the different types of workflow instances in that dataset. So we choose to manually introduce two variants of NAM weather forecast workflow (See Figure 10(a) and Figure 10(b)). Figure 10(a) and Figure 10(b) are two different variants of the normal NAM weather forecast workflow (See Figure 7). Compared with the normal NAM workflow, Figure 10(a) has two intermediate data generated for last processing step, which leads to two final data outputs; Figure 10(b) 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 put them together with the provenance representation of normal (complete) NAM workflow instances.

The second issue comes with the apriori algorithm: it is is less efficient when dealing with long sequences (there are 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. After applying apriori algorithm (the representation sequences need to be discretized first), we look into the resulting association rules and find some rules related to the two variants. Table 4 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

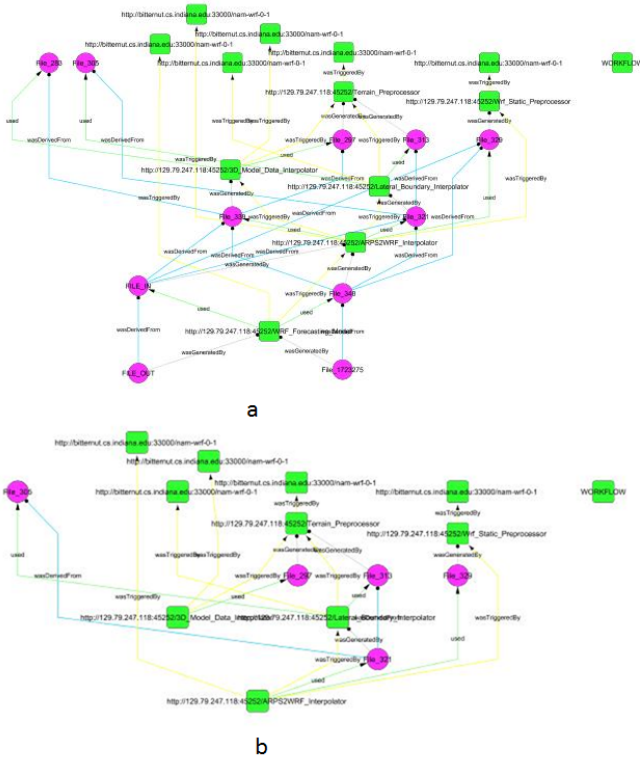


**Fig. 10.** Figure 10(a): the provenance graph of a NAM weather forecast workflow instance that have two data outputs. Figure 10(b): the provenance graph of a NAM weather forecast workflow instance that have one input data missing.

(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) means 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 variant in Figure 10(a). Rule 3) says that if the number of nodes in subset 2 is equal to or less than 1.1, then there will be no node in subset 8 (number of nodes is equal to or less than 0.2) , which reveals the variant in Figure 10(b). Notice that, if we have those variants during the workflow execution, we can use these association rules for runtime prediction. That is, following Rule 1) and Rule 2), if we have one data input for the last process, we will have one data output; but if we have more data inputs, we will have more data outputs. We can also predict the failure of execution by checking the number of pre-required files using Rule 3).

**Table 4.** Sampling of association rules mined by Apriori method provide by Weka

| **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.numberOfNodes_8 $=' (0.8 - 1]'$ ==> numberOfNodes_10 $=' (0.8 - 1]'$<br>2.numberOfNodes_8 $=' (1.8 - \text{inf})'$ ==> numberOfNodes_10 $=' (1.8 - \text{inf})'$<br>3.numberOfNodes_2 $=' (-\text{inf} - 1.1]'$ ==> numberOfNodes_8 $=' (-\text{inf} - 0.2]'$ |

To demonstrate the power of the association rules in describing a cluster, we choose 4 clusters of time-domain provenance representations (all have 10 subsets), mine 50 rules for each of them, and examine how well the rules can distinguish between different clusters. That is, 4 set of associations rules are mined from 4 cluster of provenance representations, and for each set of association rules, we validate all the provenance representations against them, to see whether it can distinguish the provenance representations from other clusters.

**Table 5.** Validation provenance representations against mined rules. The ratio represents how many representations passed the validation

| | **Provenance representations from Cluster 0** | **Provenance representations from Cluster 1** | **Provenance representations from Cluster 2** | **Provenance representations from Cluster 3** |
|---|---|---|---|---|
| **Validating against the rules from Cluster 0** | 100% | *0%* | *0.02%* | 100% |
| **Validating against the rules from Cluster 1** | 100% | 99.7% | *0.02%* | 100% |
| **Validating against the rules from Cluster 2** | 100% | 100% | 100% | 100% |
| **Validating against the rules from Cluster 3** | 100% | *0%* | *0.02%* | 100% |

Table 5 shows that when validating the provenance representations of other clusters against one cluster's association rules, there can be very low passing ratios (marked in italic). For example, none of the provenance representations of Cluster 1 can pass the validation of the association rule set mined from Cluster 0. Thus we can use this association rule set to distinguish provenance representations from Cluster 1 by validating the representation against it.

In conclusion, the time domain provenance representation with reduced number of features supports the apriori algorithm well: The association rules can show us different variants during execution, based on what we can have runtime prediction; They describe the cluster well so that they can be further used to distinguish different clusters.

**Table 6.** Summary of temporal mining

| **Approach** | **Evaluation** |
|---|---|
| Unsupervised clustering on time-domain representations | **Pros:**<br>Representation is 3 orders of magnitude smaller than original provenance (1014:1);<br>only 7.6% representations are incorrectly clustered.<br>Able to detect failed workflow instances (outliers)<br>**Cons:**<br>Need to group representations based on length; creates small clusters inside representation group. |
| Unsupervised clustering on frequency-domain representations | **Pros:**<br>Representation has a compression ratio of 4348 : 1;<br>has good WCSS (50) and purity (0.88) as number of clusters reach some number (20); generates smaller number of clusters (20).<br>**Cons:**<br>Representations do not maintain meaningful information for mining association rules. |
| Classification on frequency-domain representations | **Pros:**<br>Representation has compression ratio of 4348 : 1;<br>Distinct characteristic required to perform classification is still maintained, thus model trained for classification provides;<br>Can predicate the workflow type of new workflow instances. |
| Association rules mining on time-domain representations | **Pros:**<br>Association rules capture some causal relationship between subsets;<br>Some association rule sets can be used to distinguish different clusters.<br>Can predict the result in runtimes.<br>Can describe/distinguish different clusters.<br>**Cons:**<br>Association rules built on time-domain representations only reflect patterns on statistic features;<br>Apriori algorithm favor small representation length (less number of features). |

### 6.8 Comparing results

As shown in Table 6, the provenance representation we propose is well suited to temporal data mining tasks such as unsupervised clustering, classification and mining association rules. However, because of the abstract provenance representations have only statistic features from the original provenance graph, the association rules discovered are limited to statistical patterns. This makes the feature selection very important, and as long as we selected the right features that can reveal the variants during the execution of workflow instances, the resulting rules are sufficient to support runtime prediction well.

## 7 Conclusion and Future Work

The outcomes of this study are: we define a Logical Clock-P for provenance graph with causal dependencies, and we use it to produce partitions that preserve temporal orders between node subsets; based on the provenance graph partition, we propose a method to create its temporal representation in time domain by extracting statistical feature from each subset. The temporal representations generated by our method in our experiments are 3 orders of magnitude smaller than original provenance, and we can further reduce its size by transforming them into frequency domain; by applying different techniques from temporal data mining field against the generated provenance representations. We can 1) Create high level overview of unknown provenance dataset, and detect failed workflow instances (outliers) from the result of unsupervised clustering. 2) Predicate the type of new workflow instances using the model trained from frequency-domain representations. 3) Mine association rules capturing causal relationship between subsets from each cluster of time-domain representations. The association rules are able to perform runtime prediction and describe/distinguish different clusters.

Future work is to check other temporal data mining techniques on our provenance representations. Besides, the quality of provenance data is very important, and we will try to apply our methodology in determining the quality of provenance data. For example, we hypothesize that the smaller size the workflow instance cluster has, the more stable workflow model that cluster has. However, this needs to be tested.

## 8 Acknowledgments

We thank You-Wei Cheah of Indiana University for substantial guidance in conducting experiments on the 10GB provenance database. 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. Davidson, S.B. and Freire, J.: Provenance and scientific workflows: challenges and opportunities. SIGMOD Conf. 1345–1350, 2008.
8. Davidson, S. et al.: Provenance in scientific workflow systems. IEEE Data Eng. Bull. 30, 44–50, 2007.
9. 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.
10. Groth, P. and Moreau, L.: the Open Provenance Model XML Schema. http://openprovenance.org/model/opmx-20101012.xsd, 2010.
11. Hall, M. et al.: The WEKA data mining software: an update. ACM SIGKDD Explorations Newsletter. 11, 10–18, 2009.
12. Jung, J.Y. and Bae, J.: Workflow clustering method based on process similarity. Computational Science and Its Applications-ICCSA 2006. 379–389, 2006.
13. Kahn, A. B., Topological sorting of large networks", Communications of the ACM 5 (11): 558562, 1962.
14. 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/
15. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Communications of ACM. 21, 558–565, 1978.
16. 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.
17. Margo, D. and Smogor, R.: Using provenance to extract semantic file attributes. USENIX 2nd Conf on Theory and practice of provenance. 7, 2010.
18. Moreau, L. and et al.: The open provenance model core specification (v1. 1). Future Generation Computer Systems. 27, 743–756, 2011.
19. 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.
20. 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.
21. Silva, C.T. and Freire, J. and Callahan, S.P.: Provenance for visualizations: Reproducibility and beyond. IEEE Computing in Science & Engineering. 82–89, 2007.
22. Simmhan, Y.L. and Plale, B. and Gannon, D.: A survey of data provenance in e-science. ACM SIGMOD Record. 34, 31–36, 2005.
23. Simmhan, Y.L and Plale, B. Using Provenance for Personalized Quality Ranking of Scientific Datasets, Artem Chebotko, Yogesh Simmhan and Paolo Missier, eds. Intl Journal of Computers and Their Applications: Special Issue on Scientific Workflows, Provenance and Their Applications, 18(3), pp. 180-196, 2011

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.