Synthesizing Scenario-based Dataset for User Behavior Pattern Mining

Weina Ma
Dept. of Electrical, Computer and Software Engineering
University of Ontario Institute of Technology
Oshawa, Canada
Email: weina.ma [AT] uoit.ca

Kamran Sartipi
Dept. of Electrical, Computer and Software Engineering
University of Ontario Institute of Technology
Oshawa, Canada

Abstract—**User behavior pattern mining has drawn great attention in business and security areas. Realistic and accurate datasets are required for evaluating various user behavior pattern mining approaches, their implementations and optimization results. Synthetic datasets are crucial due to restricted access to production datasets, security and privacy issues, meeting specific needs of consumers, or the high costs of real datasets. This paper presents a synthetic dataset generator that effectively assists data scientists and analysts in designing scenario-driven datasets with embedded user behavior patterns, and visually analyzing the quality of the generated datasets. We developed an interactive data exploration environment to such a design-generate-visualize-analyze-optimize process. An abstract representation of the real-world user behavior pattern is proposed, which allows data analysts to create datasets with both intended and random patterns injected. Dataset generation is controlled by both data statistics (e.g., data size, and attribute distribution) and scenario-based user behavior patterns (e.g., association pattern, sequential pattern and time constraint). A prototype toolkit has been developed to synthesize and analyze the datasets in different application domains.**

Keywords-behavior pattern; synthetic dataset generation; data mining; visualization; sequential pattern mining; clustering

I. INTRODUCTION

User-system interactions are routinely recorded in audit logs, and are used to study the performance of the enterprises and the activities of their users. Identifying user behavior patterns from such data is valuable to e-commerce companies for consumer analysis, marketing strategy design, and security system for terrorist and criminal detection [1]. Unfortunately there is a general lack of access to real-world audit logs, and in particular in sensitive and mission-critical industries, such as healthcare, banking, and military. Moreover, finding or constructing a useful dataset from real-world systems is difficult due to the nondeterministic nature of the real-datasets especially in the early stages of the system during the production operation [2]. In general, the performances of different data mining approaches depend heavily on the testing datasets. Some algorithms are very sensitive to the target patterns and critical features. To decrease the algorithm evaluation time at the early stages, the data analysts can take advantage of a controlled simulation environment which allows them to acquire various simulated datasets containing embedded interesting patterns and features.

Creating datasets that represent real-world systems is challenging. A representative synthetic dataset of user-system interactions should exhibit realistic features such as: the frequency of user access requests that mimic both normal work days and busy system times; statistic biases towards certain type of actions (e.g., more reads than writes); highly associated features (e.g., specific users normally work at specific locations); and more importantly, the sequence that reflects the continuous or connected series of actions (e.g., what a user does next is impacted by what he or she did in the past few steps). Accurate and comprehensive user behavior pattern abstraction is necessary but has been less studied. To design a controllable dataset generator with realistic features, we analyzed the scenarios in several public and private datasets in the fields of healthcare, banking and traffic collision. Based on the analysis results, we extract sequencing, timing and association rules to define a scenario: sequencing requires that a series of steps occur in a certain order; timing limits the occurrence frequency of certain values; and association enforces that the occurrence of one value should result in one or more other values.

We developed a dataset generator toolkit for controllable dataset generation, suitable for unbiased evaluation of user behavior pattern mining algorithms. The proposed dataset generator toolkit mainly consists of three layers: i) behavior pattern representation layer; ii) dataset generation layer; and iii) dataset visualization and analysis layer. The behavior pattern representation layer defines a scenario as a behavior pattern based on sequencing, timing and association rules. This representation layer allows data analysts to design interesting features and patterns that will be injected into the dataset. The dataset generation layer creates datasets that are controlled by data size, data distribution, and the designed behavior patterns. The visualization and analysis layer provides an interactive exploration environment for visual analysis of the quality of generated datasets and a means to revise and enhance the generated dataset. Without such a generator it is impossible to test, evaluate and calibrate the algorithms that explore a system's production dataset, as the nature of a production dataset is unknown, and most probably it is not even available.

Our distinctive contribution in this paper is proposing a user behavior pattern representation technique based on sequencing, association and timing of the system's events. Data analysts can easily design and generate different testing datasets as benchmarks using our automated controllable dataset generator toolkit. The developed toolkit also provides clustering and visualization utilities to assist data analysts in obtaining deep insight into the generated dataset. The proposed approach significantly decreases the dataset development time, by applying a visual analysis method that helps to optimize the dataset design process. To demonstrate the usefulness of the approach, we developed a full case study for the healthcare domain, and two short case studies for banking and automobile traffic domains.

The remaining of this paper is organized as followings. Related work is discussed in Section II, and the relevant background knowledge is presented in Section III. In Section IV our scenario-based dataset generator for user behavior pattern mining is explained. Section V is allocated to case studies, and finally conclusion is presented in Section VI.

II. RELATED WORK

Most of the existing dataset generators create sequential dataset or clustering dataset with no specific goal, however we target to develop controllable user-behavior-based datasets. IBM Quest Synthetic Generator [3] is an open source market-basket synthetic dataset generator, capable of creating sequence dataset with randomly injected sequential patterns based on user specified parameters: sequence count, sequence pattern count, average sequence pattern length, sequence correlation, etc. The generated dataset is only suitable to evaluate the performance of data mining algorithms because of the extremely limited control on injected sequential patterns. SPMF (Sequential Pattern Mining Framework) [4] is an open-source data mining library that offers implementation of 75 data mining algorithms for sequential pattern mining, association mining, and frequent itemset mining, and clustering. SPMF also provides a sequence dataset generator by using completely random method. Another synthetic dataset generator for clustering and outlier analysis creates datasets based on different distribution and transformation [5]. Given the number of points and number of clusters, datasets could be generated according to the user specified distribution, density level, difficulty level and outlier level which are based solely on the mathematical parameters. In contrast to our approach, their synthetic datasets target for testing clustering and outlier analysis algorithms instead of user behavior pattern mining.

The user behavior pattern mining typically involves in sequential pattern mining, clustering and visualization. Sequential pattern mining is the process of applying data mining techniques to a sequential database for the purpose of discovering frequent subsequences as patterns. Sequential pattern mining has been widely used to obtain user navigational behavior via analyzing Web log data [6]. In [7] a behavior-based access control for distributed healthcare systems was proposed: an access control model captures user's dynamic behavior, and determines access right through comparing with the expected behavior pattern. Ideally, the distance between observed behavior and expected behavior is significant if the user acts abnormally. Similarity-based sequence clustering methods define and utilize similarity metrics to determine the closeness between all pairs of sequences to group similar sequences [8]. Clustering of sequential patterns allows to model common characteristics of sequences and also to seek for outliers [9]. An approach of approximate mining of consensus sequential patterns uses clustering as a preprocessing step to group similar sequences, and then mines the underlying consensus patterns in each cluster through alignment [10].

R is a programming language and open source software environment for statistical computing and graphics [11]. It provides a wide variety of data mining techniques and graphical facilities, such as time-series analysis, clustering and classification. An R package includes a set of powerful functions, which enables data analysts to go deep data mining by writing only several lines of code. Some of these packages are especially useful in our approach: "arules" is an R-package that provides the infrastructure for representing, manipulating and analyzing frequent itemset pattern and association rules among transaction data [12]; "arulesSequences" implements mining frequent sequence pattern algorithms [13]; "TraMineR" is a toolbox for mining, describing and visualizing sequences of events [14].

III. BACKGROUND

In this section, we provide an overview of different background knowledge required for our approach.

A. Frequent Sequential Pattern Mining

Agrawal [15] introduced sequential pattern mining of frequently occurring ordered events or subsequences. Consider a database of customer transactions where each transaction contains a customer-id, transaction time, and the items bought in the transaction. The collection of a customer's transactions, where each transaction contains a set of items and the transactions are ordered by increasing transaction time, is together viewed as a *customer-sequence*. Mining sequential patterns is the process of finding the frequent sub-sequences that appear in the customer-sequences more than a user specified threshold namely minimum support (*minsup*). The support for a sequence is the number of customer-sequences that contain that sequence as a sub-sequence. The sequential pattern mining algorithm passes the sequence database again and again. In the first pass, the algorithm finds the frequent "*1-length"* sequences. In each subsequent pass, the algorithm generates the candidate *k-length* sequences by joining frequent *(k-1)-length* sequences found in the previous pass, and then measures their support to determine whether they are frequent k-length.

B. Similarity-based Sequence Clustering

Data clustering is a method of grouping objects in a way that objects in one cluster are very similar to each other, but they are dissimilar to the objects in other clusters [16].

Similarity-based clustering algorithms define and utilize similarity metrics to describe quantitatively how close two data points are. The similarity problem of sequential data requires determining whether different sequences have similar behavior. Due to the unique characteristic of sequential data, most classic clustering algorithms and similarity measurement do not work well for sequence clustering. An obvious measure of the closeness of two sequences is to find the maximum number of identical items in these two sequences (preserving the symbol order), which is defined as Longest Common Subsequence (LCS) of the sequences [17]. Formally, let $X=(x_1, x_2, \ldots, x_m)$ and $Y=(y_1, y_2, \ldots, y_n)$ be two sequences of lengths m and n, respectively. A common subsequence *cs* of *X* and *Y* represented by *cs(X, Y)* is a subsequence that occurs in both sequences. The longest common subsequence *lcs* of sequence X and Y, *lcs (X, Y)* is a common subsequence of both sequences with maximum length. The length of *lcs(X, Y)* is denoted by *R(X, Y).* Solving *R(X, Y)* is to determine the longest common subsequence for all possible prefix combinations of the two sequences *X* and *Y*. Let *r(i, j)* be the length of the *lcs* of x_i and y_i, where $x_i = (x_1, x_2, \ldots, x_i)$ and $y_i = (y_1, y_2, \ldots, y_j)$. Then *R(X, Y)* can be defined recursively as following [17]:

$$r(i, j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ r(i-1, j-1) + 1 & \text{if } x_i = y_j \\ \max\{r(i-1, j), r(i, j-1)\} & \text{if } x_i \neq y_j \end{cases}$$

Based on the recursive definition, the LCS is found by a backtracking programming algorithm with time complexity O(m+n) [17].

C. Extracting Representative Sequences

Clustering decomposes a dataset into a set of disjoint clusters based on a pairwise similarity metrics. Data analysts are interested in the summarization of these clusters of sequences by describing the patterns that characterize them. More specifically, extracting representative sequences is the determination of the smallest possible number of representative subsequences that ensures a given coverage of the whole cluster [18]. This study is similar to the elimination of redundancy from data of clusters. Representative sequences are a set of non-redundant typical sequences that largely cover the observed sequences in a cluster. Such representative sequences are regarded as the most common and broad covering patterns which reveal the main traits of the cluster. The extracting algorithm requires a representativeness criterion such as the pairwise similarity metrics (e.g., LCS) under which two sequences are considered as identical or not. Neighborhood density, frequency, centrality are alternatives for measuring the redundancy. The result is also controlled by coverage level. You can increase the coverage level to explore more comprehensive traits from each cluster; alternatively, you can decrease the coverage level to get less but the most significant and typical characteristic of each cluster.

IV. APPRAOCH

In this section, we present the design of a highly configurable synthetic dataset generation toolkit that is characterized by: i) a behavior pattern definition interface that allows data analysts to design intuitive and interesting behavior patterns; ii) flexible parameters to customize datasets that vary in size, distribution, variety and complexity; and iii) pattern visualization capability that allows the analysts to interact and supervise the generated datasets.

A. Definitions of Terms

The following terms are used in the proposed approach.

Attributes: are extracted characteristics from application domain. They are fundamental elements that can be ascribed to events. An attribute is represented by a name and a domain of attribute values. For example, "action", "location" and "resource" are generic attributes extracted from audit logs. "Read exam" and "update report" are values of attribute "action" in the medical domain.

Event: records a single user-system interaction (i.e., any communication with the system such as storing and retrieving a resource). An event is represented by a set of "attribute, value" pairs. Whenever an "attribute, value" pair changes, a new event is recorded.

Behavior: is a consistent observation of a sequence of events performed by the same user, under certain environment during a specific time interval. Behavior is extracted from a collection of user-system interactions (events). In our approach behavior is represented as a quadruple:

Behavior=<Actor, Sequence, Context, Time Interval>

where i) *Actor* issues a behavior; ii) *Sequence* is the sequence of events performed by the Actor; iii) *Context* is the circumstances in which a behavior takes place; and iv) *Time interval* is the time duration within which the behavior is recovered.

Scenario: is a narrative that captures the interactions between users and systems, which is often considered as a chain of events for a specific purpose. In our approach, we represent an abstraction of a scenario as a behavior pattern.

Synthetic dataset: is a sequence of events that are composed of predefined attributes, with intended and random behavior patterns embedded.

B. Synthesizing Process

Figure 1 illustrates our proposed synthetic dataset, generation and visual analysis process. The first important task is how to easily define the expected dataset. We propose a behavior pattern representation using data mining concepts (association pattern, and sequence pattern) to constitute the behavior pattern, and mathematical tools (probability distribution) to indicate the statistical characteristics of data. The dataset generation process produces a dataset that contains predefined attributes (representing the intended behavior patterns) and ensures that such behavior patterns will be

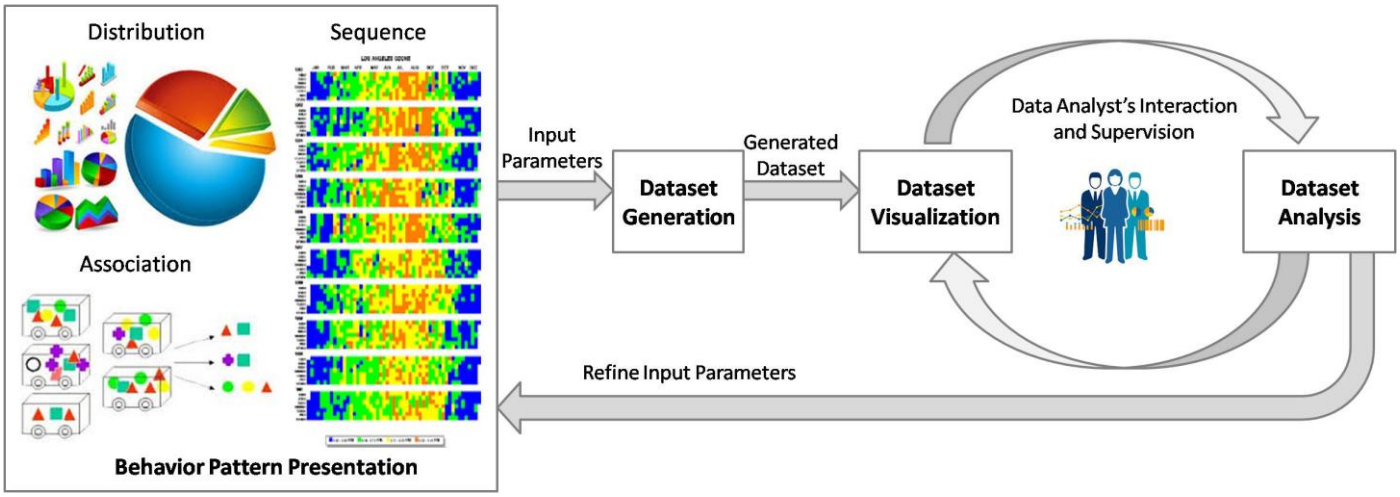

Figure 1. Our proposed dataset design, generation and visual analysis process.

embedded into the generated dataset. Such patterns are represented through associations of attributes, and sequences and time constraints of events. After producing the dataset according to the input parameters, the visual analysis step allows the data analysts to explore and verify the injected behavior patterns. The primary objective of the visualization is to extract simplified workable information from the dataset to effectively summarize and identify the embedded behavior patterns. The generated dataset allows the data analysts to apply distinct data mining techniques (e.g., association mining, sequential pattern mining, and clustering) to verify the differences between the generated dataset and the expected dataset. Using an iterative process, data analysts investigate the properties of the recovered behavior patterns, and finally refine the input parameters to optimize the dataset generation.

C. Dataset Design and Generation

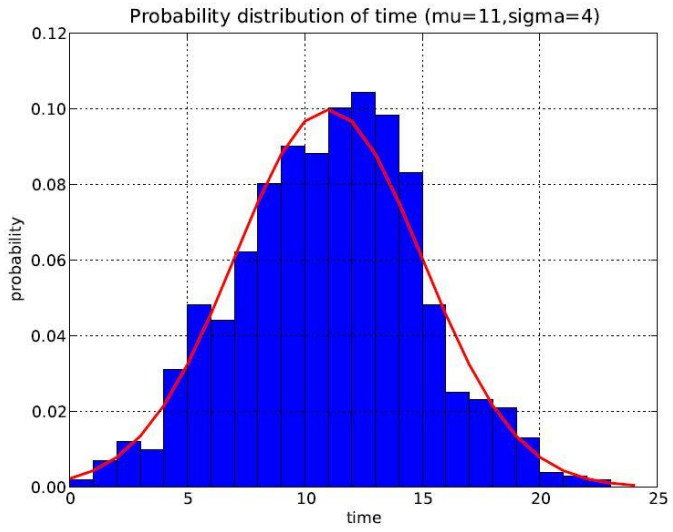

Figure 2. Normal distribution of attribute "time" with mean (μ =11) and devation (σ = 4)

Dataset generation process is controlled by: attribute distribution, and behavior pattern definition. Various continuous probability distributions (e.g., normal distribution, Poisson distribution, beta distribution) can be applied to indicate the desired distribution of attribute values. We select normal distribution to explain our method because it is remarkably useful with a well-defined mean value and well-defined variance. We propose a new abstraction of behavior pattern to assist data analysts in designing a desirable dataset.

1) Normal Distribution

Normal distribution [19] is specified by two parameters: the median value μ and standard deviation σ. The probability density function of random variable x (the attribute value) with a normal distribution is as follows:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Our approach generates events with randomly selected attribute values but following specified distribution. By given an attribute name, attribute domain (the scope of allowed attribute values), mean value and standard variety value, data analysts are able to define desired attribute distribution in generated events. Figure 2 indicates an example of attribute distribution which simulates the time of accessing patient's health records in a hospital. A real system may have the following features: the hospital has to access patient's records the whole day (24 hours) as patient may come to hospital anytime; most of access requests are from 7:00 to 20:00; and the daily rush hour of system access is 11:00am. To simulate a dataset with such features, the data analysts can define an attribute named "time" with 24 values; the mean value is 11:00am; and the deviation is 4 to simulate that daytime covers the most of access requests. Figure 2 shows the extracted attribute distribution of "time" from the generated dataset using our proposed generator.

2) Behavior Pattern Representation

The following statements are sample user-system interactions that can be expressed as the user behavior patterns:

- User u_i in a workflow process, reads resources r_a and r_b that belong to owner o_l, and then updates resource r_b.
- User u_i in a day takes roles in the following order: r_k, r_l ,r_h.
- User u_i in a week works at loc_a from Monday to Friday, and works at location loc_b on Saturday.
- User u_i accesses resources r_a, r_b about n_1 times on average between times t_1 and t_2.

In the above, the typical user behavior patterns can be expressed using our proposed behavior pattern representation "*Behavior=<Actor, Sequence, Context, Time-Interval, Support>",* where *Actor* is a specific user or a group of users; *Context* is explained by association to enforce that the occurrence of one attribute value should result in one or more other attribute values; *Sequence* requires that attribute values occur in a certain sequence; *Time-Interval* restricts the duration of behavior such as hourly behavior, daily behavior, or weekly behavior; and *Support* indicates the percentage of generated events should constitute this behavior pattern.

```
{
    {
        "id": "P-00001",
        "description": "typical doctor's workflow in radiology
        department",
        "context": {
          "role": "doctor",
          "department": "radiology"
        }
        "sequence": {
          "action": "create an order",
          "action": "read historical exams",
          "action": "create an exam",
          "action": "create a report"
        },
        "interval": "within 3 days",
        "support": "10%"
    }
    {
        "id": "P-00002",
        "description": "daily nursing ward-round",
        "context": {
          "role": "nurse"
        }
        "sequence": {
          "location": "ward-A",
          "location": "ward-B",
          "location": "ward-C"
        },
        "interval": "daily",
        "support": "15%"
    }
}
```

Figure 3. An example of behavior defined in JSON format

Figure 3 shows an example representation of user behavior patterns in healthcare system. The first behavior pattern *P-00001* describes a typical doctor's workflow in radiology department, of taking a medical exam for a patient in the following order: order an exam; read patient's historical exams; create a new exam for the patient; write diagnostic report for the new exam. This medical examination workflow should finish within 3 working days. The support specifies the percentage of events in which this behavior pattern exists. The second behavior pattern *P-00002* indicates daily nursing ward-round in hospital. In this example, nurses normally perform ward-round in order: ward-A, ward-B, ward-C. The support means around 15% of generated events should include behavior pattern *P-00002*.

In the example, the behavior pattern is expressed in JSON (JavaScript Object Notation) format [20]. JSON is built on two types of structures that fit our requirements very well: i) a collection of name-value pairs; and ii) an ordered list of values. The data analysts can design and express various behavior patterns in this simple JSON format. All the examples listed at the beginging of this subsection can easily be expressed in JSON format. Our generator would automatically transform the behavior definition to association patterns, sequence patterns and time constraints. These patterns and constraints control the dataset generation process.

3) Dataset Generation Algorithm

In this subsection, we formally define the proposed attribute distributions and behavior pattern definition. Let $A=\{a_1,a_2,\ldots,a_m\}$ be a set of attributes designed by the data analyst. Let $V = \{V_1, V_2, \ldots, V_m\}$ denote the collection of the allowed domains of values for different attributes, where $V_i = \{v_{i1}, v_{i2}, \ldots, v_{ix}, \ldots, v_{in}\}$ denotes the attribute domain of attribute a_i, and v_{ix} denotes a specific value of attribute a_i. Let $B=\{B_1, B_2, \ldots B_{j,\ldots}\}$ be a set of behavior patterns designed by the data analyst. Each behavior pattern is represented as a tuple $B_j=<B_j\text{-}a, B_j\text{-}s, B_j\text{-}c, B_j\text{-}t, B_j\text{-}sup>$, where $B_j\text{-}a$ is actor, $B_j\text{-}s$ is sequence, $B_j\text{-}c$ is context, $B_j\text{-}t$ is time interval, and $B_j\text{-}sup$ is support.

Figure 4 presents the Algorithm for the proposed dataset generator which consists of three main steps: i) parse and convert behavior patterns into association patterns, sequential patterns and time constraints, and randomly select users to support these behavior patterns; ii) build biased random attribute value selection function; iii) generate event dataset according to the rules constructed in steps i) and ii). First, the Algorithm loads the data analysts' input parameters including the attribute definition *A*, attribute domain *V*, user behavior pattern design *B*, all system users U_{All}, and the average event number per user per day *avg* and all event days D_{All} to control the dataset size. The description of the Algorithm with reference to its line numbers is as follows:

Line 2 to 7: the Algorithm transforms each behavior pattern into association patterns and constraint-based sequence patterns. As each behavior pattern is defined with a support parameter, the algorithm randomly selects $B_j\text{-}sup$ users from the system users U_{All} to support the behavior pattern B_j.

Line 8 to 10: the Algorithm builds a set of biased random attribute value selection functions that follow the attribute distribution definition.

TABLE I. VARIABLES OF ALGORITHM

Variables	Description
$E = \{e_1, e_2, \ldots, e_i, \ldots\}$ $e_i = <v_{i1}, v_{i2}, \ldots v_{im}>$ u_o-d_p-s_x = $<e_{op1}, e_{op2}, \ldots, e_{opx}>$	E is a set of events. An event e_i is a tuple of attribute values. Each event constitutes m attributes. u_o-d_p-s_x denotes a daily event sequence s_x (length is x) of user u_o at day d_p.
$B = \{B_1, B_2, \ldots, B_j, \ldots\}$ B_j = $<B_j$-a, B_j-s, B_j-c, B_j-t, B_j-$sup>$	B is a set of predefined behavior patterns B_j-a denotes the actor of behavior B_j; B_j-s denotes the sequence; B_j-c denotes the context; B_j-t denotes the time constraint; B_j-sup denotes the support.
$I = \{i_1, i_2, \ldots, i_j, \ldots\}$ $i_j = <v_{j1}, v_{j2}, \ldots>$	I denotes a set of association patterns. Each association pattern i_j enforces a group of attribute values (e.g., v_{j1} and v_{j2}) to occur together within a group of events, where the behavior context constraint B_j-c determines i_j and the selected events.
$S = \{s_1, s_2, \ldots, s_j, \ldots\}$ $s_j = <v_{j1}, v_{j2}, v_{j3}, \ldots>$	S denotes a set of sequence patterns. Each sequence pattern s_j enforces a group of attribute values (e.g, v_{j1}, v_{j2} *and* v_{j3}) to occur in time order and be inserted into an event sequence u_o-d_p-s_x, where the behavior sequence constraint B_j-s determines s_j and the selected event sequence.
$U_{All} = \{u_1, u_2, \ldots, u_n\}$ $U = \{U\text{-}B_1, U\text{-}B_2, \ldots, U\text{-}B_j, \ldots\}$ $U\text{-}B_j = \{u_{j1}, u_{j2}, \ldots u_{jl}\}$	U_{All} is the set of all users in the system. U denotes the collection of users that are selected to insert different behavior patterns. Un-selected users do not have behavior patterns. U-B_j denotes a group of selected users $\{u_{j1}, u_{j2}, \ldots u_{jl}\}$ to have predefined behavior pattern B_j. l is the number of selected users, which is controlled by the support of the behavior pattern B_j-sup.
F_k	F_k denotes a distribution function for generating random values for attribute a_k.
avg	avg denotes the average number of the generated events per user per day.
D_{All}	D_{All} denotes the collection of days of generated dataset. avg and D_{All} are used to control the dataset size.

Algorithm: dataset-generator
Input: *A, V, B, avg. D_{All}, U_{All}*
Output: *E*

```
1  I = Ø, S= Ø , E = Ø , U = Ø
2  for Bj in B do begin
3     association pattern ij = Bj-c
4     sequence pattern sj=Bj-s
5     apply time constraint Bj-t to sj
6     randomly select Bj-sup users U-Bj
7  end for
8  for ak in A do begin
9     build biased random value select function Fk
10 end for
11 for each user uo in UAll do begin
12    for each day dp in DAll do begin
13       x = randomly selected an integer around avg
14       generate an empty event sequence uo-dp-sx
15       for U-Bj in U do begin
16          if uo exists in U-Bj then
17             insert constraint-based sequence pattern sj to
               uo- dp-sx
18             insert association pattern ij to uo-dp-sx
19          end if
20       end for
21       for each event ei in uo-dp-sx do begin
22          for each empty attribute ak in ei do begin
23             call function Fk to assign random value vik to ei
24          end for
25       end for
26    end for
27 end for
```

Figure 4. Dataset Generator Algorithm

Line 11 to 14: the Algorithm starts to generate the events user by user, day by day. At line 13, the Algorithm assigns a random integer x around the value of avg as the number of events for user u_o at day d_p. At line 14, the Algorithm generates an empty event sequence u_o-d_p-s_x for user u_o at day d_p with length of x.

Line 15 to 20: if the user u_o is selected to support one or more behavior patterns, the algorithm inserts the corresponding association patterns and constraint-based sequence patterns into the event sequence u_o-d_p-s_x. Based on the time-constraints, the Algorithm must restrict the duration of events when inserting sequence patterns into event sequence. After inserting the patterns, the event sequence is only partially generated since some attributes that are not defined in patterns are still empty.

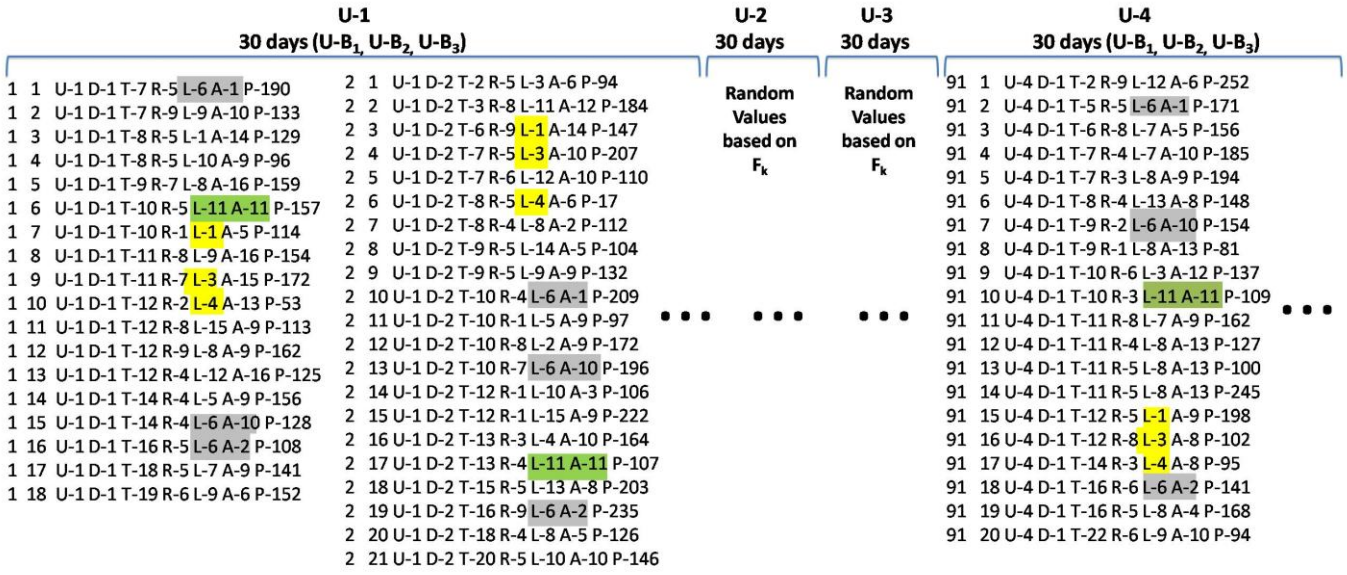

Figure 5. A slice of generated event dataset for 4 users where the averageg evernts per user per day is 20; day *D-1* of user *U-1*has 18 events; day *D-2* of user *U-1* has 21 events; day *D-1* of user *U-4* has 20 events. Users *U-1* and *U-4* are selected for insertion of 3 behavior patterns B_1, B_2 and B_3, which are highlighted with different corlors. These three event sequences represent u_1-d_1-s_{18}, u_1-d_2-s_{21} and u_4-d_1-s_{20} in the Algorithm of Figure 4. The events for users *U-2* and *U-3* are randomly selected based on the normal distribution function F_k in the Algorithm.

Line 21 to 25: the Algorithm calls the biased random attribute value selection function to assign values to the remaining attributes. If the user is not selected to insert any pattern, the algorithm executes section from line 21 to 25 to randomly assign all attribute values to each event.

Figure 5 shows a slice of the generated events for users *U-1, U-2, U-3* and *U-4* for a month (30 days), using our dataset generator Algorithm. Each row represents an event in the format of "*seuqnece_id event_id user date time role location action patient*". The first column is sequence number and the second column is event number of the sequence. Each attribute value is encoded in the format of a representation letter followed by an integer. For example, *D-2* represents the second day of the month. Consider three predefined behavior patterns B_1, B_2 and B_3 such that B_1-*s = {L-1, L-3, L-4}*, B_1-*t = {4 hours}*, B_2-*s = {A-1, A-10, A-2}*, B_2-*c = {L-6}*, B_3-*c = {A-11, L-11}*. The average of events per user per day is 20. When performing the event generator Algorithm, consider users *U-1* and *U-4* are selected to contain these three behavior patterns (i.e., U-B_1 *= {U-1, U-4}*, U-B_2 *= {U-1, U4}*, U-B_3 *= {U-1, U-4}*). First, the Algorithm parses these 3 behavior patterns and converts them into: association patterns i_2 *= {A-1, L-6}*, i_2 *= {A-10, L-6}*, i_3 *= {A-2, L-6}*, i_4 *= {A-11, L-11};* and constrained-based sequence pattern s_1 *= {L-1, L-3, L-4}* which happened within 4 hours and sequence pattern s_2 *= {A-1, A-10, A-2}*. Then the Algorithm generates the event sequences for users *U-1, U-2, U-3* and *U-4* day by day, respectively. It generates an empty event sequence for user *U-1* at day *D-1* with the length of 18. Then the Algorithm randomly inserts association patterns i_1, i_2, i_3, i_4 and sequence patterns s_1 and s_2 into the event sequence. As shown in Figure 5, the inserted behavior patterns are highlighted: B_1 is marked as yellow, B_2 as gray, and B_3 as green. After that, the Algorithm assigns values to the remaining empty attributes based on their attribute distributions. Following the same flow, the Algorithm generates event sequence for User *U-1* at day *D-2* with 21 events, and after 30 days it generates event sequences for user *U-2*, hich is not selected to insert any behavior pattern so that all attribute values of events for U-2 are randomly selected by function F_k based on the intended attribute distributions. The events for user *U-3* are also randomly generated based on random attribute selection function F_k. After 30 days for user *U-3*, the Algorithm generates 20 events for user *U-4* at day *D-1* with embedded behavior patterns, as shown in Figure 5.

D. Dataset Visualization and Analysis

After the process of dataset generation, the visual analysis step enables the data analysts to examine the details of the generated datasets. We developed a toolkit that can produce the following visual graphs for analyzing the dataset: sequence overview, frequent sequential patterns, and clustering representatives.

1) Sequence Overview

We generated an example dataset to simulate the medical system scenarios. We defined an attribute "action" with 16 values, where the attribute distribution follows normal distribution with mean value of "read exam" and the variety value (standard deviation) of 3. We added three typical medical behavior patterns as sequences: i) "read exam, read report, update report"; ii) "read exam, read order, create exam"; and iii) "create profile, read profile, update profile". We also defined two more parameters U_{All} and D_{All}: 100 users and 30 days. On average, we consider that each user has 10 access events per day (*avg* =10).

Based on the above design of attributes and behavior patterns, we utilize our developed generator to produce a

dataset with around 3,000 user sequences. A "user sequence" is an ordered list of events performed by one person per day. Now we verify whether the designed behavioral features are actually injected into the generated events, or not. Figure 6 illustrates a visualization of the user sequences produced by our dataset generator. Each row is a user sequence. An attribute value in an event is encoded as a colored rectangle, and each attribute value is colored differently. Each user sequence is represented by a series of colored rectangles. The total area of rectangles with a specific-color reflects the frequency of the corresponding attribute value in the dataset. The existence of a group of successive rectangles having the same sequence of colors reveals frequent subsequences. By visually comparing the overall areas of different colors, it reveals that "read exam" and "read report" are the most frequent actions, since the proportion of green color is larger than other colors. By visualizing user sequences, data analysts will get the first impression about the features of the generated dataset.

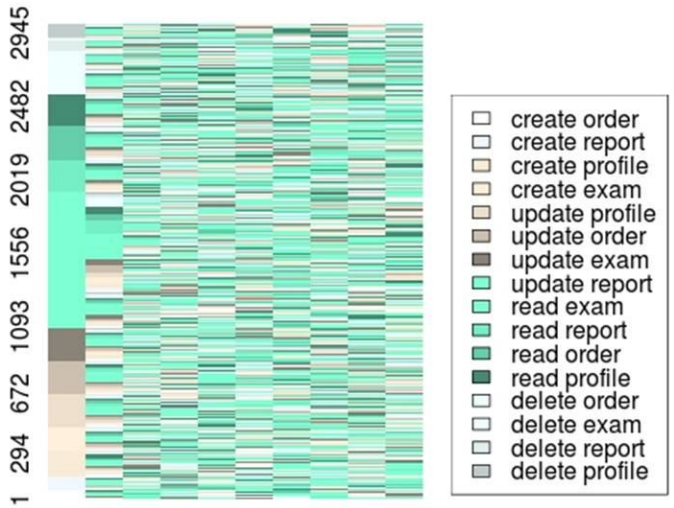

Figure 6. User sequences of action

2) Frequent Sequential Patterns

The graph of sequence overview in Figure 6 provides an overall impression of the most significant attribute values for the data analysts. While this overall view is important, due to the large population of the colored rectangles, it fails to provide a detailed and accurate analysis of the user behavior patterns. In other words, it is impossible to identify the sequential patterns and transitions of actions. The problem of mining user's frequent behavior pattern is to find the subsequences that appear frequently among all user sequences. Given a minimum support value, our analysis toolkit performs frequent sequential pattern mining operation on the generated dataset and displays the identified frequent subsequences in a graph. A subsequence is regarded as frequent if its support is greater than minimum support. Figure 7 shows the view of frequent action sequences produced by our generator toolkit from the same dataset in Figure 6. The analysis toolkit totally discovered 35 action sequences whose lengths are 3 and their supports are greater than 15% (it means over 15% of users have this behavior pattern). Each row presents a frequent action subsequence. By decreasing the minimum support, the data analysts could get more frequent sequences; and by increasing the minimum support, the data analysts may find less but more significant sequences.

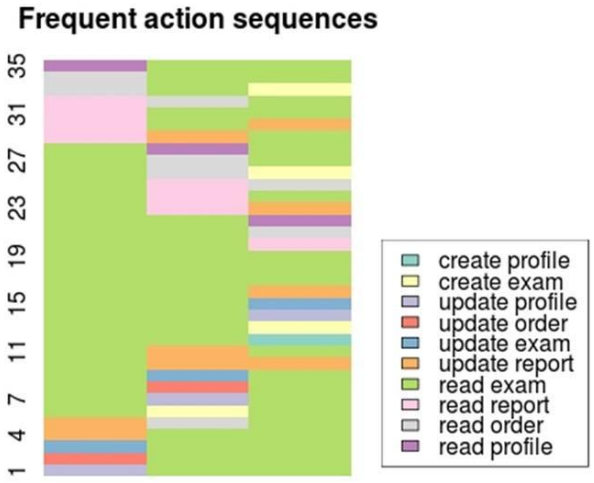

Figure 7. Frequent action sequences (minsup = 15%)

3) Clustering and Representative

Based on the visualization of sequence overview (Figure 6) and frequent sequences (Figure 7), the data analyst acquires the knowledge of significant actions and action sequences in the dataset. However, this is not enough. To obtain deeper knowledge, the data analyst may ask the following questions: How many types of behavior patterns do exist in the frequent behaviors? Which behavior patterns are similar? What are the common characteristics among similar behaviors? Data analysts could get answers by dividing the dataset into a number of clusters and exploring the representative patterns of each cluster. Our dataset generator allows the data analysts to designate different similarity measurement for clustering and input the number of expected partitions. By selecting LCS (Longest Common Subsequence) as sequence similarity measurement and inputting the expected partition as 3 clusters, our analysis toolkit is able to perform clustering operation on frequent action sequences. Figure 8 shows the result of cluster visualization produced by our analysis toolkit. It can be seen that cluster 3 is very small with only 3 sequences, but cluster 1 is comparatively large with 26 sequences. To eliminate the redundancy of sequences in each cluster, the data analyst can utilize the toolkit to explore the representative sequences. Figure 9 presents a set of non-redundant representative sequences for each cluster. The vertical size of rectangle presents the sequence frequency. A desired percentage of coverage of sequences in the cluster is required as user input. By increasing the coverage parameter, the data analyst could get more sequences with higher coverage; and by reducing the coverage parameter, the data analyst may find sequences with high representative scores.

The explored attribute frequency, frequent sequence patterns, cluster patterns and representative patterns provide useful knowledge for data analysts to optimize the dataset design and generation.

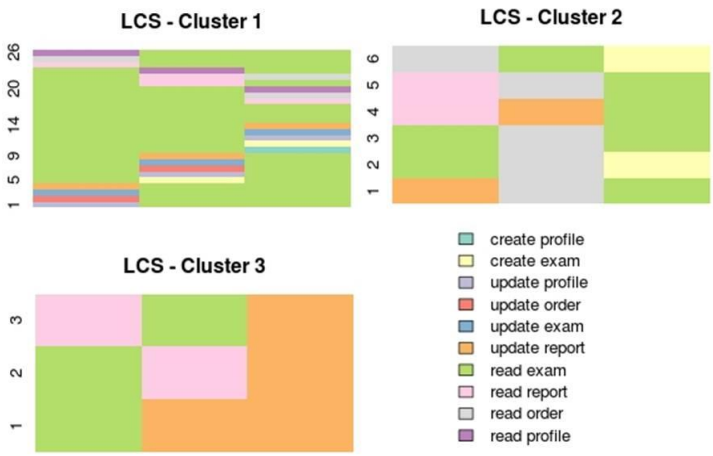

Figure 8. Clusters of frequent action sequences based on similarity measurement of LCS

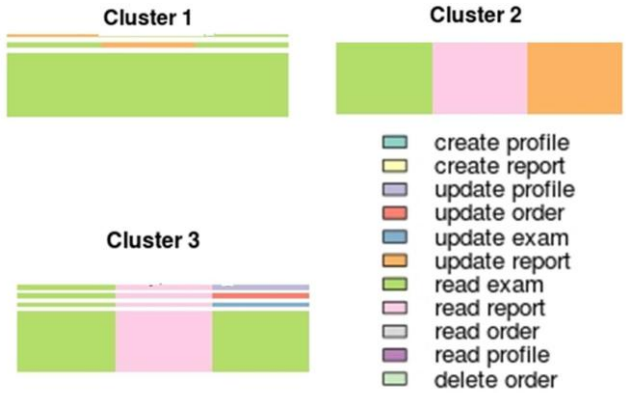

Figure 9. Cluster representative with coverage of 30%

V. CASE STUDY

To evaluate the functionality and feasibility of our dataset generator, we produced a dataset to simulate user-system interactions in distributed medical imaging systems. The attribute characteristics and designed user behavior patterns are selected after analyzing the audit log specification in distributed PACS systems (Picturing Archiving and Communication System). We use this case study to demonstrate the dataset design, dataset generation, and visual analysis. Besides, we designed two more datasets to simulate banking services and traffic collisions to prove that our dataset generator toolkit is useful for generating datasets in different application domains.

A. Dataset Design and Generation

It is very challenging to access the production audit logs from healthcare organizations due to the patient's sensitive information, privacy issues, and ethics board's approval. Therefore, synthetic dataset is required to assist researchers and developers for systematic testing, analyzing and evaluating different techniques and software solutions to be approved for healthcare domain. Distributed PACS systems follow RFC3881 "Security Audit and Access Account ability Message XML Data Definitions for Health Applications" [21]. This document defines the minimum set of attributes that need to be captured for security auditing in healthcare application systems. In our experiment, an event is composed of a group of attributes based on RFC3881 definition, as follows: *Event=<User, Location, Action, Patient, Date, Time>*, where *User* is the user identifier; *Location* is the current location of user; *Action* is the service that the user requested; *Patient* is the patient identifier indicating the owner of the requested resource; *Date* and *Time* record the timestamp.

TABLE II. ATTRIBUTE DISTRIBUTION DEFINITION

Attribute	Rep	Domain	Type	Mu	Sigma
User	U-	100	Random	-	-
Location	L-	15	Normal	8	4
Action	A-	16	Normal	8	3
Patient	P-	300	Normal	150	50
Date	D-	30	Normal	15	5
Time	T-	24	Normal	11	4

Table 2 indicates the input parameters of the attribute distribution. Attribute value is encoded by an "identifying letter" plus an integer number (e.g., U-1, D-2, L-4). The attribute domain specifies the scope of the allowed attribute values. In our experiment, we simulated a system consisting of 100 users with different roles (physicians, lab specialists, etc.) and 300 patients. The timeline of the simulated events is one month. The attribute Time records the hour when the event occurs. Minute and second can be added if data analysts require more accurate timestamps. The attributes with normal distributions need to be configured with two parameters: mean (mu) and variance (sigma). For example, the attribute Time with normal distribution and "mu=11" means "11:00am" is the rush hour in real systems within the hospitals.

TABLE III. ANALYST DEFINED BEHAVIOR PATTERNS

Pattern Id	Sequence	Support
P-00001	office-1-Juravinski-Hamilton, office-3-Juravinski-Hamilton, office-4-Juravinski-Hamilton	30%
P-00002	office-3-Lakeridge-Oshawa, office-4-Lakeridge-Oshawa, office-5-Lakeridge-Oshawa	25%
P-00003	office-2-McMaster-Hamilton, office-1-McMaster-Hamilton, office-3-McMaster-Hamilton	20%
P-00004	read exam, read report, update report	30%
P-00005	read exam, read order, create exam	25%
P-00006	create profile, read profile, update profile	20%
P-00007	11:00, 12:00, 14:00	30%
P-00008	10:00, 11:00, 12:00	25%
P-00009	14:00, 15:00, 16:00	20%

Table 3 shows 9 typical user behavior patterns that constitute ordering, timing, sequence and combination relationships among events. Pattern *P-00001* expresses that a user in a single day works at three different locations in the following order: office-1-Juravinski-Hamilton, office-3-Juravinski-Hamilton, office-4-Juravinski-Hamilton. *Support* is the percentage of users whose event sequences include this location sequence pattern. Pattern *P-00002* and *P-00003* are two more examples of location sequence patterns with different supports. *P-00004* shows a typical physician workflow in radiology department in the following order: read a new exam; read its diagnostic report; update the diagnostic report. *P-00005* and *P-0006* are also two typical workflows in a radiology department with different supports. *P-00007 – P-00009* indicate three time sequence patterns during which physicians have accessed the medical images.

Our dataset generator transforms these 9 user behavior patterns onto the JSON formats as the input parameters. By adding more parameters with average event count per user per day to control the size of the dataset, our generator engine generated 30,000 events with embedded designed features.

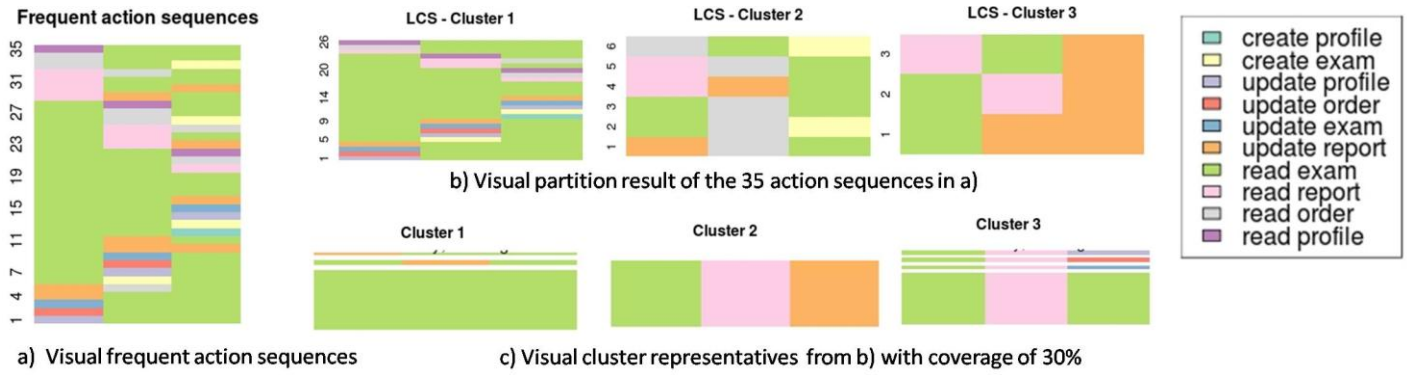

Figure 10. Visual analysis of action sequences

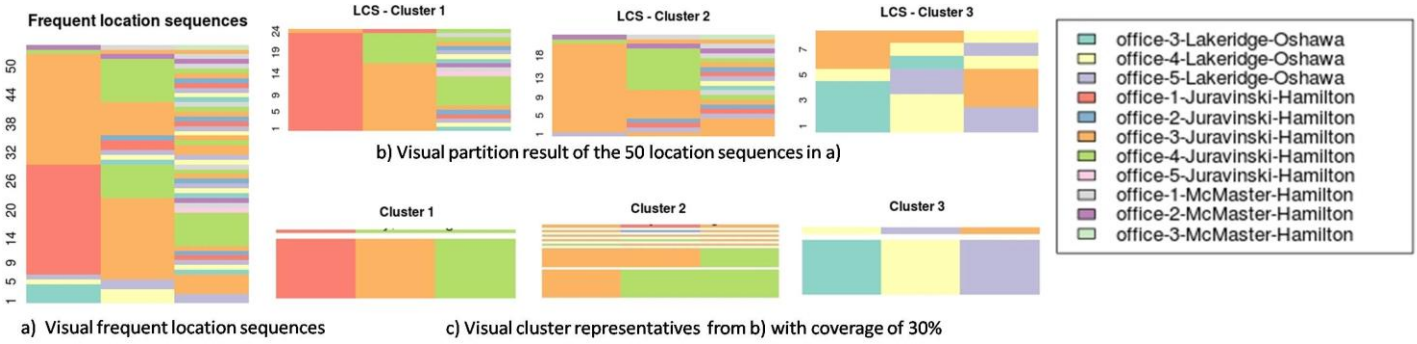

Figure 11.Visual analysis of location sequences

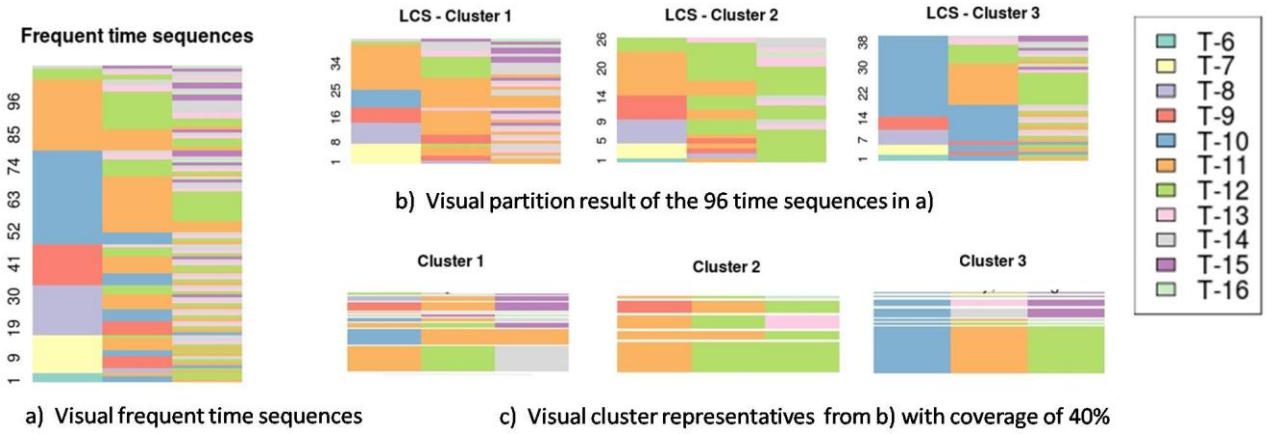

Figure 12. Visual analysis of time sequences

B. Dataset Visualization and Analysis

To verify whether the generated dataset contains the embedded designed features, we use our toolkit to analyze the dataset in three aspects: "action sequence analysis", "location sequence analysis", and "time sequence analysis". Figures 10 to 12 illustrate the visualization of the behavior patterns to analyze the results.

As we see in Figure 10(a), 35 frequent action sequences are extracted with minimum support of 15%. Using a hierarchical clustering operation with LCS pairwise similarity metric between sequences, the frequent action sequences in Figure 10(a) are partitioned into three clusters as shown in Figure 10(b), where Cluster 1 with 26 action sequences is populated essentially by the "read exam" operation; Cluster 2 with 6 action sequences gathers more "read exam" and "update profile" operations; and Cluster 3 mainly gathers "read exam" and "update report" operations. Figure 10(c) shows the representative sequences that are extracted using the neighborhood density criterion with a coverage parameter of 30%. As we can see, Cluster 2 is represented by a sequence of "read exam, read report, update report" operations, which is exactly the predefined pattern "P-00004". The extracted representative sequences do not include patterns "P-00005" and "P-00006". However, if we increase the coverage parameter to 80%, "P-00005" is one of the representatives of Cluster 3 and "P-00006" appears in representative sequences of Cluster 1. Similarly, Figures 11 and 12 are visual analysis of location sequences and the time sequences in terms of: a) visual frequent sequences; b) visual sequence partitioning with a three-cluster parameter; and c) visual explored representative sequences of each cluster with a coverage parameter of 30% and 40% for Figure 11 and 12, respectively. All predefined user behavior patterns about location sequences are explored: "P-00001" in Cluster 1, "P-00002" in Cluster 3, and "P-00003" in Cluster 2. As for predefined behavior patterns of time sequences, we can see "P-00007" appears in Cluster 1, and "P-00008" exists in Cluster 3. However, "P-00009" does not exist in any cluster. By adjusting the parameters such as minimum support, cluster number and representative coverage, data analysts are capable of visually analyzing the characteristics of the generated datasets.

C. More Application Domains

Various application domains can benefit from our designed toolkit by generating datasets with domain specific features and behavior patterns. For example, in [22] a Consultant Service in a banking organization is proposed that is capable of matching the customer's attributes with those of previous customers under similar circumstance to provide the new customer with the best possible recommendations. Our dataset generator can be used to design and generate the desired dataset to train the recommendation system. Based on the investigation of several services offered by a Banking organization, we designed the attribute-tuple for banking event as <*Status, Occuption, TypeOfService, Age, AmountOfMoney, UseOfMoney, CreditHisotry, Degree*>.

TABLE IV. BANKING SERVICE BEHAVIOR PATTERNS

Pattern Id	Sequence	Time Constraint	Support
P-00001	BB, BBB, AA, AAA	4 months	10%
P-00002	AAA, BB, CCC	3 months	5%
P-00003	Good, Good, Excellent	3 years	15%
P-00004	Fair, Good, Poor	3 years	10%
P-00005	Travel, Travel, House, House	4 years	15%
P-00006	Tuition, Tuition	2 years	5%

Using our dataset generator interface, the system designer is able to define the distribution of different attributes of the banking event and the banking service scenarios to be injected into a training dataset. Table 4 indices 6 example behavior patterns in the banking service systems. Suppose that a *CreditHisotry System* for bank customers records the customer's credit ratings from excellent to poor, as: *AAA*, *AA*, *A*, *BBB*, *BB*, *B*, *CCC*, *CC*, *C* and *D*, respectively. Pattern *P-00001* expresses that a pattern of credit rating is improving since the past 4 months. On the other hand, pattern P-00002 is a pattern of credit rating is deteriorating in the past 3 months. People's income *AmountOfMoney* is divided into ranges of *Excellent*, *Good*, *Fair*, and *Poor*. Pattern *P-00003* and *P-00004* are two example patterns of income changing in recent 3 years. Attribute *UseOfMoney* tracks customer's expenses. Pattern P-00005 reveals young people mostly spend money on travel when they are single, and then spend a large amount of money on purchasing a house after marriage. P-00006 indicates normally students bear the burden of tuition.

TABLE V. TRAFFIC COLLISION BEHAVIOR PATTERNS

Pattern Id	Association	Support
P-00001	Snowing, Climbing lane	5%
P-00002	Young, Sport, Injury	10%
P-00003	8:00am, No injury	20%

Table 5 presents another example regarding the traffic safety recommendation, which requires a detailed testing dataset when collisions happen, such as: collision level, vehicle's attributes, driver's attributes, weather and location. After investigating several Canada's traffic collision reports, we designed the attribute-tuple for traffic collision event as <*CollisionTime, CollisionLocation, Critical, VehiclesInvolved, VehileModel, Weather, RoadSurface, DriverAge, DriverGender*>. Depending on our generator interface, the safety recommendation system designer can produce various testing datasets with designed collision features. Table 5 shows 3 example behavior patterns based on association pattern to describe traffic collisions. Pattern *P-00001* reveals that traffic collision is easy to happen in the climbing lane during snow fall. Pattern *P-00002* explains more young people driving sport vehicles have injury in traffic collision because of over speed. And many traffic collisions happen at 8:00am but with few injuries as 8:00am is rush hour.

VI. CONCLUSION

In this paper we proposed a general purpose, scenario-oriented synthetic dataset generator, and provided for the data analysts a series of utilities for visualizing, analyzing and verifying the generated dataset in an integrated environment. The proposed dataset generator assists data analysts to evaluate their algorithms by creating rich datasets with the desired statistical characteristics and behavior patterns. Our designed system offers a user-friendly interface and comprehensive parameters to create datasets that vary in size, frequency, variety, and complexity. With the proposed behavior pattern representation, data analysts can easily define user behavior patterns that are constructed using timing, association, sequencing, and combination relationships among events. Our visualization and data mining toolsets that are built on top of the R packages are very helpful to evaluate the quality of the generated datasets. The interactive visual exploration approach helps uncover data characteristics and behavior patterns step by step, deeper and deeper. The extracted knowledge will help data analysts design better testing datasets, leading to shorter dataset development time.

There are still a few aspects of our approach to be further enhanced in the future. The current sequence visualization capability presents one attribute category, such as action sequences or location transition sequences. Looking for a way to visualize multi-dimension sequences is challenging. Furthermore, we will continue working on enhancing the behavior pattern representation. Besides of distribution, timing, association and sequence patterns, we plan to include more statistic and data mining techniques and constraints.

ACKNOWLEDGMENT

This research was funded by an ORF grant for the project ”Secure Intelligent Content Delivery System for Timely Delivery of Large Data Sets in a Regional/National Electronic Health Record”. Also, an early version of the proposed toolkit was developed by Mr. Duane Bender, Director of the MEDIC Lab at Mohawk College, Hamilton, Canada.

REFERENCES

[1] Phua, C., Lee, V., Smith, K., and Gayler, R. 2010. A comprehensive survey of data mining-based fraud detection research. arXiv preprint arXiv:1009.6119.

[2] Whiting, M. A., Haack, J., andVarley, C. 2008. Creating realistic, scenario-based synthetic data for test and evaluation of information analytics software.In Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization. ACM.

[3] Market-Basket Synthetic Data Generator, https://synthdatagen.codeplex.com/

[4] An Open-Source Data Mining Library, http://www.philippe-fournier-viger.com/spmf/

[5] Pei, Y., and Zaïane, O. 2006. A synthetic data generator for clustering and outlier analysis. Technical report, Department of Computing science, University of Alberta, edmonton, AB, Canada.

[6] Vijaylakshmi, S., Mohan, V., and Suresh Raja, S. 2010. Mining of users access behavior for frequent sequential pattern from web logs. International Journal of Database Management System (IJDM), 2.

[7] Yarmand, M. H., Sartipi, K., amd Down, D. G. 2013. Behavior-based access control for distributed healthcare systems. Journal of Computer Security, 21(1), 1-39.

[8] Xu, R., and Wunsch, D. 2005. Survey of clustering algorithms. Neural Networks, IEEE Transactions on, 16(3), 645-678.

[9] Saneifar, H., Bringay, S., Laurent, A., and Teisseire, M. 2008. S 2 MP: similarity measure for sequential patterns. In Proceedings of the 7th Australasian Data Mining Conference-Volume 87 (pp. 95-104). Australian Computer Society, Inc..

[10] Kum, H. C., Pei, J., Wang, W., and Duncan, D. 2003. ApproxMAP: Approximate Mining of Consensus Sequential Patterns. In SDM (pp. 311-315).

[11] The R project for statistical computing website, http://www.r-project.org/

[12] R-package arules: Mining Association Rules and Frequent Itemsets, http://cran.r-project.org/web/packages/arules/index.html

[13] R-package arulesSequences: Mining frequent sequences, http://cran.r-project.org/web/packages/arulesSequences/index.html

[14] Gabadinho, A., Ritschard, G., Mueller, N. S., and Studer, M. 2011. Analyzing and visualizing state sequences in R with TraMineR. Journal of Statistical Software, 40(4), 1-37.

[15] Agrawal, R., &Srikant, R. 1995. Mining sequential patterns. In Data Engineering, Proceedings of the Eleventh International Conference on(pp. 3-14). IEEE.

[16] Aggarwal, C. C., and Reddy, C. K. (Eds.). 2013. Data clustering: algorithms and applications. CRC Press.

[17] [14] Bergroth, L., Hakonen, H., and Raita, T. 2000. A survey of longest common subsequence algorithms. In String Processing and Information Retrieval, 2000. SPIRE 2000. Proceedings. Seventh International Symposium on (pp. 39-48). IEEE.

[18] Gabadinho, A., Ritschard, G., Studer, M., and Müller, N. S. 2011. Extracting and rendering representative sequences. In Knowledge Discovery, Knowledge Engineering and Knowledge Management (pp. 94-106). Springer Berlin Heidelberg.

[19] Dixon, W. J., and Massey, F. J. 1969. Introduction to statistical analysis (Vol. 344). New York: McGraw-Hill.

[20] Introducing JSON website, http://json.org/

[21] RFC 3881, Security Audit and Access Accountability Message XML Data Definition for Healthcare Applications, https://tools.ietf.org/html/rfc3881

[22] Yarazavi, A, and Kamran S. "Consultant-as-a-service: an interactive and context-driven approach to mobile decision support services." CASCON. 2013.