Template-based Time series generation with Loom

Lars Kegel, Martin Hahmann, Wolfgang Lehner
Technische Universität Dresden
01062 Dresden, Germany
{firstname.lastname}@tu-dresden.de

ABSTRACT

Time series analysis and forecasting are important techniques for decision-making in many domains. They are typically evaluated on given sets of time series that have a constant size and specified characteristics. Synthetic datasets are relevant because they are flexible in both size and characteristics. In this demo, we present our prototype Loom, that generates datasets with respect to the user's configuration of categorical information and time series characteristics. The prototype allows for comparison of different analysis techniques.

Categories and Subject Descriptors

I.6.7 [**Simulation and modeling**]: Simulation Support Systems; H.2.8 [**Database management**]: Database Applications—*Statistical databases*

Keywords

Time series analysis, Data generation

1. INTRODUCTION

Time series describe the dynamic behavior of a monitored object, parameter, or process over time and are one of the most popular and useful data types. They can be found in a multitude of application domains, e.g. as item sales in commerce, various sensor readings in manufacturing processes, or as demand and production in the energy domain. Obviously, this makes them a valuable source for diverse data analysis techniques, such as forecasting [8]. Especially in the domain of renewable energy production, where the fluctuating character of renewable energy sources makes accurate forecasts vital in order to match electricity production and demand. Further applications on time series data include querying, classification, efficient storage and much more. The ubiquity of this data type and the ongoing trend for data collection, storage and analysis have led to a substantial amount of research that is dedicated to the handling and processing of large sets of time series data. While all these research endeavors can differ greatly with respect to their individual goals and application scenarios they have one thing in common: they require large amounts of time series data in order to evaluate, verify, and optimize their findings. Although there are many stakeholders that have a substantial interest in using and exploiting time series, acquiring sophisticated data is not easy. Basically there are two sources: First, public open repositories or single datasets [10], that are tailored to specific applications and only offer a small selection. Second, ”real” data owned by companies/organizations that is sometimes made available to partners in the context of closed research projects but rarely to the general public. Moreover, obtaining real data can be tedious due to the time and cost that is necessary to collect them [11]. Based on our own experience we can state that some data is always available, that allows to conduct basic evaluations. This situation normally becomes problematic when scalability, versatility, and robustness have to be examined. These require a more versatile selection of data, containing datasets with varying size, time series length, trends, seasonality, or just a different blend of time series characteristics. In general, this is not available which often leads to researchers using workarounds to create more data, e.g. duplication to increase the number of time series or their length.

To cope with this problem, we demonstrate Loom which is a user-friendly and flexible approach to generate sets of time series for the evaluation of arbitrary analysis techniques or the benchmarking of time series management systems. Loom stands for the process of weaving different time series generators, datasets of arbitrary size. In addition, users can generate categorical information to structure the time series hierarchically. Thus, they form a data cube that can be explored by usual OLAP queries, such as roll up or drill down. Generated datasets can be directly exported to relational databases or flat file formats in order to easily utilize them in different applications. The usage of Loom is template-driven at its core. This means, given datasets can be analyzed in order to extract a template containing their defining characteristics. These templates are then used to create different variants of datasets that are still similar to the template. This approach eases the application of our tool as users do not have to specify a completely synthetic time series model. In addition, this mechanism offers a certain degree of anonymization for otherwise closed data.

In the remainder of this article, we present a general system overview in Section 2, before we describe our demonstra-

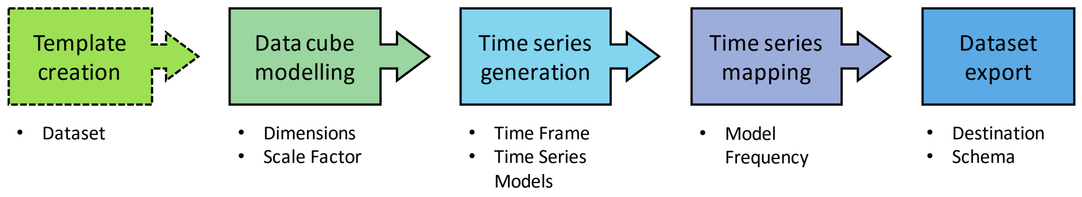


Figure 1: Workflow overview

tion in detail in Section 3. Previous work related to dataset generation and time series models is presented in Section 4 before the concluding remarks and pointers to future work are given in Section 5.

2. SYSTEM OVERVIEW

The main workflow of Loom is depicted in Figure 1 and shows all steps necessary to create a set of time series. In this section, we give an outline of the idea behind each step before we describe its implementation in Section 3.

Template creation. This optional step at the beginning of the workflow allows the user to upload and analyze given time series data in order to create templates that can be used during the latter steps of the data generation process. Currently, Loom employs three types of template creation: (1) If present, existing hierarchies of categorical information are extracted and stored. In order to anonymize the data, the original attributes are replaced with synthetic ids. (2) Given time series are extracted and taken as samples for time series generation. (3) The whole set of time series is analyzed to create a template that represents its characteristics and can be used to create multiple datasets that are similar to the original. While the first two types are straightforward, the third one is more complex. In order to create the described template, Loom uses an approach based on the hierarchical divisive analysis clustering (DIANA) [12]. With this method, the dataset is partitioned into groups of similar time series. From each of these clusters, the time series with the lowest average distance to the remaining members is selected as a prototype. By fitting an analytical time series model, e.g. ARIMA, to this time series, a generator that represents the characteristics of its underlying cluster is created. This generator can be used to create multiple variations of the original time series. To complete the template, the size of each group in relation to the whole dataset is stored. With the collected information it is possible to create multiple datasets that are different but still share the characteristic time series of the original and their distribution.

Data cube modelling. Usually, a set of time series does not only contain sequences of measured values, but also categorical information, e.g. geography, purpose, or color. Sets of these attributes are organized in hierarchies, called dimensions, while sets of these dimensions form the skeleton of a data cube. The goal of this step is to allow users the configuration and creation of such cubes. In short, data cubes can be formally described as follows [17]:

A data cube skeleton consists of a set of *dimensions*. A dimension is a lattice of *levels* $L = \{l_1, l_2, \ldots, l_m\}$. The constraint of this lattice states that the values of a level, called *category attributes* functionally determine the values of its parent level, e.g., $l' \to l''$. More formally, for each level l, l', l'' of the same dimension:

$$\begin{aligned} &l \to l && \text{(reflexivity)} \\ &l \to l' \wedge l' \to l \Rightarrow l = l' && \text{(antisymmetry)} \\ &l \to l' \wedge l' \to l'' \Rightarrow l \to l'' && \text{(transitivity)} \end{aligned}$$

For the sake of simplicity, we implemented totally ordered dimensions in Loom; this will be discussed in Section 5. A total order has the following additional condition:

$$l \to l' \vee l' \to l \quad \text{(totality)}$$

As an example, Figure 2 shows the totally ordered dimension *Geography* of Australia with the two levels *State* and *Region*. The category attributes of *Region* functionally determine the category attributes of *State*, e.g. Melbourne and Ballarat determine Victoria.

Three parameters are necessary to configure a data cube skeleton: the number of dimensions, the number of levels per dimension, and the outdegree of a category attribute. While the first two parameters are straightforward, the last one needs more explanation. The outdegree of a category attribute defines the number of subcategories within a category and thus describes the branching between the levels of a dimension. To illustrate this, we regard the example from Figure 2 where we observe an outdegree of 2. This means a category attribute on the State level, e.g. New South Wales, is related to two category attributes on the lower Region level, e.g. Sydney and Blue Mountains. Category attributes that have no further subcategories are called base category attributes and form the leaves of a dimension's hierarchy. Thus, a data cube skeleton is the cross-product of all base category attributes of all dimensions.

In the energy domain, categorical information is also beneficial because forecast models that are built for categories may lead to more accurate and robust forecast results. For

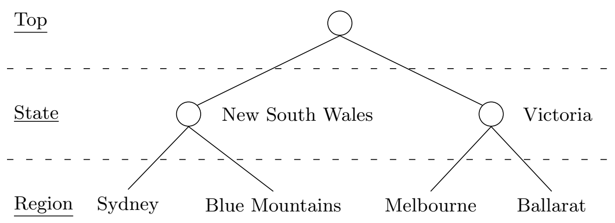


Figure 2: Dimension with levels State and Region

instance, the Irish Smart Metering Project [1] gathers time series of smart meters in over 5,000 Irish households and businesses. In a survey, the owners give additional information, such as Social class, House type and Age of the house. We identify 8 dimensions, each of whose consisting of one dimension level. Thus, forecast models can be created for individual time series or aggregated time series along one or more dimensions.

Time series generation. The data cube skeleton that was configured and created in the previous step must now be filled with facts, which in the case of our system are time series. In the following, we again give a quick formalization of time series and explain what is necessary to configure their creation. A time series is a sequence of successive observations x_t $(1 \leq t \leq T)$ recorded at specified time instances t. For this demonstration, we assume that observations are complete and equidistant, i.e. there exists an observation for every time instance and all time instances have the same distance. For configuration a time frame must be defined that consists of start and end time instance as well as the distance between time instances. In addition one or more measure columns must be defined, depending whether univariate or multivariate time series should be generated. To fill these measures with actual values, our system uses time series models and samples.

As an example, the user can create a synthetic model with a base b_t, season s_t and error component e_t:

$$x_t = b_t \cdot s_{(t \ mod \ L)} + e_t$$

The season length is given by L. The component s_t is a seasonal mask of length L. Thus, the weights repeat every L time instances. The error component e_t is normal distributed $\mathcal{N}(0, \sigma^2)$ and overlays the "perfect" model $x_t^\star = b_t \cdot s_{(t \ mod \ L)}$. The standard deviation σ depends on the user's accuracy expectation, expressed as *mean average percentage error (MAPE)*:

$$MAPE = \frac{1}{T}\sum_{t=1}^{T}|\frac{x_t^\star - x_t}{x_t^\star}| = \frac{1}{T}\sum_{t=1}^{T}|\frac{e_t}{b_t \cdot s_{(t \ mod \ L)}}|$$

The error distribution is calculated such that the user given $MAPE$ holds on average of the whole time series.

Datasets may also be created from template. During template creation, a set of ARIMA models is created that is based on user given data. Synthetic time series are generated by those ARIMA models, incorporating normally distributed errors or sampled errors from the template [4].

Time series mapping. The mapping links time series to the data cube, i.e. populates the skeleton with facts. As manual insertion of thousands of time series into a large data cube would not be feasible, Loom features an automatic mapping that randomly distributes the generated time series over the data cube skeleton. The user may set the model frequency by weighting each time series model. If templates are used, their distribution is used as default.

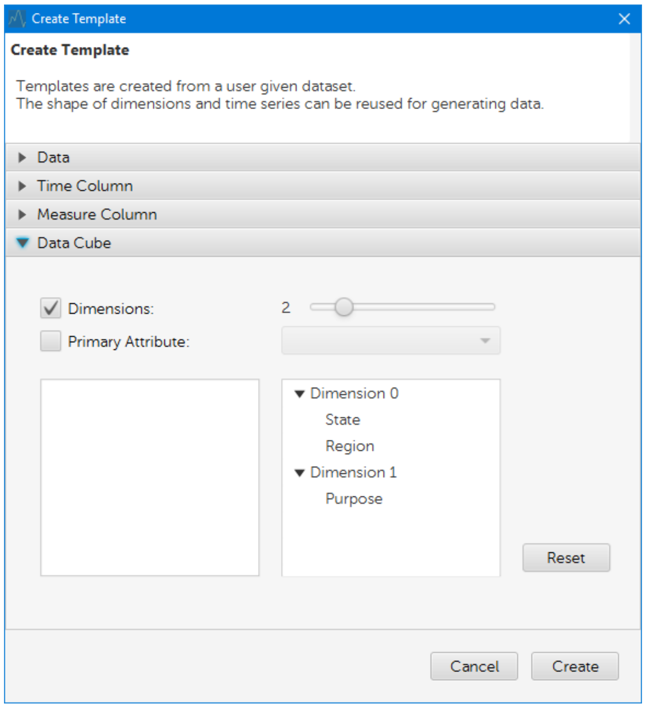
Dataset export. After the configuration and mapping steps are done, the actual data is created and can be exported in a suitable format for further use. Our application offers two different export destinations: either file or database. File export offers general formats like CSV and SQL script, allowing the use of our generated data in almost every application. In addition, we offer export as RData which is the data format *data.table* for the popular statistical workbench R [3]. The database export transforms the created data into fact tables that can be imported into any RDBMS. These tables have one time and at least one measure column, categorical information may be stored in fact tables as well as in dimensional tables. As we deal with a high amount of structured data, it is necessary to bring the dataset into an appropriate schema depending on the chosen export option.

3. DEMONSTRATION

This section demonstrates the usage of Loom. Starting the application, the user sets a workspace directory that is used for configuration files and generated datasets. Below, we describe the configuration of a template, a dataset and further generation steps. These steps correspond to the workflow shown in Figure 1.

3.1 Template configuration

The optional template configuration annotates a user given dataset with information that is further needed for the template creation. The user inputs a CSV file and annotates each column with the corresponding semantic (time, measure or category). Moreover, the time column needs information about the time type (integer, date, time) and the respective format. As shown in Figure 3, the user configures dimensions by Drag and Drop of the category names. Optionally, the user may indicate the primary attribute that is the lowest level category in every dimension. Once this configuration is finished, the template may be used for time series and/or data cube generation.


Create Template

Create Template

Templates are created from a user given dataset.
The shape of dimensions and time series can be reused for generating data.

- Data
- Time Column
- Measure Column
- Data Cube

[✓] Dimensions: 2
[ ] Primary Attribute:

- Dimension 0
  - State
  - Region
- Dimension 1
  - Purpose

Figure 3: Template configuration

3.2 Dataset configuration

After login, the user is greeted by an overview window that displays all datasets that he/she has already generated and those that are queued for generation, Figure 4. Clicking “Create new”, opens a wizard dialog that guides the user through the configuration. The first input by the user is a dataset name which is needed to reference and handle the configuration and its results.

3.2.1 Data cube configuration

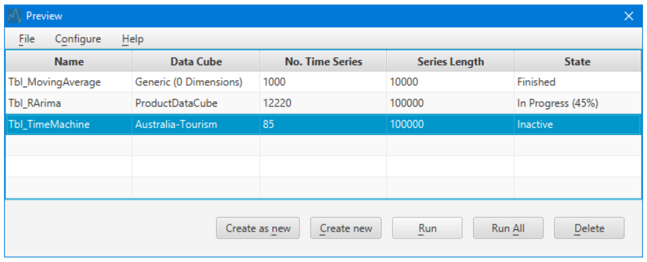
The next dialog, Figure 5, allows the configuration of the data cube. Loom offers two ways of configuration: template and synthetic.

- *Template configuration*: The user can select a template as basis for the data cube skeleton. Templates are derived from real world datasets or from existing data cubes and are ready-made configurations featuring default values for number of dimensions, number of levels and outdegrees. Thus, this type of configuration is more user friendly than the synthetic one. Users can still customize the configuration by making selective adjustments to the template. It is possible to create different variants, e.g. smaller, larger, highly branched, etc., of an existing data cube.

- *Synthetic configuration*: Alternatively, this type of configuration allows users the full manual customization of the data cube skeleton. Users specify the number of dimensions, the number of levels per dimension and the outdegree per level. These parameters can be provided fixed for each element or the whole cube. In addition, Loom offers a random parameter distribution, allowing a probabilistic setting with randomly structured dimensions.

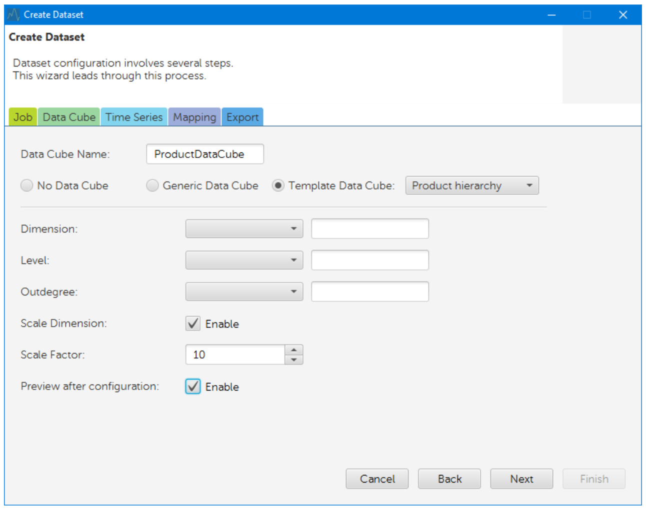
Both template and synthetic configurations, employ random distribution of outdegrees to a certain extent. This means the specific number of facts that can be accommodated by the data cube skeleton is not known during configuration. As the user must know the actual structure of the data cube in order to configure an appropriate number of time series, our system offers a preview of the data cube skeleton directly after configuration. This preview is depicted in Figure 6 and shows all category attributes ordered by their size. The user may restart the modeling or accept the generated result.

In many benchmarks such as TPC-H [2], it is common to set a *scale factor* SF of a database size, such as 10, 30, 100. Loom adopts this functionality by using a scale dimension that consists of exactly one level with $SF \geq 1$ category



Name	Data Cube	No. Time Series	Series Length	State
Tbl_MovingAverage	Generic (0 Dimensions)	1000	10000	Finished
Tbl_RArima	ProductDataCube	12220	100000	In Progress (45%)
Tbl_TimeMachine	Australia-Tourism	85	100000	Inactive

Figure 4: Dataset overview



Create Dataset

Dataset configuration involves several steps.
This wizard leads through this process.

Data Cube Name: ProductDataCube

No Data Cube [✓ Template Data Cube: Product hierarchy] Generic Data Cube

Dimension:

Level:

Outdegree:

Scale Dimension: ✓ Enable

Scale Factor: 10

Preview after configuration: ✓ Enable

Figure 5: Data cube configuration

attributes and can be used to adapt the size of the data cube as desired. Thus, the resulting data cube is inflated by this factor.

While the parameters and configuration types described in this section provide users with a versatile and comfortable way of modeling the categorical information of a data cube, its configuration is not mandatory. If a user wishes for a plain set of unlabeled data, only a primary attribute is generated to identify the created time series.

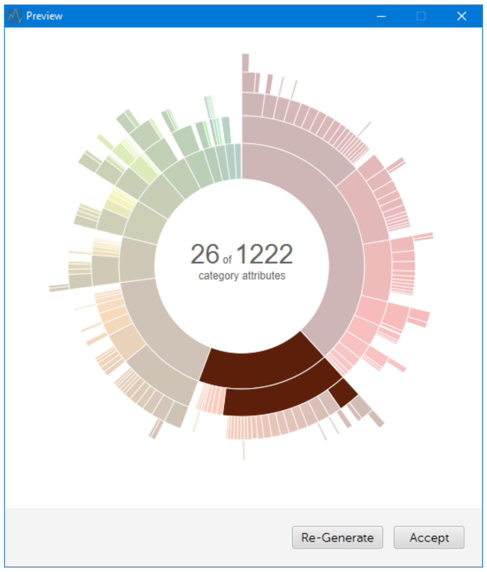


Figure 6: Preview of generated dimension

3.2.2 Time series configuration

Time series configuration is split into three separate dialogs: time attribute, measure attributes, and time series models. The time attribute needs parameters such as the time type and the timeframe, i.e. start time, granularity, end time.

In the measure dialog, the user sets the number of measure columns of the dataset. For each measure attribute, the user sets a data type. Usually, a measure is of double-precision floating-point format but in order to decrease dataset size in a database, the user can also set single-precision floating-point or integer format.

Most importantly, time series models have to be added to the configuration. For this, Loom offers four options:

- *Sampled from Template*: All time series data is generated "as is" by taking values from given time series from existing datasets uploaded by users. According to the timeframe configuration data is extracted from the original time series and added to the generated one. Mismatches with the timeframe or data cube size are resolved using duplication, cutting, granularity conversion etc.

- *Recombined from Template*: Time series are created via decomposition and recombination. Classical decomposition strategies like *decompose* [13] and *stl* [6] are used to extract the defining components trend, seasonality, and noise from an existing set of time series. By recombining these components, new time series are created and can be used to add volume and variety to a dataset.

- *Modeled from Template*: This option uses the third type of templates we described in Section 2. Users can load the time series generators and their distributions of an existing dataset. Customization is possible by changing the number of time series an individual generator creates or by removing/adding certain generators.

- *Synthetic time series*: The user configures a time series model from scratch, without relying on any given measures. Time series properties are defined freely and are synthetically generated, e.g. with a linear rising trend, a regular gauss-shaped seasonal component, and a normally distributed error series.

3.2.3 Export configuration

In the last dialog, the user sets the export configuration. The export destination has different options. CSV, SQL and RData create flat files within the workspace. Alternatively, time series are exported to a database via JDBC driver. Thus, the user sets up a database connection with a database location and login credentials. Finally, the user sets the schema. Loom supports different schemas: (1) Basic unnormalized export in a *Universal schema*, which creates high redundancy as it stores the categorical information with each value of a time series. (2) The partly resp. fully normalized *Star* and *Snowflake schema*, which allows more compact exports and are common in database design. (3) The *Parent-Child schema*, which stores each functional dependency as a pair of parent attribute and child attribute in the respective dimension table.

Particular attention has to be paid to the export when dimensions are unbalanced, i.e. their base categories are not on the same level. Figure 7 shows an example for such a dimension, where base category attributes are either on the region level (Darwin, Alice Springs) or on the state level (Australia Capital Territory). While both the universal schema and the parent-child schema support unbalanced dimensions, the star and snowflake schema only support unbalanced dimensions when referential integrity can be guaranteed. To achieve this, the user may add a primary attribute column and a minimum outdegree. If the user did not set a data cube skeleton, then a primary attribute is automatically generated.

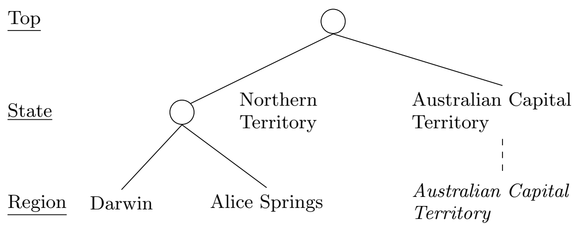


Figure 7: Unbalanced dimension with two levels

3.3 Table generation

Closing the configuration window brings the user back to the initial overview where a new entry has been added, see Figure 4. Now, the generation process can be invoked by clicking „Start Selected“. The state switches to „In Progress“ and indicates the current amount of data that has been generated.

4. RELATED WORK

Workload generation has been studied in many papers each of which focus either on the generation process or time series modelling. To our knowledge, Loom is the first application that integrated both techniques and allows for flexible data cube and time series characteristics. In the following, we present selected sources that relate to our prototype.

The IDAS dataset generator [11] offers data generation based on statistical distributions. Attributes form dependency graphs that are not necessarily lattices. The goal is the creation of a synthetic dataset for testing data mining algorithms. The workflow is similar to Loom since the user creates a dataset by specifying the number of tables, setting the attributes and initiating the data generation. Moreover, the authors experienced similar shortcomings with real data, such as privacy issues, a lack of training data or unsatisfying categorical information. Still, measures do not depend on time, thus time series generation is not supported.

Schaffner and Januschowski [14] focus on benchmarking of databases under varying request rates. Request rates can be seen as time series of aggregate tenant traces. Since there is not enough real data available, they provide two methodologies for generating synthetic tenant traces. (1) The modelling approach fits a function as a model for a given aggregate tenant trace. The function's shape has been determined empirically. By adding an error term, they create diversity among the synthetic tenant traces. (2) Another way is the decomposition of time series by bootstrapping. Thus, a given trace is split into windows that are randomly shifted and result in synthetic traces. Both approaches are similar to Loom's template creation in that synthetic time series are either modelled or recombined from template.

The F-TPC-H benchmark [7] is a modified TPC-H benchmark for time series generation. This work reuses the given TPC-H schema in that customers submit orders of products for a certain quantity. While this quantity does not depend on time in TPC-H, the modified F-TPC-H adds dependency for representing trend and seasonal effects via ARIMA models. Thus, this work represents a subset of Loom because it consists of a given schema and allows for synthetic time series in the sales domain. Loom also supports schema flexibility and allows for composing different time series generators.

A specific use case for managing energy data is given by [16]. This work proposes a unified Data Warehouse schema for storing workloads, given as information about actors like producers and consumers, offers, and time series about past measures. Their time series schema involves measures from different types such as energy, power and price. Categorical information is necessary in order to store special annotations such as aggregation level of time series or additional information for each time series type. A time series is represented by several tables: (1) a time series table stores the primary attribute that identifies the time series and that links to each category, (2) another dimensional table stores time frames with an identifier and the resp. time frame information, (3) the fact table itself consists of the primary attribute, a measure column and a foreign key to the time frame. Thus, this schema is not a traditional star of snowflake schema and cannot directly be covered by Loom. Moreover, Loom keeps time and measure together as fact. This may increase redundancy but we opt for this solution for several reasons: (1) there is no need for an additional description of time frames, (2) a time frame is encoded either as an integer or a short string, thus the space for storage is still affordable, (3) there is no join operation needed in order to retrieve a time series. After all, time series from the energy domain may be generated by models integrated in Loom.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced Loom as a tool for generating large sets of synthetical time series data. Our prototype utilizes different time series generators to create multiple time series that share certain characteristics. In addition, our prototype allows the creation of dimensional categorical information for the description of time series. Besides the full manual definition of a dataset, Loom features a template driven approach that analyses given datasets and allows the creation of synthetic variants of this template data.

Currently, our approach only generates complete time series with equidistant time stamps. This is done, as forecast methods like Exponential smoothing [9] and ARIMA [5] rely on these properties, except few models like [15]. Part of our future work will be the integration of functions for generating incomplete time series with configurable gap patterns.

Regarding data cube modelling, we assume that a dimension is a totally ordered set of levels, which is the case in most real-world datasets. However, there are exceptions, such as the modelling of a time dimension with levels: day, week and month. There, a day functionally determines a week and a month, but a week does not determine the month. Such lattices are not supported by our prototype.

Further future work will be focused on time series mapping to the data cube. Right now, we use a very simple approach for this and randomly distribute our time series over the data cube skeleton. This approach will be replaced with a more sophisticated method that allows the configuration of the distribution. For example, time series from a certain generator only occur in a specified subset of the data cube skeleton.

6. REFERENCES

[1] CER Smart Metering Project. `http://www.ucd.ie/issda/data`, 2010.

[2] TPC Benchmark H. `http://www.tpc.org/TPC_Documents_Current_Versions/pdf/tpch2.17.1.pdf`, 2014.

[3] R data.table package. `https://cran.r-project.org/web/packages/data.table/index.html`, 2015.

[4] R forecast package. `https://cran.r-project.org/web/packages/forecast/index.html`, 2015.

[5] G. E. P. Box and G. M. Jenkins. *Time series analysis forecasting and control*. Holden-Day, San Francisco, 1970.

[6] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6:3–73, 1990.

[7] U. Fischer. Forecasting in database systems, 2014.

[8] U. Fischer, F. Rosenthal, and W. Lehner. F2DB: The Flash-Forward Database System. In *ICDE*, pages 1245–1248, 2012.

[9] C. C. Holt. Forecasting trends and seasonal by exponentially weighted averages. *Office of Naval Research Memorandum*, 52, 1957. Reprinted in: International Journal of Forecasting, 20(1):5-10, 2004.

[10] R. J. Hyndman. Time series data library. `http://data.is/TSDLdemo`. Accessed on 9-24-15.

[11] D. R. Jeske et al. Generation of synthetic data sets for evaluating the accuracy of knowledge discovery systems. In *Proc. of KDD*, pages 756–762, 2005.

[12] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data*. Wiley, 1990.

[13] M. Kendall and A. Stuart. *The Advanced Theory of Statistics*, volume 3. Griffin, 1983.

[14] J. Schaffner and T. Januschowski. Realistic tenant traces for enterprise DBaaS. In *Workshops Proc. of ICDE*, pages 29–35, 2013.

[15] R. H. Shumway and D. S. Stoffer. *Time Series Analysis and Its Applications*. Springer, 2011.

[16] L. Siksnys, C. Thomsen, and T. B. Pedersen. MIRABEL DW: managing complex energy data in a smart grid. In *Proc. of DaWaK*, pages 443–457, 2012.

[17] P. Vassiliadis. Modeling multidimensional databases, cubes and cube operations. In *Proc. of SSDBM*, pages 53–62, 1998.