Final Report - On Repairing Timestamps for Regular Interval Time Series

Varun Totakura - VT22E

Sri Vennala Kandibedala - SK22BV

Abstract:

Time series data are frequently quite full, but occasionally they may be incomplete due to challenges with storing the data or problems at the source. In order to improve the effectiveness of the machine learning algorithms or the application using the data, we try to formalize a method that can recognize missing data points in time series data and can repair them. Time series data quality measures can be assessed using the repair results.

Source Code and Environment:

- GitHub: https://github.com/varuntotakura/DatabaseSystemsProject/tree/main

We have used C++ Programming language for implementing the algorithms. The datasets are CSV files, which we have taken as input and parsed by the program we have developed. The parsed data points are then sent to the corresponding algorithms to generate the results. In the end, the produced results are sent into the metrics functions to calculate the RMSE, Cost, and Accuracy values.

To build the code, we have used the GNU GCC Compiler of Version 14 in Windows 11 environment. In addition, we have also used CMake to build and create the executable file of the program.

Introduction:

The data must be complete in order to increase the effectiveness of using it. By contrasting the algorithms presented in this research study with other algorithms, we would like to draw attention to them. To locate the data points that need to be fixed, they've developed methods like the "Match Searching Algorithm" and the "Traceback Algorithm." They have utilized the "Exact & Approximate Regular Interval Repair RIR-Exact Algorithm" to repair the data after tracing the data points that require repair.

Problem:

Nowadays a lot of time-series data is being stored which were being collected from various data sources like sensors in automobiles, IoT devices in houses, and sensors placed in industries to monitor the workload and effectiveness. Even though there is a high chance of recording the whole time-series data continuously from various sources, there is a slight chance that sometimes the sensors can be in trouble for some time, and some of the data can be lost or no data is generated for a certain amount of time. This issue can bring a disaster to its application. For example, if sensors in the Self-Driving car are lost for a moment, the vehicle can cause accidents as it will lose the decision-making capacity due to the lack of data. So, it is very important to deal with missing time-series data. This project will implement a procedure to detect and repair the data quality issues in timestamp data. The project will also solve a few

sub-problems involving match searching that aims in finding regular interval time series that minimize the repair cost. It also solves the length determination problem that finds the best length to minimize the repair cost. Further solves start determination to find the best start which minimizes repair cost. Lastly, solves interval determination to find the best interval where repair cost is minimized. The pictorial representation of all these problems is presented in the figure-1.

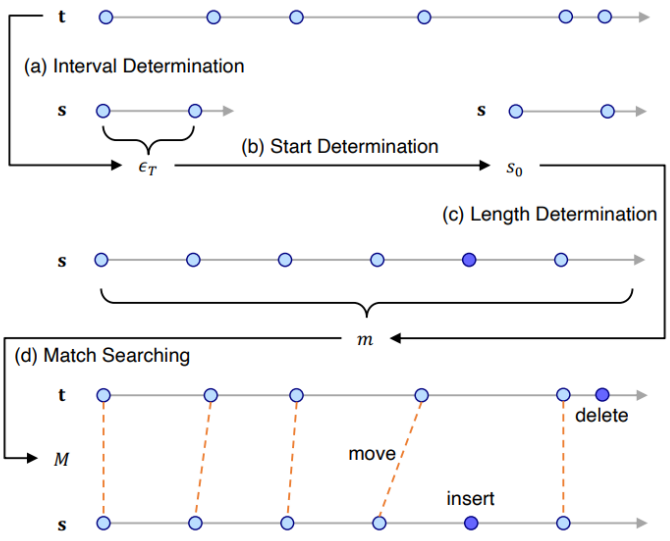

Figure-1: Sub-problems and pipeline for time series

Contributions:

- To reduce the cost of the move, insert, and delete operations, we formalize the timestamp fixing problem for time series with regular intervals. In regular interval time series, determining the interval, start, and length is regarded as a sub-problem.
- In order to locate the irregular and regular interval time series matching with the lowest distance cost and to calculate the length of the regular interval time series, we develop an exact method based on dynamic programming. Interestingly, we establish the lower bounds of the repair cost in terms of the interval and the start, leading to the creation of sophisticated pruning techniques.
- Bi-directional dynamic programming is used to create an approximation technique that considerably reduces time expenditure.
- We thoroughly compare different existing techniques in both repaired timestamps and downstream tasks, such as frequency-domain analysis and data compression.

Related Work:

CTTC(Cleaning Timestamps with Temporal constraints) introduced by Song et Al [1], is a novel problem of repairing inconsistent timestamps that do not conform to the required temporal constraints. It is possible to use temporal restrictions to assess how accurate timestamps are. They have developed a solution that is influenced by these restrictions, and the key is to identify a limited number of candidates

for timestamp restoration. They tried to create precise, heuristic, and randomized timestamp correction techniques after they had identified them. However, it does not account for the addition or subtraction of points, which causes a significant shift when certain points are missed or repeated.

The Heuristic Approach by Bohannon et al. [2] notes that database reconciliation is a crucial application of data integration. Because of its advantages, they have introduced a heuristic technique based on class equivalence in their work. The operation can be ended with the aid of class equivalence, which also helps to distinguish the relationship between characteristics and the values assigned during the repair. Additionally, they have devised a greedy algorithm to fix the time-stamp data, drawing on their heuristic approach and a new low-cost framework. However, due to the clear distinction between integrity constraints and temporal constraints, such direct procedures that can change data directly cannot be used.

SCREEN (Stream Data Cleaning under Speed Constraints) by Song et Al., [3] was formulated to repair problems under speed constraints by considering the entire sequence as a whole. Even the algorithm has been changed to accommodate the arrival of out-of-order data. It takes into account the restrictions on the rate of data updates. The setting for this issue is different, with the speed restriction set to the specified time interval in order to fix the inconsistent interval timestamps in each window.

Holistic Approach to Clean Data Points: In order to fix incorrect timestamps, Chu et al. [4] presented a holistic technique that expresses regular time periods as denial restrictions. They've included a compilation technique to put denial constraints into the currently running instance and capture the interplay between constraints as overlaps of violations on the data instance. With regard to a single, unified objective function, their approach corrects all violations simultaneously. Holistic repair, however, does not address missing and duplicated points, much like CTTC.

Methodology:

The Exact methodology begins with estimating the best values for match searching, length determination, Start determination, and Interval Determination. In all the above-mentioned estimates the major goal is to find the best solution that minimizes the time cost. Further, in approximate match searching, we propose a median approximation that directly regards the median timestamp of the original time series, inspired by the median's robustness. Here, we modify the dynamic programming algorithm proposed to a bi-directional median approximation program to measure the correctness of dynamic programming in both directions. Though we proposed efficient pruning by various bounds in exact interval determination it is still very costly to consider a huge number of possible intervals to find the optimal interval. Therefore, we propose to compute an approximation interval by using the medians of the intervals. At this step, for an approximate algorithm to prove the error bound, we prove triangle inequality over the proposed repair cost that restricts the difference between approximate and exact costs. Merging the repaired time series discussions we derive an error bound for the approximate algorithm.

1) Match Search Algorithm – Considering the match of a set of point pairs from the original time series t to the target time series s and repair cost for the move, delete, or insert cost we devise the Match Search Algorithm. It is mainly used to find the time interval of the given original data which has a minimum cost, even when the operations are performed on it.

2) Traceback Algorithm – To find the data points with proper start and time intervals, after seeing the minimum cost of the data points using the Match Search Algorithm

3) Exact Regular Interval Repair RIR-Exact – To repair the found minimal cost-effective time-stamp datapoints in the given time-series data

4) Approximate Regular Interval Repair RIR-Appr – Is also like RIR-Exact but it is motivated by the robustness of the median for noisy time-series data

Experiments and Evaluation:

We use the function timestamprepair to implement the effective approximation of timestamp repair. The findings of the timestamp correction could also be used to assess the timeliness, completeness, and consistency of the time series data. The data quality issues like delayed, missing or repeated data points lead to three-time series data quality measures that include timeliness, completeness, and consistency in the time dimension. We use the repair results and match them to profile data quality measures for the time series databases as depicted in Figure 2 below.

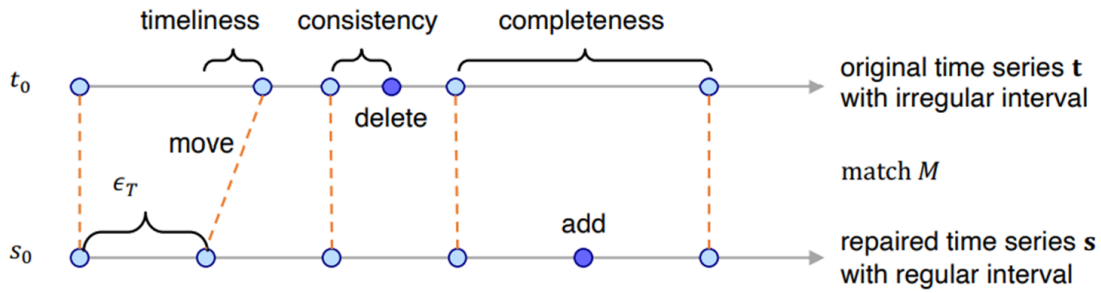

Figure 2: Data quality measures

Datasets: Features of the datasets. Quality issues are the data quality issues existing in the original datasets. Truth denotes the ground truth factors of the dataset that we have.

Dataset	Data quality issues	Truth	#Points
Engine	delayed	E_T, s_0, m	43,954
Turbine	Delayed, missing, redundant	E_T, s_0, m	28,000
Vehicle	Delayed, missing, redundant	-	111,790

Energy	-	E_T, s_0, m	19,735
PM	-	E_T, s_0, m	43,824
Air Quality	-	E_T, s_0, m	9,357

Table 1: Details about the datasets used

We have used 3 different datasets, to compute the results and calculate the metrics to compare the performance of the algorithm as there were only 3 complete datasets which are available. And the other three were not available. For the 3 datasets, we have used 3 different types of metrics, they are RMSE, Time Cost of the resulting data, and Accuracy.

RMSE(Root Mean Squared Error): A quadratic scoring rule called RMSE also calculates the average error magnitude. It is the average of the squared discrepancies between the prediction and the observed data. In our experiment, we consider both the RMSE loss between the time series and the RMSE between the factors. The RMSE between the factors evaluates the regular interval time series through the three factors i.e., interval, start, and length. We normalize the above three factors as they lie in different scales and finally compute the average according to their weights.

Time Cost: The metrics determine how efficient the algorithm is based on its execution time. The cost could also differ depending on the datasets. Higher the number of datasets, high the computation time.

Accuracy: The metric accuracy defines how well the model is performing compared to benchmark values or competitive models. However, these metrics could vary their values depending on various factors like the algorithm implemented, the datasets used, etc.

After calculating all the results using the mentioned metrics, we have compared our results with another Time-Stamps Repairing algorithm, which is SCREEN. After comparing the results it is very clear that our algorithms have outperformed the SCREEN algorithm on all datasets. In the table-2, you will be able to see the results which we got from the algorithms with the corresponding datasets. We have mentioned the best results we got while we tested with various datasets.

Dataset	RMSE - Exact	RMSE - Approx	Accuracy - Exact	Accuracy - Approx	SCREEN RMSE - Exact	SCREEN RMSE - Approx
Energy	0.7	2	99.97	99.94	4.27	31.5
PM	2	3.2	76.84	60.78	10.31	122.0

Air Quality	4	3	52.84	61.94	43.9	26.7

Table 2: Performance Comparision between RIR and SCREEN Algorithms

From the table-2, it is very clear that our algorithms have outperformed the SCREEN by nearly 4 times. In the paper, the authors have compared their algorithm with some other algorithms as well, they were able to get a huge difference in the results. Overall, in all the cases the RIR-Exact and RIR-Approx have outperformed all other existing time-series data fixing algorithms.

Project Applications:

Frequency-domain Analysis: While the values of delayed and deleted points are processed naturally when timestamp correction is applied to FFT, we must impute the corresponding values for the missing points, for example, using linear interpolation[5]. When the missing values are also imputed, we are able to compare the time series repaired in terms of FFT results. In other words, we measure the frequency domain distance between the corrected time series and the original data. Better performance results from lower RMSE. As a baseline, NUFFT [6] with integrated interpolation is also mentioned. Here it proves that RIR-Exact outperforms analogous other results. Additionally, RIR-Appr clearly outperforms other baseline approaches in RMSE, confirming the effectiveness of our approximation correction once more.

Data Compression: Regular interval time series also has the advantage of being compressed, especially in time series databases that use second-order differences[8]. In this experiment, we store original and corrected time series data from 0.6 to 3 million data points in the time-series database. We enter information into the database for each size and keep track of the increases in storage space. The fact that the restored time series use up less database space is not surprising. We also provide information on how much space each option saves.

Handling Missing Data Points: While this proposal fixes data point timestamps, one may still use the current methods[7] to interpolate or impute the values of the added points. In this way, our approach is an addition to any interpolation or imputation techniques used to handle missing data points. We combine the existing data imputation techniques with our precise and appropriate timestamp repair algorithms, and then we calculate the value imputation error (RMSE) of the data points. In most cases, RIR-Exact outperforms RIR-Appr in terms of imputation/interpolation performance thanks to more precise timestamp fixes. In all of the datasets, RIR-Exact with Interpolation exhibits the best value imputation performance; as a result, it is advised for accuracy in practical applications.

Conclusion and Future Work:

We suggest moving, inserting, and deleting data points in order to fix dirty timestamps in a time series for predictable time periods. The start, length, and interval of the time series must all be determined for the repairing problem if they are not known beforehand, which makes it difficult. We look into the repaired lower boundaries for effective pruning. Additionally, a bi-directional dynamic programming-based approximation is developed for a more effective repair. Extensive tests show how advantageous our suggestions are for downstream applications like frequency-domain analysis and data compression, as well as for repair accuracy.

The RIR-Exact algorithm and RIR-Approx algorithms have performed very well when compared with other existing time-series data fixing algorithms. There is one more thing that we can observe from the table-2, the same algorithm has shown different results with different datasets. This is because even though the algorithm has been designed to work with greater accuracy, it is not performing very well when the data is highly noisy or ambiguous. So, this problem can be taken as one of the issues or drawbacks of these algorithms, and by fine-tuning the existing RIR algorithms, a newer version of the algorithms can be designed which can perform well on noisy data as well.

Consider employing a machine learning model to forecast the subsequent timestamp using the previous ones in order to correct timestamps for time series with irregular intervals. However, the issue is more intricate. It is challenging to select how far the forecast deviates from the observation as a timestamp inaccuracy in this study rather than a regular time interval. Another way to make the project better is to further integrate the data values with the issue; for example, try to include data characteristics in the repair prices rather than only using them to make match decisions. As data properties, which vary among datasets, are more difficult to take into account than timestamps, we, therefore, leave this intriguing but difficult subject for further research.

Contribution - Teamwork:

Varun Totakura has worked majorly on developing the C++ code on implementing the mentioned algorithms RIR-Exact and RIR-Approx. He also has worked on reporting the experimental results and comparing the results with other algorithms. SriVennala Kandibedala has worked on fixing the bugs in the developed code and has majorly contributed to writing the project proposal, survey, status report, and final report. We both have collaboratively worked together on all the other miscellaneous works.

References:

[1]. S. Song, R. Huang, Y. Cao, and J. Wang. Cleaning timestamps with temporal constraints. VLDB J., 30(3):425–446, 2021.

[2]. P. Bohannon, M. Flaster, W. Fan, and R. Rastogi. A cost-based model and effective heuristic for repairing constraints by value modification. In SIGMOD Conference, pages 143–154. ACM, 2005.

[3]. S. Song, A. Zhang, J. Wang, and P. S. Yu. SCREEN: stream data cleaning under speed constraints. In T. K. Sellis, S. B. Davidson, and Z. G. Ives, editors, Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015, pages 827–841. ACM, 2015

[4]. X. Chu, I. F. Ilyas, and P. Papotti. Holistic data cleaning: Putting violations into context. In ICDE, pages 458–469. IEEE Computer Society, 2013

[5]. M. Lepot, J.-B. Aubin, and F. H. Clemens. Interpolation in time series: An introductive overview of existing methods, their performance criteria, and uncertainty assessment. Water, 9(10):796, 2017.

[6]. J. A. Fessler and B. P. Sutton. Nonuniform fast Fourier transforms using min-max interpolation. IEEE Trans. Signal Process., 51(2):560–574, 2003.

[7]. R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. J. Mach. Learn. Res., 11:2287–2322, 2010.

[8]. C. Wang, X. Huang, J. Qiao, T. Jiang, L. Rui, J. Zhang, R. Kang, J. Feinauer, K. Mcgrail, P. Wang, D. Luo, J. Yuan, J. Wang, and J. Sun. Apache iotdb: Timeseries database for internet of things. Proc. VLDB Endow., 13(12):2901–2904, 2020.