A review of the paper "M4: A Visualization-Oriented Time Series Data Aggregation"

Md Johirul Islam
mislam@iastate.edu

Lakshay Ahuja
lakshay@iastate.edu

I. Introduction

Visualization of large scale time series data is a crucial need of modern exploratory bigdata analysis [1]. But the huge size of the data is a barrier to visualization [2], [3], [4]. To address this challenge of bigdata different data reduction and sampling strategies are used to overcome the barrier [5], [6]. But for preserving the semantics of trend line of time series data these sampling strategies show huge limitations [7].

In this review paper we present a review of the paper [7] which address this issue of preserving the semantic of time series data and present some related works in the line. The paper appeared in the **Proceedings of the VLDB Endowment, 2014**.

The authors present M4, an aggregation based time series data reduction strategy that guarantees error free visualization of time series data as line chart as well as higher rate of data reduction. The approach is generic to any visualization system as long as the visualization systems uses RDBMS as data source.

II. Controbutions of the paper

The authors of the paper rewrite visualization queries Q using data reduction operator M_R such that the visualization of the original data from query Q and the visualization from the query $Q_R = M_R(Q)$ are similar and error free. As

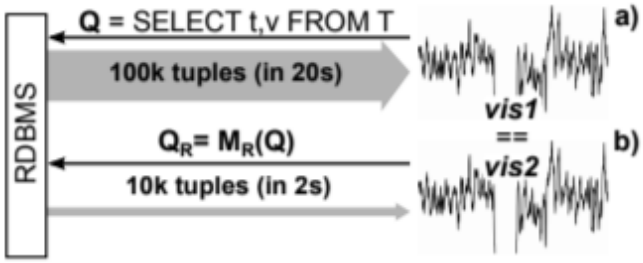


Figure 1. Time series visualization: a) based on original data; b) Using data reduction operator;

shown in Figure 1 Q_R produced the same visualization as Q with almost 10 times less tuples and 10 times reduced time. The main contributions of the paper are following:

- Proposed a visualization driven query rewriting technique relying on relational operators and parameterized with width and height of the desired visualization
- Focusing on the detailed semantics of the line charts, they propose a visualization driven aggregation strategy that only select necessary points needed for visualization. For visualization, in every time interval which corresponds to a pixel column in the visualization they select four tuples. The starting tuple, ending tuple, max tuple and the min tuple.

A. Query Rewriting

Most queries for time series visualization are of the form **SELECT time, value FROM SERIES WHERE** $time > t_1$ **AND** $time < t_2$. In addition to the query the visualization parameters like width and height are also passed for query rewriting. The rewritten query Q_R contains the following subqueries:

1) Original Query Q
2) A cardinal query Q_C on Q
3) a cardinality check (conditional execution)
4) to either use the result of Q directly or to execute an additional reduction Q_D on Q.

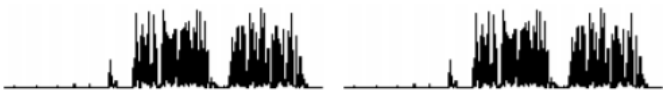
$M4$ system composes all those subqueries into single SQL query to avoid bandwidth consumption.

B. M4 Aggregation

M4 is a value preserving aggregation strategy for time series data. It divides the entire time series dataset into w

a) value-preserving M4 aggregation query

```
SELECT t,v FROM Q JOIN
(SELECT round($w*(t-$t1)/($t2-$t1)) as k, --define key
        min(v) as v_min, max(v) as v_max, --get min,max
        min(t) as t_min, max(t) as t_max  --get 1st,last
        FROM Q GROUP BY k) as QA          --group by k
ON k = round($w*(t-$t1)/($t2-$t1))        --join on k
   AND (v = v_min OR v = v_max OR         --&(min|max|
        t = t_min OR t = t_max)           --  1st|last)
```



b) resulting image == expected image

Figure 2. M4 query and visualization

equal groups and thus each pixel column in the visualization takes only one group. For each group M4 select the aggregates $min(v)$, $max(v)$, $min(t)$,$max(t)$ and that is why it is

called M4 aggregation and then it joins the aggregated data to the time series and add missing timestamps t_{bottom},t_{top} and missing values v_{first}, v_{last}. In Figure 2 an example M4 query and the corresponding visualization is shown.

Complexity of M4: The grouping and computation of aggregated values can be done in $O(n)$ time where n is the number of tuples in the original query Q. Then the subsequent joining of the $4.w$ aggregated tuples with Q requires $O(n + 4.w)$ using hash join.

C. M4 Upper Bound

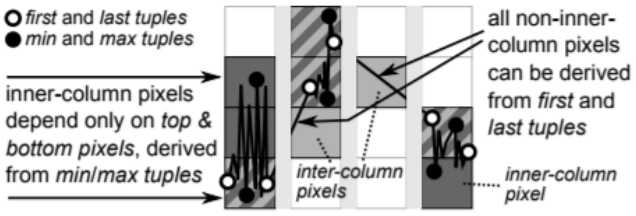


Figure 3. Illustration of Theorem 1

The question is common to be asked, whether selecting only four extreme tuples from each group provide error free visualization. The authors of the paper provide some proofs on the upper bound on the number of tuples required for error free visualization. The authors prove two theorems to show that the four extreme tuples are sufficient enough for error free visualization of time series data.

Theorem 1: Any two-color line visualization of an arbitrary time series T is equal to the two-color line visualization of a time series T' that contains at least the four extrema of all groups of the width-based grouping of T, i.e., $vis_{wh}(G_{M4}(T)) = vis_{wh}(T)$.
The illustration of theorem 1 is given Figure 3. The authors provide a detailed proof in the paper.

Theorem 2: There exists an error-free two-color line visualization of an arbitrary time series T, based on a subset T' of T, with $|T'| \leq 4w$.

III. Evaluation

The authors use three different real life time series data and evaluate their results with some existing state of the art data reduction approaches. The authors use the following datasets:

1) the price of a single share on the Frankfurt stock exchange over 6 weeks (700k tuples)
2) 71 minutes from a speed sensor of a soccer ball [8](ball number 8, 7M rows)
3) one week of sensor data from an electrical power sensor of a semiconductor manufacturing machine [9](sensor MF03, 55M rows)

The approaches compared are following: 1) a baseline query that selects all tuples to be visualized, 2) a PAA-query that computes up to 4w average tuples, 3) a two-dimensional rounding query that selects up to w.h rounded tuples, 4) a strati- fied random sampling query that selects 4w random tuples, 5) a systematic sampling query that selects 4w first tuples, 6) a MinMax query that selects the two min and max tuples from 2 w groups, and finally 7) M4 query selecting all four extrema from w groups.

A. Query performance

The query performance of different approaches in terms of execution time of the queries is shown in Figure 4. It shows that aggregation based approaches perform better compared to baseline approaches. Figure 5 shows exemplary results of performance for varying row counts on soccer data. The aggregation based queries perform much than baseline queries as the size of rows increase.

B. Visualization quality and Data Efficiency

Authors in [10] have shown that for visualization quality MSE (Mean Square Error) doesn't perform well and they proposed $SSIM$ (Structural Similarity Index) for the measurement of image quality. The SSIM yields a similarity value between 1 and 1. The authors use $DSSIM$, the normalized distance between two visualizations for measuring the visualization quality. The formula is given below:

$$DSSIM(V_1, V_2) = \frac{1 - SSIM}{2} \tag{1}$$

In Figure 6, the authors plotted the measured, resulting visualization quality (DSSIM) over the resulting number of tuples of each different groupings $n_h = 1$ to $n_h = 2.5w$ of an applied data reduction technique. The lower the number of tuples and the higher the DSSIM, the more data efficient is the corresponding technique for the purpose of line visualization. The Figures 6aI and 6bI depict these measures for binary line visualizations and the Figures 6aII and 6bII for anti-aliased line visualizations. On average M4 provides a visualization quality of $DSSIM > 0.9$.

C. Evaluation of pixel errors

The authors also show an evaluation of the pixel errors incurred due to data reduction. In a visualization of 100*20 pixels, MinMax results in 30 false pixels, RDP in 39 false pixels, and PAA in over 100 false pixels. M4 stays error-free. Which shows that M4 was able to produce error free visualization even after reduction of data.

IV. Prior Works

In this section, various prior works in the field of visualization systems has been discussed.

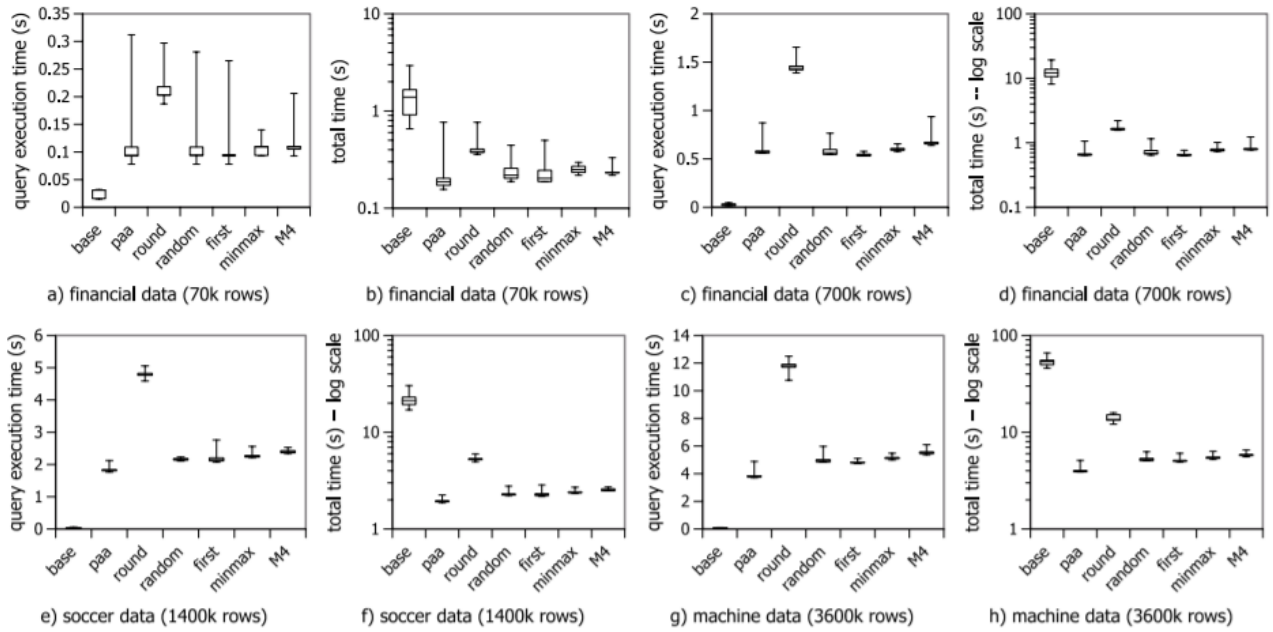


Figure 4. Query performance: (a,b,c,d) financial data, (e,f) soccer data, (g,h) machine data

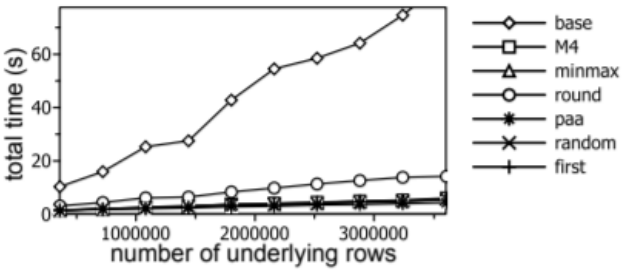


Figure 5. Performance of different queries with varying row counts

A. Visualization Systems

Current Visualization systems are categorized into three categories 1) The ones which do not used data reduction 2) The ones that compute and send images instead to data visualization 3) The ones which rely upon data reduction outside of the database.

These three approaches have been compared with the authors proposed solution.

Visual Analytic Tools Tools such as Tableu, SAP Lumira, QlikView, and Datawatch fall into first category, i.e. they do not apply any data reduction related to visualization, even though they contain the most recent and advanced data engines. None of these tools can efficiently process and visualize data having more than 1 million rows. There is a great opportunity to implement the proposed solution in this paper along with these softwares.

Client Server Systems Online data visualization websites like Yahoo Finance, or Google Finance come under second category. They reduce data volumes by generating images instead of actual data visualization. They are dependent upon client systems to interact with these images to explore data. These systems rely on additional data reduction or image generation components between the data-engine and the client. Transferring large query results to external image generation or data reduction components will negatively impact the system performance as data transfer is one of the costliest operations.

Data-Centric Systems - The third category of systems consist of rich-client visualization systems which are also described in the section above. Authors proposed system can prevent the costly data transfer by running the expensive data reduction operations directly inside the data engine. The system modifies the original query by adding in some data reduction operations and then runs the new query. The data engine can then execute this new query, thus performing the additional data reduction task without the need of data transfer.

B. Data Reduction

In this section, the author talks about the pre-existing data reduction methods and how they are related to visualization.

Quantization - Most visualization systems reduce the

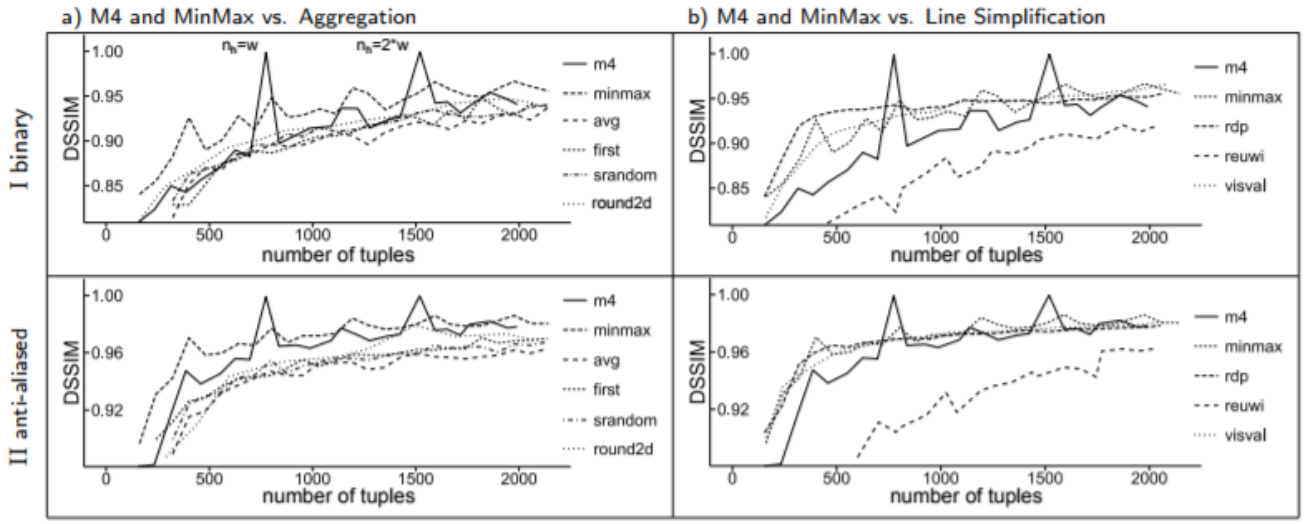


Figure 6. : Data efficiency of evaluated techniques, showing DSSIM over data volume

continuos time series data into discrete values, by generating images or rounding of the decimal integers. This does not allow correct reproduction of original data.

Time-Series Representation - The goal of existing works on time-series representation is to obtain a much smaller representation of the complete time-series. This is often achieved by dividing the time series into various intervals and calculating the average of those intervals. This result is the approximation of original time series. The authors in addition to these methods have focused on relational operators and incorporated the semantics of visualizations.

In addition to the above pre-existing methods there are some other methods which focus on *offline Aggregation and synopsis, Data Compression, Content Adaption, and various statistical approaches*. In offline aggregation technique, some aggregation methods like sum, avg, count of data are implemented which approximately represent the data, but they still are subjected to approximation errors. Data Compression is applied on application level data till now. Authorstechniques can also be applied to transport level data such as data packet compression. There are different data reduction techniques for different kind of contents. The proposed solution in this paper can be applied to any kind of content like videos, images, or text in web based systems.

C. Visualization-Driven Data Reduction

Burtini et al. [11] has described the usage of visualization parameters for data reduction. Howeverm they describe a client-server system as described earlier in category 3, i.e., they apply data reduction outside of the database. In the proposed solution, all the data processing including the data reduction is processed by the means of modified queries. Also, previous works use aggregation techniques for data reduction thus losing the important details in the vertical extrema and do not discuss the semantics of rasterized line visualizations as covered in this paper.

V. Our Proposal

Though the M4 aggregation provides an error free visualization it has the following limitations:

1) It doesn't work with other data sources besides RDBMS
2) It doesn't provide solution to map-reduce based big data frame works like Hadoop, spark etc.
3) and it doesn't handle the data file system like CSV, JSON and most visualization systems take data directly from these file systems [12].
4) It also doesn't provide solution for streaming data visualizations. But visualization of streaming time series data with appropriate reduction in data size is a crucial need of moder exploratory data analysis etc.

We propose some improvements and future works that can be done on top of this works.

- To cover the modern NoSQL databases like MongoDB and cluster based daat sources like Hadoop and spark we need to modify the M4 aggregation strategies and adapt those to MongoDB NoSQL queries and Hadoop Map-reduce queries [13]. It can be easily shown that the M4 query can be adapted to these platforms after slight modifications. For example if we write map-reduce queries using Apache Hive [14], [15] that comes with Hadoop stack the same SQL queries used by M4 can be used.
- For streaming data we can use incremental data processing techniques [16], [17] and apply the concepts of

M4 and make some modifications to propose an incremental version of M4 for real time data visualization with reduced data.

- For processing file system data we can create an intermediate data operator that will convert the M4 queries into a query that can be applied on CSV, JSON or relevant file types.

VI. Conclusion

We presented a review of the paper [7]. The paper presents a technique of time series data reduction that provides error free visualization and higher time performance. The work is only suitable for visualization systems where the data source is a RDBMS. We also identified few limitations of this work and proposed some future research directions.

References

[1] T.-c. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011.

[2] A. Labrinidis and H. V. Jagadish, "Challenges and opportunities with big data," *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 2032–2033, 2012.

[3] J. Fan, F. Han, and H. Liu, "Challenges of big data analysis," *National science review*, vol. 1, no. 2, pp. 293–314, 2014.

[4] C. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on big data," *Information Sciences*, vol. 275, pp. 314–347, 2014.

[5] G. Cormode and N. Duffield, "Sampling for big data," 2014.

[6] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE transactions on knowledge and data engineering*, vol. 26, no. 1, pp. 97–107, 2014.

[7] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl, "M4: a visualization-oriented time series data aggregation," *Proceedings of the VLDB Endowment*, vol. 7, no. 10, pp. 797–808, 2014.

[8] C. Mutschler, H. Ziekow, and Z. Jerzak, "The debs 2013 grand challenge," in *Proceedings of the 7th ACM international conference on Distributed event-based systems*. ACM, 2013, pp. 289–294.

[9] Z. Jerzak, T. Heinze, M. Fehr, D. Gröber, R. Hartung, and N. Stojanovic, "The debs 2012 grand challenge," in *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*. ACM, 2012, pp. 393–398.

[10] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[11] G. Burtini, S. Fazackerley, and R. Lawrence, "Time series compression for adaptive chart generation," in *Electrical and Computer Engineering (CCECE), 2013 26th Annual IEEE Canadian Conference on*. IEEE, 2013, pp. 1–6.

[12] M. Bostock, V. Ogievetsky, and J. Heer, "D^3 data-driven documents," *IEEE transactions on visualization and computer graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[13] A. B. Patel, M. Birla, and U. Nair, "Addressing big data problem using hadoop and map reduce," in *Engineering (NUiCONE), 2012 Nirma University International Conference on*. IEEE, 2012, pp. 1–5.

[14] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, "Hive: a warehousing solution over a map-reduce framework," *Proceedings of the VLDB Endowment*, vol. 2, no. 2, pp. 1626–1629, 2009.

[15] E. Barbierato, M. Gribaudo, and M. Iacono, "Modeling apache hive based applications in big data architectures," in *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013, pp. 30–38.

[16] B. De Seabra and R. Ravindran, "Incremental data processing," May 22 2015, uS Patent App. 14/720,448.

[17] C. Yan, X. Yang, Z. Yu, M. Li, and X. Li, "Incmr: Incremental data processing based on mapreduce," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. IEEE, 2012, pp. 534–541.