Data Series Management: Fulfilling the Need for Big Sequence Analytics

Kostas Zoumpatianos [*,#1], Themis Palpanas [#2]

[*] *Harvard University*
[#] *LIPADE, Paris Descartes University*
[1] kostas@seas.harvard.edu
[2] themis@mi.parisdescartes.fr

***Abstract*—Massive data sequence collections exist in virtually every scientific and social domain, and have to be analyzed to extract useful knowledge. However, no existing data management solution (such as relational databases, column stores, array databases, and time series management systems) can offer native support for sequences and the corresponding operators necessary for complex analytics. We argue for the need to study the theory and foundations for sequence management of big data sequences, and to build corresponding systems that will enable scalable management and analysis of very large sequence collections. To this effect, we need to develop novel techniques to efficiently support a wide range of sequence queries and mining operations, while leveraging modern hardware. The overall goal is to allow analysts across domains to tap in the goldmine of the massive and ever-growing sequence collections they (already) have.**

I. Introduction

Massive data series collections are becoming a reality for virtually every scientific and social domain, and there is an increasingly pressing need by relevant applications for developing techniques able to index and mine very large collections of sequences, or data series[1]. Examples of such applications come from biology, astronomy, entomology, engineering, the web, and other domains. It is not unusual for these applications to involve numbers of data series in the order of hundreds of millions to billions, which are not analyzed in their full detail due to their sheer size [1].

Despite the strong increasing interest in data series management systems (see Figure 1), existing approaches (e.g., based on DBMSs, Column Stores, TSMSs, or Array Databases) do not provide a viable solution, since they have not been designed for managing and processing sequence data as first class citizens: they do not offer a suitable storage model, declarative query language, or optimization mechanism. Moreover, they lack auxiliary data structures (such as indexes), that can support a variety of sequence query workloads in an efficient manner. For example, they do not have native support for similarity search [2], [3], and therefore, cannot efficiently support complex analytics, or machine learning, on very large data series collections.

Current solutions for processing data series collections, in various domains, are mostly ad hoc (and hardly scalable), requiring huge investments in time and effort, and duplication of effort across different teams. For this reason new data management technologies should be developed; albeit ones that will meet their requirements for processing and analyzing very large sequence collections.

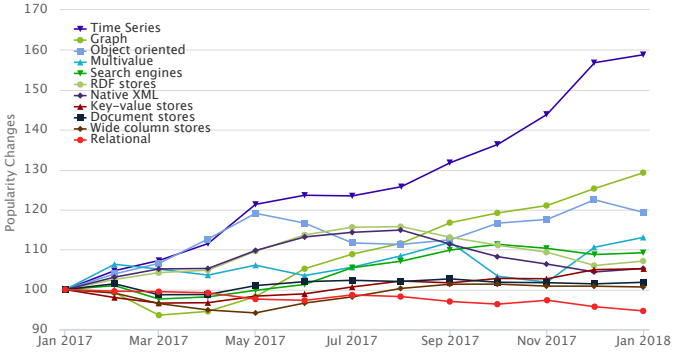

Fig. 1. DBMS category popularity change trend [17]

Our vision is to design and develop a general purpose Sequence Management System, able to cope with big data series (very large and continuously growing collections of data series with diverse characteristics, which may have uncertainty in their values), by transparently optimizing query execution, and taking advantage of new management and query answering techniques, as well as modern hardware [2], [18]. Just like databases abstracted the relational data management problem and offered a black box solution that is now omnipresent, the proposed system will enable users and analysts that are not experts in data series management to tap in the goldmine of the massive and ever-growing data series collections they (already) have. Our preliminary results, including the first data series similarity search benchmark [4], and indexing algorithms that can be efficiently bulk-loaded [5], [6], [7], adapt to the query workload [8], [9], [10], support similarity queries of varying length [11], [12], take into account uncertainty [13], [14], and exploit multi-cores [15] and distributed platforms (e.g., Apache Spark) [16], are promising first steps.

II. Data Series Management: Open Challenges

In this section, we briefly provide an overview of the necessary components of a Sequence Management System

[1]A data series, or data sequence, is an ordered sequence of points. If the dimension that imposes the ordering of the sequence is time then we talk about time series, but it could also be mass (e.g., in mass spectrometry), angle (e.g., in astronomy), or position (e.g., in biology).

(SMS), and we discuss the envisioned functionality and open research problems (for more details see [2], [18]).

Data Series Queries: There are various types of data sequence queries that analysts need to perform: (a) simple Selection Projection-Transformation (SPT) queries, and (b) more complex Data-Mining (DM) queries (similarity queries, clustering, classification, outlier patterns, frequent sub-sequences, etc.). The majority of the SPT queries could eventually be handled by current database management systems, albeit not optimally. DM queries on the other hand cannot be supported by current data management systems, since they require specialized data structures and algorithms.

Data Model: Neither the relational model nor the array model can adequately capture the characteristics of sequences. In the case of relational data, there are various options available for translating sequences into relations and each one of them has significant limitations. On the other hand, in Array Databases we lack the expressive power to define collections of sequences, and are restricted to defining large multi-dimensional matrices. An ideal sequence model should instead be able to effectively describe collections of sequences and allow us to do operations on them (e.g., select sequences based on metadata or their values, project them as complete sequences or sub-sequences, and join them in a variety of ways).

Data Structures: Our experience with massive sequence collections shows that creating an index data structure can become a significant bottleneck in a data sequence analysis process [1]. It is imperative therefore, to develop novel indexing techniques that interactively and adaptively build parts of the index, focusing on the data necessary to answer the queries. Other promising directions should also be explored, such as methods that rely on fast scans of the data [19]. Relevant data structures should also be re-designed to support uncertain series (i.e., the raw data have an inherent uncertainty in their values) and queries of varying length, and to take into account the trends of modern hardware (i.e., multi-cores, SIMD, and GPUs).

Distributed Processing: During the last years there has been a lot of research on cluster computing. Nevertheless, up to this point work on sequential data query processing has mainly concentrated on efficiently performing parallel scans of the complete dataset, while all indexing-related studies only consider read-only operations. Even though various approaches have been proposed for speeding up iterative algorithms, none of the proposed models is a suitable match for the algorithms and techniques we need, where timely communications among workers play a crucial role in reducing the amount of total work done. Therefore, there is need for work in this area [18], taking into consideration new paradigms as well [20].

Cost based optimization: We may have multiple different execution strategies for answering the same query, including the various choices of serial scans, indexes, and processing methods (e.g., parallelization, GPU, etc.). The challenge in choosing the right execution strategy is to estimate the amount of data that such a query will need to access before executing it. For example, a fast parallel SIMD-enabled scan on compressed data might be a better option than the use of a non-optimized index when SIMD instructions are available, but not a better choice when such instructions are not available. This problem becomes even more challenging when complex queries involving several DM and SPT operators need to be executed. No methods have been proposed so far to solve this problem, and ground-breaking work needs to be done.

III. Conclusions

Very large data series collections are becoming common place for applications across many domains, and the need for systems that can support big sequence analytics is now more pressing than ever. Despite the efforts of the community, and the progress achieved during the past two decades, we are still far from fulfilling this need, with several challenging and exciting research opportunities calling for action.

Acknowledgements

We would like to thank our collaborators: R. Akbarinia, A. Camerra, M. Dallachiesa, N. Dayan, Y. Fatourou, J. Gehrke, S. Idreos, I. Ilyas, E. Keogh, H. Kondylakis, M. Linardi, Y. Lou, F. Masseglia, K. Mirylenka, B. Nushi, B. Peng, F. Roncallo, J. Shieh, and D.-E. Yagoubi.

References

[1] T. Palpanas, "Big sequence management: A glimpse of the past, the present, and the future," in *SOFSEM*, 2016.

[2] ——, "Data series management: The road to big sequence analytics," *SIGMOD Rec.*, 2015.

[3] S. K. Jensen, T. B. Pedersen, and C. Thomsen, "Time series management systems: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 11, pp. 2581–2600, 2017.

[4] K. Zoumpatianos, Y. Lou, T. Palpanas, and J. Gehrke, "Query workloads for data series indexes," in *SIGKDD*, 2015, pp. 1603–1612.

[5] A. Camerra, T. Palpanas, J. Shieh, and E. J. Keogh, "isax 2.0: Indexing and mining one billion time series," in *ICDM 2010*, 2010.

[6] A. Camerra, J. Shieh, T. Palpanas, T. Rakthanmanon, and E. J. Keogh, "Beyond one billion time series: indexing and mining very large time series collections with isax2+," *KAIS*, 2014.

[7] H. Kondylakis, N. Dayan, K. Zoumpatianos, and T. Palpanas, "Coconut: A scalable bottom-up approach for building data series indexes," in *PVLDB*, 2018.

[8] K. Zoumpatianos, S. Idreos, and T. Palpanas, "Indexing for interactive exploration of big data series," in *SIGMOD*, 2014.

[9] ——, "RINSE: interactive data series exploration with ADS+," *PVLDB*, 2015.

[10] ——, "ADS: the adaptive data series index," *VLDB J.*, 2016.

[11] M. Linardi and T. Palpanas, "ULISSE: ULtra compact Index for Variable-Length Similarity SEarch in Data Series," in *ICDE*, 2018.

[12] M. Linardi, Y. Zhu, T. Palpanas, and E. J. Keogh, "Matrix profile X: Valmod - scalable discovery of variable-length motifs in data series," 2018.

[13] M. Dallachiesa, B. Nushi, K. Mirylenka, and T. Palpanas, "Uncertain time-series similarity: Return to the basics," *PVLDB*, vol. 5, no. 11, pp. 1662–1673, 2012.

[14] M. Dallachiesa, T. Palpanas, and I. F. Ilyas, "Top-k nearest neighbor search in uncertain data series," *PVLDB*, vol. 8, no. 1, pp. 13–24, 2014.

[15] B. Peng, Y. Fatourou, and T. Palpanas, "Adaptive data series indexing in parallel," *Technical Report*, 2018.

[16] D.-E. Yagoubi, R. Akbarinia, F. Masseglia, and T. Palpanas, "Dpisax: Massively distributed partitioned isax," 2017.

[17] "Db-engines," https://db-engines.com/en/ranking_categories, 2018.

[18] T. Palpanas, "The parallel and distributed future of data series mining," in *High Performance Computing & Simulation (HPCS)*, 2017.

[19] T. R. et al., "Searching and mining trillions of time series subsequences under dynamic time warping," in *SIGKDD*, 2012.

[20] P. A. Bernstein, M. Dashti, T. Kiefer, and D. Maier, "Indexing in an actor-oriented database," in *CIDR*, 2017.