I J R B A T, Issue (XI) Vol (II) May 2023 : 161-163
A Double-Blind Peer Reviewed & Refereed Journal
OPEN ACCESS
e-ISSN 2347 – 517X
Original Article
INTERNATIONAL JOURNAL OF RESEARCHES IN BIOSCIENCES, AGRICULTURE AND TECHNOLOGY
 www.ijrbat.in

DATA STREAMING AND ITS APPLICATION TO STREAM PROCESSING

Mayuri G. Ghodmare[1], M.N.Quadri[2] and S.B.Kishor[3]

[1]Department of Computer Science, Gondwana University, Gadchiroli
[2]Department of Computer Science, N. S. Sci. College Bhadrawati
[3]Department of Computer Science, Sardar Patel. Mahavidyalaya, Chandrapur.

Communicated : 16.01.2023 | Revision : 21.02.2023 & 02.03.2023 Accepted : 27.03.2023 | Published : 30.05.2023

ABSTRACT:
This tutorial presents recent research results in the field of data streaming as applied to stream processing systems. In particular, we introduce data streaming models and discuss the results of key algorithms in this research area. We then detail how such algorithms can be applied to modern distributed stream processing systems to improve their efficiency. Finally, we outline open research directions in this area. Today's data is generated from a myriad of sources, including IoT sensors, servers, security logs, applications, and internal/external systems. It is almost impossible to control the structure, data integrity, or the amount or speed of data generated.

Keywords :- Data, Computing, Stream processing, Cloud computing.

INTRODUCTION :

Stream analytics is now used in many situations where the amount of data and the speed of production precludes other approaches (such as batch processing). Industrial control systems, sensor networks, and business intelligence are examples of industries that currently rely heavily on stream analytics to extract actionable information from large, near-real-time data streams. Application scenarios in these contexts have recently required the development of new technologies and frameworks to support the desired requirements. A data streaming model provides a randomized and/or approximation solution for computing a given function over a (distributed) stream of data items in the worst-case scenario, with the goal of reducing resource usage increase. Stream processing, which is somehow complementary and more practical, provides efficient and highly scalable frameworks to perform so real-time generic computation on streams, relying on cloud computing platforms. While data streaming algorithms and stream processing systems were born separately and have so far seen parallel evolutions, their duality opens the opportunity for making stream processing systems more efficient and performant through data streaming solutions. This tutorial paper provides a brief overview of the data streaming model, and describes how current limitations of stream processing systems may be surpassed or at least mitigated through data streaming solutions. This new path in the evolution of stream processing systems others promising results that have immediate applicability in real-world problems; however, it still presents complex open problems that may possibly appeal the research community. The rest of this paper is structured as follows. Section 2 introduces the general system model considered for data stream algorithms tailored to the stream processing model. Section 3 applies to the introduction to data streaming. In particular, we provide an overview of the key algorithmic results of this study.

http://doi.org/10.29369/ijrbat.2023.02.1.0023

SYSTEM MODEL :

Data streaming focuses on estimating metrics across streams. This is an important task in data-intensive applications that need to process large amounts of data quickly and accurately. In particular, this is applicable to stream processing at both the system level and the application level. Here we formalize a system model that bridges the gap between traditional distributed data streaming models (that is, distributed function monitoring) and stream processing models. Consider a distributed power processing system "PLC". Multiple compute nodes are deployed in a cluster that exchanges data via messages sent over the network. SPS runs stream processing applications represented by topologies. In other words, it is a directed acyclic graph that connects operators, represented by nodes, to data streams, represented by edges. Each topology contains at least one source. H. Operators connected only by outgoing data streams and sinks i. H. Operators connected only by incoming data streams. Data from the source is encapsulated in units called tuples, and each stream is a sequence of tuples. Without loss of generality, we assume here that each tuple is a unique set of key-value pairs that can be customized to represent complex data structures. For ease of discussion, the rest of this work deals with streams of unary tuples with single non-negative integer values. A us tuple in a data stream can be represented as an element in the stream σ = ht1, . . . , tj, . .tmi, each element t comes from a universe of size n.

IMPROVING STREAM PROCESSING SYSTEMS

Stream processing systems (PLCs) are gaining importance today as tools for performing analysis on continuous data streams. These applications have become ubiquitous due to increased automation in telecommunications, healthcare, transportation, retail, science, security, emergency response, and finance. As a result, various research communities have developed their own programming models for streaming. The communities most committed to streaming optimization are digital signal processing, operating systems and networks, complex event processing, and databases. Hirsel et al. proposes a catalog containing 11 stream processing optimizations developed by these communities.

FUTURE RESEARCH DIRECTIONS :

The work described in the previous section mainly focuses on balancing the load assigned to the parallelized instances Oj ∈ K of operator O. Of course, there are other ways in which stream processing could benefit from the use of data streaming, opening up some promising research directions. This includes the applicability of other data stream algorithms, their application to other system-level problems, and investigation of application-level problems.

Application level :

Approximate queries are one of the methods developed in the database community to provide fast (but not exact) answers to queries. This is similar to the Lambda architecture being considered in the stream processing space. Earlier, we introduced the concept of load shedding. An interesting contribution by Reiss et al. is said to switch from exact to approximate calculations when the stress becomes too great. In either case, data streaming can provide a way to introduce these concepts rigorously into stream processing.

System level :

Data streaming has demonstrated its strengths in the database community, such as providing efficient join size estimation. There are many query plan optimizations performed by database systems based on such estimates. The stream processing community may want to port these to stream processing systems and use data streaming to gather relevant information and dynamically optimize deployment phases/query

http://doi.org/10.29369/ijrbat.2023.02.1.0023

plans. Similarly, data streaming can be applied to online adaptive planning and automatic parallelization, two topics of much ongoing research, mostly based on machine learning techniques. Focusing on specialized stream processing systems shows other possible applications. B. Improved partitioning in graph-based computations. In this area, a recent contribution showed how the split quality depends on the topological properties of the input graph (such as the vertex degree distribution). Such functionality can be tracked using a data streaming solution.

Other metrics :

There are many other, more complex metrics being explored in the data streaming community. B. Compute the entropy of the stream [8]. This may find innovative applications in stream processing. For example, the similarity distance estimate between two streams can be used to detect redundancy in stream processing applications.

CONCLUSION :

Today's data is generated from a myriad of sources, including IoT sensors, servers, security logs, applications, and internal/external systems. It is almost impossible to control the structure, data integrity, or quantity or speed of the data generated. While traditional solutions are designed to ingest, process, and structure data before processing it, streaming data architectures offer the ability to consume, store, enrich, and analyse data in motion. Offers. Therefore, applications that manipulate data streams always need two main functions: storage and processing. Storage must be able to record large streams of data continuously and consistently. Processing must be able to interact with storage, consume data, analysed it, and perform computations.

REFERENCES:

N. Aron, Y. Matthias, M. Szegedy. e Spatial complexity of the frequency moment approximation. Proceedings of the 28th ACM Symposium on Computing Theory, STOC, 1996.

E. Ansome and his Y. Bussner. Distributed Information Difference Estimation on Data Streams. IEEE Transactions on Parallel and Distributed Systems, 25(2):478–487, 2014.

R. Ben-Basat、G. Einziger、R. Friedman und Y. Kastner. Heavy highs of streams and sash windows. IEEE INFOCOM 2016 - e 35th Annual IEEE International Conference on Computer Communications, page 1–9, 2016.

M. Caneill, A. El Rheddane, V. Leroy, N. De Palma. Location-based routing in stateful streaming applications. Proceedings of the 17th International Middleware Conference, Middleware `16, 2016.

V. Cardellini 、 E. Casalicchio 、 M. Colajanni 、 P. S. Yue State-of-the-art technology for locally distributed web server systems. ACM Computing Surveys, 34(2):263–311, 2002.

Cardellini V, Grassi V, F. Lo Presti and M. Nardelli. Optimal operator placement for distributed stream processing applications. Proceedings of the 10th ACM International Conference on Decentralized and Event-Based Systems, DEBS, 2016

http://doi.org/10.29369/ijrbat.2023.02.1.0023