Ilkogretim Online - Elementary Education Online, Year; Vol 20 (Issue 5): pp. 424-430
http://ilkogretim-online.org
doi: 10.17051/ilkonline.2021.05.44

Modelling Concerns in Data Streaming Systems

Shailender Kumar, Department of Computer Science & Engineering, Delhi Technological University, New Delhi, India, shailenderkumar@dce.ac.in
Pankaj Lathar, Department of Computer Science & Applications, Bhai Parmanand Institute of Business Studies, New Delhi, India, pankajlathar@gmail.com

Abstract: This paper discusses the necessityand analysis of theproblems emerging from a model of evaluating stream data. In these models, informationdoesn't take the shape of persistent relations, however rather arrives in multiple, perpetual, fast, time dependentinformation streams. Additionally retrospection of the past work significant to data stream management systems and recent comeswithin thespace, the paper delves the topics in stream question languages, new needs and challenges in questionprocess, and recursiveproblems.

Keywords: data streams,data stream management system, database management system, continuous queries, one-time queries, timestamping, temporal database

I. INTRODUCTION

Data intensive applications are gaining wide recognition in recent times. These are the applications in which data is demonstrated as transient stream of data in addition to the persistent data relations. Some of the examples of these types of applications are security, web applications, financial applications, weather monitoring, sensor networks, telecommunications data management, manufacturing and many more [1]. Individual data items stored as rowsin these application's models like call records, sensor readings, web page visits, network measurements, etc. But rapid and perpetual incomingof data that is time dependent with absolute as well as capricious streams, generates some new fundamental research problems.

In all these data-intensive applications, simply loading arriving data in database management system is not a feasible solution. Traditional database management systems are designed to evaluate only one-time queries and not for continuous queries [2]. And data stream applications demand the loading of all the rapid and continuous data arriving for individual data items. The key ingredients identified in executing queries and processing of them are approximation and adaptivity. These are used over rapid data streams for data analysis and mining purpose while in traditional database management system, focus is on to compute precise answers for stable queries.

In this paper, all the fundamental models and issues related to Data Stream Management System are considered. In Section 2, recent data stream processing projects and the research done in past in field of data streams are reviewed like sequence databases, active databases, filtering systems, continuous queries and so on. While most of the work done till now has applications in data stream processing, but there are still many problems left which needs to be addressed in order to realize the complete functionality of Data Stream Management System.

In Section 3, data stream model and stream queries are considered. Streams are appended with the transient tuples in relations queries used are SQL operating queries. In later sections, issues which are responsible to complicate the models and query languages are being discussed like sliding windows, timestamping and ordering.

Section 4 delves the area of processing the queries and discovers some more important issues comprising of:Unbounded amount of memory, Query processing using sliding window, approximation technique in query language.

The paper is concluded in section 5 with a brief summary on the analysis done on issues of stream data system and the models discussed.

II. RELATED WORK

This section discusses various current and past projects associated with the data stream management system has been discussed.

The Tapestry System uses append only databases for content based filtering over email and bulletin board messages. This system uses continuous queries for this purpose. To provide efficient and append only

results for the query fired, the query language used is a restricted subset of SQL [10]. The Alert System uses continuous queries which are defined over append-only active tables using the conventional SQL database which uses event-condition-action style triggers to perform certain action on occurrence of particular event with aspecific condition.
Continuous queries in XPath language were used in XFilter content based filtering system based on user profiles toevaluate efficient filtering of XML documents. Another content based filtering systemis Xyleme which uses restricted query language to enable very high throughput [12]. A stream database manager named as Tribeca uses network packet streams to provide restricted querying capability. A stream query processing system, known as Tangram, was used to analyse the large amount of data store in database by using different stream processing techniques.
Continuous queries were used in OpenCQ and NiagaraCQ systems over a wide area network to monitor persistent data sets. Algorithm used by OpenCQ which is based on incremental view maintenance, is query processing algorithm [9]. NiagaraCQ uses grouped continuous queries for efficient evaluation and addresses the issue of scalability in large number of queries [2]. For NiagaraCQ project, an issue of blocking operators supporting the query plans over the data streams has been addressed by Shanmugasundaram [6]. A new methodology for optimization which is based on arrival rate of stream and rate of data processing, known as Rate-based optimization, proposed by Viglas and Naughton, over data streams [11].
A data model introduced a form of data streams which is an ordered sequence of tuples known as chronicles. That data model is named as the Chronicle data model which operates on traditional relations together with the chronicles. This model defines a new algebra known as chronicle algebra which is a modifiedversion of user specific views with their mapping on conceptual schema[13]. The main aim was to maintain the views incrementally, defined in the chronicle algebra, without the need of loading those chronicles. Seshadri, Ramakrishnan, along with Livny proposed a supplementary query language and algebra to query ordered relations which are normally called as sequences [7].
Materialized views are related to the continuous queries that gets incrementally updated or re-evaluated every time when the base data changes. In case the base data is not available, enough data must be saved to sustain a view which can be known as self-maintenance. If a base data comprises with the maintenance of view then the system must know the data expiration to remove that base data [2].
Some basic technical ideas and target applications of data stream management system are shared by the Telegraph project. For efficient processing of queries in unpredictable and volatile environments, an adaptive query engine is used. Autonomous data sources perform dynamic data integration which also supports this adaptive query engine, and this new system is known as the Tukwila system [4].
A new data processing system has been designed by the Aurora project to exclusively target the stream monitoring applications. It contains a large amount of triggers. Agraph can be used to represent each trigger in which every single node represents unique built-in operation amidall of the pre-defined operations. Both types of optimization, run-time and compile-time, can be performed by Aurora [12].

III. MODEL FOR STREAM DATA

This section discusses a process in which the complete data entry arrives as single or multipleperpetual data streams rather than all of the input data is available on disk or memory for random access. There are many ways in which data streams differs from conventional relational model. First one is, in the stream data elements arrives online in random order so there is no control of the system over that order of the data needs to be operated either athwart the data stream or innards a data stream, data stream magnitude is unbounded, Data element is discarded or archived once it get processed and retrieval of that stream is not possible until and unless it is stored in memory explicitly. But the memory size is small relative to the data stream which is to be stored in it.
Operation of continuous data elements in this model does not preclude that of in conventional stored relations [2]. The process of stream data model is depicted in Figure 1. Joins between the both, current data and already loaded data, can be performed by thecontinuous queries frequently. So, an assumption is made in that the contents of the stored relations remains static. By doing this, if any updates to the stored relations occur concurrent tothe processing of the data streams, all the transaction processing issues can be prevented.

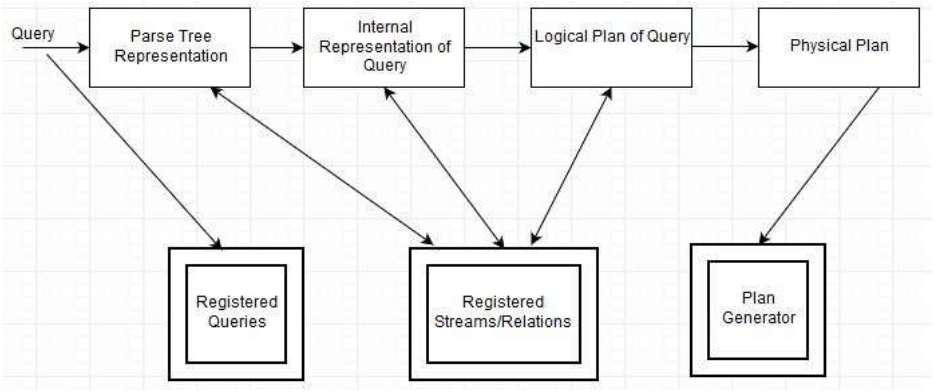

Fig.1. Process for Data Stream Model

3.1 Distinction Between Queries

Continuous data streams queries have many similarities with the queries of the traditional database management system. Though, they still have two important discrepancy inimitable to the model of data stream. The foremost discrepancy is between the continuous queries and the one-time queries. Traditional database management system queries are one-time queries, evaluated on point in time snapshot of data which returns the answer to the user. While continuous queries run over the continuous data streams arriving at the same time [5]. Continuous queries produce the answer, reflecting all the streamed data, over the time. Answers of the continuous query can be loaded and contemporize as unprocessed data continues to arrive, or can be yield in the form of data streams. Example can be an aggregation query which continuously changes the answers stored in the tuples.

Another discrepancy is between the ad hoc queries and the predefined queries. Any query that is given as input to the system afore the arrival of felicitous data is known as pre-defined query [1]. And the queries that are released in real-time since the beginning of streaming are known as Ad hoc queries. There are two serious issues with ad hoc queries because of that they complicate the data stream management system design, first is ad hoc queries are unbeknownst in earlier phases of query optimization purpose and also for common sub expressions identification across the queries and the second one and the most important one is that ad hoc query requires to reference those data elements that are already arrived in the system and may get discarded also to generate the correct answers.

3.2 Examples of Data Stream System

Some examples are discussed that motivates the necessity of data stream systems that can be realized by numerous technical fields including networking, funding, security, web based applications and signal detection.

Traderbotworks on the real-time streaming financial data like stocks and news feeds to evaluate the queries [8]. It is a commercial search engine. Customer of the traderbot posed some queries on their website which can be taken as illustration forperpetual queries and one-time queries.

There exist so many applications that are emerging in the area of sensor networks and monitoring. In these applications, several sensors are distributed geographically which generates a stream of data that needs to be firstly combine them, then analyse them and after that monitoring is performed [4].

Many security applications restrict the network packet stream by applying various sophisticated rules. Like, iPolicy Networks which provides services for instance firewalls, obtrusive detection for gigabit conglomerate package stream over an integrated security platform [3]. Processing of complex streams that include URL filtering and correlation across multiple networks traffic can be done by this type of platform.

Monitoring of online web logs for applications like load balancing, monitor the performance and personalization. Widely distributed web-servers aids some websites that are needed to coordinate with the analyses of distributed clickstream.

Application domain of Network Traffic Management is taken as example for detailed explanation. It tries to foster data related to configuration of traffic flow which can be done by monitoring the information of header of the network packet athwart a coalition of routers. In this type of applications domain, continuous queries arise naturally which cannot be supported by database management system technology.

3.2.1 General Structure of Continuous Queries

Considera network traffic administration system consisting of immense number of networks, just like an Internet Service Provider network. Traces of network packets are put together which are being collected from various links of the network. There are two specific links, one is customer link CL, connecting the customer to the internet service provider's network, and another is backbone link BL, connecting internet service provider's backbone network to two routers. Two streams of traces of packet, analogous to those two links, can be denoted by let CL and BL. An assumption is made in order to trace the packets that only five fields of header are contained in the trace: s is the sender's IP address, d is the destination IP address, ID is the unique identification number for each packet given by the sender, length is Packet length and t is the time recorded when packet header arrives.
Firstly compute continuous query Q1, to evaluate the load of link BL over an interval of one-minute and notify when the load crosses the threshold th, to the network operator. Two functions are used: notify_operator and get_minute, having their natural interpretation.

```
Q1:  SELECT          notify_operator(sum(length))
FROM          BL
     GROUP BY    get_minute(t)
     HAVING       sum(length)>th
```

In second query Q2, evaluation for the generation of amount of traffic for every flow is done by isolating all the flows in the backbone link.

```
Q2: SELECT          fID, s, d, sum(length) AS flength
    FROM          (SELECT s, d, length, t
                   FROM BL
                   ORDER BY t)
    GROUP BY     s, d, get_fID(s, d, t)
AS fID
```

get_fID is a customized function, taking source &destination IP address and a packet timestamp as arguments and returns a value of flow identification to which a specific packet belongs.
Now in query Q3, evaluate fraction of the traffic flows through backbone link that can be accredited to network of the customer. This query attempts to identify the likely cause of congestion registered during different time periods on the customer network. It is a kind of ad hoc continuous query.

```
Q3: (SELECT     count(*)
     FROM       CL, BL
WHERE     CL.s=BL.s and CL.d=BL.d
and CL.ID=BL.ID)/
     (SELECT count(*) FROM BL)
```

Final query example Q4 is to monitor the pairs of source-destination, in terms of backbone traffic, in top 5 percent. It is a continuous query. It uses WITH construct of SQL-99 for ease of exposition.

```
Q4: WITH Load AS
     (SELECT        s, d, sum(length) AS trafic
      FROM          BL
GROUP BY    s, d)
     SELECT      s, d, trafic
     FROM       Loads AS L3
     WHERE     (SELECT count(*)
FROM Loads AS L2
          WHERE L2.trafic<L3.trafic)>
(SELECT    0.90 * count(*) FROM Loads)
     ORDER BY traffic
```

IV. COMPUTATION CHALLENGES

4.1 Query Processing on Streamed Data

Query handling in the information stream model of calculation accompanies its own extraordinary difficulties. In this area, the most intriguing of these difficulties are considered, and a few elective methodologies for settling them are depicted.

4.1.1 Need of IndefiniteStorage

As streamed data isconceivably indefinitely large so the amount of the capacity neededfor the processing of a correct reply to a stream question can likewise develop extensively. Whilst secondary storage calculations [9] that are dealing with the data sets bigger than fundamental storage has been considered, similar calculations are not appropriate to stream applications as they don't bolster nonstop questions and are commonly too moderate for continuous reaction. New stream data is always thriving even as the paststreamed data is handled, a measure of calculation of count per data component have to be less, or moreover the inertness of calculation will be raised up.
Arasu et al. [7] made some underlying strides towards recognizing questions that can be addressed precisely utilizing a given limited measure of storage that can be imprecise if circle gets to are permitted. They conceive the constrained orderof questions and give a total portrayal of the questions in that order which demand for a conceivably indefinite measure of storage for replying. Their outcome demonstrates that having no clue ofspan of the info data element, it will be difficult to put the cutoff on the storage prerequisites for the most normal questions including joins.

4.1.2 Group Processing, Sampling, and Synopses

Assume that a stream of data question is addressed utilizing an organised data that may belooked after progressively [14]. The best broad depiction of similar organized data is, to the point that it bolsters two methods, computeAnswr() and updte(tuple).The refresh process is conjured to refresh the organized data as all new data component how up, andcomputeAnswr strategy creates refreshed outcomes to the question. When handling nonstop inquiries, the best situation is that the two methods are quick with respect to the entry rate of components in the data streams.

Group Processing

The principal situation is that the refresh process is quick however the computeAnswr methodis moderate. The normal arrangement is to progress the data in clumps. Instead of delivering a persistently contemporary reply, the data components are supported as they show up, and the response to question is figured occasionally as time grants. This approach of estimate through group preparing is appealing in light of the fact that it does not cause any vulnerability about the precision of the appropriate response, giving up convenience.

Sampling

In the second situation, computeAnswr might be quick, however the refresh operation is moderate, and it need larger time than the normal between landing time instant of data components. It is pointless to endeavour to make utilization of the considerable number of data when registering an answer, since data arrives speedier than it may be prepared. Rather, some rows can be passed over by and large, with the goal that the inquiry is assessed over a specimen of stream instead of over the whole data stream. Outlining examining based calculations that can deliver surmised response that are validated near the correct response is an essential and dynamic zone of research.

Synopsis Structures

Obviously, organized data where both the refresh and computeAnswr methods are quick and generally attractive. For division of streamed data questions where no correct data with the coveted attributes exists, someone may regularly plan a rough data structure that keeps up a little summary or draw of the data as opposed to a correct portrayal, and subsequently can keep calculation per stream data component to a least. Performing data lessening through summation organized data as a contrasting option to cluster handling or testing is a productive research region.

4.1.3 Extraction from Past Data

In streaming model of calculation, when a data component has been streamed by, it can't be returned to. This restriction implies that specially appointed inquiries that are produced later, a few data has just been disposed of might be difficult to answer precisely [3]. One basic answer for this issue is to stipulate that

impromptu questions are just permitted to cross-reference upcoming data: they are assessed as if the streams started right when the question was put out, and any former stream components are overlooked.
A more aggressive way to deal with taking care of specially appointed questions that cross-reference former data is to look after rundowns of streams that can be utilized to give estimated answers to future impromptu inquiries. Adopting this strategy requires settling on a choice ahead of time about the best approach to utilize memory assets to give great rough responses to an expansive scope of conceivable future inquiries.
There is an imperative distinction: in a customary database framework, at the point when a file or view is deficient with regards to, it is conceivable to refer the basic connection, though at an expanded cost.
In stream model of calculation, if the proper rundown structure is absent, at that point no further plan of action is accessible. In these streaming applications, where the utmost questions are extensive ceaseless questions as opposed tofleeting single-timeinquiries,the additions that can be accomplished by multi-inquiry advancement can be fundamentally more prominent than what is conceivable in conventional database frameworks.

4.2 Problems in Creating Summary Structures

The main focus will be on the issues in creating synopses for stream model. Some concerns are discussed which can be occur as a challenge while implementing algorithms and techniques in data stream model.

4.2.1 Prevention from Stale Data

Analysis and statistics can be affected by some stale data present in historical database. So to prevent that data from manipulating the statistics and analysis, sliding windows can be used. One technique used by the researchers, Datar, Babock and Motwani, is to implement the reservoir sampling algorithm with the sliding window. Another technique proposed by Koudas and Guha is to use V-optimal histograms on the organised stream of data and buffer each element in sliding window model.

4.2.2 Contingent Samples

Arbitrary instance may be utilizedas condensed structures in various situations where a tiny instance is estimated to acquire the crucial properties of collection of data. It may be the simplest form of characterization in a stream management system and a brief outlinemay be fabricated from a sample itself [5]. A new type of sampling has been suggested as a substitute of homogeneous sampling in recent times, to limit the error occurs because of the discrepancy in data and similarly to lower the error caused by group-by query. Sampling algorithm given by Vitter, named as reservoir sampling algorithm is appropriate for stream model.

4.2.3 Stream Reduction

Some researchers has introduced, in recent times, the concept of stream reduction in the streaming algorithms. To implement better reductions, new streaming algorithms known as list-efficient algorithm is needed to be employed [4]. This algorithm defines a concept that rather than processing single data element at an instant of time, a sequence of data elements can be presented in a concise form at the same time. If that list of data item is processed efficiently within time, which is a responsibility of brief depiction of size, then it will be called as list-efficient algorithm. Certain list-efficient procedures have already been developed to cope with problems like to count total number of rectangles present in the graph which is taken as a stream for processing, computing frequency moments.

V. CONCLUSION

This paper discusses the possible challenges that can emerge when management of data, their processing, and problems related to algorithms are considered. A brief review of all the current and past work associated with data streams is also been considered, along with the architecture of the data stream model process.
Some elementary issues are also been discussed in this paper, including random samples, prevention from stale data using sliding windows, stream reduction. A major issue is that how to deal with multiple streams as it is not feasible to divert all the high frequency streams on a single position for query evaluation. Hence it is now essential to implement certain type of evaluation at the entrance point ofmultiple streams which will raise all the issues at each and every level of data stream system. Lastly, questions of query optimization, structuring of synopses, management of resources, approximation processing of query remain opened for many systems.

From theoretical perspective, there is need to define extension of relational operators in order to handle the streamed data and study of that stream algebra with its properties. This will lead to development of generic query processor for all types of streamed data.

REFERENCES

1. D. Terry, D. Goldberg, D. Nichols, and B. Oki. Continuous queries over append-only databases. In Proc. of the 1992 ACM SIGMOD Intl. Conf. on Management of Data, pages 321–330, June 1992.
2. R. Avnur and J. Hellerstein. Eddies: Continuously adaptive query processing. In Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data, pages 261–272, May 2000.
3. S. Babu and J.Widom. Continuous queries over data streams. SIGMOD Record, 30(3):109–120, Sept. 2001.
4. L. Liu, C. Pu, and W. Tang. Continual queries for internet scale event-driven information delivery. IEEE Trans. on Knowledge and Data Engineering, 11(4):583–590, Aug. 1999.
5. U. Schreier, H. Pirahesh, R. Agrawal, and C. Mohan. Alert: An architecture for transforming a passive DBMS into an active DBMS. In Proc. of the 1991 Intl. Conf. on Very Large Data Bases, pages 469–478, Sept. 1991.
6. S. Viglas and J. Naughton. Rate-based query optimization for streaming information sources. In Proc of the 2002 ACM SIGMOD Intl. Conf. on Management of Data, June 2002.
7. A. Arasu, B. Babcock, S. Babu, J. McAlister, and J. Widom. Characterizing memory requirements for queries over continuous data streams. In Proc. Of the 2002 ACM Symp. on Principles of Database Systems, June 2002.
8. D. Carney, U. Cetinternel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul, and S. Zdonik. Monitoring streams – a new class of dbms applications. Technical Report CS-02-01, Department of Computer Science, Brown University, Feb. 2002.
9. B. Nguyen, S. Abiteboul, G. Cobena, and M. Preda. Monitoring XML data on the web. In Proc. of the 2001 ACM SIGMOD Intl. Conf. on Management of Data, pages 437–448, May 2001.
10. Aggarwal, Disha; Kumar, Shailender. A Comparative Study of Tuple Timestamped Data Models. International Journal of Advanced Research in Computer Science, [S.l.], v. 8, n. 5, p. 72-78, june 2017. ISSN 0976-5697.
11. M. Sullivan. Tribeca: A stream database manager for network traffic analysis. In Proc. of the 1996 Intl. Conf. on Very Large Data Bases, page 594, Sept. 1996.
12. J. Chen, D. J. DeWitt, F. Tian, and Y. Wang. NiagraCQ: A scalable continuous query system for internet databases. In Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data, pages 379–390, May 2000.
13. S. Madden and M. J. Franklin. Fjording the stream: An architecture for queries over streaming sensor data. In Proc. of the 2002 Intl. Conf. on Data Engineering, Feb. 2002.
14. Guha and N. Koudas. Approximating a data stream for querying and estimation: Algorithms and performance evaluation. In Proc. of the 2002 Intl. Conf. on Data Engineering, 2002.