Technology of Continuous Query Optimization over Data Streams*

FENG Wei-bing[1] LI Zhan-huai[1]
College of Computer Science
Northwestern PolyTechnical University
Xi'an 710072 , China
f_w001@163.com
lzh@co-think.com

Abstract :**The characteristic of data stream is that it has a huge size and its data change continually, which need to be responded quickly, since the times of query is limited. The continuous query and data stream approximate query model are introduced in this paper. Then, the query optimization of data stream and traditional database are compared. Finally, Technology of continuous query optimization over data streams was investigated.**

KeyWords: *streaming data; query optimization ; sliding Window ; continuous query*

1. Introduction

Continuous query and continuous query optimization over data streams have become a hot spot in database research. Series of continuous and orderly data produced in financial information monitoring, network monitoring, security, web applications manufacturing, telecommunications data management, sensor networks, are called data streams[1]. It has the characteristics of great magnitude, changing frequently, limited times for query. Therefore, it is difficult to control the order in which it flows out and it is impossible to save all the data. Thus, its query and storage must be converted from a traditional mode to a new method, namely continuous query and dynamic processing historical data.

There is much difference between the query of data stream and the query of traditional database. On the one hand, the query of data stream is real-time and continuous. It is an initiative process which returns the query result along with the arrival of data. The query is driven by the arrival of data, as is different from the traditional database which is driven by query requirements of users. On the other hand, its query is unpredictable, while the structure of database is known in traditional database, which can be adjusted dynamically by the size of the database and optimization. Nevertheless, the data of data stream is changing continuously. The optimization of query must be processed dynamically and the query results are approximative. It is impossible and unnecessary for the data to be processed precisely.

Though some research on query optimazation has been done [2,3,4,6], it is not mature. Technology of continuous query optimization over data streams was investigated in this paper.

2. Data stream approximate query model

2.1 Data stream query

A streams[1][9][12] S is a bag of elements $<s,t>$, where s is a tuple belonging to the schema of the stream and $t \in T$ is the timestamp of the

* This work has been supported by the National Natural Science Foundation of China (Grant No. 60573096).

978-0-7695-3494-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ISISE.2008.36

element. The query of data stream, which is also called continuous query, is real time query and process of the data stream. It means that the data flowing into the system are queried while the new data detect continuously the query in query system, match the query qualification and return the query results. Compared with traditional query, it is more valuable in practice.

The query of data stream has some similarities with the query provide by RDBMS. However, unlike the traditional database query, the data elements are not acquired by "pull" but by "push". It demands that the query mechanism of data streams have a high processing efficiency, better adaptability , the ability of dealing with outburst flux and load balance.

2.2. Approximate query model

The approximate query model of data steam is shown in Fig.1. when the user send out the query Q, an equivalent query Q' is obtained by query rewriting. The query is conducted by the data stream processing engine. In the process, the data stream processor refresh data with the memory sketch when the data are processed to keep the validity and accuracy of the memory sketch. The query processor obtains query information from the sketch according to the sketch computation rule. The fast approximate query results are finally returned to the user. The memory sketch is usually computed by the reservoir sample, histogram and wavelet method and so on.

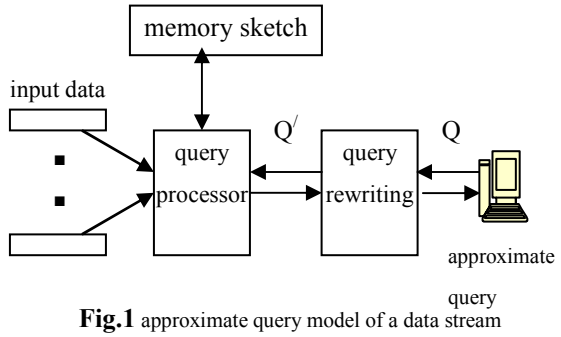

Fig.1 approximate query model of a data stream

However, In the traditional database, all the data will be saved in the medium and the precise query results are obtained by submitting the DML sentence. But because the quantity of the data is magnitude, the real time processing can not be implemented by the traditional database. The properties of a data stream determine that adaptive and approximate query is its vital technology.

3. Difference of the query optimization between data stream and traditional database

Query optimization is extremely important in database system. The optimization based on cost is adopted in traditional RDBMS, in which all parameters, including cardinalities, the access path, the buffer size, distribution of data, are taken into account by the optimizer. The executing costs of all query schemes by a cost model and the scheme of the lowest cost is performed. As the number of tuples is impossible to be known in data streams, the optimization method based on cost is not appropriate for online process of an infinite data stream. For example, in a traditional database, $\sigma_1(\sigma_2(E)) = \sigma_2(\sigma_1(E))$, which means that the query cost of $\sigma_1(\sigma_2(E))$ and $\sigma_2(\sigma_1(E))$ is equivalent. While for a data stream, they have different output rate, as shown in Fig. 2(a), (b). The output rate of $\sigma_2(\sigma_1(E))$ is 1 tuple/sec. and that of $\sigma_1(\sigma_2(E))$ is 10 tuples/sec.. Thus, in the query plan, $\sigma_1(\sigma_2(E))$ is better than $\sigma_2(\sigma_1(E))$.

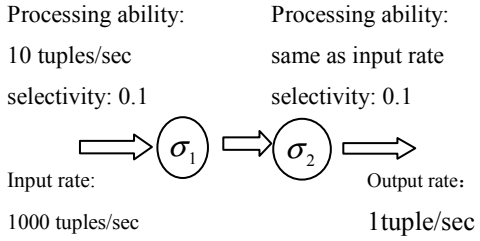

Fig. 2(a) output rate 1tuple/sec.

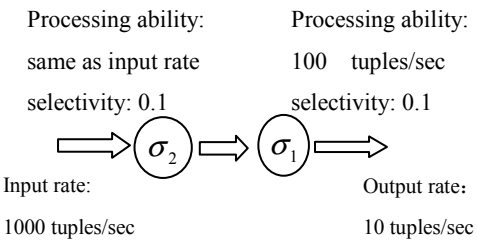

Fig. 2(b) output rate 10 tuples/sec.

The selection operation in traditional RDBMS is usually pushed down and leaf nodes are approached, which doesn't suit a data stream because join operation can not be shared. A strategy for the selection operation is proposed for optimizing continuous query in Ref. [8]. There is much difference in the query optimization methods of a data steam and traditional methods.

A query plan which consumes time as little as possible is selected by RDBMS through meta data in the data dictionary, the properties of relation algebra and system performance. It is static optimization before the query is conducted. Nevertheless, burst flux and abnormal circumstance will occur because of the rate of a data steam and the properties of the data in a data stream management system. The system performance will fluctuate with the variation of the data's properties. So the query plan should be optimized dynamically to adapt the rate of the data stream and the variation of the data.

Measure method of query optimization over data stream

The measure methods of a data stream include: the method based on rate, based on resource consumption,and based on Qos. Ref. [9] presented a measure method based on unit time in which delay and storage capacity are chosen as the measures. S. Viglas proposed an optimization method which is based on rate. The experiment results show that it can maximize the output rate. The algorithm takes into account different transmission rate of different information sources in non-blocking operation. Each operation is expressed by a non-blocking query plan. Given the output rate of a plan $r(t)$ then the number of results the plan will produce at any point in time t_{p_i} is given by the integral of the rate over time[5]:

$$\#Outputs = \int_0^{t_{Pi}} r(t)dt \qquad (1)$$

The query optimization is to find a query plan in the collection P_i which consumes the least time if the output rate $r_{P_i}(t)$ and the output N are presented.

4. Query optimization techniques

4.1. Query Rewriting

Query rewriting is a very useful technique to minimize the cost of the join operation in relation databases. In Ref. [5] some preliminary research in the join order of a data stream is conducted, the output rate is maximized by query rewriting. Ref. [6] did some work in main-memory window join. Query rewriting is introduced in many query languages, such as the exchangeability of the selection and projection operation in a sliding window [10].

4.2. Adaptability [7,4,13]

The adaptability means that the query plan can be changed with the change of query process time, selectivity of predication and arrival rate of the data stream [4]. The original adaptive query plan is the re-estimation of medium query. To further improve the adaptability, instead of maintaining a rigid tree-structured query plan, The Eddies method performs scheduling of each tuple separately by routing it through the operators that make up the query

plan. In fact, the query plan is dynamically re-ordered to match current system conditions. This is accomplished by tuple routing policies that attempt to discover which operators are fast and selective, and those operators are scheduled first. There is, however, an important balance between the resulting adaptivity and the overhead for routing each tuple separately.

4.3. Load shedding

If a burst flux exceed the processing capacity of the system and no corresponding measures are taken, the throughput and response time will deteriorate. Load shedding discards a certain amount of data to guarantee the system performance through sacrificing the accuracy and integrity of the output. It can be devised to two kinds: the random load shedding algorithm and the semantic algorithm.

The random load shedding means that parts of the tuples will be discarded randomly to guarantee the run of the system when the input data exceed the processing capacity of the system. The semantic load shedding discarded a part of tuples selectively to minimize the influence of the tuple loss on the performance and output of the system. It is relevant to the context of the system. The main problem to be dealt with is when, where and how to carry out load shedding. D.Carney, et al. proposed the radom load shedding by dropping some tuples and the semantic load shedding which is implemented by filtering tuples. B.Babcock, M.Datar, et al. pointed out the deficiency of the method proposed by D.Carney. which can not guarantee the accuracy of query. A method to improve the accuracy is presented and a concrete semantic laod shedding algorithm. Load shedding is utilized in both the Stream and Aurora system.

Although sampling method and load shedding can reduce the consumed memory, the error of query is augmented. A query evaluating standard can be established if the accuracy and storage of every two operations are given.

4.4. Materialized View

A more efficient query scheme can be obtained if the query overwriting of a set of materialized views if found in query optimization. The query processing rate can be improved by the computation of the advance materialized view because the computation for the query has been accomplished when the view is queried. It is significant when the view and the query comprise grouping and aggregation operations. The aggregation computation of the views can be reduced because the query is computed by the computed aggregated views. The computation of a query execution plan is correlative with the query computation based on views [11]. The query set Q contains another query set Q′ if for a couple of instances I1 , I2, when the query ability of Q is equivalent, the query ability of Q′ is equivalent

4.5. Schedule of operation

The schedule of operation alters the rate of a data stream, harmonizes operation alterations, query of response time and QoS, load shedding and array storage. Its goal is that for a given query plan and selectivity, minimum total array storage requirements and tuple response time is realized by operating chain-schedule tuples. Meanwhile, the peril of hunger of some tuples should to considered and the storage boundaries can not be exceeded in the operation of the chain scheduling tuples. The tradeoff of the schedule of operation and resources should be carried out. The resources include storage space, computation and I/0. Accuracy is a function of the resources allotted to query operation. The global optimization can be expressed that the inputs are many selective query plans and the output is the selected program which allots the resources appropriately and optimizes the accuracy of query results.

4.6. Other strategies

Query optimization over data stream is a complex problem, needs to consider many aspects such as resources, Qos, input rate of data, the processing rate of operation and so on. Therefore, the following points should be noted in optimization.

① In the continuous query base on increment database, only the changed data are processed. Not all of the information is refreshed.

② In the muti-thread parallel operation, bottlenecks should be reduced to improve the query rate.

③ Similar queries should be merged to avoid computation bottlenecks.

④ Shared computation is important when many queries are carries out in data streams. The expansibility of the system can be achieved by shared computation and multi-query optimization measures. In the NiagaraCQ system, The execution of some queries having same sub-expressions is shared and the performance is effectively improved. Madden et al. expanded the adaptive Eddy structure, in which computation and memories in many queries can be shared.

5. Summary

Query is the most importance operator in database and its optimization is vital for improving the system performance. The continuous query processing is discussed in this paper. Then, the query optimization of data stream and traditional database is compared. Finally, techniques of query optimization over data steam are investigated.

6. Reference

[1] B. Babcock, S. Babu, M. Datar, etal. Models and issues in data streams[C]. In: Proc. ACM Symp on Principles of Database Systems. New York: ACM Press, 2002. 1-16

[2] M.A.Hammad, M.J.Franklin,W.G.Aref, and A.K.Elmagarmid. Scheduling for shared window joins over data streams[C]. In Proc.of VLDB,2003, 297-308

[3] J.Kang,J.F.Naughton,and S.Viglas. Evaluating Window Joins over Unbounded Streams[C]. In Proc.of ICDE,2003, 341-352

[4] S. Madden, M. Shah, J. Hellerstein, V. Raman. Continuously Adaptive Continuous Queries Over Streams[C]. Proc. ACM Int. Conf. on Management of Data, 2002, 49-60

[5] S. Viglas, J. Naughton. Rate-Based Query Optimization for Streaming Information Sources[C] In Proc.of SIGMOD, 2002:37-48.

[6] L. Golab, M. T. Ozsu. Processing Sliding Window Multi-Joins in Continuous Queries over Data Streams[A]. In: Freytag JC, Lockemann PC, Abiteboul S, eds. Proc. of the 29th Int'l Conf. on Very Large Data Bases[C]. Berlin: Morgan Kaufmann Publishers, 2003. 500-511.

[7] R. Avnur, J. Hellerstein. Eddies: Continuously Adaptive Query Processing[C]. In Proc. ACM Int. Conf. on Management of Data, 2000, 261-272.

[8] J.Chen, D.J.DeWitt, and J.F.Naughton. Design and evalution of alternative selection placement strategies in optimizing continuous queries[C]. In Proc.of ICDE, 2002. 345-356.

[9] S. Chandrasekaran, M. J. Franklin. Streaming Queries over Streaming Data[C]. In Proc. Int. Conf. On Very Large Data Bases, 2002, 203-214.

[10] A. Arasu, S. Babu, J. Widom. An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations[C/OL]. Technical Report, S~nford University Database Group 2002(2006-11-10). http:// dbpubs.stanford.edu:8090/pub/2002-57.

[11] Chen Xin, Song Han Tao. Based on data stream approximate query computation and applied research [J]. Application research of Computer, 2003(11):113-116

[12] Liu Jing chun, Wang Yongli. Data stream processing technology [J].Journal of Jiamusi University: Natural sciences version .2004, 22(4):556-558.

[13] V. Raman, A. Deshpande, J. M Hellerstein. Using state modules for adaptive query processing[A]. In: Proc. 19th Int' l Conf.Data Engineering(ICDE)[C]. Los Alamitos: IEEE Computer Society Press, 2003. 353-364