Research on View Incremental Calculation Method in Information Integration Services

Xufang Li[1,2, a], Wei Liang[3, b]

[1]School of Management, Shanghai University of Engineering Science, Shanghai, 201620, China

[2]Shanghai Key Laboratory of Data Science, Fudan University, Shanghai, 201203, China

[3]Shenzhen audio-visual education center, Shenzhen 518008, China

[a]email: lucylxf@163.com, [b]email:sues619@163.com

Keywords: Information Integration; Materialized View; View Maintenance; View Incremental Calculation

Abstract. Using incremental calculation methods on materialized views maintenance can be beneficial for quick access to views, query optimization and so on in the digital library. This paper presents a new method of view incremental calculation named PCA. And this method can efficiently compensate for the error caused by deferred calculation using partial compensation. It compensates only once for the data of every base table. It's time complexity is only $O(n^2)$ verified by the experiment.

Introduction

Information integration service is current research hotspot, and view incremental calculation is the key point of integrated information online maintenance and real view application. The incremental calculation methods [1, 2, 3] usually calculate increment over a period of time. These methods use the base tables involved in the view definition at some moment and the increment of the base tables in computation time to calculate view increment. Base table increment can be obtained by reading the database transaction log or snapshot differential algorithm[7]. These methods have some problems: firstly, the incremental calculation must be in sync with the computing time. In other words, the calculation must be carried out at the end of the period of time, no delay. However many applications (such as enterprise decision analysis) usually requires lagging calculation. And obtaining the increment by reading the database transaction log or snapshot differential calculation should take time. Secondly if there is a change in the process of base table calculation, calculation result error will occur. Therefore, we can use the asynchronous incremental calculation methods, which the view incremental calculation for a period of time interval can be carried out after the interval. These methods can flexibly deal with the materialized view lagging behind the data source. And they are very useful for enterprise decision analysis, data mining and real-time applications, such as securities trading.

Rolling join propagation algorithm (RJP) is one of asynchronous incremental methods[4]. RJP algorithm uses the compensation method to implement the asynchronous incremental calculation. But it may introduce new errors to compensate false errors, and may also execute multiple compensation leading to time complexity be O (n!) .

This paper presents a new asynchronous incremental calculation algorithm, which is PCA algorithm (part compensation algorithm, PCA). The characteristics of the algorithm is to view incremental calculation, whenever the query a base table for computing the required data, namely to compensate this part of the data, the data of each base table just compensation time, its time complexity is O (n^2).

Relevant definition and assumption

Database is a collection of the tables, and the table is made up of many tuples. Database data changes caused by update transactions, which include inserting, deleting and updating the tuples. In

this paper, the PCA algorithm used incremental query to calculate view increment. An incremental query of view V has the same form as the definition of V, just one or more base tables in V are replaced with the corresponding increment table. For example, if the definition of V has the following: $R^1R^2...$ R^n, and its corresponding incremental query can be expressed as: R^1 $R^2{}_{a,b}...$ R^n. Usually, the view increment is a combination of several such incremental query results. In this paper, Q^V is an incremental query, and $Q^V[i]$ is a base table R^i or an incremental table $R^i{}_{a,b}$.

Part compensation algorithm

Part compensation algorithm (PCA) is proposed in this paper. The main improvement of PCA is that the data is compensated immediately every time obtaining required data by querying a base table for calculating.

PCA algorithm description is given below.

input: Q is a query defined on n base tables, t_{old} is start time of the time interval, t_{new} is end time of the time interval

output: The increment $Q_{old,new}$ of Q from t_{old} to t_{new}

step 1: Let $Q_{old,new}$ is NULL. For each base table R^k ($1 \le k \le n$) in V repeat the following step 2 to step 8.

step 2: Dealing with the left of R^k. Let $Q^V[k][k]= R^k{}_{old,new}$, and j is from k-1 to 1. Repeat step 3 and step 4.

step 3: Executing $R^jQ^V[k][j+1]$. t_{exec} is for execution time, and assign the execution result to $Q^V[k][j]$.

step 4: Compensate for $Q^V[k][j]$. $Q^V[k][j]= Q^V[k][j]- R^j{}_{old,exec}Q^V[k][j+1]$.

step 5: Dealing with the right of R^k. Let $Q^V[k][k]= Q^V[k][1]$, and j is from k+1 to n. Repeat step 6 and step 7.

step 6: Executing $Q^V[k][j-1]$ R^j. t_{exec} is for execution time, and assign the execution result to $Q^V[k][j]$.

step 7: Compensate for $Q^V[k][j]$. $Q^V[k][j]= Q^V[k][j]- Q^V[k][j-1]$ $R^j{}_{new,exec}$

step 8: $Q^V[k][n]$ is assigned to $Q^V[k]$, and add $Q^V[k]$ to $Q_{old,new}$. Namely $Q_{old,new}=Q_{old,new}+ Q^V[k]$.

step 9: output $Q_{old,new}$.

Performance analysis

Time complexity

In PCA algorithm, increment query operations of step 3 and step 6 are basic operations. For a base table R^k($1 \le k \le n$) in the view definition, the correspond incremental table is $R^k{}_{a,b}$(time from t_a to t_b is the interval for calculation). When calculating the left of R^k, step 3 needs to execute k-1 times. When calculating the right of R^k, step 6 needs to execute n-k times. Therefore, the basic operations of step 3 and step 6 need to execute n-1 times. If the value of k is from 1 to n, the basic operations of step 3 and step 6 need to execute n*(n-1) times. So the time complexity of PCA algorithm is $O(n^2)$.

The number of incremental tuples

As mentioned earlier, for compensating calculation error we need to add additional incremental tuples to the results. And the more incremental tuples, the more the time needed to complete the calculation.

For the convenience of description, here introduce a few parameters as follows.

C: The number of tuples participating in connection. The tuples is in the corresponding increment tables to base tables. For the convenience of comparison, it is assumed to be constant here.

J: Connecting fator. A connecting factor $J(R^i, a)$ is the number of tuples when the attribute a of R^i equal to a fixed value. For the convenience of comparison, we assume that the connecting factor J is a constant which not changes with the base table.

In RJP algorithm, each incremental query will generate CJ^{n-1} incremental tuples. However, in PCA algorithm, each incremental query will generate 2CJ incremental tuples. There are $nC(2J)^{n-1}$ incremental tuples in total in PCA algorithm.

Test results

The paper simulated calculating one day increment of materialized view a_toy_sales by using RJP and PCA respectively.

Assumed that there is a database to store sales data of a chain store, and the database has the following base table:

store(store_id, city, province, manager)

sale(sale_id, store_id,day, month, year)

line(line_id, sale_id, item_id, sales_price)

item(item_id, item_name, category, supplier_name)

In each above base table, the first attribute is the primary key of the base table. Base table “store” contains the address of each chain store and information of the sales manager; base table “sale” recorded the address and time of each transaction; each transaction may involve several items, so in each row in base table “line” represents commodity information of a deal; base table “item” contains all inventory information of each chain store.

For the convenience of observation, we assume that all of the base tables are empty at 1 o 'clock and the materialized view a_toy_sales also is empty. In one day, a new chain open, and a batch of toy cars are purchased. At the same time, there were two businesses to buy the toy cars. So in the day, a tuple is inserted into base table “store” and base table “item”, two tuples are inserted into base table “sale” and base table “line”. Two tuples are inserted into materialized view a_toy_sales. For each case we have repeated the experiment many times and got average value, to guarantee the accuracy of the results as far as possible.

Table 1 Base tables did not change in the process of calculation

	Running Time	Number of Increment tuples
RJP	1734ms	30
PCA	1469ms	30

Table 2 Base tables changed in the process of calculation

	Running Time	Number of Increment tuples
RJP	2297ms	114
PCA	1516ms	54

The comparison results of RJP and PCA are shown in table1 when base tables did not change in the process of calculation. The comparison results of RJP and PCA are shown in table2 when base tables changed in the process of calculation. From the above two tables, RJP algorithm is affected by base table change more than PCA algorithm.

Table 3 Running Time of RJP and PCA in processing several base tables

Number / Algorithm	5	6	7
RJP	7128ms	47187ms	705984ms
PCA	2422ms	7765ms	42140ms

As shown in table 3, along with the increase in the number of base table, although PCA algorithm running time also obviously increase, but far less than RJP algorithm.

The experimental results show that PCA algorithm is superior to RJP algorithms, especially in situations where the number of base table for calculating is more.

Conclusion

Information integration services can be more reasonably and effectively organize, storage, retrieval and access to all kinds of digital information, and make the information being spread and used on the Internet. Using the methods of materialized view incremental maintenance can quickly access the data source view, and optimize the query process.

Aiming at the problems, such as existing view incremental calculation method cannot delay calculation, calculation is not accurate and time complexity is high, this paper puts forward a new incremental calculation method called PCA algorithm. The algorithm use part compensation method to effectively compensate errors caused by delay calculation, and it's time complexity is O (n^2).

The paper compared RJP algorithm with PCA algorithm in aspect of time complexity, the number of increment tuples, and running time. The experimental results show that PCA algorithm is superior to RJP algorithms, especially in situations where the number of base table for calculating is more.

Acknowledgement

In this paper, the research work was financially supported by Humanities and Social Science Fund of Ministry of Education (13YJCZH122) and Course Construction of SUES (Z20140303, P201403002).

References

[1] L S Colby，T Griffin，L Libkin，I S Mumick. Algorithms for deferred view maintenance.The ACM SIGMOD Int'l Conf on Management of Data，1996：469-480.

[2] A Gupta，I S Mumick，V S Subrahmanian. Maintaining views incrementally.The ACM SIGMOD Int'l Conf on Management of Data，1993：157-167.

[3] JOSE A.，Blakeley,Per-Ake Larson，Frank Wm. Tompa. Efficiently materialized views. In Proeeding of the ACM SIGMOD International Conference on Management of Data,1986：61-71.

[4] J. L. Wiener. "What is data warehousing and what is Stanford doing about it?" An overview talk given in the Stanford DB Seminar series, Fall, 1997.

[5] Zou Xianxia, etc. Delay Part Compensation Algorithm of view increment calculation [J]. Computer Integrated Manufacturing Systems, 2011,17(5):1024-1031.

[6] Timothy Griffin, Leonid Libkin, and Howard Trickey. An improved algorithm for the incremental recomputation of active relational expressions [J]. IEEE Transactions on Knowledge and Data Engineering, 1997,3(9):508-511.

[7] W. J. Labio and H. Garcia-Molina. "Efficient Snapshot Differential Algorithms for Data Warehousing." In Proceedings of VLDB Conference, Bombay, India, September 1996: 63-74.

[8] Kang Chunpeng. Library portal design in information integration service mode [J]. Journal of Modern Information,2011,31(4): 33-34,39.

[9] Yang Rui. Digital library information integration service in the web2.0 environment [J].Journal of Intelligence, 2010,29(z1):327-329.

[10]P. A. Bernstein, L. M. Haas. Information integration in the enterprise [J].Communications of the ACM, 2008,51(9):72-79.