A Preemptive Batching Schedule Method Based on Multifactor in DSMS

Lijie Zhang1

Abstract. We propose a preemptive batching schedule method based on multifactor over data stream. We divide operators sequence into some different operator units in my schedule approach, the different priorities, which are changed as operator units' output rate, are assigned based on operator units' output rate. We adjust the query plan according to the altered priority, and adopt preemptive batching schedule method which not only improve the system's performance, but also decrease the waiting time of tuples and improve the output rate of tuples greatly.

Keywords: Data stream · Rate batching · Preemptive

1.1 Introduction

In a growing number of information processing applications, data takes the form of continuous data streams rather than traditional stored databases. Examples include stock ticks in financial applications, performance measurements in network monitoring and traffic management, log records or click-streams in Web tracking and personalization, data feeds from sensor applications, network packets and messages in firewall-based security, call detail records in telecommunications, and so on. However, their continuous arrival in multiple, rapid, time-varying, possibly unpredictable and unbounded streams appears to yield some fundamentally new research problems [1,2]. Systems that seek to give rapid or real-time query responses in such an environment must be prepared to deal gracefully with bursts in data arrival without compromising system performance.

Aim at several variation factors about environment and data stream itself, such as date stream rate, process cost of operator, memory usage of data stream management system (DSMS), in order to improve query performance, a preemptive batching re-optimized method based on multi-factors is generated. This method divides operators into some different operator units. The different priorities, which

[1] Lijie. Zhang (✉)
School of Software, Dalian International Studies University, Dalian, China
e-mail: lijie8111@126.com

are changed as operator units' output rate, are assigned based on operator units' output rate. We adjust the query plan according to the altered priority, and adopt preemptive schedule, which improves efficiency of continuous query. Experiments have proved that this method improve system overall performance, meanwhile validly increases output rate of tuples and reduces waiting time latency.

1.2 Related Work and Concept

1.2.1 Related Work

Generally speaking, query process cost of data stream system contains four parts below: (1)cost to call scheduler.(2)cost to call operators.(3) cost to load /unload data queue.(4)cost to execute operators.

At present, many studies had developed about re-optimized method over data stream. For example, a reference proposes an optimized schedule strategy based on rate [2]. The operators' box and proposes operators batching is defined in references [3]. A reference proposes a preemptive schedule method about operators [4]. A detail task definition of scheduler in DSMS is proposed in reference [5].

On the base of tupels output rate, aiming at improve overall performance and shorten waiting time of tuples, we try to batching schedule operators, preemptive execute operator unit with lower output rate and re-optimize query plan.

In a growing number of information processing applications, data takes the form of continuous data streams rather than traditional stored databases. Examples include stock ticks in financial applications, performance measurements in network monitoring and traffic management, log records or click-streams in Web tracking and personalization, data feeds from sensor applications, network packets and messages in firewall-based security, call detail records in telecommunications, and so on. However, their continuous arrival in multiple, rapid, time-varying, possibly unpredictable and unbounded streams appears to yield some fundamentally new research problems. Systems that seek to give rapid or real-time query responses in such an environment must be prepared to deal gracefully with bursts in data arrival without compromising system performance.

1.2.2Ralated concepts

Data Stream: streaming date model based on relational schema R is <R, T>, where T is the time field (logical time or physical Time) in data stream system. Time granularity is usually defined by user, the minimum granularity is second. The streaming data element can be expressed as <r, τ>. Where r is a tuple in data stream, τ represents timestamp. Data stream can be regarded as a data set that is composed of streaming data elements in order. At τ_i time point, data stream can be described: $S(\tau_i) = \{\langle r, \tau \rangle | r \in R \wedge \tau, \tau_i \in T \wedge \tau \le \tau_i\}$. Data stream discussed in this paper take the form of append-only style: $\forall \tau \le \tau_i : S(\tau) \subseteq S(\tau_i)$.

Operator' Selectivity: the count of tuples through an operator to a tuple. For monocular operator (projection or selection), operator' selectivity $S_i = n_{out} / n_{in}$, n_{in} is the count of tuples into the operator O_i, n_{out} is the count of tuples out from the operator O_i. For binocular operator (join), it will be more complex than projection and selection because it has two inputs. Operator' selectivity $S_i = n_{out} / n_l \times n_r$, n_l and n_r are the count of tuples into left branch and right branch of operator O_i.

Operate Unit: operator set scheduled batched by scheduler, that is U= $\{O_1, O_2, O_3 \ldots, O_i, \ldots\}$.

Execute Cost of Operation Unit: firstly, the time of performing an operation on an input object, that is the cost of O_i is C_i. Secondly, the cost sum of all operators in an operation unit. Operation unit U contains N operators, then the cost of N, that is $E(U) = \sum_{i=1}^{N} C_i$ (this formula is not adapted to operation unit including join operator.

Input and Output Rate: the input and output tuple count in defined time. Output rate (Out (U)): Out (U) =output tuples/the time cost to finish output for an operation unit U.Input rate (I): I is the count of arrival tuples in defined time.

Tuple'Waiting Time: $W_i = D_i - A_i$, W_i is the waiting time to tupe , D_i is the leaving time to tuple i, A_i is the arriving time to tupe . The average waiting time to N tuples is $\frac{1}{N}\sum_{i=1}^{N} W_i$.

1.3 Cost Estimate Based on Output Rate

Given that the data stream input rate is I, we can think a tuple arrives every 1/I time unit. All the formula of this paper is applicable to this premise, that is1/I<E(U). in this case, The cost of performing one operation is greater than the inter-arrival time for input objects, a lot of tuples can't be operated immediately. Therefore the middle queues are needed to set up to store temporary tuples which are not processed.

1.3.1 Monocular Operator

For monocular operator (projection or selection), computing output rate is relative easy. Given that C_σ and C_π mean the execute cost of selection operator and projection operator, S_σ and S_π mean the selectivity separately, n is the tuple count of input queue, Set operators, the number of tuples in the input queue for, according to the above definition, output rate of selection operator and projection operator can be expressed as respectively $Out(O_\sigma) = \frac{n \times s_\sigma}{n \times c_\sigma}$, $Out(O_\pi) = \frac{n \times s_\pi}{n \times c_\pi}$,that is $Out(O_\sigma) = \frac{s_\sigma}{c_\sigma}$, $Out(O_\pi) = \frac{s_\pi}{c_\pi}$.

To sum up, the unified expression of the output rate about the selection and projection operations is $Out(O\ ) = \frac{S}{C}$.

Given that operation unit consists of selection and projection operation , that is U={O$_1$,O$_2$,…,O$_i$}, among this formula, O$_i$ is the input operator of U, O$_i$ is the output operator of U, so the output rate of U is: $Out(U) = \frac{S_1 \times S_2 \times ... \times S_i}{C_1 + S_1 \times C_2 + ... + S_1 \times S_2 \times ... \times S_{i-1} \times C_i}$.

1.3.2 Binocular Operator

Assuming input rates I_l and I_r are the two input streams, the C is the execute cost of operator O, S is the selectivity of join operator. Then the numbers of result objects from these inputs in the first time unit will be $S \times I_l \times I_r$. Now consider the second time unit. At the end of this second time unit, the total number of output elements generated by arrivals during the second time unit will then be $2 \times 2 \times S \times I_l \times I_r - S \times I_l \times I_r$, that is $3 \times S \times I_l \times I_r$. Using the same logic, the total number of output elements generated by arrivals during the third time unit will then be $5 \times S \times I_l \times I_r$. So the total number of output elements generated by arrivals during the tth time unit will be shown as follows:
$S \times I_l \times I_r + 3 \times S \times I_l \times I_r + 5 \times S \times I_l \times I_r + 7 \times S$
$\times I_l \times I_r + ... + (2t-1) \times S \times I_l \times I_r = S \times I_l \times I_r \times t^2$

The total execute cost of join operator is $t \times (I_l \times \frac{C}{2} + I_r \times \frac{C}{2})$, so the output rate of operator O is $Out(O\ ) = \frac{S \times I_l \times I_r \times t^2}{t \times (I_l \times \frac{C}{2} + I_r \times \frac{C}{2})} = \frac{2 \times S \times I_l \times I_r \times t}{C \times (I_l + I_r)}$.

Given that operator O_l and O_r is the left child and right child of join operator, the C_l and C_r is the execute cost of O_l and O_r ,the S_l and S_r is the selectivity of O_l and O_r .so $I_l = Out(O_l) = \frac{S_l}{C_l}$, $I_r = Out(O_r) = \frac{S_r}{C_r}$., according above formula , we can deduces that: $Out(U) = \frac{2 \times S \times S_l \times S_r \times t}{C \times (S_l \times C_r + S_r \times C_l)}$.

From this formula, we can get the conclusion that the output rate of operation unit with join operator is increasing through the time.

1.4 A Preemptive Batching Schedule Method

1.4.1 Operation Unit

We adopt batch processing method to schedule operators, because (1) using operation unit contains a lot of operators can reduce switch cost to schedule different operators. (2) Beach distribution memory space to operation unit can reduce cost to manage memory. So we define two types of operation unit to execute a query plan.

Binocular Operator Unit (BOU): left and right child of join unit are consist of the binocular operator unit, for example {2,3,4} in Fig. 1. Join operation on two relations can be seen as a first do generalized cartesian product, again according to the connection conditions for the choice of operation, so the generalized cartesian also can be regarded as a join operation without condition.

Monocular Operator Unit (MOU):MOU contains projection and selection operators in the same path, for example {10, 11} in Fig. 1.

According to the classification of the operation unit method, a query tree in the Fig.1 can be divided into BOU :{2,3,4},{6,7,8}, MOU :{1},{5},{9},{10,11}.

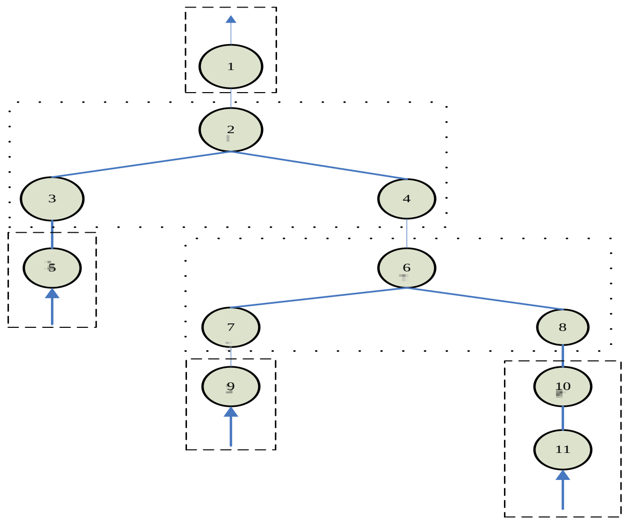

Fig. 1 Query Tree

1.4.2 Operating Unit for Batching Strategy of Tuples

Scheduler distributes the corresponding queue in each input of operating unit. Input queue size can be set according to the input flow rate. We will deal with a collection of tuples one-time called a Bus, The size of a Bus is $Size(Bus)$ defined as being processed on the number of tuples with the ratio of the total number of tuples in the input queue.

According to the memory overhead of the current system with the response time requirements, scheduler can define appropriated the value of $Size(Bus)$. Due to space limitations, this problem will be discussed further in future research.

1.4.3 The Preemptive Scheduling of Query Plan within The Operating Unit

According to the above classification method of operation unit, we can divide the query tree in Fig.1 into several operating units based on the operating unit output rate method for scheduling, besides, we introduces "BOU can grab MOU and MOU cannot preempt the BOU" principles of preemption. We call this scheduling method based on the rate of preemptive batching method (PRB), as shown in algorithm 1.

Algorithm 1: preemptive batch process operation unit
Input: SQL query
Output: Stream query result
Steps:

Step1: Convert the registered SQL query into a query tree, then divided it into several MOU and BOU according to the classification method of operation unit.

Step2: Order separately the operation unit of BOU set $V(BOU)$ and $V(MOU)$ to fulfill that: $V(BOU)$ = { $U_1, U_2, U_3 ...$ } , $U_1, U_2, U_3 ...$ is operation unit, and $Out(U_1) \geq Out(U_2) \geq Out(U_3) \geq ...$ .meanwhile, $V(MOU)$ ={ $U_1, U_2, U_3 ...$ }, $U_1, U_2, U_3 ...$ is operation unit, and $Out(U_1) \geq Out(U_2) \geq Out(U_3) \geq ...$.

Step3: In the initial phase of the query began, according the above formula, we get that $Out(MOU) > Out(BOU)$,so scheduler will select the first MOU to execute form the $V(MOU)$ .by parity of reasoning, scheduler will select the second MOU to execute form the $V(MOU)$.

Step4: With the passage of time, the value of $Out(BOU)$ is also increasing, BOU grabs MOU when $Out(BOU)$ of the first BOU in $V(BOU)$ greater than $Out(MOU)$ are being executed.

Step5: Parameter t will be cleared when b is the end of execution. Then scheduler will compare the $Out(BOU)$ of the second JOU in $V(BOU)$ with $Out(MOU)$ of grabed MOU, if $Out(BOU) > Out(MOU)$, then the second BOU will be executed, or else, grabbed MOU will go on executing.

Step6: When present MOU completes, the next MOU in $V(MOU)$ will be executed if any one of the BOU' $Out(BOU)$ in $V(BOU)$ are no more than the next MOU in $V(MOU)$ .The rest circular schedule about operation in $V(BOU)$ and $V(MOU)$ can be done in the same manner until query is stopped by user.

1.5 Performance Analysis

Performance analysis is done to test the benefit of the preemptive batch method in schedule process. In order to better demonstration effect, we introduced the concept of distribution: $f(x) = \frac{1}{\ln m} x^{-1}$, m is the parameter of the Zipf. Zipf parameter value is greater, the greater the degree of difference between the values of the generated random. In these experiments, the selectivity of operators is 1, the execute cost meet the Zipf distribution. We analyze capability according changing the Zipf parameter value. Here are some of the other scheduling policies.

FIFO: Operating units did not finish to a tuple, and then to the tuple is not processed.

RR: Expired when the time slice assigned to a single operating unit, the operating unit is preempted, the next operation unit is scheduled.

RB: When a high output rate MOU is running, a higher output rate BOU must wait.

From Fig.2, we can find that the performance of the FIFO is superior to the RR when the Zipf parameter is small, namely the difference of the execution cost of each operator is not big. While, with the Zipf parameter increase, preemptive RR will be superior to the FIFO. RB and PRB based on the input rate has been better than RR and FIFO, the which the preemptive PRB own the best performance.

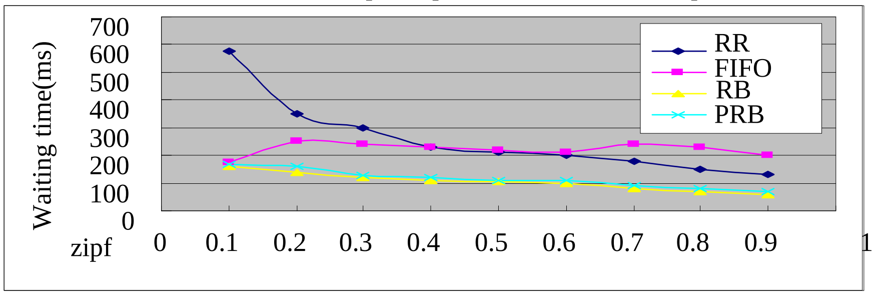

Fig. 2 Waiting time Comparison between schedule strategies

This paper emphasizes the operator scheduling strategy based on multifactor, adopts batch processing of operator, preemptive schedules operation unit, and finally proposes a pre-emptive batching schedule method based on multifactor. This method can not only maximize output rate of tuples, and greatly reduce the system overhead. Experiments show that this method has good performance.

1.6 References

1. B. Babcock, S. Babu, M. Datar, et al. Models and issues in data stream systems. In Proc. 21st ACM SIGACT-SIGMODSIGART Symp. on Principles of Database Systems, pages 1, Madison,Wisconsin, May 2002.
2. D J Abadi, D Carney, et al., Aurora: a new model and architecture for data stream management, The VLDB Journal, 2003, 12(2): 120.
3. D. Stratis, F. Viglas Jeffrey, Naughton: Rate-Based Query Optimization for Streaming Information Sources. Department of Computer Sciences, University of Wisconsin-Madison, 1210 W Dayton St. Madison, WI, 53706, USA
4. B. Brian, B. Shivnath, and D. Mayur et al. Operator Scheduling in Data Stream Systems, June 2004.
5. Mohamed A. Sharaf, Panos K. Chrysanthis, Alexandros Labrinidis:Preemptive Rate-based Operator Scheduling in a Data Stream Management System ,Department of Computer Science,University of Pittsburgh,Pittsburgh, PA 15260, USA