Load Shedding for Aggregation Queries over Data Streams

V. Mahesh Kumar

04305906
Computer Science and Engineering

Authors: Brian Babcock , Mayur Datar and Rajeev Motwani
Proceedings of the 20th International Conference on Data Engineering (ICDE04)

logo

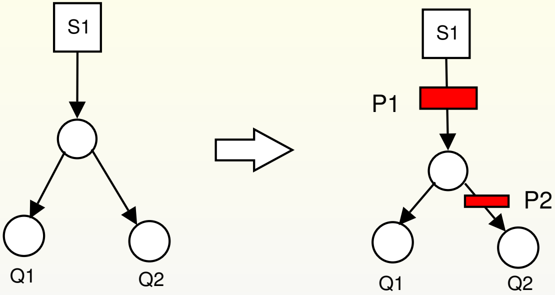
Introduction

- This paper is related to STREAM paper discussed earlier in the course.
- A continuous data stream S is a potentially unbounded sequence of tuples that arrive over time.
- Continuous monitoring queries over data streams.

- Many sources of Streaming data are quite bursty.
  - Spikes in Traffic experienced by news websites and telephone networks on sept 11, 2001.
  - Spikes in Traffic at a corporate web following the announcement of New Product.

- Handling of peak load is generally impractical.
- Systems processing continuous monitoring queries over data streams must be able to adaptive
- As overloaded system will be unable to process all of its input data. So **Load shedding i.e discarding some fraction of the unprocessed data** becomes necessary.
- The Question we study is which tuples to drop, and where in the query plan to drop them, so that the inaccuracy in query answers introduced as a result of load shedding is minimized.

Overview of the Approach



- Introduction of Load shedders
- Each load shedder is parameterized by a sampling rate p.
- Compensation of the lost tuples.
- The above decisions are based on the statistics of the data streams and operators.

Problem Formulation

Sliding Window queries.

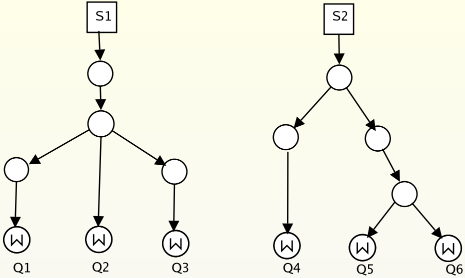
- Time-based.
- Tuple-based.

In this paper

- Sliding Window aggregate queries with possibly foreign key joins with stored relations, over continuous data streams are only considered.
- No joins between different Data streams.
- The aggregate functions considered are SUM and COUNT.
- Sharing of common subexpressions .

The input to the load shedding problem consists of

- A set of queries $q_1, q_2 \ldots q_n$
- Data streams $S_1 \ldots S_m$
- A set of Query operators $O_1 \ldots O_k$
- Some associated statistics.



DATA FLOW DIAGRAM

- The operators are arranged into a data flow diagram
- As joins between streams are not considered, dfd consists of trees.
- The query path for Query Q_i
- $T(S_j)$ denote the tree of operators st stream source S_j

Associated Statistics with input

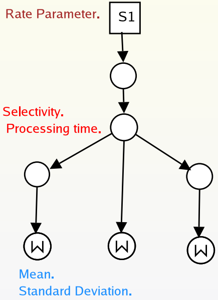
Every operator consists of

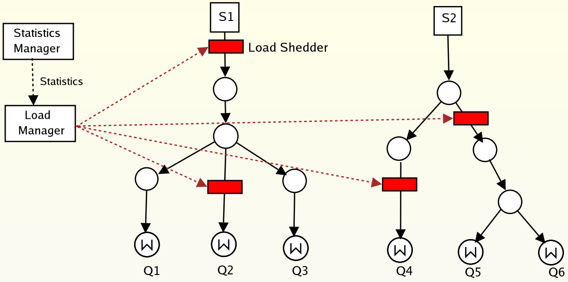
- selectivity s_i
- Processing time per tuple t_i

Every SUM aggregate operator is associated with two additional parameters.

- Mean μ_i
- Standard deviation. σ_i

The final parameters are the rate parameters r_j, i.e average rate of tuple arrival , one for each data stream S_j, measured in tuples per unit time.





DATA FLOW DIAGRAM

- STREAM data stream management system, had a Statistics Manager module that estimates the parameters.
- As Steam arrival rate and data characteristics change, the appropriate amount of load to shed and the right place to shed it may change.
- Statistics Manager estimates are periodically refreshed, and load shedding decisions are periodically revisited.

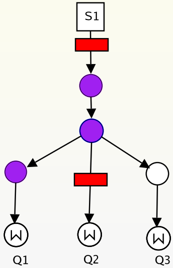
Accuracy Metric

- Let $\widehat{A_1} \ldots \widehat{A_n}$ be the answers to queries by the system to $q_1 \ldots q_n$, at some point, and $A_1 \ldots A_n$ be the actual answers.
- The relative error is more important then absolute magnitude.
- For query i relative error is $\epsilon_i = |A_i - \widehat{A_i}|/|A_i|$
- For Multiple queries the aim is to minimize maximum error across all queries, $\epsilon_{max} = max_{1 \leqslant i \leqslant n} \epsilon_i$

Load Equation

- The rate at which tuples are processed must be as high as the rate at which new tuples are arriving on the data streams.
- U_i denote the *Upstream* operators of O_i
- Let p_i be the sampling rate of load shedder introduced immediately before O_i
- The effective input rate for O_i
  $r(O_i) = r_{src(i)} p_i \prod_{O_x \in U_i} s_x p_x$
- Load Equation

$$\sum_{1 \leqslant i \leqslant k} \left(t_i r_{src(i)} p_i \prod_{O_x \in U_i} s_x p_x \right) \leqslant 1$$



DATA FLOW DIAGRAM

Problem Formal Statement

Formal Statement

Given a data flow diagram, the parameters $s_i, t_i, \mu_i, \sigma_i$ for each operator O_i, and the rate parameters r_j for each data stream S_j , select load shedding sampling rates p_i to minimize the maximum relative error $\epsilon_{max} = max_{1 \leqslant i \leqslant n} \epsilon_i$, subject to the constraint that the load equation,must be satisfied.

Load shedding Algorithm

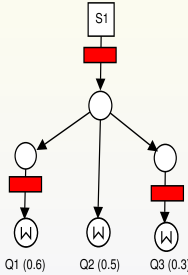
The algorithm has two steps.

Step 1

Determine the effective sampling rates for each query that will distribute error evenly among all queries.

Step 2

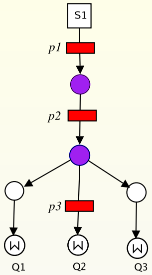
Determine where in the data flow diagram load shedding should be performed to achieve the appropriate rates and satisfy the load equation.



Step 1

- It is impossible to precisely predict the relative error.
- We can only get estimate of relative error using probabilistic techniques.
- The approach to compare alternate load shedding policies is as follows.
- For a fixed small quantity δ (say 0.01), load shedding policy achieves error ϵ if,
  for each query q_i , the relative error resulting from using the policy to estimate the answer to q_i exceeds ϵ with probability at most δ.
- $$Pr\left(\frac{|\widehat{A}_i - A_i|}{|A_i|} \geqslant \epsilon\right) \leqslant \delta$$

- Effective sampling rate P_i for query Q_i (eg: $P_2 = p_1 p_2 p_3$)
- Let N_i denote the Number of tuples in the CSW that would pass all selection conditions.
- Let $v_1, v_2 \ldots v_{N_i}$ denote the values of the attribute summed, and let A_i be their sum.
- The appx answer $\widehat{A}_i$ of system will be sum of $v_i s$ for the tuples that get included in the sample, scaled by the inverse of the effective sampling rate $(1/P_i)$.



DATA FLOW DIAGRAM

Hoeffeding Theorem Result

Let $X_1, X_2 \ldots X_N$ be N random variables, such that each random variable X_j takes the value v_j/p with probability P and the value zero otherwise. Let $\widehat{A}_i$ be the sum of these random variables and let $A_i = \sum_{j=1}^{N} v_j$.If we denote by SS_i the sum $\sum_{j=1}^{N} v_j^2$, then

$$Pr\{|\widehat{A}_i - A_i| \geqslant \epsilon |A_i|\} \leqslant 2e^{(-2P^2 \epsilon^2 A_i^2 / SS_i)}$$

- Thus $2e^{(-2P^2\epsilon^2A_i^2/SS_i)} \leqslant \delta$,
  which occurs when $P_i\epsilon_i \geqslant C_i$, where $C_i = \sqrt{\frac{SS_i}{2A_i^2}\log\frac{2}{\delta}}$
- The ratio $SS_i/2A_i^2$ is equal to $(\sigma_i^2+\mu_i^2)/(N_i\mu_i^2)$. Thus
  $C_i = \sqrt{\frac{(\sigma_i^2+\mu_i^2)}{(N_i\mu_i^2)}\log\frac{2}{\delta}}$
- For a load shedding policy to achieve relative error ϵ_i, we must guarantee that $P_i \geqslant C_i/\epsilon_i$
- To set P_i correctly, we need to estimate C_i.
- C_i is larger for queries that are more selective, for queries over smaller sliding windows,more skewed attributes.
- Count Aggregate.

- The objective that we seek to minimize is the maximum relative errorϵ_i across all queries q_i.

Observation 1

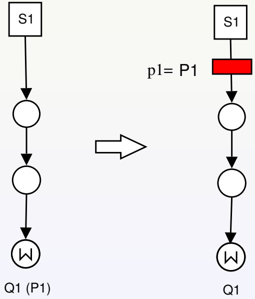
In the optimal solution, the relative error ϵ_i is equal for all queries.

- $P_i = C_i/\epsilon_i = C_i/\epsilon_{max}$
- The problem is reduced to determining best achievable ϵ_{max} and inserting load shedders such that eff. sampling rate P_i is equal to C_i/ϵ_{max}

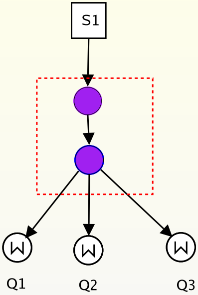
Step 2 : Placement of Load shedders

- Given a dfd along with a set of target effective sampling rates P_i for each query q_i
- Modify the dfd by inserting the load shedding operators
- Set their sampling rates such that total processing time is minimized

- If No sharing of operators.



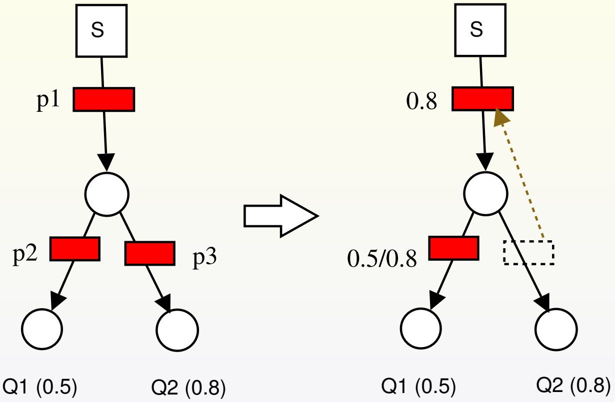
- Sharing among query plans.
- *Shared Segment*
- *branch point, parent segment, child segments*

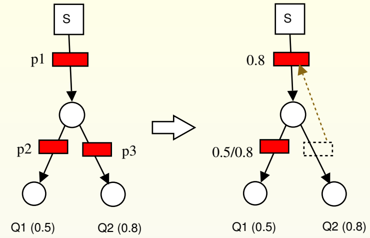


Observation 2

In the optimal solution, load shedding is performed at the start of the shared segments

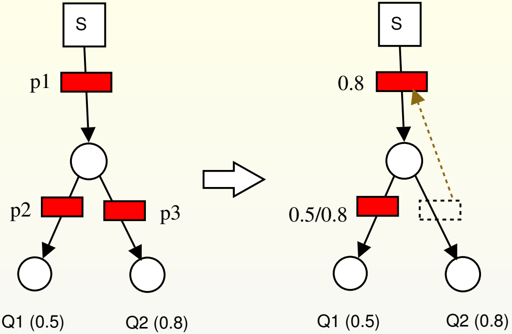
Example





Observation 3

Let q_{max} be the query that has the highest effective sampling rate among all queries sharing the parent segment of a branch point B. In the optimal solution, the child segment of B that lies on the query path for q_{max} will not contain a load shedder. All other child segments of B will contain a load shedder with sampling rate P_{child}/P_{max}, where q_{child} is defined for each child segment as the query with the highest effective sampling rate among the queries sharing that child segment.



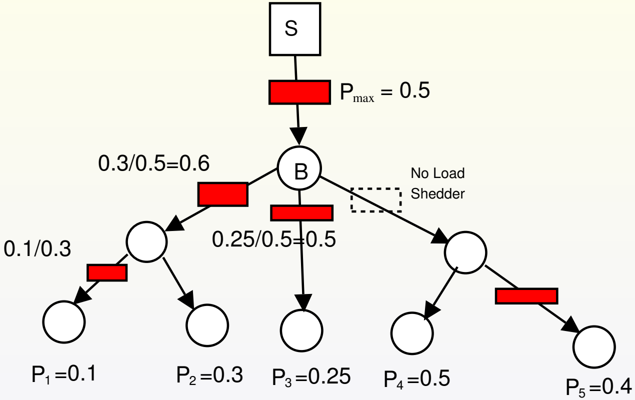
Observation 4

Let q_{max} be the query that has the highest effective sampling rate among all queries sharing an initial segment S. In the optimal solution, S will contain a load shedder with sampling rate P_{max}.

Procedure SetSamplingRate(x,Rx)

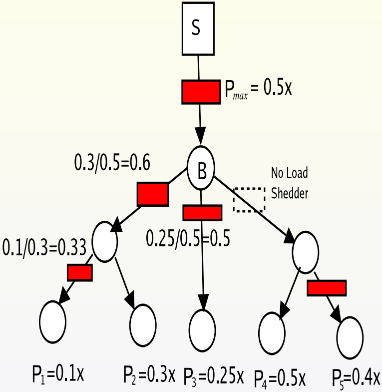
```
begin
**if** x is a leaf node **then**
    return
**end if**
Let $x_1, x_2, ...xk$ be the children of x
**for** i = 1 to k **do**
    **if** $P_{x_i} \leq R_x$ **then**
        Shed load with $p = P_{x_i}/R_x$ on edge (x, $x_i$)
    **end if**
    SetSamplingRate($x_i$, $P_{x_i}$)
**end for**
End
```

Example


S
P max = 0.5
B
0.3/0.5=0.6
No Load
Shedder
0.1/0.3
0.25/0.5=0.5
P1 =0.1
P2 =0.3
P3 =0.25
P4 =0.5
P5 =0.4

Determining the value of ϵ_{max}

- The first load shedder sampling rate depends on the actual P_i value.
- All other nodes depends only on the ratios between effective sampling rates.
- The load equation becomes a linear function of X i.e ϵ_{max}
- Care to be taken when there is no relative error.



Extensions

- Quality of Service
  - Query weights w_i
  - Minimize the maximum weighted *relative* error $(P_i \propto C_i w_i)$
- More General Query Classes
  - Group by query to be multiple queries, one query for each group.
  - Queries of type Set valued Answers.
- Incorporating Load Shedding Overhead
  - Emperical results show it is negligible.
  - Associating a processing cost per tuple with load shedding operators and including their cost in the load equation.

Experiments

- Do the load shedding decisions made by our algorithm, produce good approximate answers on realworld data streams
- Does our algorithm provide an appreciable benefit over a basic load shedding scheme that simply discards a fraction of arriving tuples when stream rates exceed system capacity?

To the degree that the future does not resemble the past, this could be a cause of inaccuracy.

Experiments

- Single Processor
- Linux server
- 512 MB RAM
- Network Domain Data
- 7 concurrently running Queries

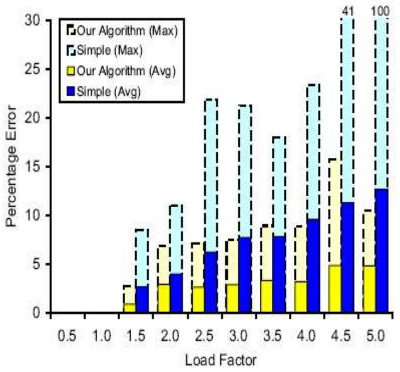


Figure 4. Accuracy at various levels of load

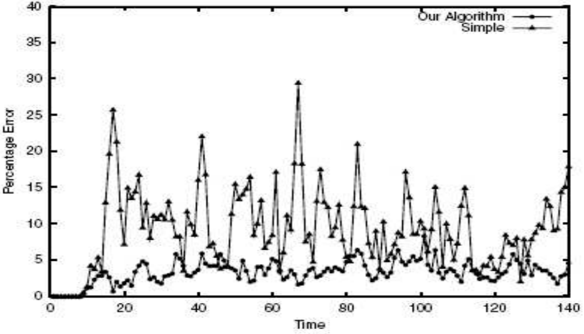


Figure 5. Accuracy of load shedding

- System Load Factor: 3

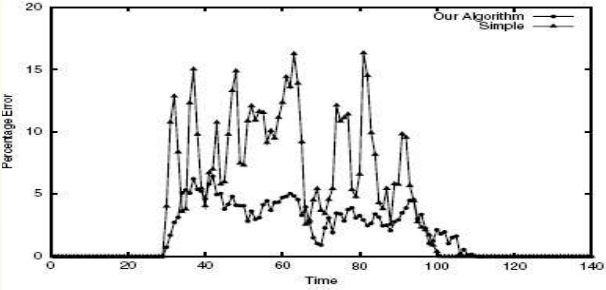


Figure 6. Adjusting to variable stream rates

- Initially the load is less then the System Load
- Suddenly increased to 3 times and finally to the original level

Conclusion

- It is important for computer systems to able to adapt changes in the operating environments.
- we have described a framework for one type of adaptive data stream processing, namely graceful performance degradation via load shedding.
- Our solution to the load shedding uses probabilistic bounds to determine the sensitivity of different queries to load shedding.

THANK YOU !