2009 IEEE International Advance Computing Conference (IACC 2009)
Patiala, India, 6–7 March 2009

Survey on Query Estimation in Data Streams

Sudhanshu Gupta[1], Deepak Garg[2]
[1]Research Scholar,[2] Asstt. Professor
Computer Science & Engineering Department,
Thapar University, Patiala.

Abstract
Query estimation plays an important role in query optimization by choosing a particular query plan. Performing Query estimation becomes quite challenging in case of fast, continuous, online data streams. Different summarization methods like Sampling, Histograms, Wavelets, Sketches, Discrete cosine series etc. are used to store data distribution for query estimation. In this paper a brief survey of query estimation techniques in view of data streams is presented.

Keywords: Query Estimation, Data Streams.

1. Introduction

The query estimation problem is important for optimization of the queries, as most of the queries have to be resolved in online time. The effectiveness of query optimization depends on the system's ability to assess the execution costs of different query execution plans. For this purpose, the sizes and data distributions of the intermediate results generated during plan execution need to be estimated accurately.

Besides query optimization there is a number of additional applications of query estimation in data streams such as decision support, fraud detection, quality and performance maintenance etc. Queries executing in data-mining try to monopolize a server. Therefore, knowledge of their running times is necessary to assign the proper priorities to each query. In case of multiple servers, cost estimations can be used for load balancing.

The size of the result of a query when applied on a given data distribution is expressed as number of tuples returned. This is referred to as query result size estimation. The fraction of tuples from the original data present in the query result is referred to as the selectivity of the query. Also estimation of the result data itself can be done in the form of a data distribution, from which all the information is extracted.

For this purpose, the sizes and data distributions of the intermediate results generated during plan execution have to be estimated accurately. This estimation requires the maintenance of statistics on the data, which is referred to as data synopses. Because data synopses reside in main memory, they compete for available space with the database cache and query execution buffers. Consequently, the memory available to data synopses needs to be used efficiently. So it becomes challenging to determine the best set of synopses for a given combination of datasets, queries, and available memory.

In this paper we will provide general view of the query estimation, along with its related work. We begin in section 2 with discussing the challenges posed by fast, continuous, evolving data streams.

In section 3 we review recent work using different techniques in query estimation. We conclude in section 4 with the directions on future work.

2. Challenging Data streams

Advances in hardware technology have facilitated the ability to collect data continuously. Most of everyday activities such as using a phone or browsing the web lead to automated data storage. Large volumes of data flow over the network continuously. These large volumes of data can be mined for interesting information in a wide variety of applications. This online monitoring of data streams poses a challenge in many data-centric applications including network traffic management, trend analysis, web-click streams, intrusion detection, and sensor networks.

With large volume of the data, it is not possible to process the data efficiently by using multiple passes. So a data item can be processed at most once. This leads to constraints on the implementation of the underlying algorithms to be designed to work with one pass over the data.

Data streams are potentially unbounded in size[3], so it is not possible to store the whole data stream in the main memory. The amount of storage required to compute an exact answer to a data stream query may also grow without bound. Due to inability to store a complete stream, approximate summary structures are used. Algorithms compute approximate answers as it is difficult to compute answers accurately with limited memory.

In data streams data may evolve over time. Therefore, a straightforward modification of one-pass mining algorithms may not be an effective solution to the problem. Stream mining algorithms need to be carefully designed with a clear focus on the evolution of the underlying data. For better efficiency and speedy processing of data streams, algorithms should be incremental, such that as new data are produced intermediate results should be reused.

3. Methods of Query Estimations

Query estimation is required for number of basic operations such as selection, projection and joins as sequence of these operation forms a query execution plan. A common approach to query estimation in data bases is to store summary information for each attribute that can be used to estimate the resulting size of the query. One of the main problems of applying query estimation techniques is unknown characteristics of distribution of data in a given data stream. A number of statistical methods have been employed to estimate queries efficiently. These methods differ in time and type of collection information and how the information is stored. Almost all of these methods use statistical distributions of the attribute values as the base for estimation. In order for such techniques to be useful the estimation techniques should be effective for different kind of data distributions.

In this section we will discuss works done to estimate queries using different summarization techniques.

3.1 Sampling

The samples based approach uses random samples of tuples as a significantly scaled down copy of original data, to get an estimate of results at runtime. With sufficient samples, these methods can provide accurate estimation as the information collected is the most recent one. The focus of an effective sampling method is to obtain the highest possible accuracy with minimum number of samples. Sampling techniques are simple to implement, do not require storage for statistical summary information.

Chen et. al. [6] gave an approach for estimating the record selectivities of database queries. The method approximates the attribute value distribution using query feedbacks. The idea is to use subsequent query feedbacks to regress the distribution, in the hope that as queries proceed, the approximation becomes more accurate. The adaptive approximation learns from the query executions, it remembers, recalls and predicts the selectivities of query predicates. This method doesn't take into account query information when approximating the value distribution and doesn't continuously adjust the distribution in case of updates.

Concise samples and counting samples [14] are summery statistics based on sampling. These samples are incrementally maintained under insertion and deletion regardless of the data distribution. The key idea is to increase the confidence for approximate answer by increasing the size of the samples. Any changes in the data distribution are reflected in the sampling frequency. The expected sample size increased with the skew in the data. However, counting samples method has more overheads as compared to traditional samples.

Acharya el al. [15] computes one Join synopsis for each relation for answering queries with foreign key joins. Available memory space is allocated optimally among join synopsis. The join synopsis for each relation is created by joining the sample of that relation with other relations. Join synopsis are incrementally maintaining in the presence of updates to base relation. This approach requires large samples for correct results.

Golden estimator [22] is designed to estimate the size of single dimensional range queries. The method adapts the golden rule given by Von Neuman for sampling cumulative probability distributions to discrete domains. It uses samples based on frequency distribution of the cumulative frequency distribution. The generated samples reflect the distribution without a priori knowledge. But the method is applicable only on range queries over a single attribute and not on join queries and multidimensional range queries. The golden estimator gives better approximation than MaxDiff(V,F) based histogram and wavelet based approaches under the same space requirement.

In case of dynamically changing query pattern, only focus on data reduction may not produce efficient results. According to method given by [21] the storing user query patterns instead of data distribution can be more effective. The method starts without a priori information of query patterns and randomly generates a sample set according to the data distribution with the hope that the user queries follow the same distribution. Instead of picking sample points with respect to data distribution, sample points are picked using query distribution. This technique is based on sampling of the cumulative distribution function (cdf) and can be applied to multiple attribute domains.

3.2 Histograms

Histogram is a popular way to approximate a data distribution. The histogram based approach divide attribute values to buckets, to store the information that can summarize the distribution of attribute values. The assumption for constructing histograms is that the data set to be approximated is finite and the size can be easily derived by performing a single pass over the finite data set. To reduce estimation error of selectivity histograms the enclosed domain region of each bucket has near uniform tuple density distribution. Histograms approximate a data distribution using a fixed amount of space, and under certain assumptions strive to minimize the overall error incurred by the approximation. The main challenge for histograms on multidimensional data is to capture the correlations among different attributes.

Histogram construction of an infinite data set on which data arrive continuously forming an infinite data stream is placed by Guha et al.[17] They gave first efficient single pass data stream histogramming algorithms for the fixed window data stream model, supporting incremental maintenance of the histograms. The method concentrates on finite fixed window data stream algorithms. These algorithms complement agglomerative data stream histogram algorithms presented in [16] and together try to solve the one pass histogram construction problem. The method improve over the optimal histogram algorithm of Jagadish et al. [10], which is used in developing fixed window algorithm.

To provide an approximated description of the cumulative frequencies in the bucket a 32 bits index is introduced in [8]. The approximate value of the cumulative frequency is stored in 7 intervals inside the bucket. The frequency values are organized in a 4-level tree over the bucket, they are called 4-level tree (4LT) index. The 4-level tree index tries to provide the best frequency estimation inside a bucket. The 4LT index when added to MaxDiff and VOptimal histogramming techniques the frequency estimation improves over inter-bucket ranges. The method shows improvements in the estimation of range queries.

Gunopulos et al.[7] address the problem of estimating the selectivity of multidimensional range queries for the datasets have numerical attributes with real values. The range queries considered are intersections of ranges, each range being defined on a single attribute. The technique defines buckets of variable size and allows the buckets to overlap. The use of overlapping buckets allows a more compact approximation of the data distribution. The method uses more and smaller buckets to approximate the data distribution where the data density is higher and fewer and larger buckets where the density decreases. GENHIST is a good technique for space dimensionalities but initially takes more than 5 passes to create histograms.

Another histogram used for query estimation is called nLT[9]. This is a non-bucket-based histogram. The idea underlying the nLT takes its origin from the above discussed 4LT method and extends the application of the approach to the construction of the entire histograms instead of single buckets. It is based on a hierarchical decomposition of the original data distribution kept in a full binary tree. This tree, containing a set of pre-computed hierarchical queries, uses bit saving for representing integer numbers, so that the reduced storage space allows to increase the tree resolution which in turn increase accuracy. The method improves significantly the accuracy in estimating range queries. Opposite to bucket based histogram update of an nLT is easy as partitions are not re-constructed.

3.3 Wavelets

Wavelet transforms are mathematical transforms that attempt to capture the trend in numerical functions by decomposing them. Most of the time, very few of the wavelet coefficients of empirical data sets are significant and majorities are small and hence insignificant. While implementing, a small number of significant coefficients are needed to capture the trends in numerical functions. For different type of functions of interest such image, curve, or surface, wavelets offer an elegant technique for representing the various levels of detail of the function in a space-efficient manner.

Matias et. al. [23] uses technique based upon multi- resolution wavelet decomposition for building histograms on the underlying data distributions. Histograms built on the cumulative data distributions are used on-line for selectivity estimation with limited space usage. Only best coefficients are kept by thresholding method based on a logarithm transform that significantly reduces the errors in wavelet-based approximation of high dimensional data. Wavelet approximation is more effective for selectivity estimation of range queries. Although Input/Output communication can be a problem, wavelet-based histograms using linear bases perform well over most of query sets and data distributions. These wavelet based histograms can be extended to multiple attributes by multidimensional wavelet decomposition and reconstruction.

The wavelet transform is able to compress a histogram into a small number of coefficients and offers a space efficient representation of the data distributions. However, in a dynamic streaming environment, it could require large space to calculate the wavelet coefficients and thus not directly applicable to data stream processing. The problem of summarizing the signal represented by the data stream in small space so that aggregate queries on the signal can be answered with reasonable accuracy is addressed by [2]. If the underlying signal has a small, highly accurate, transform based representation, the methods provide a high quality approximation.

Maintaining wavelet transforms is a not simple task as it requires tracking significant wavelet coefficients over time. When a data item changes in value, many coefficients may get affected and the set of significant coefficients could change quite extensively. Also to compute the highest B-term approximation to a signal the highest B-wavelet basis coefficients are maintained accurately. Nearly all of the signals must be in the auxiliary store in order to calculate the highest B-term approximation in data streaming models. So it is not possible to provide good data streaming algorithm for constructing wavelet approximations to the signal. To solve this problem a sketch of each signal is maintained so that any linear projection of it can be estimated, and the large projections get estimated accurately. As data items get read, the sketch gets updated. Since this sketch size is rather small compared to the signal size, the sketch is stored in the auxiliary store.

In sketches, linear projections can be generated of the signal with a small number of vectors quite accurately, provided the dot product of the corresponding unit vectors is large. Although sketches can be directly used to estimate the point query and range queries, wavelet coefficient approximations are generated from the sketch, which are in turn used for better query estimations on the signal. This wavelet based method can be scaled for multi-dimensions datasets.

3.4 Sketches

The sketch based approach uses specific functions to generate sketch as an estimate of counts of relation tuples. Its interesting randomizing algorithm and updatability make it attractive to data stream processing. Sketch has been used for aggregate query estimation over data streams.

The basic sketching technique was introduced for on-line self-join size estimation by Alon et al.[12].They use groups of such independent atomic sketches to improve the join size estimate in limited storage. They maintained signatures for each relation instead of maintaining synopsis on joining relations. The join is computed using these individual signatures. To estimate the join size of two relations first the size of the join of their signatures is computed then result is scaled. With small samples results are not accurate, so technique requires large sample to produce good estimations. Their results are extended and generalized by the Alin et al.[1] using algorithms that provide probabilistic accuracy guarantees for queries containing any number of relational joins and by considering a wide range of aggregate operators. This approach is based on randomizing techniques that compute small, pseudo-random sketch summaries of the data stream can then be used to provide approximate answers to aggregate queries with provable guarantees on the approximation error. The method intelligently partition the domain of the underlying attributes and decomposes the sketching problem in a way that gives efficient results. It minimizes the self-join sizes of the resulting partitions and intelligently allocates space to independent sketches for each partition. This approach requires a priori knowledge of the data distributions which may not be feasible for data stream environment. This sketch-based method is quite accurate when estimating the results of complex aggregate queries. This method performs better in accuracy than on-line histogram-based methods.

The problem of efficiently processing multiple concurrent aggregate SQL queries over a collection of input data streams is considered as extension of above described work by Alin et al.[1] The basic idea is to share sketch computation and sketching space across several queries in the workload that can effectively use the same sketches over their input streams. Sharing sketches among queries can significantly reduce the number of sketches needed to compute estimates. This, in turn, results in better utilization of the available memory, and much higher accuracy for returned query answers.

Ganguly et al. [19] gave better estimation results than the basic sketch [12,13].They stores the atomic sketches as hash structures. Firstly the dense frequencies that are greater than a certain threshold are skimmed from these hash-sketches. The join size is estimated as sum of subjoins calculated from dense frequencies and the skimmed sketches that is sparse frequencies.

Ganguly et al. [18] find the natural join over two data streams using parallel sketch structures. The structures use pair of parallel hash tables. First the join values for the most frequent top-k items are estimated. Then deleting the contribution of most frequent items from hash tables, the join value is estimated for remaining items as the median of averages.

Ganguly et al. [17] proposed algorithmic solution to the general Join-Distinct estimation problem over continuous data streams that can handle insertions and deletions. These estimators are probabilistic in nature and rely on efficient building and combining of hash-based synopses for individual update streams. These synopses are called JD-sketches. They make use of 2-level hash sketch structures while imposing an additional level of hashing that is needed in order to effectively project and count on the attributes for distinct-count. Estimator constructs several independent pairs of parallel JD sketch synopses for the input update streams. Then, at estimation time, each such pair of parallel JD sketches is composed in a manner to build a synopsis for the number of distinct pairs in the join result. The space usage of these estimators is within small factors of the best possible for the Join-Distinct problem.

3.5 Cosine Series

Discrete cosine series provides a good way to approximate data distributions. Discreet cosine transforms (DCT) requires only a small amount of space to store the data distributions. Another advantage of the method is that its coefficients can be updated easily and dynamically to manage with the rapid flow of data streams. The discrete cosine transform are used to approximate signals and images of various forms. The DCT generally can provide concise and accurate approximations to data distributions and its coefficients can be updated easily in the presence of insertions and deletions. These features make the DCT suitable for dynamic data stream environments.

Discrete cosine transform are used for estimating join size over the data streams [23]. Using DCT approximate aggregation query processing is done over data streams with limited storage space. Cosine series is used to approximate the data distributions of the data streams and then use them to estimate the size of equi-join queries. The method works well for on-line approximate aggregation equi-join queries over continuous data streams. The cosine transform yields better estimates than the sketches most of the time using the same amount of storage space.

Jiang et al. [24] developed a non-parametric selectivity estimation approach for range queries based on the data density approximation method inspired by Rao [3]. A data density function represents the actual data distribution of a relation. To store such a potentially complex distribution, to approximate it by a cosine series is proposed. This technique approximates data distributions directly and yields accurate estimates, especially in multi-dimensional cases. The estimator can be updated easily and remains accurate even when the underlying data has undergone substantial changes. In comparison to other methods, it is fast to construct, and easy to maintain.

S. No.	Type of Technique	Basic Idea	Advantages	Limitations
1.	Sampling	It uses random samples of tuples as a significantly scaled down copy of data.	1. Easy to implement. 2. Does not require storage of statistical information.	1. Does not work well under updates. 2. Difficult to work with complex queries 3. Multidimensional data cannot be handled easily with Sampling.
2.	Histograms	It partitions attribute values into buckets to summarize the data distribution.	1. Approximate the data distribution using a fixed amount of space.	1. Does not work well in the presence of updates. 2. It is not easy to handle large domains of attributes. 3. Difficult to work with multidimensional data.
3.	Wavelets	This method attempts to capture broad trends in data by decomposing data into more significant coefficient.	1. Requires a small number of significant coefficients for capturing trends in numerical functions. 2. Can be easily applied to multiple attributes.	1. In dynamic streaming environment it could require large space to calculate the wavelet coefficients. 2. Can not be applied directly to data stream processing. 3. Updating wavelets is difficult.
4.	Sketches	This approach uses simple randomized linear projection of underlying data vectors for query estimation.	1. Highly space efficient. 2. Works well with complex queries. 3. Good updation technique under insertion and deletion.	1. Difficult to apply to arbitrary applications. 2. Does not work well for multidimensional data.
5.	Cosine series	The method approximate data distribution by maintaining significant transform values.	1.Requires less space to store data distribution 2. Works well with complex queries. 3. Updated easily in the presence of insertions and deletions.	Does not perform well for strong positively correlated data as compared to sketches.
6.	Micro-clustering	The method stores micro-clusters at snapshots in time which follow a pyramidal pattern.	1. Provides a tradeoff between the storage requirements and the ability to recall summary statistics from different time horizons. 2. Works well with complex queries. 3. Multidimensional data is handled easily. 4.Adjusts well with evolution of data	

Table I: Comparison of Query Estimation Techniques

4. Conclusion and Future Work

Sampling is a simple and dynamic approach. It can be easily adapted to the continuous data stream environment. But its accuracy for more complex queries e.g. Join queries is not that good. Histogram provides a concise and efficient way to represent distributions of low-dimensional data. However, as the number of dimensions increases, the space required increases significantly. The large domains of attributes in data stream applications make it difficult to use Histograms.

Wavelets have been used for range, point queries. As the number of dimensions increases, the accuracy degrades sharply unless the number of coefficients used increases considerably. The update of the wavelet coefficients also faces a challenge as new data keep flowing in a data stream environment. Moreover wavelet may not be directly applicable to data streams because it could require large space to generate the highest coefficients. Sketches and cosine series based method works better than other approaches.

Micro-clusters [5] can be used to encode summary information about the data stream more efficiently than other techniques. As micro clusters store data distribution more efficiently, query estimation can be performed more efficiently. This technique can be applied on different type of queries.

References

[1] A. Dobra, M. Garofalakis, J. Gehrke, and R. Rastogi. “Processing complex aggregate queries over data stream”, In *ACM-SIGMOD,* Madison, Wisconsin, June 2002, pp 61-72.

[2] A. Gilbert, Y. Kotidis, S. Muthurkrishan, M. Straus, “Surfing wavelets on streams: one-pass summaries for approximate aggregate queries”, in: Proceedings of the VLDB Conference, 2001, pp. 79–88.

[3] B. Rao, Nonparametric Functional Estimation, Academic Press, 1983.

[4] Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. “Models and Issues in Data Stream Systems”. *21st ACM SIGACTSIGMOD-SIGART Symposium on Principles of Database Systems*, Madison, June 2002, pp 1-16.

[5] Charu C. Aggarwal ,Philip S.Yu, “Data streams : Models and Algorithms” , Springer ,2006 .

[6] Chungmin Melvin Chen,Nick Roussopoulos, “Adaptive Selectivity Estimation Using Query Feedback”, SIGMOD, ACM,1994

[7] Dimitrios Gunopulos, George Kollios, Vassilis J. Tsotras, Carlotta Domeniconi,” Selectivity estimators for multidimensional range queries over real attributes”,The VLDB Journal (2005) 14: Published online: March 4, 2004 Springer-Verlag 2004,pp 137–154.

[8] F. Buccafurri, L. Pontieri, D. Rosaci, D. Sacca. “Improving range query estimation on histograms”, in: Proceedings of the International Conference on Data Engineering, ICDE, 2002, pp. 628–638.

[9] Francesco Buccafurri , Gianluca Lax ,”Fast range query estimation by N-level tree histograms”,Data & Knowledge Engineering (2004) 257–275.

[10] H.V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K.C. Sevcik, T. Suel, “Optimal histograms with quality guarantees”, in: Proceedings of the 24th International Conference on Very Large Data Bases, VLDB, 24–27 1998,pp. 275–286.

[11] K. Chakrabarti, M. N. Garofalakis, R. Rastogi, and K. Shim. “Approximate query processing using wavelets”. In *Proceedings of 26th International Conference on VLDB*, Cairo, Egypt, 2001, pp 111-122.

[12] N. Alon, P.B Gibbons, Y. Matias and M.Szegedy. “Tracking Join and Self-join Sizes in Limited Storage”, In *proc of the 18th ACM SIGACT-SIGMOD-SIGART Symp. on the Principles of Database Systems*, May 1999, pp.10-20.

[13] N. Alon, Y. Matias and M.Szegedy. “The Space Complexity of Approximation the Frequency Moments”, In *Proc of 28th Annual ACM Symp.on the Theory of Computing*, May 1996, pp 20-29.

[14] P. Gibbons and Y. Matias. “New sampling-based summary statistics for improving approximate query answers”. In *ACM SIGMOD 1998*, Seattle, Washington , June 1998, pp331-342.

[15] S. Acharya, P. Gibbons, V. Poosala, and S. Ramaswamy. “Join synopses for approximate query answering”, In *SIGMOD. ACM Press*, 1999, pp275-286.

[16] S. Guha, N. Koudas, “Approximating a data streams for querying and estimation: algorithms and performance evaluation”, ICDE (2002) 567–576.

[17] Sumit Ganguly IIT , Minos garofalakis , Amit kumar , Rajeev Rastogi., “Join–Distinct Aggregate Estimation over Update Streams”, ACM ,june 2005,pp 259-270.

[18] Sumit Ganguly, Deepanjan Kesh, and Chandan Saha,” Practical Algorithms for Tracking Database Join Sizes” FSTTCS 2005, Springer-Verlag Berlin Heidelberg 2005 ,pp. 297–309.

[19] Sumit Ganguly, M. Garofalakis, R. Rastogi, “Processing Data-Stream Join Aggregates Using Skimmed Sketches”, *Proc of EDBT*, Heraklion-Crete, Greece, March 2004, pp. 569-586.

[20] Y. Matias, J. S. Vitter, and M. Wang. “Wavelet-Based Histograms for Selectivity Estimation”, In *Proceedings of the ACM*
SIGMOD Conference 1998, Seattle, Washington , June 1998, pp 448-459.

[21] Yi-Leh Wu ,Divyakant Agrawal Amr El Abbadi , “Query Estimation By Adaptive Sampling” Data Engineering, 2002. Proceedings. 18th International Conference on Volume , Issue , 2002 Page(s):639 - 648 .

[22] Y-L Wu, D. Agrawal and A. E. Abbadi, “Applying the Golden Rule of Sampling for Query Estimation”, In ACM SIGMOD 2001, Santa Barbara, California, May 2001, pp 449- 460.

[23] Zhewei Jiang, Cheng Luo, Wen-Chi Hou, Feng Yan, Qiang Zhu, Chih-Fang Wang **“**Join Size Estimation Over Data Streams Using Cosine Series **“,**International Journal of Information Technology, Vol. 13 No. 1 2007,pp 27-46.

[24] Zhewei Jiang, Cheng Luo, Wen-Chi Hou, Feng Yan, Qiang Zhu,”Selectivity Estimation of Range queries based on data density approximation via cosine series”,Data & Knowledge Engineering ,may 2007,pp 855-878.