A Survey on Continuous Monitoring of Preference Queries Using Sliding Window

Vineetha Sara Abraham [*], S.Deepa Kanmani [**]

[*] Post-Graduate Student, Department of Computer Science and Engineering, Karunya University, India
[**] Assistant professor, Department of Computer Science and Engineering, Karunya University, India

Abstract- Preference queries are used in multi criteria decision making applications where a number of contradictory criteria are involved to select the most commodious answers to the user. Many modern applications agree to the streaming model of computation and therefore continuous query processing algorithms are required to refresh the query result. Examples of such emerging applications are technical data analysis, data organization in sensor networks, article filtering in information retrieval, web-based alerts, issue/subscribe services. Continuous processing of the preference queries is performed by using the sliding window technique which generates fairly accurate answer to data stream query by evaluating the recent data. A Brief overview of how the sliding window technique can be used with preference queries comprised in this survey.

Index Terms- Top-k, Skyline, Top-k dominating, data streams, algorithms, analysis.

I. INTRODUCTION

Preference based queries rank the items of a database according to the significance of their attributes.In addition to hard constraint (eg price<100) the result must satisfy some additional specific properties related to the attribute values associated with each tuple [1]. The preference queries are broadly classified into top k query, skyline query and top k dominating query. In a top k query user defined ranking function is used which assign a value to each tuple. It bound the output size. If two or more tuples have the matching score then all these tuples added or use a tie-breaking criterion. The most important limitation of the top k query is that a user defined ranking function is used.

Skyline query is one of the most widely used preference query. The result of a skyline query is composed of the points that are not dominated by any other point. A dominant tuple is defined as tuple t_x dominates another tuple t_y ,if and only if t_x is smaller than or equal to $t_{y.}$in all dimensions [5].The dominance relationship counts on the semantics of each attribute. In some cases, small values are preferred (e.g., distance) but in other cases large values are suitable (e.g., quality). The key advantage of the skyline query is that it does not require any user-defined information or parameter. The limitation of the skyline query is that it does not bind the output size and therefore in extreme cases it is possible that all tuples be the part of skyline tuples.
A new concept has been proposed which combines the idea of domination with the notion of ranking functions. This new query is termed as top-k dominating query. It is a combination of top-k and skyline queries. It uses the dominant relationship rather than a user defined score function. It uses an intuitive score to rank the tuples that can be interpreted easily by a non expert. The score associated with a point p_i, denoted as score (p_i), equals to the number of points that p_i dominates. The incentive of this thought is to define a preference query that maintains the compensation and eliminates the restrictions of both top-k and skyline query. The top-k dominating query has the following desirable properties:

- The amount of result is assured.
- The outcome is scaling invariant.
- User-defined position function is not necessary.
- It uses an intuitive score to rank the tuples.

The top-k query, skyline query and top-k dominating query are answered by using proficient algorithms. These algorithms operate in an ad hoc fashion meaning that they lead up a query processing task only if a query is issued .This is adequate for applications functioning on static or almost static data sets, where updates are rare. Many modern applications agree to the streaming model of computation, and therefore continuous query processing algorithms are required to refresh the query result. Examples of such upcoming applications are computer network monitoring, scientific data analysis, data management in sensor networks, document filtering in information retrieval, web-based alerts, publish/subscribe services, The common property of these applications is that updates are very frequent, rendering query re-execution a nonviable solution.

The continuous monitoring of preference queries is performed by using a sliding window. Sliding window technique is the process of generating an approximate answer to a data stream query by evaluating the query not over the entire past history of the data stream, but rather only over a sliding window of the recent data [7].For example, only data from the last week could be considered in producing the query answers. The sliding window technique has many attractive properties. It is definitely and easily understood. It is deterministic. It emphasizes on recent data. The two basic sliding window types are count based sliding window and time based sliding window. In a count based sliding window, the number of dynamic points remains constant. If n new points arrive the n oldest points expire. In a time-based sliding window the number of active points remains constant. The termination time of a point does not depend on the reaching or termination of other points. The set of dynamic points is composed of all points reached at the last T time instances.

The rest of the paper is organized as follows. Section 2 and 3 study different processing techniques of top k query and skyline query respectively. Section 4 presents top k dominating query. Finally the conclusions are given in the section 5.

II. Top k Query

Consider a dataset *D* and a preference function f, a top-k query contains the k tuples with the highest scores according to f. The problem is well-studied in conventional databases but the existing methods are incompatible to highly dynamic environments involving numerous long running queries. K.Mouratidis et.al [3] proposed algorithms for the continuous monitoring of top-k queries over a fixed-size window W. The sliding window size can be articulated either in terms of the count based or time units. To achieve real-time query estimation the suitable tuples are stored in main memory. The valid records are arranged by using grid based index schema. Grid based index preserves a book keeping structure.

The top k query is important for several online applications such as communication and sensor networks, stock market trading, and profile based marketing etc. Top k query evaluation can be performed by using the count based and time based sliding window. The count based window W contains the most recent items and the time based window W contains all tuples that arrived within a fixed time instances. The task of the query processor is to constantly report the top k set of every monitoring query among the valid data. When a query *q* first arrives at the system, its result is computed by the top-*k* computation module which searches the minimum number of cells that may contain result records. Two algorithms are used for the continuous evaluation of Top k monitoring. The algorithms are Top k Monitoring Algorithms (TMA) and Skyband Monitoring Algorithm (SMA).The Top k Monitoring Algorithm re-computes the answer of a query wheneve*r* some of the current top-*k* points expire. The Skyband Monitoring Algorithm (SMA) partially precomputes future results by reducing the problem to sky band maintenance over a subset of the valid records.

The Top k Monitoring Algorithm consists of three modules such as grid based index structure, top k computation module and maintenance module. The grid based index is represented by using 2-dimensional space,. The grid structure contains cells Each grid cell contains the points. Each point p consist of following attributes <p.id, p.x, p.y, p.t> where id is the unique identifier, x and y are the attributes and t is the arrival time. The grid based index schema allows the continuous processing of multiple queries [4]. It avoids expensive reorganization costs. It can be broadly classified into regular grid structure and irregular grid structure. The benefit of the regular grid is that insertions and deletions are processed efficiently. It is significant to supply an efficient mechanism for evicting the expiring records. In the count and time based sliding window the tuples are evicted in First In First Out (FIFO) manner .All the records are stored in a single list .The new arrivals are placed at the end of the list. The tuples that fall out of the window are discarded from the head of the list.

The running queries q is stored in a query table. Query table maintains for each query q contains a unique identifier q.id,its scoring function q.f, the number of tuples required q.k and its current list q.top_list. The score of the kth point in q.top_list is referred to as q.top_score. To restrict the scope of the top k maintenance algorithms each cell is associated with an influence list IL_c.

In computation module the result of a query q is obtained by sorting all the score of the cell c according to the maxscore(c) and processes them in descending order. The search terminates when the cell c under the consideration has maxscore(c) ≤ q.top_score (q.top_score is the score of the kth element in the q.top_list) .

The operation on the maintenance module occurs after the computation of the initial result. When a new tuples arrives at the system the oldest tuples expires. Let P_{ins} be the set of incoming tuples and P_{del} be the set of evicted ones. For each $p \in P_{ins}$ it initially insert into the point list of the corresponding cell c.Then it scan the influence list IL_c of c and updates the result of every $q \in IL_c$ for which score(p) ≥ q.top_score . The expunged point p may be part of the result for some of the queries in IL_c . For each query q in IL_c , If $p \in$ q.top_list , q is marked as affected ,implying that it result has to be computed from scratch when the processing of P_{del} is completed.

Skyband Monitoring Algorithm applies the reduction of top k to k-skyband queries in order to avoid computation from scratch when the results expire. The skyband maintenance procedure only handles tuples p with score(p) ≥ q.top_score. When such a tuple arrives at the system, it is inserted into q.skyband increasing its cardinality.

SMA is expected to be faster than TMA, because it involves less frequent calls to the top k computation module. The space requirements of SMA are higher than the TMA, because it maintains the skyband of each query. TMA recomputed the result from the scratch and the SMA maintains a superset of the current answer in the form of a k skyband.

III. Skyline Query

The skyline consists of the tuples not dominated by other tuple. The skyline computation has received considerable attention in relational database but the existing algorithms of the skyline computation are inapplicable to stream application. The irst reason is they assume static data that are stored in the disk. The second reason is they focus on “one-time” execution that returns a single skyline .The third reason is they aim at reducing the i/o overhead. The skyline computation in streaming environment is performed by using a sliding window. Y.Tao [2] proposes algorithms that continuously monitor the incoming data and maintain the skyline incrementally. These algorithms utilize several interesting properties of stream skylines to improve space/time efficiency by expunging data from the system as early as possible. The skyline computation in data stream system that consider only the tuples that arrived in a sliding window covering the W most recent timestamps, where W is a system parameter called the window length.

The architecture of the skyline system consists of four sections such as input buffer, pre-processing modules, the database and maintenance module. The arriving tuples are placed in the input buffer which is processed by processing module in ascending order and the sorted tuples are placed in the database to place the arriving tuples. The database is further divided into DB_{sky} and DB_{rest} storing points that are and are not in the current skyline. Whenever a skyline points expires some points in the DB_{sky} may appear in the new skyline. The Maintenance module is responsible for expunging the obsolete data from the database and outputting the skyline streams.

The two general frameworks for online monitoring of skyline query are Lazy method and Eager method. The Lazy method delays most computational work until the expiration of the skyline point. The eager method takes the advantage of precomputation to memory consumption.

The continuous evaluation of the skyline query can occur only when a new tuple arrives or some skyline points expire. Lazy approach handles these situations in its pre-processing module and maintenance module. Given an arriving point p the pre-processing module checks if it is dominated by any point in the DB_{sky} , then it is placed on the DB_{rest}. On the other hand, if a new point p is not dominated by any skyline point, it is added to DB_{sky}. The point p may dominate some skyline point that may expunge from the system. The disadvantage of the lazy method is that the DB_{rest} need to store obsolete data and points that will never appear in the skyline.

The Eager method aims at minimizing the memory consumption by keeping only those tuples that are or may become part of the skyline in the future and reducing the cost of the maintenance module. It achieves these goals by performing additional work in the pre-processing module. Eager maintains an event list that contains entries of the form e=<e.ptr,e.t,e.tag>.Field e.ptr is a pointer to the tuple involved in the event, e.t specifies the event time and e.tag indicates the event type. if the tuple r referenced by e.ptr belongs to the skyline currently, then e.tag ='EX' indicating the expiry of the skyline point or e.tag='SK' indicating the inclusion of the skyline. The key benefit of the eager method is it reduce the memory consumption.

IV. Top k Dominating Query

Top k dominating query returns the finest top k points to the user based on the dominance power. It maintains the advantage and eliminates the disadvantage of Top k query and skyline query. The continuous monitoring of top-k dominating queries agree to the sliding window approach in which only the n most recent points called active points are occupied into account. The count based sliding window technique is used here. Each point p associate two time instances p.arr is the onset time of p, whereas p.exp is the resultant termination time. When a point terminate, it is separated from the set of dynamic points.

M. Kontaki et.al [9] proposed algorithms for continuous processing of top k dominating query using sliding window technique. The algorithms are Brute Force Algorithm (BFA), Event based Algorithm (EVA), Advanced Algorithm (ADA), Advanced Hoeffding Bound Algorithm (AHBA) and Advanced Minimum Score Algorithm (AMSA).

The naive approach to evaluate a top-k dominating query continuously is to perform all domination checks using points. For a new point p_x, its score (P_x) is computed by counting the number of points dominated by p_x. Moreover, the score of a point p_y, $y \neq x$ should be increased if p_y dominates p_x. When a point expires, the scores of other points need to be updated. This simple algorithm is called Brute Force Algorithm (BFA).It is expected that BFA will invoke a large number (O(n)) of domination checks between points.

An Event-Based Algorithm (EVA) which uses event scheduling and rescheduling toward avoiding the examination of points for inclusion in top k. Let p_i be a point that is not part of top k and therefore, score $(p_i) < score_k$ (score of top k points) . In each update, the value of $score_k$ can be reduced at most by 1 and the value of score (can be increased at most by 1. Therefore, p_i cannot be in top k in less than $[(score_k - scorep_{i)}/2)$ time instances, unless a top-k dominating point expires during this period. Event Based Algorithm has two disadvantages. The first is that all points that are not part of top k should be examined at the expiration time of a top-k dominating point. The second is that it is possible that many points have a score close to $score_k$, resulting in consecutive exact score computations.

To avoid the disadvantage of Event Based Algorithm a new algorithm called Advanced Algorithm can be used. In advanced algorithm it continuously evaluate the score of some special points called candidate points, whose event processing time is in the near future.

To trade accuracy for speed two approximate algorithms such as Advanced Hoeffding Algorithm (AHBA) and Advanced Minimum Score Algorithm (AMSA) are used. The effectiveness of an approximate solution is measured by considering the accuracy of the result, which corresponds to the fraction of the correct top-k dominating points returned by the algorithm over the actual number of top-k dominating points. AHBA offers probabilistic guarantees regarding the accuracy of the result based on the Hoeffding bound. AMSA performs a more aggressive computation resulting in more efficient processing

Table I: Comparison between Preference Based Queries

Query Type	Method	Sliding Window	Merits	Demerits
Top k Query	Top k Monitoring Algorithm Skyband Monitoring Algorithm	Count and Time	Simple implementation	No efficient handling of an expired tuple
Skyline Query	Lazy and Eager method	Count and Time	Handle Skyline Expiration	Processing time per tuple.
Top k Dominating Query	Advanced Algorithm and Approximate Algorithms	Count	Based on Event Time computation.	Different methods used for computation.

V. CONCLUSION

The main aim of this survey is to analysis of the methods used by preferences queries to find out the top k result. The sliding window technique is used for continuous monitoring of data streams. In top k query the continuous monitoring of the data is performed by using Top k Monitoring Algorithm and Skyband Monitoring Algorithm. In skyline query continuous monitoring of skyline points is performed by lazy method and eager method. In top k dominating query the advanced algorithm and approximate algorithms are used to return the top k result. Finally all preference queries methods are compared and concluded that top k dominating query with advanced algorithm show the best performance.

ACKNOWLEDGMENT

I express my thanks to my guide who have helped me to do this survey and, all staffs and my friends who helped me with their knowledge and experiences to work on this paper “A Survey on Continuous Monitoring of Preference Queries using Sliding Window”.

REFERENCES

[1] W. Kiessling, “Foundations of Preferences in Database Systems,”Proc. 28th Int’l Conf. Very Large Data Bases (VLDB), pp. 311-322,2002.
[2] Y. Tao and D. Papadias, “Maintaining Sliding Window Skylines on Data Streams,” IEEE Trans. Knowledge and Data Eng., vol. 18, no. 3, pp. 377-391, Mar. 2006.
[3] K. Mouratidis and S. Bakiras, , “Continuous Monitoring of Top-k Queries over Sliding Windows,” Proc. ACMSIGMOD Conf. Management of Data ,pp. 635-646, 2006.
[4] M. Kontaki, A.N. Papadopoulos, and Y. Manolopoulos, “Continuous Top-k Dominating Queries in Subspaces, ” Proc. Panhellenic Conf. Informatics (PCI), pp. 675-689, 2008.
[5] D. Papadias, Y. Tao, G. Fu, and B. Seeger, “Progressive Skyline Computation in Database Systems” ACM Trans. Database Systems,vol. 30, no. 1, pp. 41-82, 2005.
[6] M.L. Yiu and N. Mamoulis, “Multidimensional Top-k Dominating Queries,” VLDB J., vol. 18, pp. 695-718, 2009.
[7] B. Babcock, M. Datar, R. Motwani, and J. Widom,“Models and Issues in Data Stream Systems,” Proc. ACM SIGMOD Symp. Principles of Database Systems), pp. 1-16, 2002.
[8] V. Hristidis, N. Koudas, and Y. Papakonstantinou, “PREFER: A System for the Efficient Execution of Multi-Parametric Ranked Queries,” Proc. ACM SIGMOD Int’l Conf. Management of Data (SIGMOD), pp. 259-270, 2001.
[9] M. Kontaki, A.N. Papadopoulos, and Y. Manolopoulos , “Continuous Monitoring of Top k Dominating Queries” IEEE Transactions on Knowledge and Data Engineering, pp, 840 - 853 ,2012.
[10] S. Borzsonyi, D. Kossmann, and K. Stocker, “The Skyline Operator,” Proc. Int’l Conf. Data Eng. (ICDE), pp. 421-430, 2001.

AUTHORS

First Author – Vineetha Sara Abraham, Post-Graduate Student, Department of Computer Science and Engineering, Karunya University, India
Second Author – S.Deepa Kanmani, Assistant professor, Department of Computer Science and Engineering, Karunya University, India