Multicriteria Data Retreival in Database using Advanced Database Opeartor

[1]Dr. Anil Rajput, [2]Ms. Sunita Dwivedi
[1]*Department of Mathematics and Computer Science, Chandra Shekhar Azad Govt.P.G.Nodal College, Sehore(M.P)*
[2]*Makhanlal chaturvedi National University of Jouranalism and Communication, Bhopal(M.P.)*

ABSTRACT

Database management systems have been increasingly used in decision support applications. One of the features of such application is query with multiple, and sometimes conflicting, goals. People recently are interested in an advanced query operator for such queries named skyline which returns the objects that are not dominated by any other objects with regard to certain measures in a multi-dimensional space. The skyline query is frequently used to find a set of dominating data points (called skyline points) in a multidimensional dataset and finds a set of interesting objects, satisfying a set of possibly conflicting conditions.

In this paper we present a study of this interesting and still evolving research area so that readers can easily obtain an overview of Skyline query. We presented history of this concept and then evaluation of Skyline query in database.

II INTRODUCTION

Preferences are ubiquitous in everyday private and business life. In many situations for decision making, users need to select one or more data from database in accordance with their interest. Preference queries have a wider range of applications such as personalized search engine and e-shopping. The need for preference queries arises because traditional queries, which ask for results that match users' criteria exactly, cannot cope well with real users' demands. Many extensions to the SQL language have been proposed, such as Preference SQL [17] [18] [19].

Many applications in multi-criteria decision making, database management systems have been used where there is no clear preference function over the attributes, and the user wants an overall big picture of which objects dominate (equivalently, are better than) other objects in terms of preferences. In the presence of huge amounts of data that today's systems are providing access to; it is a tedious task for a user to find the most interesting available data without using advanced query types. In order to support those multi-criteria decision making applications effectively, an extension is proposed [4] in database systems by a skyline operation. Skyline query is one of the most extensively studied sub-problems of preference query. It corresponds to the Pareto preference constructor, where every criterion is equally important. Also, standard skyline query assumes that the records can be mapped to points in the Euclidean space, i.e., there is a total order in any single dimension.

A commonly cited example for multi criteria decision making is assisting a tourist in choosing a set of interesting hotels from a larger set of candidate hotels. Each hotel is identified by two attributes: a distance from a specific point (such as a location on a beach), and the price for the hotel. Such applications are featured by following

1. Queries are typically based on multiple, and sometimes conflicting, goals, as hotels with less distance to beach tend to be costly.
2. As opposed to conventional applications, there may be no single optimal answer (or answer set) to these applications, there is no one hotel exists which will fit best on both the needs so list of hotels is expected.
3. Due to the second feature of these applications, users are typically looking for all the satisfied answers. One can find best as final decision from those "good" candidates by weighing his personal preferences for price and distance to the beach.
4. For the same query, different users, dictated by their personal preferences, may find different answers meeting their needs.

Traditionally, the DBMS supports these applications by returning all answers that may meet the user's requirement. In our tourist example, if the user specifies "price" to mean in the range of $120-$200, and "distance" to mean within 5 km, then the system may return all hotels that satisfy these predicates. This is not very helpful because users may be overloaded with too much information. Decision making will be typically more complex when selection of best choice may have many more criteria with various amenities as star rating, swimming pool etc. As no best choice can be produced one can wish at least all interesting hotels may be selected based on the users choice. Interesting are all hotels that are not

worse (dominated) than any other hotel in all requested dimensions.

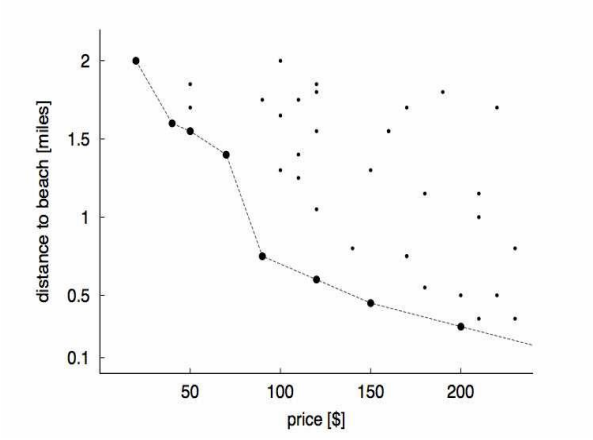

Figure 1: Skyline of "Cheap hotels near the beach"

The same considerations also hold for a variety of applications (e.g., electronic marketing places or real-estate databases for houses), where the user is interested in mobiles, cars, houses, or other products. The user might, for instance, be looking for a new mobile supporting all fancy features such as Wi-Fi, high resolution camera and display, but with long talk/standby times and still minimum weight and size. Likewise, a user who is interested in buying a car wants to find a good trade-off between minimum mileage, minimum age, and minimum price.

II THE SKYLINE QUERY

Skyline operation is defined as notion of comparing the goodness along each dimension. A skyline query returns a set of data points that are not dominated by any other points in a given data set. A point dominates another point if it is no worse in all concerning dimensions and better in at least one dimension. To assist a tourist in narrowing down the choices, the skyline operator can be used to find the set of all hotels that are not dominated by another hotel on both the criteria of selection.

Given a set of points, "skyline query" returns the points in the set, which are not "dominated" by another point. In all the dimensions included. These points are skyline objects. Here the dominance can be defined as :

for a dataset D consisting of data points t_1, t_2, ..., t_n the skyline S for a defined comparison function, is the set of all ti such that there is no t_j that dominates t_i.

t_i is said to dominate t_j if t_i is better than t_j in at least one dimension and not worse than t_j in all other dimensions.

For each table tuple $t_i = (a_0,a_1,\ldots,a_d)$ dominates tuple

$t_j = (a'_0,a'_1,\ldots,a'_d)$

If $(\exists\ m\ (0 \le m \le d)\ ,\ (a_m > a'_m)$ and

$\forall\ a_k \ge a'_k$ for $0 \le k \le d)$

III MAXIMAL VECTOR

The popularity of the skyline operator is mainly due to its applicability for decision making applications in database. Skyline queries help users make intelligent decisions over complex data, where different and often conflicting criteria are defined for decision.

Whereas the problem is known as skylines in database research, in other areas it was already known before, Exploration of the problem to search for such "best" items has a long history and can be traced back to the 1960s in the theory field as the maximum vector problem or the Pareto optimum [12][20][25]. The maximal vector problem is to find the subset of the vectors such that each is not dominated by any of the vectors from the set. One vector dominates another if each of its components has an equal or higher value than the other vector's corresponding component, and it has a higher value on at least one of the corresponding components.

One may equivalently consider points in a k-dimensional space instead of vectors. In this context, the maximals have also been called the admissible points, and the set of maximals called the Pareto set. The maximal vector problem has been rediscovered in the database context with the introduction of skyline queries. Instead of vectors or points, this time it is to find the maximals over a set of tuples. For example Skyline queries ask for a set of interesting points from a potentially large set of data points. Hotel A dominates hotel B if A is at least as close as B and at least as cheap as B, and offers either a better price, or is closer, or both compared to B. Criterias of selection may have multiple dimensions as star rating, food quality and so on.

However, algorithms developed in maximal vector based on a divide and conquer approach, cannot be directly applied in a database scenario, since they do not take into account main memory limitations that prevent the whole dataset being loaded before the actual skyline computation starts. This was observed in [4], where a divide and conquer algorithm, D&C, adapted to work with external memory was proposed.

IV ADAPTATION IN SQL

Computing the skyline is equivalent to determining the maxima of a set of vectors, a well-known problem in computational geometry [25]. The syntax and semantics of skyline queries were first formally presented in [3]. The basic syntax of skyline queries is defined using the following extension to SQL. SQL adaptation of skyline of is implemented in [11].

The SQL's SELECT statement by an optional *SKYLINE OF* clause as follows:

SELECT … FROM …WHERE …

GROUP BY … HAVING …

SKYLINE OF [DISTINCT d_1 [MIN |MAX | DIFF],..,d_m [MIN | MAX | DIFF]

ORDER BY ...

d_1,...,d_m denote the dimensions of the Skyline; e.g., price, distance to the beach, or rating. MIN, MAX, and DIFF specify whether the value in that dimension should be minimized, maximized, or simply be different. For example, the price of a hotel should be minimized (MIN annotation) whereas the rating should be maximized (MAX annotation). It does not matter in which order the dimensions are specified in Skyline. DIFF can be explaining with another skyline example "The Mostly used Skyline of Manhattan as an visualization database problem which is to compute building higher and near to Hudson river. Two buildings that have different x coordinates can both be seen and therefore both may be part of the skyline; as a result, the x dimension is listed in the SKYLINE OF clause of that query with a DIFF annotation. The optional DISTINCT specifies how to deal with duplicates. If two tuples have the same values for all attributes listed in the SKYLINE OF clause and they are not dominated by any other tuple, then they are both part of the result. With DISTINCT, however, either of them is retained.

Let p, q be two tuples in a database of dimension n and the skyline points of interest is to evaluate from m of d dimension i.e.

Tuple p = (p_1,...,p_k, p_{k+1},..., p_l, p_{l+1},..., p_m, p_{m+1},..., p_n)

Tuple q = (q_1,..., q_k, q_{k+1},..., q_l, q_{l+1},..., q_m, q_{m+1},..., q_n)

And we have the following Skyline Query:

SKYLINE OF d_1 MIN,..., d_k MIN,

d_{k+1} MAX,..., d_l MAX,

d_{l+1}DIFF,...,d_mDIFF

where, MIN ,MAX, DIFF indicates that we are looking for a minimum maximum, different point in that specific dimension. Then we can say that tuple p dominates tuple q if the following 3 conditions hold:

$p_i \leq q_i$ for all i = 1, ..., k

$p_i \geq q_i$ for all i = (k + 1), ..., l

p_i = qi for all i = (l + 1), ...,m.

The SKYLINE OF clause is executed after the SELECT ... FROM ... WHERE ... GROUP BY . . . HAVING. . . part of the query, but before the ORDER BY clause and possibly other clauses that follows The SKYLINE OF clause selects all interesting tuples; i.e., tuples which are not dominated by any other tuple.

So for the example discussed of finding a close and cheap hotel can be expressed within the scope of SKYLINE OF clause is as follows:

```
SELECT * FROM Hotels
WHERE city = 'Nassau'
SKYLINE OF price MIN, distance MIN;
```

The equivalent can be expressed in existing SQL [3] [32] which need to work with nested loop execution that's why database community advocated adopting SKYLINE OF as database operator.

```
SELECT * FROM Hotels h
WHERE h.city = 'Nassau' AND
NOT EXISTS (SELECT * FROM Hotels hl
 WHERE hl.city = 'Nassau'    AND
 hl.distance <= h.distance   AND
  hl.price <= h.price   AND
 (hl.distance < h.distance OR
 hl.price < h.price));
```

The skyline operator is within the expressive power of SQL and can be re written in standard SQL however there are reasons for advocating inbuilt skyline operator into SQL and building into RDBMS.

- It is easier to specify and grasp what's going on with user perspective in compare to similar syntax with nested sub query.
- It is faster to evaluate as number of algorithms now available, while sub query substitute suffers with nested loop execution.
- Preference query [18] initiated a new set of optimization opportunities to adopt in many applications.

- As skyline operator can be accommodated with other skyline operator the query optimizer [26] requires a little updating to adopt it in integration of other operator but need not to update concurrency or transaction management.
- The Skyline operator encapsulates the implementation of the SKYLINE OF clause. The implementation of other operators (e.g. join) need not be changed. According to the semantics of Skyline queries, the Skyline operator is typically executed after scan, join, and group-by operators and before a final sort operator.
- The optimization mechanism even can select most efficient skyline evaluation technique in different situation if a set of skyline computation mechanism included in RDBMS.

Computing the Skyline is easy if the SKYLINE OF clause involves only two dimensions. A two dimensional Skyline can be computed by sorting the data. If the data is topologically sorted according to the two attributes of the SKYLINE OF clause, the test of whether a tuple is part of the Skyline is very cheap: you simply need to compare a tuple with its predecessor. More precisely, you need to compare a tuple with the last previous tuple which is part of the Skyline. If sorting needs to be carried out in two (or more passes) because the data does not fit into main memory, then tuplescan be eliminated while generating each run and in the merge phase of the sort

Skyline queries can also involve more than two dimensions and they could depend on the current position of a user. For instance, (mobile) users could be interested in restaurants that are near, cheap, and have good food (according to some rating system). The distance is based on the current location of the user. Again, the idea is to give the user the big picture of interesting options and then let the user make a decision. If the user moves on, the Skyline should be re-computed continuously in order to give the user a choice of interesting restaurants based on the user's new location. Since skyline operator is costly, optimizing the queries is important.

V EVOLUTION OF SKYLINE QUERIES OVER TIME

Recently, there has been much interest in processing skyline queries for various applications that include decision making, personalized services, and search pruning. Skyline queries aim to prune a search space of large numbers of multidimensional data items to a small set of interesting items. Incorporating the skyline operator in a relational server first need to add the physical implementation of the skyline operator as part of the execution engine. In addition, it needs to change the query optimizer to allow interaction of skyline with other operators and provide cardinality and cost estimation modules for the skyline operator.

Since the introduction of skyline queries [4] in 2001, the researchers provide Skyline Computation on various directions in both centralized and distributed database. [4] first introduced the skyline operator and presented two basic main memory algorithms: BNL (Block Nested Loops) and D&C (Divide & Conquer). The BNL algorithm uses a block nested loop to compare each tuple of the database with every other tuple. A tuple is reported as a result only if it is not dominated by any other tuple. The D&C algorithm recursively divides the set of input tuples into smaller sets (regions), computes the individual skyline for each region separately, and merges them into the Improvement on above is on sorted data. Sort based computation was proposed in [7][12] with main principle that if the tuples are ordered based on a monotone scoring function, then no tuple can be dominated by subsequent tuples. Skyline query processing with the use of index structures was first proposed by [4], but elaborated on in later works [19][23][27]. The key idea is to use an index to determine dominance between tuples and to prune tuples from further consideration at an early stage. These algorithms first compute the nearest neighbor to the origin, which is guaranteed to be part of the skyline result set. Papadias et al. [23] first introduced different variants of the skyline operator, such as constrained, subspace and dynamic skyline queries. Constrained skyline queries were also discussed in [8][30]. Subspace skyline queries were discussed primarily from the view of query semantics in [24]. Then, SKYCUBE [31] was defined as the union of all skyline points of all possible non-empty subspaces. Subspace skyline retrieval was also studied in [28], which proposed the SUBSKY algorithm. Aiming to restrict the skyline cardinality, Chan et al. [6] proposed the k-dominant skyline query.

Reverse skyline queries have been studied in [10]. Given a query point q, the reverse skyline set contains all points p whose dynamic skyline set contains q. For data point p, the dynamic skyline query employed by the definition of the reverse skyline set, uses d dimension functions defined as the absolute difference between the attribute values on each dimension. This corresponds to the skyline set of a transformed data space where p becomes the origin and all other data points are represented by their coordinate-wise

distances to point p. Thus, the reverse skyline query retrieves the data objects that are at least in one dimension more similar (in terms of absolute di fference of attribute values) to q than all other data objects. Apart from the efficient computation of the skyline operator and its variants, skyline queries have been studied in different domains, such as probabilistic skyline queries over uncertain data , skyline queries on incomplete data [16] and partially-ordered domains [5]. Furthermore, efficient skyline computation over streams has first been studied in [29]. Moreover, in [22], bandwidth-constrained skyline queries over mobile devices were studied.

Skyline queries have originally been proposed for centralized environments [4],i.e single database environments. As now a day's data is increasingly stored and processed in a distributed way, skyline processing over distributed data has attracted much attention .Skyline computation in highly distributed systems, such as P2P systems, where each server stores a fraction of the available data. Skyline queries have also been studied in other distributed environments, such as web information systems [1, 21] or parallel shared-nothing architectures [14][15], and with respect to different data types, such as streamed [26] or uncertain data [14].

VI APPLICATION AREA OF SKYLINE QUERY

The rapid growth in the number of Internet users has resulted in the development of a variety of on-line services to facilitate commerce, social networking and information sharing. This phenomenon has highlighted the need for supporting complex multi-criteria decision support (MCDS) queries [4]. The intuitive nature of specifying a set of user preferences has made Pareto-optimal (or skyline) queries a popular class of MCDS queries . In such applications, the database tuples are represented as a set of multidimensional data points and the skyline set contains those points that are the best trade of between the different dimensions.

Well known application areas of skyline queries are –

Customer information systems, travel agencies and mobile city guides are one of the application area for which Skyline queries are useful .Skyline has to be computed when user move on.

- Decision Support (Business intelligence) is another area, For instance, a Skyline query can be used in order to determine customers who buy much and complain little.
- The Skyline operation is very useful for data visualization. With the help of the Skyline, the outline of a geometric object can be determined; in other words, the points of a geometric object that are visible from a certain perspective can be determined using a Skyline query .The Skyline of Manhattan, for instance, can be computed as the set of buildings which are high and close to the Hudson River.
- Another application is distributed query optimization: the set of interesting sites that are potentially useful to carry out a distributed query can be determined using a Skyline query: those interesting sites have high computing power and are close to data needed to execute the query.
- In many applications, similarity search is more practical than exact match. For instance, in image retrieval, there may not exist any image in the database which is exactly the same as the query image. To increase the search accuracy, one may use multiple query images, such as those extracted from a video taken during crime. Different query images may be the most similar to different images in the database. The question is: what are the most similar images to ALL the query images? The skyline concept can be used here to answer the above question. In particular, the query result should be the images in the database which are not dominated by any other image. Image A is dominated by image B, if B is more (or equally) similar to all query images than A is. Here the images are mapped to a metric space and the similarity between two images is often captured.

VII CONCLUSION

The skyline operator is an elegant summary method over multidimensional datasets. A skyline query returns a set of data points that are not dominated by any other points in a given data set. In this paper, we highlighted the skyline query concept, its history and Application areas. We also presented the key contributions related to skyline computation in centralized and distributed databases.

REFERENCES

[1] Balke, W., Guuntzer, U., Zheng, J.X., " Efficient distributed skylining for web information systems.", Proceedings of International Conference on Extending Database Technology (EDBT), pp. 256-273 , 2004

[2] Bartolini I., Ciaccia P., and Patella M. "Salsa: computing the skylinewithout scanning the whole sky.",In CIKM, pages 405–414, 2006.

[3] Bayross Ivan. (2008): SQL, PL/SQL The Programming Language of Oracle. 3rd Revised Edition, BPB Publications. [ISBN: 81- 7656-964-X].

[4] Borzsonyi S.,Kossmann D. and Stocker K., "The skyline operator.", Proceedings of the 17th International Conference on Data Engineering (ICDE), pages 421-430, 2001.

[5] Chan, C., Eng, P., Tan, K. "Stratied Computation of Skylines with Partially-Ordered Domains." Proceedings of International Conference on Management of Data (SIGMOD), pp. 203-214, 2005.

[6] Chan C., Jagadish H., Tan K., Tung A., Zhang Z. "Finding k-dominant skylines in high dimensional space.", Proceedings of International Conference on Management of Data (SIGMOD), pp. 503-514 2006 .

[7] Chomicki J. , Godfrey P., Gryz J., and Liang D. "Skyline with presorting." Technical Report, Computer Science, York University, Toronto, ON, Canada, Oct. 2002.

[8] Cui B., Lu H., Xu Q., Chen L., Dai Y., Zhou Y. "Parallel distributed processing of constrained skyline queries by ltering.", Proceedings of International Conference on Data Engineering (ICDE), pp. 546-555 ,2008.

[9] Dalvi N., Choudhary S., and Kaushik R., "Robust cardinality and cost estimation for skyline operator.", Proceedings of the 22ndt International Conference on Data Engineering (ICDE), pp. 64-74, 2006.

[10] Dellis, E., Seeger, B., " Efficient computation of reverse skyline queries." ,Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 291-302 ,2007.

[11] Eder H., "On extending PostgreSQL with the skyline operator." Master's Thesis Vienna University of Technology, 2009.

[12] Godfrey P., Shipley R., Gryz J. " Maximal vector computation in large data sets." , Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 229-240 ,2005.

[13] Guttman A., "R-trees: A dynamic index structure for spatial searching." Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 47-57, 1984.

[14] Hua M., Pei J., and Zhang W., "Ranking Queries on Uncertain Data A Probabilistic Threshold Approach," Proceedings of ACM SIGMOD, ACM, New York, pp. 673-686., 2008.

[15] Huang Z., Jensen C.S., Lu H., and Ooi.B. C., " Skyline queries against mobile lightweight devices in manets.", Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE) , 2006.

[16] Khalefa M., Mokbel M. and Levandoski J., "Skyline Query Processing for Incomplete Data.", Proceedings of International Conference on Data Engineering (ICDE), pp. 556- 565 ,2008.

[17] Kiessling W. "Foundations of preferences in database systems.", Proceedings of the 28th International Conference on Very Large Data Bases Hong Kong, China. MorganKaufmann, San Francisco, CA, pp.311–322,2002

[18] Kiessling W. and Koestler G., "Preference SQL: Design, implementation, experiences." In VLDB, Aug. 2002.

[19] Kossmann D., Ramsak F., and Rost S., "Shooting stars in the sky: an online algorithm for skyline queries.", Proceedings of 28th International Conference on Very Large Data Bases (VLDB), pp. 275-286, 2002.

[20] Kung H. T, Luccio F., and Preparata F. P. J. "On finding the maxima of a set of vectors.", ACM, 22(4):469–476, 1975.

[21] Lo, E., Yip, K.Y., Lin, K.I., Cheung, D.W. "Progressive skylining over web-accessible databases." Data Knowledge Engineering (DKE) 57(2), 122-147 ,2006.

[22] Lin, X., Yuan, Y., Wang, W., Lu, H., "Stabbing the sky: Effcient skyline computation over sliding windows.", Proceedings of International Conference on Data Engineering (ICDE), pp. 502-513 ,2005.

[23] Papadias D., Tao Y., Fu G., Seeger B. " An optimal and progressive algorithm for skyline queries." , Proceedings of International Conference on Management of Data (SIGMOD), pp. 467-478 ,2003.

[24] Pei J., Jin W., Ester M., Tao Y., " Catching the best views of skyline: A semantic approach based on decisive subspaces.", Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 253-264 ,2005.

[25] Preparata F. P. and M. I. Shamos. "Computational Geometry: An Introduction." Springer-Verlag, New York, Berlin, etc., 1985.

[26] Sun, S., Huang, Z., Zhong, H., Dai, D., Liu, H., Li, J,."Efficient monitoring of skyline queries over distributed data streams." Knowledge and Information System25,575-606 ,2010.

[27] Tan K.L., Eng P.K., Ooi B.C. " Efficiecient progressive skyline computation." Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 301{310 (2001)

[28] Tao Y., Xiao X., Pei J. " Subsky: Efficient computation of skylines in subspaces.", Proceedings of International Conference on Data Engineering (ICDE), p. 65 ,2006.

[29] Vlachou, A., Nrvag, K.,"Bandwidth-constrained distributed skyline computation. In: Proceedings of the International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE), "pp. 17-24 ,2009.

[30] Wu P., Zhang C., Feng Y., Zhao B., Agrawal D., Abbadi,A. "Parallelizing skyline queries for scalable distribution.", Proceedings of International Conference on Extending Database Technology (EDBT), pp. 112-130 ,2006.

[31] Yuan Y., Lin X., Liu Q., Wang W., Yu J.X., Zhang Q. " Efficient computation of the skyline cube. ", Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 241-252 ,2005.

[32] .http://en.wikipedia.org/wiki/SQL.