Scalable Processing of Dominance-Based Queries

Skyline query and *top-k* dominating queries

|  |  |
| --- | --- |
| Emmanouil Kalyvas  School of Informatics  Aristotle University  Thessaloniki, Greece  kalyvase@csd.auth.gr | Vasileios Ntaoulas  School of Informatics  Aristotle University  Thessaloniki, Greece  ntaoulasv@csd.auth.gr |

ABSTRACT

The information age has brought about unprecedented amounts of data, which has in turn led to the development of big data and the use of analytics to make sense of it. Data management has become a critical aspect of this process, as it involves organizing and storing data in a way that makes it accessible and usable. Query operators are a type of tool that allow users to retrieve specific data from a database by specifying certain criteria. Overall, the proliferation of data and the ability to analyze it has had a huge impact on a variety of fields and is likely to continue to shape the way we live and work in the future. In this paper, we study the skyline query, a decision support mechanism, and *top-k* queries, a rank-aware approach. Skyline query retrieves the value-for-money options of a dataset by identifying the objects that present the optimal combination of the dataset’s characteristics [1]. On the other hand, *top-k* queries define a cumulative scoring function in order to retrieve the best results of a dataset since this will reduce the potential multidimensional comparisons of data to a single scalar value.

KEYWORDS

Scala; Spark; Big Data; Dominance; top-k; skyline; queries

1 INTRODUCTION

As decision support systems have grown in popularity and multidimensional data has become larger, researchers have sought ways to efficiently process data and gain insights from it. Operational research, which involves using advanced analytical methods such as mathematical models and data mining to support decision making, is one area that has developed a number of rank-aware approaches, including those used in top-k queries. However, these approaches can sometimes be limited by their reliance on a single scoring function, which may not capture the full complexity of multi-dimensional data.

Skyline queries offer an alternative approach that is more intuitive for humans to understand. Unlike *top-k* queries, which use specific ranking functions and criteria, skyline queries consider the preferences of users over the attributes of the data. These preferences, which might include things like personal likes and dislikes or preferred vacation destinations, are used to identify the subset of data that is most interesting and preferred based on the preferences of all users. This subset, known as the skyline set or pareto optimal set, represents the most interesting and preferred items in the dataset.

In recent years, the processing of skyline queries has become a significant topic in database research, as they are useful for extracting interesting objects from multi-dimensional datasets in a way that takes into account the preferences of users. The simplicity and applicability of the skyline operator to multi-criteria decision support based on user preferences have made it a popular choice for many applications.

1.1 Skyline

Skyline processing was first studied in a single-database environment, i.e., in a centralized setup. As nowadays data are increasingly stored and processed in a distributed way, skyline processing over distributed data has attracted much attention recently [2]. They have been an active area of research in computer science, and over the years, there have been numerous advances in the algorithms and techniques used to support skyline queries. These advances have made it possible to process large datasets more efficiently, and to support more sophisticated preference criteria.

Some of the most important benefits of using skyline queries are:

* They allow users to easily identify the most preferred objects in a dataset, based on multiple criteria.
* They can be used to personalize search results, by allowing users to specify their own preference criteria.
* They can be used to support decision making by presenting a ranked list of the most preferred options.
* They do not require the use of a specific ranking function; their results only depend on the intrinsic characteristics of the data.

Skyline computation has recently received considerable attention in the database community, especially for progressive methods that can quickly return the initial results without reading the entire database [3]. For example, consider a database that contains information about cars. Each tuple of the database is represented as a point in a data space consisting of numerous dimensions (such as the car’s price, the car’s highspeed etc.). Assume a user is looking for cars with low price and not so old. In this case, it is not obvious if the user would prefer: a car with a low price but older, or a new one but more expensive. The skyline query retrieves all cars for which no other car exists that is cheaper and less old.

A more clear definition of skyline has been given in [2]:

(Skyline): A point p ∈ S is said to dominate another point q ∈ S, denoted as p ≺ q, if (1) on every dimension di ∈ D, pi ≤ qi and (2) on at least one dimension dj ∈ D, pj < q j. The *skyline* is a set of points SKY(S) ⊆ S that are not dominated by any other point. The points in SKY(S) are called skyline points.

Fig. 1 shows an example of how skyline points are represented in 2-d space (green dots). Of course, this can be extended for more dimensions.

Chart, scatter chart

Description automatically generated

**Fig. 1:** Example of 3 skyline points

1.2 *Top-k* dominating query

Top-k dominating queries retrieve the *k* data objects that dominate the highest number of data objects in a dataset. The *top-k* dominating query was first introduced by Papadias [4] as an extension of the skyline query, but nowadays they have become an important tool for decision support, data mining, web search, and multi-criteria retrieval applications and have also been studied by different perspectives, such as in indexed and non-indexed multi-dimensional data using efficient exact computation algorithms, in uncertain data using randomized algorithms and in data streams [5].

As a *top-k* query, the user can bound the number of returned results through the parameter *k* and provide a ranking score function (usually monotone) to order the objects by their scores and, thus, retrieve the top-k best objects. They do not provide an objective order of importance for the points, because their results are sensitive to the preference function used [6]. For example, consider the same database that contains information about cars. Each tuple of the database is represented as a point in a data space comprising numerous dimensions (such as the car’s price, the car’s highspeed etc.). Assume a user is looking for a car with low price and not so old. Here, it is not obvious if the user would prefer: a car with a low price but older, or a new one but more expensive. The *top-k* dominance query searches among all cars and yields as a result *k* cars that provide the best combination of low price and new in ascending order, i.e., *top-1* being the best and *top-k* being the worst among those *k* cars.

The authors of [5] defined *top-k* queries as:

The object p = (p..x1, p..x2, …, p..xd) ∈ D dominates another object q = (q..x1, q..x2, …, p..qd) ∈ D, i.e., p ≺ q, when: ∀i ∈ {1, …, d} : p..x1 ≤ q..x1 Λ ∃i ∈ {1, …, d} : p..x1 < q..x1. This means that p is as good as q in all dimensions, and it is strictly better than q in at least one dimension. Then, the domination score of p, dom(p) is defined as: dom(p) = | {q ∈ S : p ≺ q} |. A top-k dominating query returns the k objects with the maximum domination scores in D.

Fig. 2 shows an example of *top-4* dominant points represented in 2-d space. Of course, this can be extended for more dimensions.

Chart, scatter chart

Description automatically generated

**Fig. 2:** Example of *top-k* dominant points (k = 4)

The rest of the paper is organized as follows. Section 2 reviews the implementation of the case problem, for each given task: Given a potentially large set of d-dimensional points, where each point is represented as a d-dimensional vector, we need to detect interesting points. Section 3 presents the results and, finally, Section 4 summarizes the experiment and concludes the paper.

2 IMPLEMENTATION

2.1 Task 1

2.2 Task 2

2.3 Task 3

3 RESULTS

4 CONCLUSION

In the below paragraph, it is explained how alt-txt value is placed in **MS Word 2010**. To add alternative text to a picture in Word 2010, follow these steps:

1. In a Word 2010 document, insert a picture.
2. Right click on the inserted picture and select the **Format Picture** option.
3. Select the **Alt Txt** option from the left-side panel options.
4. In the "Title:" and "Description:" text boxes, type the text you want to represent the picture, and then click "Close".

Below are steps to place alt-txt value in **MS Word 2013/2016**. To add alternative text to a picture in Word 2013/2016, follow these steps:

1. In a Word 2013/2016 document, insert a picture.
2. Right click on the inserted picture and select the **Format Picture** option.
3. In the settings at the right side of the window, click on the "Layout & Properties" icon (3rd option).
4. Expand **Alt Txt** option.
5. In the "Title:" and "Description:" text boxes, type the text you want to represent the picture, and then click "Close".

*1.1.1 Heading Level 3.* Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

*1.1.1.1 Heading Level 4.*Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

ACKNOWLEDGMENTS

Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

REFERENCES

[1] Christos Kalyvas and Theodoros Tzouramanis. 2017. A survey of skyline query processing. arXiv preprint arXiv:1704.01788 (2017).

[2] Hose, K., Vlachou, A. A survey of skyline processing in highly distributed environments. The VLDB Journal 21, 359–384 (2012). https://doi.org/10.1007/s00778-011-0246-6.

[3] Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. 2005. Progressive skyline computation in database systems. ACM Transactions on Database Systems (TODS) 30, 1 (2005), 41–82.

[4] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive Skyline Computation in Database Systems. TODS, 30(1):41–82, 2005.

[5] Eleftherios Tiakas, A. Papadopoulos, and Y. Manolopoulos. 2015. Top-k Dominating Queries : a Survey.Conference Name:ACM Woodstock conference

[6] Yiu, M.L., Mamoulis, N. Multi-dimensional top-k dominating queries. The VLDB Journal 18, 695–718 (2009). https://doi.org/10.1007/s00778-008-0117-y

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00