Journal of King Saud University – Computer and Information Sciences (2016) **28**, 447–456

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Skyline computation for frequent queries in update intensive environment

R.D. Kulkarni *, **B.F. Momin**

Department of Computer Science and Engineering, Walchand College of Engineering, Sangli, Maharashtra, India

Received 27 March 2014; revised 7 March 2015; accepted 14 April 2015
Available online 2 December 2015

KEYWORDS

Skyline queries;
Frequent queries;
Query Profiler

Abstract The skyline queries produce the tuples which are 'promising' on the dimensions of the user's interest. The popular datasets often get queried by the users where dimensions of the user queries often overlap. For such frequent, overlapping skyline queries repeating computations on large datasets result in unacceptable response time. In the scenarios where, there exists a little deviation in the query dimensions than those of the popular dimensions or when the dataset gets updated, the re-use of the previous results can help in either avoiding or reducing further computational costs.

In this paper we focus exactly on this problem and aim at optimizing the response time of frequent or near to frequent skyline queries raised against the static and the update intensive dataset. We propose two novel, simple yet efficient algorithms namely the QPSkyline and the QPUpdateSkyline algorithm which make use of the proposed data structure called as 'Query Profiler' which aims at preserving the metadata of the skyline queries. The QPSkyline algorithm works in static environment and the QPUpdateSkyline algorithm is applicable for the datasets which experience frequent updates. The experiments performed on the real life dataset demonstrate the effectiveness and scalability of the proposed algorithms.

1. Introduction

The computational geometry has highlighted 'maximum vector problem' which aims at finding the 'maximum like hood' of some of the dimensions of the data. The concept of skyline queries stands up on similar lines. The skyline query helps to identify the "best" objects in a multi-attribute dataset. To understand the concept of the skyline query consider an example of a person going on holiday to Goa and looking for a hotel that is both cheap and close to the beach. This means the person is interested in all such hotels that are not worse than any other hotel on both dimensions. This set of interesting hotels is the "Skyline". From the skyline, one can make the decision after weighing the personal or the imposed preferences. The concept of the skyline was proposed for the first

* Corresponding author.
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2015.04.003

time by Borzsonyi et al. (2001). They defined the skyline as a set of points which are not dominated by any other points in the dataset. This definition uses the concept of 'dominance'. A point is said to dominate other points when it is better on at least one dimension than the other points. As an example, consider the following sample dataset of hotels shown in Fig. 1(a) which includes dimensions such as hotel-name prize and distance to the beach.

The tuple $h3$ dominates all other tuples on dimensions of distance and the tuple $h1$ dominates all other tuples on the dimension of prize. However due to the multiple preferences given in the query, both the tuples $h1$ and $h3$ formulate the skyline. Every other point is said to be 'out of the sky'! Fig. 1(b) demonstrates this concept.

A common observation is that, the dataset often gets queried on popular dimensions! The users may generate queries which are specific to certain dimensions only or a slight deviation from those of the popular dimensions is observed. In such scenarios, peculiar to the dataset there exists a set of frequent queries and a set of near to frequent queries. By near to frequent queries, we mean all those queries where dimensions the user queries often overlap. That is either new dimensions are involved or a few dimensions are skipped from the set of popular dimensions. In this scenario, the cost of recomputation of the skyline queries can either be saved or minimized by maintaining the profile of the frequent queries in some way. This can be done if we save the results of the frequent queries in some way and make use of them to optimize the response time of the frequent or near to frequent queries which follow in future. Also the dataset undergoes the updates like addition of new records, deletion of the records or updates on the field values. Upon such updates on the dataset, repeating the skyline computation for a previously processed skyline query is a useless task as the updates may not always affect the existing skyline. Hence the problem is to avoid or reduce the re computational efforts whenever the updates made to a dataset do not affect the existing skylines of various queries. This is the motivation behind our research. We aim at optimizing the response time of the frequent and near to frequent skyline queries in update intensive environment though means of preserved statistics of the queries. To serve this purpose we propose the novel data structure called "*Query Profiler*". We also propose the algorithms '*QPSkyline'* which aims at minimizing the skyline computation efforts of the frequent and near to frequent queries and the algorithm '*QPUpdateSkyline'* which has the similar aim of that of the *QPSkyline'* algorithm however it works in the update intensive environment.

Through this paper we contribute as follows:

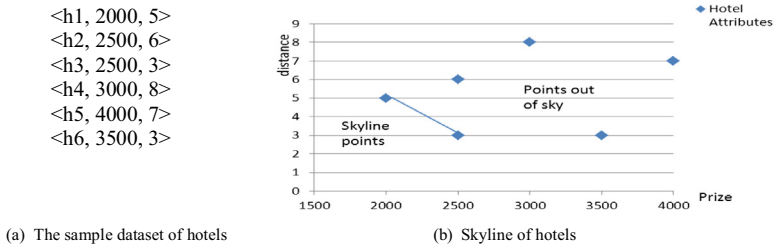
(1) We introduce the concept of *'Query Profiler'* for maintaining the metadata of the skyline queries.
(2) We propose a simple yet efficient algorithm *QPSkyline* which makes use of the *Query Profiler* to optimize the response time of frequent and near to frequent skyline queries.
(3) We also propose the *QPUpdateSkyline* algorithm which aims at optimizing the response time of such skyline queries in update intensive environment.
(4) We present the experimental results to assert the effectiveness of the proposed algorithms

The rest of the paper is organized as follows. Section 2 presents a brief review of the previous techniques related to the skyline computation. In Section 3, the proposed concept of *Query Profiler* (*QP*) has been discussed along with the *QPSkyline* algorithm, related experimentation and the analysis of the obtained results. Section 4 covers the *QPUpdateSkyline* algorithm along with the experimental results and related discussion. Section 5 concludes the paper.

2. Background and related work

Given a n dimensional relation R, a tuple t_i $(v_{i1}, v_{i2}, \ldots, v_{in})$ dominates other tuple t_j $(v_{j1}, v_{j2}, \ldots, v_{jn})$ if on all the dimensions t_i is as good or better than t_j and on at least dimension d, v_{id} is better than v_{jd}. The dominance between two tuples is denoted by $t_i > t_j$. The preferences of the dimensions are specified in the skyline query. (for example minimum prize, minimum distance). A tuple t is said to be in skyline of the relation, if there does not exist any other tuple in R, which dominates t.

Borzsonyi et al. proposed the concept of skyline operator (Borzsonyi et al., 2001). They proposed two basic algorithms namely 'Block Nested Loop' (*BNL*) and 'Divide and Conquer' (*D&C*) which compute the skyline by reading and processing all the input tuples. *BNL* (used by QPSkyline and QPUpdateSkyline) maintains a *window* of incomparable tuples in the main memory. When a tuple p is read from the input, it is compared with all the tuples in the window and if it is found to be a dominated tuple, it is eliminated from the consideration. In the other case (when it is found to be a dominating



(a) The sample dataset of hotels (b) Skyline of hotels

Figure 1 Sample dataset and skyline of it.