Information Systems 65 (2017) 22–35

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

An efficient approach to finding potential products continuously

Yu-Ling Hsueh[a], He Ma[b,*], Chia-Chun Lin[a], Roger Zimmermann[c]

[a] Department of Computer Science & Information Engineering, National Chung Cheng University, Taiwan
[b] Sino-Dutch Biomedical Information and Engineering School, Northeastern University, Shenyang, China, 110169
[c] School of Computing, National University of Singapore, Singapore

ARTICLE INFO

Keywords:
Skyline Queries
Query Processing
Multi-dimensional Databases
Data Management

ABSTRACT

Skyline points and queries are important in the context of processing datasets with multiple dimensions. As skyline points can be viewed as representing marketable products that are useful for clients and business owners, one may also consider non-skyline points that are highly competitive with the current skyline points. We address the problem of continuously finding such potential products from a dynamic d-dimensional dataset, and formally define a potential product and its upgrade promotion cost. In this paper, we propose the *CP-Sky* algorithm, an efficient approach for continuously evaluating potential products by utilizing a *second-order skyline* set, which consists of candidate points that are closest to regular skyline points (also termed the *first-order* skyline set), to facilitate efficient computations and updates for potential products. With the knowledge of the second-order skyline set, *CP-Sky* enables the system to (1) efficiently find substitute skyline points from the second-order skyline set only if a first-order skyline point is removed, and (2) continuously retrieve the top-k potential products. Within this context, the *Approximate Exclusive Dominance Region* algorithm (*AEDR*) is proposed to reduce the computational complexity of determining a candidate set for second-order skyline updates over a dynamic data set without affecting the result accuracy. Additionally, we extend the *CP-Sky* algorithm to support the computations of top-k potential products. Finally, we present experimental results on data sets with various distributions to demonstrate the performance and utility of our approach.

1. Introduction

Skyline queries have made a huge contribution to data filtering over large data sets with multiple dimensions for decision makers. The formal definition of skyline queries is given as follows: given a data set P in d-dimensional space, a distinct object set S is returned, where S contains all objects p_i which is not dominated by another object p_j in P. We say p_1 dominates p_2 ($p_1 \prec p_2$ for short), if and only if p_1 is better than or equal to p_2 on all dimensions, and p_1 is strictly better than p_2 on at least one dimension. The early work on skyline queries assumed that data objects are static [26,28]. Subsequently, the existing approaches [12,16,27,38] have addressed the efficient update support for skyline queries over moving objects with d dynamic dimensions, each of which represents a spatial or non-spatial value. As skyline points can be viewed as marketable products, one may also consider the non-skyline points that are highly competitive with the current skyline points, and we term such non-skyline points as *potential products*. We use the following examples to describe the motivation of our key idea.

Example 1 (*Finding potential products*). For hotel customers, the final decision regarding choosing hotels is made based on multiple hotel attributes (*e.g.*, lower price, closer to the city center, higher rating). With the assistance of skyline queries, the candidate hotels are provided, while most of the undesirable hotels are filtered out, even those with very competitive attributes. In Table 1, the skyline points are $\{p_1, p_2\}$. However, one may also observe that p_5, eliminated by p_2 during the skyline query processing, is very competitive with p_2. When the two skyline points p_1 with Price=$209, Distance (to the city center) =0.1 miles, and Rating=3, and p_2 with Price=$89, Distance=0.8 miles, and Rating=4 are provided, p_5 might also be a good alternative for the customer, because the price of p_5 is only $1 more than that of p_2 and it is a bit farther to the city center. From another perspective, a hotel owner might be seeking an opportunity to increase business sales. By searching for the potential products, the owner may adjust the price or other adjustable attributes of the hotel to increase its competitiveness. We refer to p_5 as a potential product, because the cost of upgrading p_5 to a marketable product is minimal among other non-skyline products.

Example 2 (*Finding Potential Products Continuously*). For stock traders, when they choose to either buy or sell a certain stock, multiple attributes (*e.g.*, current stock price, number of shares they own, binding price and number of shares offered by other traders,

* Corresponding author.
E-mail addresses: hsueh@cs.ccu.edu.tw (Y.-L. Hsueh), mahe@bmie.neu.edu.cn (H. Ma), lcc103p@cs.ccu.edu.tw (C.-C. Lin), rogerz@comp.nus.edu.sg (R. Zimmermann).

http://dx.doi.org/10.1016/j.is.2016.10.003
Received 20 October 2015; Received in revised form 14 August 2016; Accepted 16 October 2016
Available online 20 October 2016

Table 1
An example of finding potential products using a hotel data set. The skyline points are $\{p_1, p_2\}$.

Hotel ID	Price	Distance	Rating
p_1	$209	0.1 miles	3
p_2	$89	0.8 miles	4
p_3	$95	5 miles	2
p_4	$250	0.5 miles	3
p_5	$90	1.1 miles	4

company news, *etc.*) affect their final decisions. Similar to Example 1, stock traders also need to monitor these potential products and buy/sell the shares once they change to become skyline products, in order to earn maximum profits. Since these attributes are updated frequently, continuously finding potential products for an extended period of time is significantly important for stock traders.

A *promotion* [11,20,29,31,30] is a further action to upgrade a potential product (*i.e.*, to become a skyline point), when the business owners are willing to adjust one or some of its attributes. One must consider the promotion cost, which is the least cost to perform a promotion operation.

Example 3 (*Promotion*). Consider that a promotion operation is conducted on p_5. The promotion cost for p_5 is taken by decreasing the price (an adjustable attribute) by at least \$1+ϵ to turn p_5 into a marketable product, where ϵ is a small positive value. For example, the price of p_5 is set to \$88.99, the largest price strictly better (smaller) than \$89. The final marketable products (*i.e.*, skyline points) after the promotion operation are $\{p_1, p_2, p_5\}$.

The objective of the proposed algorithm is to continuously retrieve the potential products (*i.e.*, non-skyline data points) with the minimal promotion cost from a dynamic data set, as some of the data attributes change over time. These changes are caused by real time customer feedback (*e.g.*, ratings) or the adjustments made by the business owners for balancing the cost and profit. Finding potential products is useful for further decision making, especially when investigating the potential products for promotion. To achieve an efficient computation of potential products continuously, the following challenges must be addressed: (1) an effective query result update mechanism is needed to provide a short response time when reporting the current query results, and (2) an efficient strategy to reduce the search space dimensionality is also required.

Peng et al. [29,31,30] have addressed a similar idea for finding potential products. Their main emphasis is to find potential "stars" in social networks whose graphical data are transformed into a coordinate system before the search is conducted. Several pruning techniques, such as Skyboundary and Infra skyline, have been designed to reduce the search space; however, heavy computation is still incurred in order to retrieve the alternate skyline set, and the performance degrades when processing a large data set. Furthermore, none of these approaches consider dynamic data, regardless of the fact that data attributes in social networks change frequently. The use of a Skyband [27] query helps to find a set of non-skyline points which are dominated by at most k points. The potential products, however, have no direct association with the number of dominated points. In other words, the points in a 1-Skyband are not guaranteed to contain the potential product, which might be dominated by several skyline points. As a result, one must perform k-Skyband, where k is set to be as large as the number of skyline points to ensure finding the potential products correctly.

In this paper, to efficiently answer the queries of potential product computations from a dynamic data set, we utilize the *second-order skyline* as a candidate set such that the query processor can avoid accessing the entire dynamic data set with high dimensionality. An example of finding potential products is shown in Fig. 1, where the black solid points are the skyline set (*i.e.*, the first-order skyline, $S1 = \{s_1^1, s_1^2, s_1^3, s_1^4\}$), and the solid grey points represent the second-order skyline set ($S2$ for short). The potential product with the minimal promotion cost is represented by $s_2^5 \in S2$, as an arrow from ${s_2}^5$ indicates the minimal cost to promote ${s_2}^5$. The use of potential product computations enables a user to search for such potential products that are highly competitive with the existing marketable products. A promotion operation (*i.e.*, reducing the dimensional values) can be conducted afterwards to upgrade the potential products. The second-order skyline is utilized as a fundamental technique to answer such a query over dynamic data sets and enable to continuously maintain the efficient updates of $S1$ and $S2$ sets. We propose the **C**ontinuous **P**otential **Sky**line algorithm (*CP-Sky* for short) in this paper. We show that the second-order skyline technique, which avoids expensive calculations on large data sets, facilitates the potential product computations as well as the first-order skyline updates, which need to be completed first before finding potential products, because the promotion cost is computed based on the first-order skyline.

The remainder of this paper is organized as follows. Section 2 describes the related work. We formally define a potential product and address the problem statement of our work in Section 3. Sections 4.1 and 4.2 present the details of the baseline algorithm and our proposed approach to compute potential products, respectively. In particular, we detail the updated technique for $S1$ and $S2$ in Section 4.2.1, and the *CP-Sky* algorithm in Section 4.2.2. The *CP-Sky* algorithm is further extended to support the computations of top-k potential products, and the details of which are described in Section 4.2.3. We extensively verify the performance of our technique in Section 5, and finally conclude with Section 6.

2. Related work

Börzsönyi et al. [5] proposed the straightforward non-progressive *Block-Nested-Loop* (BNL) and *Divide-and-Conquer* (DC) algorithms for static skyline processing. The BNL approach recursively compares each data point with the current set of candidate skyline points, which might be dominated later. BNL does not require data

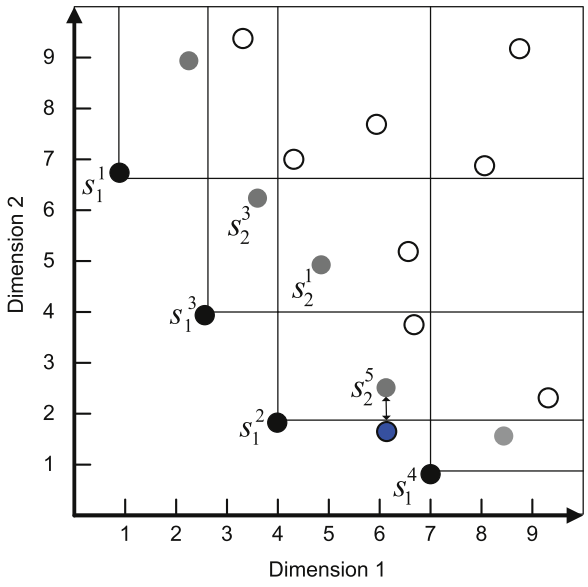

Fig. 1. An illustration of the potential product: ${s_2}^5$.