Information Systems 38 (2013) 286–301

Contents lists available at SciVerse ScienceDirect

Information Systems

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

The Farthest Spatial Skyline Queries

Gae-won You, Mu-Woong Lee, Hyeonseung Im, Seung-won Hwang*

Pohang University of Science and Technology, Republic of Korea

ARTICLE INFO

Article history:
Received 18 August 2011
Received in revised form
13 May 2012
Accepted 13 October 2012
Recommended by: F. Korn
Available online 23 October 2012

Keywords:
Pareto-optimum
Skyline query
Spatial database

ABSTRACT

Pareto-optimal objects are favored as each of such objects has at least one competitive edge against all other objects, or "not dominated". Recently, in the database literature, skyline queries have gained attention as an effective way to identify such pareto-optimal objects. In particular, this paper studies the pareto-optimal objects in perspective of facility or business locations. More specifically, given data points P and query points Q in two-dimensional space, our goal is to retrieve data points that are farther from at least one query point than all the other data points. Such queries are helpful in identifying spatial locations far away from undesirable locations, *e.g.*, unpleasant facilities or business competitors. To solve this problem, we first study a baseline Algorithm TFSS and propose an efficient progressive Algorithm BBFS, which significantly outperforms TFSS by exploiting spatial locality. We also develop an efficient approximation algorithm to trade accuracy for efficiency. We validate our proposed algorithms using extensive evaluations over synthetic and real datasets.

1. Introduction

With the advent of GPS devices, it is becoming important for facility or business location queries to identify an optimal location from massive location data, as we illustrate with the following two real-life examples.

Example 1 (*Facility location*). Consider a scenario where a desirable construction site for a new park is needed. Among potential locations P, we want to avoid locations close to unfavorable sites Q such as chemical plants.

Example 2 (*Business location*). Consider a scenario of seeking for a desirable location for a new Starbucks franchise. Among potential locations P, an optimal location will be those far from the locations of competing branches Q.

For such an optimization problem, we consider a classical notion of "pareto-optimality" [1] in economics such that a point p is *pareto-optimal*, if no other point p' is farther from all undesirable locations (or, p' dominates p). Fig. 1(a) illustrates such a scenario in Example 1. Among 10 potential locations, we aim at finding a pareto-optimal subset of locations, far from two query points represented as two solid rectangles (*i.e.*, a chemical plant and a landfill). An alternative solution is to define "hard" ranges around the two rectangles to avoid such locations, which is hard to define and often require a series of refinements, as observed in [2], with conditions being too strict (or too relaxed). In clear contrast, pareto-optimization is a "soft" query, not burdening users to elaborate such ranges.

Finding pareto-optimal solutions has been actively studied in the database literature as skyline queries [3–7]. In fact, Fig. 1(a) can be restated as a classic skyline problem. Observe from this figure that skyline point p_2 is more favorable than non-skyline point p_4, as p_2 is farther from both unfavorable sites than p_4, *i.e.*, p_2 dominates p_4. We can transform our problem into an equivalent general skyline query problem, as Fig. 1(b) illustrates, such that

* Corresponding author. Tel.: +82 54 279 2385.
E-mail addresses: gwyou@postech.ac.kr (G.-w. You), sigliel@postech.ac.kr (M.-W. Lee), genilhs@postech.ac.kr (H. Im), swhwang@postech.ac.kr (S.-w. Hwang).

0306-4379/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.is.2012.10.001

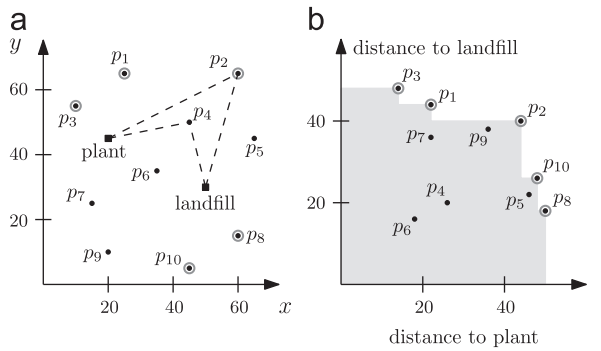

Fig. 1. Example of pareto-optimal locations. (a) Original space and (b) transformed space.

the x-axis and y-axis correspond to the distances of data points from the chemical plant and the landfill, respectively. The results of the pareto-optimal query thus correspond to the results of the general skyline query in this transformed space, *i.e.*, $\{p_1,p_2,p_3,p_8,p_{10}\}$.

However, though this "transform-then-process" approach qualifies as one naïve solution, such approach is inherently expensive due to the following two reasons. First, as attribute values in the transformed space are determined dynamically with respect to query points, it is infeasible to expect any pre-computed index structure on the transformed space. This restricts from adopting existing state-of-the-art skyline processing algorithms, such as BBS [7], building upon such indices to ensure efficiency. Second, the dimensionality increases up to $|Q|$ after the transformation, which significantly deteriorates the performance of skyline computation.

In clear contrast, our goal is to develop "sub-linear" algorithms without scanning the whole data space, achieved by fully exploiting spatial locality for efficiency in the original spatial space. The most relevant work is *Spatial Skyline Queries* (SSQs), aiming at the "opposite goal" of identifying desirable points that are "closer" to locations of interest, *e.g.*, an airport, a beach, and a conference venue, as query points. In this sense, our problem can be named as *Farthest Spatial Skyline Queries* (FSSQs). These two problems, SSQ and FSSQ, are closely related, when transformed as in Fig. 1(b), as a classical skyline algorithm can apply for both problems, for minimizing and maximizing distances respectively. However, such duality ceases to exist in original spatial space, as Fig. 2 contrasts the results for SSQ and FSSQ, for 400 uniformly distributed data points and 10 query points. Unlike the results for SSQ, clustered within and around the query convex hull, the results for FSSQ are scattered across the entire data space. As a result, many geometric properties observed for SSQ problems in [8,9] no longer hold, and algorithms building upon such properties thus cannot apply to FSSQ, which we will discuss more formally in Section 3.1.

We summarize our contributions as follows:

- We formally define the FSSQ problem for pareto-optimization of facility or business locations and propose Algorithm BBFS for FSSQ, which efficiently produces farthest spatial skyline points in a progressive manner.
- For higher efficiency, we also study an approximation algorithm with little loss of accuracy.
- We implement our proposed algorithms and extensively validate them using synthetic and real-life datasets.

The remainder of this paper is organized as follows: Section 2 briefly reviews related work. Section 3.1 discusses the duality between SSQ and FSSQ. Section 3.2 formally defines our FSSQ problem and Section 3.3 introduces some background theories. We design a baseline algorithm TFSS in Section 4.1, and propose Algorithm BBFS in Section 4.2 and its approximation algorithm in Section 4.3. Section 5 presents experimental results, and Section 6 provides our conclusion.

2. Related work

Various types of spatial queries have been extensively studied in the database literature. The most well known type is the nearest neighbor query [10,11], which retrieves the closest point to a single query point in the depth or best first search manner. Later work studies to extend such queries for multiple query points to retrieve the closest point, with respect to the aggregated distance to the query points [12].

However, such a query requires users to elaborate a desirable aggregation function, which is typically too burdensome for end-users. In contrast, SSQ enables users to identify a desirable set of data points that are closer than the remaining data points in any monotonic aggregated distance function without requiring users to elaborate his own distance function.

We first survey existing work on (1) SSQ and then expand our survey to consider (2) general skyline queries.

(1) *Spatial skyline computation*: Sharifzadeh and Shahabi [8,13] first introduced the SSQ problem and proposed two algorithms: Algorithm B^2S^2, and Algorithm VSh^2 enhancing B^2S^2 using a Voronoi diagram. Son et al. [9] later proposed Algorithm ES which reduces unnecessary dominance tests and outperforms VS^2. Meanwhile, while those studies are limited to formulating the skyline queries in Euclidean space called *constraint-free* network, Deng et al. [14] extended the problem into *constraint-based* road network, and Chen and Lian [15] defined and solved it in the metric space.

(2) *General skyline computation*: General skyline query processing has been the subject of more extensive prior work, compared to SSQ. Börzsönyi et al. [3] pioneered skyline queries in database applications, and devised block nested loop (BNL), divide-and-conquer (D&C) and B-tree-based algorithms. Chomicki et al. [4] developed sort-filter-skyline (SFS) algorithm using pre-sorted lists, which improves the BNL algorithm. Tan et al. [5] proposed progressive skyline computation with bitmaps and indices, which quickly returns partial skyline objects. Kossmann et al. [6] improved the D&C algorithm by partitioning data space using the nearest neighbor object, and pruning out the dominated partitioned area. Papadias