Journal of Korea Multimedia Society Vol. 22, No. 2, February 2019(pp. 210-224)
https://doi.org/10.9717/kmms.2019.22.2.210

Efficient Processing of Spatial Preference Queries in Spatial Network Databases

Hyung-Ju Cho†, Muhammad Attique††

ABSTRACT

Given a positive integer k as input, a spatial preference query finds the k best data objects based on the scores (e.g., qualities) of feature objects in their spatial neighborhoods. Several solutions have been proposed for spatial preference queries in Euclidean space. A few algorithms study spatial preference queries in undirected spatial networks where each edge is undirected and the distance between two points is the length of the shortest path connecting them. However, spatial preference queries have not been thoroughly investigated in directed spatial networks where each edge has a particular orientation that makes the distance between two points noncommutative. Therefore, in this study, we present a new method called ALPS+ for processing spatial preference queries in directed spatial networks. We conduct extensive experiments with different setups to demonstrate the superiority of ALPS+ over conventional solutions.

Key words: Spatial Preference Query, Spatial Network Databases, Range Constraint

1. INTRODUCTION

Recently, location-based services (LBSs) have become popular due to the rapid growth of mobile devices, availability of maps, and easy network access [1, 2]. Thus, many studies have been performed to process spatial queries, such as range queries [3], k nearest neighbor (kNN) queries [4, 5, 6], reverse k nearest neighbor queries [7], and road network distance queries [8, 9].

These spatial queries can be answered based on their distance from the query point. In this study, we investigate the spatial preference queries in directed spatial networks where each edge has a particular orientation that makes the network distance noncommutative, i.e., for two points p_1 and p_2 in a directed graph, $dist(p_1,p_2)=dist(p_2,p_1)$ is not guaranteed. Note that $dist(p_1,p_2)$ indicates the length of the shortest path from p_1 to p_2, whereas $dist(p_2,p_1)$ indicates the length of the shortest path from p_2 to p_1. A spatial preference query returns a ranked list of the k best data objects based on the scores of feature objects, such as facilities or services in the neighborhood of data objects. Spatial preference queries have a wide range of applications including spatial recommender systems and spatial decision support systems. For example, consider a real estate agent who holds a list of available apartments for lease. A customer may want to rank the available apartments with respect to the quality of their locations, quantified by aggregating nonspatial characteristics of other facilities (e.g., parks, schools, hospitals, and markets) in the spatial neighborhood of the apartments.

Fig. 1 presents a motivating example of a spatial preference query in a directed spatial network

※ Corresponding Author: Hyung-Ju Cho, Address: (37224) Gyeongsang-daero 2559, Sangju-si, Gyeongsangbuk-do, Republic of Korea, TEL: +82-54-530-1455, FAX: +82-54-530-1459, E-mail: hyungju@knu.ac.kr
Receipt date: Dec. 25, 2018, Approval date: Jan. 18, 2019

† Department of Software, Kyungpook National University
†† Department of Software, Sejong University
(E-mail: attiq85@gmail.com)
※ This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korean government(MSIP) (NRF-2016R1A2B4009793)

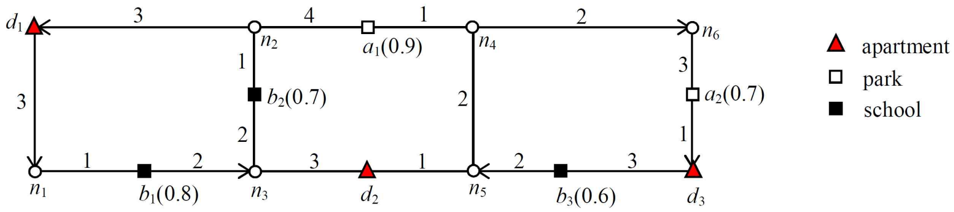


Fig. 1. Motivating example of spatial preference query in a directed spatial network.

where data objects d_1, d_2, and d_3 are represented by triangles and indicate the available apartments for lease. Feature objects a_1 and a_2 are represented by hollow rectangles, and another type of feature objects b_1, b_2, and b_3 are represented by solid rectangles, which indicate parks and schools, respectively. The number on an edge indicates the distance between two neighboring objects, e.g., $dist(d_1,n_1)=3$ and $dist(n_1,b_1)=1$. The number within the parenthesis indicates the score of the feature object beside the number. Consider a scenario where a customer finds a list of available apartments for lease that have good parks or schools in their spatial neighborhoods. For simplicity, assume that the customer has provided a spatial constraint $r=5$ to limit the distance from each available apartment to the eligible parks and schools. If apartments d_1, d_2, and d_3 are sorted based on the scores of parks only, the top-1 apartment becomes d_2 because the scores of d_1, d_2, and d_3 are 0, 0.9, and 0, respectively. Similarly, if the apartments are sorted based on the scores of schools only, the top-1 apartment becomes d1 because the scores of d_1, d_2, and d_3 are 0.8, 0.7, and 0.6, respectively. Finally, if the apartments are sorted based on the sum of the scores of parks and schools, the top-1 apartment becomes d_2 because the scores of d_1, d_2, and d_3 are 0.8, 1.6, and 0.6, respectively.

Several algorithms have been proposed to process spatial preference queries based on Euclidean distance [10, 11, 12]. However, algorithms based on Euclidean distance are not appropriate to spatial network environments. A few algorithms have been developed to evaluate spatial preference queries in undirected spatial networks where all edges are undirected. However, spatial preference queries in directed spatial networks were not yet thoroughly investigated. Our previous work referred to as ALPS [13] is an attempt to evaluate spatial preference queries in undirected spatial networks. Therefore, we propose a new method called ALPS$^+$ to evaluate spatial preference queries efficiently in directed spatial networks. In the proposed method, data objects in a directed segment are collected and then converted into a data segment. All pairs of data segments and feature objects are mapped to a distance-score space, and a subset of the pairs that is adequate to evaluate spatial preference queries is identified. To this end, we devise a mathematical formula that computes the minimum and maximum distances from the data segment to the feature object in directed spatial networks. Finally, we evaluate spatial preference queries efficiently using the materialization of this subset of the pairs, which makes it possible to avoid investigations of redundant feature objects during query evaluation.

This study is an extended version of our previous work on spatial preference query processing in undirected spatial networks. We extend the techniques in [12] to process spatial preference queries in directed spatial networks and present extensive experimental results for efficiency evaluation. The contributions of this study can be summarized as follows.

- We propose a new method called ALPS+ to process spatial preference queries efficiently in directed spatial networks.

- We present materialization strategies to improve the efficiency of the spatial preference search algorithm that exploits grouping of data objects and their skyline sets.
- We conduct extensive experiments with different setups to demonstrate the superiority of ALPS+ over conventional solutions.

The remainder of this paper is organized as follows. In Section 2, we review related studies. In Section 3, we formulate the problem and define the primary terms. In Section 4, we describe the gathering of data objects in a segment and compute the distance from the segment to a point. In Section 5, we elaborate on our solutions for processing spatial preference queries in directed spatial networks. In Section 6, we empirically compare ALPS$^+$ and conventional solutions for different setups. Finally, we conclude this paper in Section 7.

2. RELATED WORK

Several algorithms were developed to process spatial preference queries using Euclidean distance. Yiu *et al.* [11, 12] first introduced spatial preference queries based on three distinct spatial scores, i.e., range, nearest neighbor, and influence scores, and proposed different algorithms to evaluate spatial preference queries for these scores. Rocha-Junior *et al.* [10] developed a materialization technique to speed up the evaluation of spatial preference queries using Euclidean distance. They presented a mapping of pairs of the data object and feature object to a distance-score space. The minimal subset of the pairs that is adequate to answer spatial preference queries is materialized. However, the techniques based on Euclidean distance are not applicable to our problem concerning network distance-based queries.

A few algorithms were developed to answer spatial preference queries in undirected spatial networks. Our previous work called ALPS [13] is an attempt to evaluate spatial preference queries in undirected spatial networks. Similar to [10], ALPS exploits a materialization technique based on the distance-score space. ALPS$^+$ extends the functionality of ALPS. Specifically, ALPS$^+$ can evaluate spatial preference queries in directed spatial networks as well as undirected spatial networks, whereas ALPS can evaluate spatial preference queries only in undirected spatial networks. This study also presents the trade-off between query processing time and index construction time when a materialization technique is applied to process spatial preference queries. Finally, in recent years, different types of spatial queries have been studied extensively. These include range queries [3], *k*NN queries [4, 5, 6], spatial keyword queries [14, 15, 16], and spatial network distance queries [8, 9]. These studies have different problem settings from ours and their solutions are not appropriate.

3. PRELIMINARIES

3.1 Problem formulation

Given a positive integer k, a set of data objects $D=\{d_1,d_2,\cdots,d_{|D|}\}$, and a set of m feature datasets $F_i=\{f_1,f_2,\cdots,f_{|F_i|}\}$ for $1 \le i \le m$, the spatial preference query retrieves a ranked list of the best k data objects with the highest scores. The score of a data object d is determined using the scores of feature objects in the spatial neighborhood of the data object. Each feature object f has a score, denoted by $\sigma(f)$, that indicates its quality, such as user evaluation score of the feature object. The scores of feature objects are normalized in the range $[0,1]$ and can be combined using an aggregation function to derive an overall quality rating.

The score $\gamma(d)$ of a data object d is determined by aggregating the component scores $\gamma_i(d)=max\{\sigma(f)|f\in F_i, dist(d,f)\le r\}$ $(1 \le i \le m)$ with respect to a range condition and the i-th feature dataset F_i and can be formally defined as $\gamma(d)=agg$

$\{\gamma_i(d)|1 \le i \le m\}$ where $agg = \{sum, max, min\}$. The aggregation function *agg* can be any monotone function. This study mainly considers the range constraint. This is because that this study can be easily extended to the nearest neighbor constraint and the influence constraint. Recall that the component score $\gamma_i(d)$ is the highest score of feature objects $f \in F_i$ that satisfy the range constraint of a data object *d*.

3.2 Definition of terms and notations

Directed spatial network A directed spatial network can be modeled using a weighted directed graph $G = \langle N, E, W \rangle$, where N, E, and W indicate the node set, edge set, and edge distance matrix, respectively. Each edge has a positive weight and direction.

Classification of nodes Nodes can be divided into three categories based on the degree of the node. (1) If the degree of a node is equal to or larger than 3, the node is referred to as an intersection node. (2) If it is 2, the node is an intermediate node. (3) If it is 1, the node is a terminal node.

Edge sequence and segment An edge sequence $\overline{n_s n_{s+1} \cdots n_e}$ denotes a path between two nodes, n_s and n_e, such that n_s (or n_e) is either an intersection node or a terminal node, and the other nodes in the path, $n_{s+1}, \cdots, n_{e-1}$, are intermediate nodes. The two end nodes, n_s and n_e, are referred to as boundary nodes of the edge sequence. If an edge sequence forms a cycle, the boundary nodes of the edge sequence are identical. The length of an edge sequence is the total weight of the edges in the edge sequence. A part of an edge sequence is called a segment. Note that by definition, an edge sequence is also a segment defined by the boundary nodes of the edge sequence.

To simplify the presentation, Table 1 presents the notations used in this paper. Our scheme works in the same manner for undirected and directed segments and an undirected segment is used for convenience to describe the proposed scheme.

4. GROUPING AND DISTANCE COMPUTATION

4.1 Grouping of data objects in an edge sequence

The data objects in an edge sequence are gathered and are referred to as data segment. The data objects in a data segment are close to each other

Table 1. Summary of notations used in this paper

Symbol	Definition
$G = \langle N, E, W \rangle$	Graph model of a directed spatial network
$dist(p_1, p_2)$	Length of the shortest path from point p_1 to point p_2
$len(p_1, p_2)$	Length of the segment connecting p_1 and p_2, such that p_1 and p_2 are in the same edge sequence
n_i	Node in a directed spatial network
$\overline{n_i n_j}$	Edge connecting two adjacent nodes n_i and n_j
$\overline{n_s n_{s+1} \cdots n_e}$	Edge sequence where n_s (or n_e) is the start (or end) of the edge sequence and the other nodes, $n_{s+1}, \cdots, n_{e-1}$, are intermediate nodes
r	Range constraint
k	Number of data objects to be retrieved
m	Number of feature datasets
$mindist(dseg, p)$	Minimum distance from a data segment *dseg* to a point *p*
$maxdist(dseg, p)$	Maximum distance from a data segment *dseg* to a point *p*

in the spatial network; therefore, it is more effective to process them together than to process each object separately. However, feature objects in an edge sequence are not grouped because of frequent wide range of variations in their scores.

Fig. 2 shows a sample grouping of data objects in an edge sequence, which will be discussed throughout this section. As shown in Fig. 2(a), four data objects, d_1 through d_4, and four feature objects, f_1 through f_4, are in the directed spatial network. To simplify the presentation, we consider a single feature dataset F_i = $\{\langle f_1,0.9\rangle,\langle f_2,0.5\rangle,\langle f_3,0.3\rangle,\langle f_4,0.7\rangle\}$. Each feature dataset is processed independently; thus, the extension to multiple feature datasets is straightforward. The spatial network includes three edge sequences, $\overline{n_2n_1n_4n_5}$, $\overline{n_2n_5}$, and $\overrightarrow{n_2n_3n_5}$, and two intersection nodes, n_2 and n_5. Fig. 2(b) illustrates the result of grouping data objects in an edge sequence. Specifically, data objects d_1 and d_2 are grouped and transformed into the data segment $\overline{d_1d_2}$, which is represented in the bold line. Similarly, data objects d_3 and d_4 are grouped and transformed into the data segment $\overrightarrow{d_3d_4}$. Therefore, $D=\{d_1,d_2,d_3,d_4\}$ is transformed into $\overline{D}=\{\overline{d_1d_2},\overrightarrow{d_3d_4}\}$, where $\overline{D}$ denotes the set of data segments generated from the data objects in D.

4.2 Computation of minimum and maximum distances from data segment to feature object

Let $dseg\otimes f$ denote a composite object that is generated from a pair of a data segment *dseg* and a feature object *f*, where $dseg\in\overline{D}$ and $f\in F_i$. Here, $dseg\otimes f$ is represented by $dseg\otimes f=([mindist(dseg, f), maxdist(dseg, f)], \sigma(f))$, where *mindist*(*dseg*, *f*) and *maxdist*(*dseg*, *f*) indicate the minimum and maximum distances from *dseg* to *f*, respectively. We plot each $dseg\otimes f$ pair to the distance-score space as shown in Fig. 3, where the *x* value corresponds to the distance from a data segment *dseg* to a feature *f* and the *y* value corresponds to the score of a feature object *f*.

In a preprocessing step, a subset of $dseg\otimes f$ pairs is selected and indexed using an R-tree [17, 18],

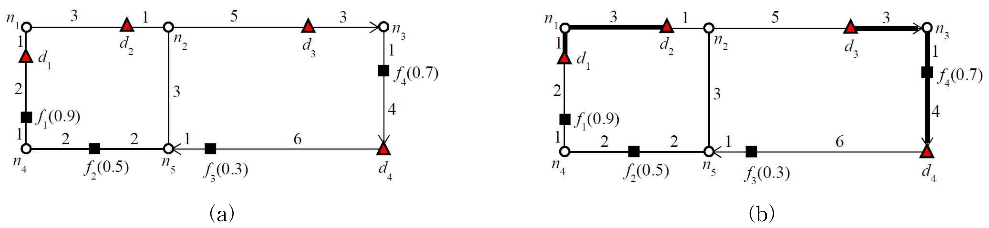


Fig. 2. Grouping of data objects in an edge sequence. (a) Data objects d1 through d4 and (b) Data segments $\overline{d_1d_2}$ and $\overrightarrow{d_3d_4}$.

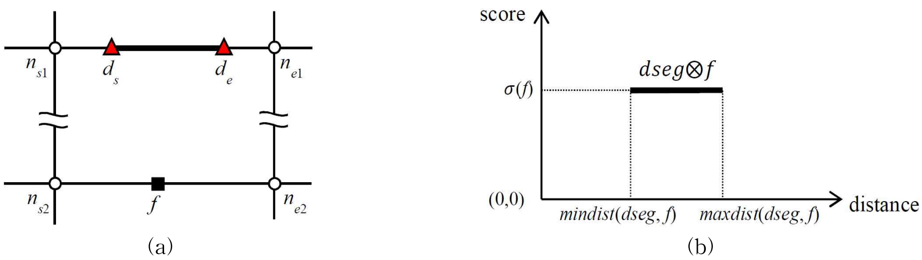


Fig. 3. Mapping of $dseg\otimes f$ to the distance-score space. (a) $dseg=\overline{d_sd_e}$ and (b) $dseg\otimes f=([mindist(dseg, f), maxdist(dseg, f)], \sigma(f))$.

Table 2. Computation of minimum and maximum distances in Fig. 2

$\overline{d_s d_e}$	f	$dist(d_s,f)$	$dist(d_e,f)$	$f \in \overline{d_s d_e}$
$\overline{d_1 d_2}$	f_1	$dist(d_1,f_1)=2$	$dist(d_2,f_1)=6$	$f_1 \not\in \overline{d_1 d_2}$
	f_2	$dist(d_1,f_2)=5$	$dist(d_2,f_2)=6$	$f_2 \not\in \overline{d_1 d_2}$
	f_3	$dist(d_1,f_3)=24$	$dist(d_2,f_3)=20$	$f_3 \not\in \overline{d_1 d_2}$
	f_4	$dist(d_1,f_4)=14$	$dist(d_2,f_4)=10$	$f_4 \not\in \overline{d_1 d_2}$
$\overrightarrow{d_3 d_4}$	f_1	$dist(d_3,f_1)=20$	$dist(d_4,f_1)=12$	$f_1 \not\in \overrightarrow{d_3 d_4}$
	f_2	$dist(d_3,f_2)=17$	$dist(d_4,f_2)=9$	$f_2 \not\in \overrightarrow{d_3 d_4}$
	f_3	$dist(d_3,f_3)=14$	$dist(d_4,f_3)=6$	$f_3 \not\in \overrightarrow{d_3 d_4}$
	f_4	$dist(d_3,f_4)=4$	$dist(d_4,f_4)=19$	$f_4 \in \overrightarrow{d_3 d_4}$

one of the most popular multi-dimensional access methods. To this end, the minimum and maximum distances from *dseg* to *f* must be computed. We now discuss the computation of the minimum and maximum distances from *dseg* to *f* in Fig. 2, where $dseg \in \{\overline{d_1 d_2}, \overrightarrow{d_3 d_4}\}$ and $f \in \{f_1, f_2, f_3, f_4\}$. Table 2 summarizes the computation of the minimum and maximum distances from *dseg* to *f*.

Fig. 4(a), 4(b), 4(c), and 4(d) illustrate the computations of minimum and maximum distances from $\overline{d_1 d_2}$ to each of f_1, f_2, f_3, and f_4, respectively. In the figures, the dashed lines denote that paths from *p* to *f* are not the shortest paths for the corresponding intervals. For data segment $\overline{d_1 d_2}$ and feature object f_1, we have $dist(d_1,f_1)=2$, $dist(d_2,f_1)=6$, and $f_1 \not\in \overline{d_1 d_2}$ as shown in Table 2. Therefore, as shown in Fig. 4(a), the minimum and maximum distances from $\overline{d_1 d_2}$ to f_1 are *mindist*$(\overline{d_1 d_2},f_1)=2$ and *maxdist*$(\overline{d_1 d_2},f_1)=6$, respectively. For data segment $\overline{d_1 d_2}$ and feature object f_2, we have $dist(d_1,f_2)=5$, $dist(d_2,f_2)=6$, and $f_2 \not\in \overline{d_1 d_2}$ as shown in Table 2. Therefore, as shown in Fig. 4(b), the minimum and maximum distances from $\overline{d_1 d_2}$ to f_2 are *mindist*$(\overline{d_1 d_2},f_2)=5$ and *maxdist*$(\overline{d_1 d_2},f_2)=7.5$, respectively. For data segment $\overline{d_1 d_2}$ and feature object f_3, we

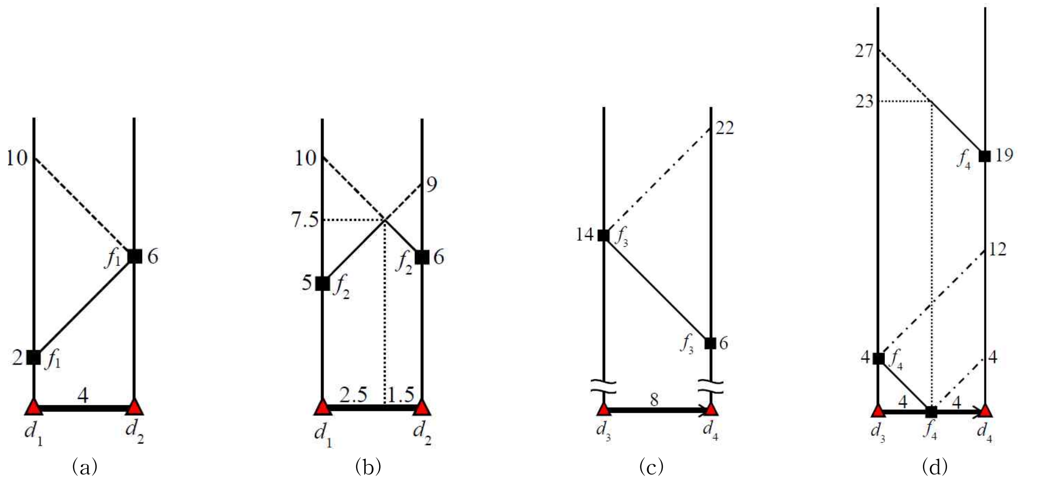


Fig. 4. Evaluation of *mindist*$(\overline{d_1 d_2},f)$ and *maxdist*$(\overline{d_1 d_2},f)$ where $f \in \{f_1,f_2,f_3,f_4\}$. (a) *mindist*$(\overline{d_1 d_2},f_1)=2$ and *maxdist*$(\overline{d_1 d_2},f_1)=6$, (b) *mindist*$(\overline{d_1 d_2},f_2)=5$ and *maxdist*$(\overline{d_1 d_2},f_2)=7.5$, (c) *mindist*$(\overline{d_1 d_2},f_3)=20$ and *maxdist*$(\overline{d_1 d_2},f_3)=24$, and (d) *mindist*$(\overline{d_1 d_2},f_4)=10$ and *maxdist*$(\overline{d_1 d_2},f_4)=14$.

have $dist(d_1,f_3)=24$, $dist(d_2,f_3)=20$, and $f_3 \notin \overline{d_1d_2}$ as shown in Table 2. Therefore, as shown in Fig. 4(c), the minimum and maximum distances from $\overline{d_1d_2}$ to f_3 are $mindist(\overline{d_1d_2},f_3)=20$ and $maxdist(\overline{d_1d_2},f_3)=24$, respectively. Finally, for data segment $\overline{d_1d_2}$ and feature object f_4, we have $dist(d_1,f_4)=14$, $dist(d_2,f_4)=10$, and $f_4 \notin \overline{d_1d_2}$ as shown in Table 2. Therefore, as shown in Fig. 4(d), the minimum and maximum distances from $\overline{d_1d_2}$ to f_4 are $mindist(\overline{d_1d_2},f_4)=10$ and $maxdist(\overline{d_1d_2},f_4)=14$, respectively.

Fig. 5(a), 5(b), 5(c), and 5(d) illustrate the computations of minimum and maximum distances from $\overrightarrow{d_3d_4}$ to each of f_1, f_2, f_3, and f_4, respectively. For data segment $\overrightarrow{d_3d_4}$ and feature object f_1, we have $dist(d_3,f_1)=20$, $dist(d_4,f_1)=12$, and $f_1 \notin \overrightarrow{d_3d_4}$ as shown in Table 2. Therefore, as shown in Fig. 5(a), the minimum and maximum distances from $\overrightarrow{d_3d_4}$ to f_1 are $mindist(\overrightarrow{d_3d_4},f_1)=12$ and $maxdist(\overrightarrow{d_3d_4},f_1)=20$, respectively. For data segment $\overrightarrow{d_3d_4}$ and feature object f_2, we have $dist(d_3,f_2)=17$, $dist(d_4,f_2)=9$, and $f_2 \notin \overrightarrow{d_3d_4}$ as shown in Table 2. Therefore, as shown in Fig. 5(b), the minimum and maximum distances from $\overrightarrow{d_3d_4}$ to f_2 are $mindist(\overrightarrow{d_3d_4},f_2)=9$ and $maxdist(\overrightarrow{d_3d_4},f_2)=17$, respectively. For data segment $\overrightarrow{d_3d_4}$ and feature object f_3, we have $dist(d_3,f_3)=14$, $dist(d_4,f_3)=6$, and $f_3 \notin \overrightarrow{d_3d_4}$ as shown in Table 2. Therefore, as shown in Fig. 5(c), the minimum and maximum distances from $\overrightarrow{d_3d_4}$ to f_3 are $mindist(\overrightarrow{d_3d_4},f_3)=6$ and $maxdist(\overrightarrow{d_3d_4},f_3)=14$, respectively. Finally, for data segment $\overrightarrow{d_3d_4}$ and feature object f_4, we have $dist(d_3,f_4)=4$, $dist(d_4,f_4)=19$, and $f_4 \in \overrightarrow{d_3d_4}$ as shown in Table 2. Therefore, as shown in Fig. 5(d), the minimum and maximum distances from $\overrightarrow{d_3d_4}$ to f_4 are $mindist(\overrightarrow{d_3d_4},f_4)=0$ and $maxdist(\overrightarrow{d_3d_4},f_4)=23$, respectively. Table 3 summarizes the minimum and maximum distances along with the scores for the $dseg \otimes f$ pairs in Fig. 2(b).

5. PROCESSING SPATIAL PREFERENCE QUERIES IN DIRECTED SPATIAL NETWORKS

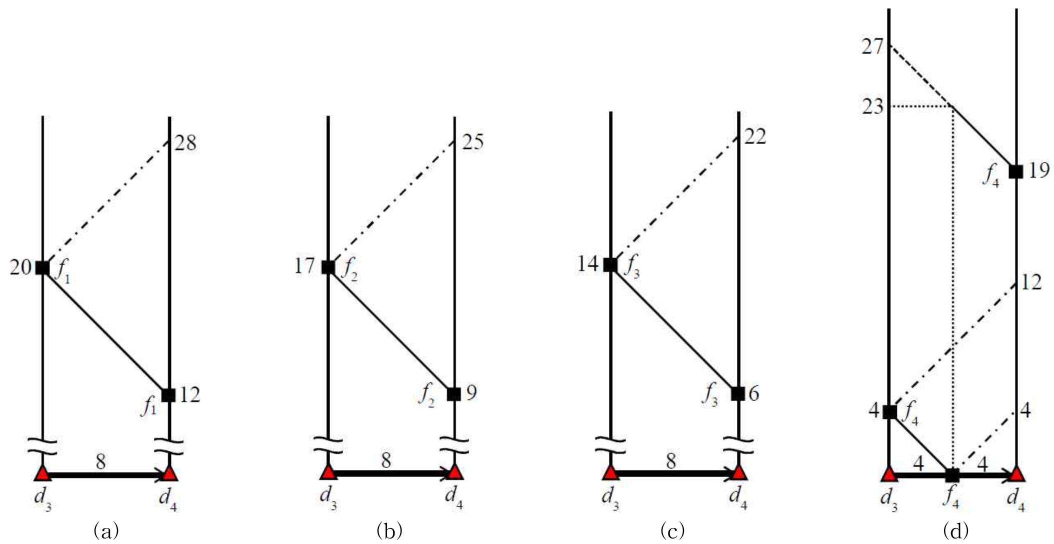


Fig. 5. Evaluation of $mindist(\overrightarrow{d_3d_4},f)$ and $maxdist(\overrightarrow{d_3d_4},f)$ where $f \in \{f_1,f_2,f_3,f_4\}$. (a) $mindist(\overrightarrow{d_3d_4},f_1)=12$ and $maxdist(\overrightarrow{d_3d_4},f_1)=20$, (b) $mindist(\overrightarrow{d_3d_4},f_2)=9$ and $maxdist(\overrightarrow{d_3d_4},f_2)=17$, (c) $mindist(\overrightarrow{d_3d_4},f_3)=6$ and $maxdist(\overrightarrow{d_3d_4},f_3)=14$, and (d) $mindist(\overrightarrow{d_3d_4},f_4)=0$ and $maxdist(\overrightarrow{d_3d_4},f_4)=23$.

Table 3. Summary of all sample $dseg \otimes f$ pairs

$dseg$	f	$dseg \otimes f$
$\overline{d_1d_2}$	f_1	$\overline{d_1d_2} \otimes f_1 = ([2,6], 0.9)$
	f_2	$\overline{d_1d_2} \otimes f_2 = ([5,7.5], 0.5)$
	f_3	$\overline{d_1d_2} \otimes f_3 = ([20,24], 0.3)$
	f_4	$\overline{d_1d_2} \otimes f_4 = ([10,14], 0.7)$
$\overrightarrow{d_3d_4}$	f_1	$\overrightarrow{d_3d_4} \otimes f_1 = ([12,20], 0.9)$
	f_2	$\overrightarrow{d_3d_4} \otimes f_2 = ([9,17], 0.5)$
	f_3	$\overrightarrow{d_3d_4} \otimes f_3 = ([6,14], 0.3)$
	f_4	$\overrightarrow{d_3d_4} \otimes f_4 = ([0,23], 0.7)$

5.1 Mapping pairs of data segment and feature object to distance-score space

We map $dseg \otimes f$ pairs to a distance-score space M, defined by the axes *distance* and *score*. Each $dseg \otimes f$ pair is mapped to either a line segment or a point in the distance-score space M. During the preprocessing step, the dominance relationship is used to remove redundant $dseg \otimes f$ pairs.

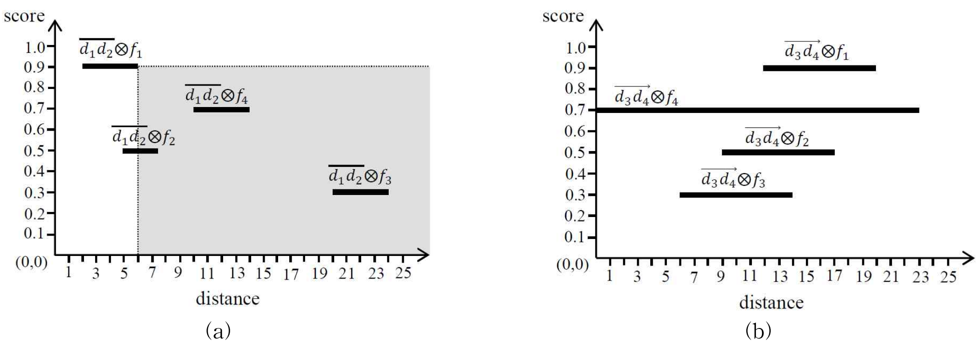
Definition 1 (Mapping of $\overline{D} \otimes F_i$ to M) The mapping of a pair that consists of each data segment $dseg \in \overline{D}$ and each feature object $f \in F_i$ to the distance-score space M is

$\overline{D} \otimes F_i = \{dseg \otimes f | dseg \in \overline{D}, f \in F_i\}$.

Definition 2 (Mapping of $dseg \otimes F_i$ to M) The mapping of a pair that consists of a data segment *dseg* and each feature object $f \in F_i$ to the distance-score space M is $dseg \otimes F_i = \{dseg \otimes f | f \in F_i\}$. $\overline{D} \otimes F_i$ is the union of all $dseg \otimes F_i$ pairs where each $dseg \in \overline{D}$.

Definition 3 (Dominance relationship $\prec$) Given two pairs of $dseg \otimes f_\alpha$ and $dseg \otimes f_\beta$, we state that $dseg \otimes f_\alpha$ dominates $dseg \otimes f_\beta$, denoted as $dseg \otimes f_\alpha \prec dseg \otimes f_\beta$, if $maxdist(dseg, f_\alpha) \leq mindist(dseg, f_\beta)$ and $\sigma(f_\alpha) > \sigma(f_\beta)$, or if $maxdist(dseg, f_\alpha) < mindist(dseg, f_\beta)$ and $\sigma(f_\alpha) \geq \sigma(f_\beta)$.

Fig. 6 shows the mapping of $\overline{D} \otimes F_i$ in Table 3 to the distance-score space M. Specifically, Fig. 6(a) and 6(b) show the mappings of $\overline{d_1d_2} \otimes F_i$ and $\overrightarrow{d_3d_4} \otimes F_i$ to M, respectively. In choosing $dseg \otimes f$ pairs, the shorter distance from a data segment *dseg* to a feature object f as well as the higher score of f is preferred. Therefore, in Fig. 6(a), the



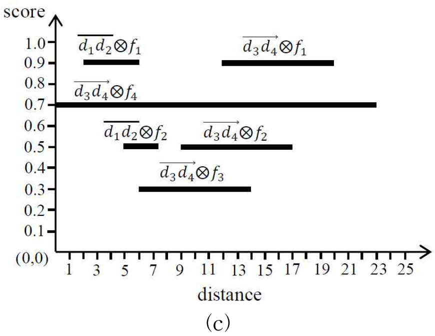


Fig. 6. Mapping of $\overline{D} \otimes F_i$ to M where $\overline{D} = \{\overline{d_1d_2}, \overrightarrow{d_3d_4}\}$ and $F_i = \{f_1, f_2, f_3, f_4\}$. (a) $\overline{d_1d_2} \otimes F_i$, (b) $\overrightarrow{d_3d_4} \otimes F_i$, and (c) $SKY(\overline{D} \otimes F_i)$.

$\overline{d_1d_2}\otimes f_1$ pair dominates both $\overline{d_1d_2}\otimes f_3$ and $\overline{d_1d_2}\otimes f_4$ pairs, which are considered redundant. This dominance is attributed to the smaller $maxdist(\overline{d_1d_2},f_1)$ than $mindist(\overline{d_1d_2},f_3)$ and $mindist(\overline{d_1d_2},f_4)$, and larger $\sigma(f_1)$ than $\sigma(f_3)$ and $\sigma(f_4)$. Thus, $\overline{d_1d_2}\otimes f_3$ and $\overline{d_1d_2}\otimes f_4$ pairs belong to the gray region, which indicates the dominance region of $\overline{d_1d_2}\otimes f_1$ pair. However, as shown in Fig. 6(b), no pair in $\overrightarrow{d_3d_4}\otimes F_i$ is dominated.

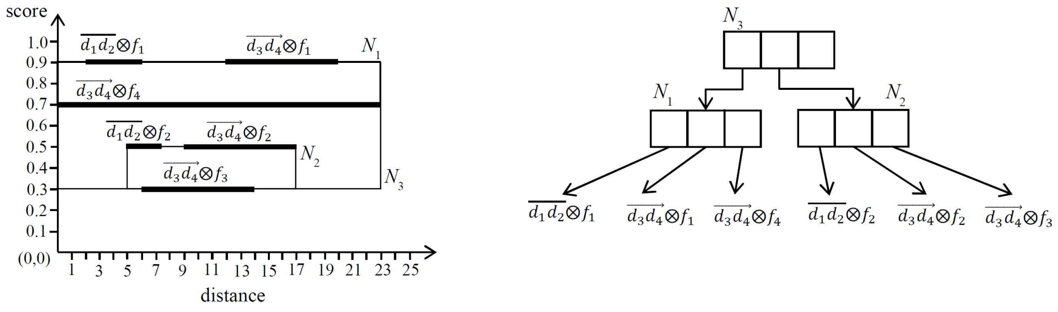
Let $SKY(dseg\otimes F_i)$ be the set of pairs that are not dominated by any other pair in $dseg\otimes F_i$. We define $SKY(dseg\otimes F_i)$ as the skyline set of $dseg\otimes F_i$. Therefore, we have $SKY(\overline{d_1d_2}\otimes F_i)=\{\overline{d_1d_2}\otimes f_1,\overline{d_1d_2}\otimes f_2\}$ and $SKY(\overrightarrow{d_3d_4}\otimes F_i)=\{\overrightarrow{d_3d_4}\otimes f_1,\overrightarrow{d_3d_4}\otimes f_2,\overrightarrow{d_3d_4}\otimes f_3,\overrightarrow{d_3d_4}\otimes f_4\}$. The pairs that are associated with different data segments (e.g., $\overline{d_1d_2}\otimes F_i$ and $\overrightarrow{d_3d_4}\otimes F_i$) cannot be dominated. Finally, the skyline set for $\overline{D}\otimes F_i$ becomes the union of the skyline sets $SKY(dseg\otimes F_i)$ where each $dseg\in\overline{D}$, i.e., $SKY(\overline{D}\otimes F_i)=\bigcup_{dseg\in\overline{D}} SKY(dseg\otimes F_i)$. Consequently, as shown in Fig. 6(c), $SKY(\overline{D}\otimes F_i)=SKY(\overline{d_1d_2}\otimes F_i)\cup SKY(\overrightarrow{d_3d_4}\otimes F_i)$.

Fig. 7(a) illustrates the mapping of the six pairs in $SKY(\overline{D}\otimes F_i)$ to M. Fig. 7(b) shows an R-tree that indexes these six pairs, assuming that the node capacity of the R-tree is set to 3. Specifically, index node N_1 includes $\overline{d_1d_2}\otimes f_1$, $\overrightarrow{d_3d_4}\otimes f_1$, and $\overrightarrow{d_3d_4}\otimes f_4$. Index node N_2 includes $\overline{d_1d_2}\otimes f_2$, $\overrightarrow{d_3d_4}\otimes f_2$, and $\overrightarrow{d_3d_4}\otimes f_3$. $SKY(dseg\otimes F_i)$ is sufficient to obtain the component score of each data object $d\in dseg$.

5.2 Processing spatial preference queries in directed spatial networks

In this section, we present an algorithm, called ALPS$^+$, for processing spatial preference queries in directed spatial networks. For ease of presentation, we focus on elaborating on the algorithm to retrieve top-k data objects based on the range score. We then describe the necessary modifications for supporting NN and influence scores. ALPS$^+$ produces the query result with sequential access to data objects in descending order of their component scores, which is similar to the NRA (No Random Access) algorithm [19]. To obtain the query result, during query processing, ALPS$^+$ retrieves qualifying data objects individually in descending order based on their component scores, which can rapidly produce a set of the k best data objects with the highest score. Recall that we use sum as the aggregation function.

Algorithm 1 returns a set of top-k data objects with the highest scores by adding the component scores of data objects retrieved from max heaps H_i $(1\le i\le m)$. For each skyline set $SKY(\overline{D}\otimes F_i)$, we employ a max heap H_i to explore data objects in descending order based on their component scores $\gamma_i(d)$. The root node of the R-tree R_i that indexes



(a) (b)

Fig. 7. $SKY(\overline{D}\otimes F_i)$ and the corresponding R-tree index. (a) $SKY(\overline{D}\otimes F_i)$ and (b) R-tree index for $SKY(\overline{D}\otimes F_i)$.

Algorithm 1 ALPS$^+$ (k, r)

Input: k: the number of requested data objects, r: range constraint
Output: a set A_k of top-k data objects with the highest score

1: $H_i \leftarrow R_i.root\_node$ $(1 \le i \le m)$
2: $C \leftarrow \varnothing$ // C is the set of candidate data objects
3: $A_k \leftarrow \varnothing$ // A_k is the current top-k set
4: $D_i^{rptd} \leftarrow \varnothing$ $(1 \le i \le m)$
5: $l_i \leftarrow 0$ $(1 \le i \le m)$ // l_i is the last component score seen in H_i
6: **while** there is H_i such that $H_i \neq \varnothing$ **do**
7: $\langle d_{top}, \gamma_i(d_{top})\rangle \leftarrow pop\_data\_object\_with\_highest\_score(H_i, r)$
8: $l_i \leftarrow \gamma_i(d_{top})$
9: $T \leftarrow sum\{l_i | 1 \le i \le m\}$
10: $\gamma_{lb}(d_{top}) \leftarrow \gamma_{lb}(d_{top}) + \gamma_i(d_{top})$
11: **if** $|A_k| < k$ **or** $\gamma_{lb}(d_{top}) > t$ **then**
12: $A_k \leftarrow A_k \cup \{d_{top}\}$
13: **if** $d_{top} \in C$ **then** $C \leftarrow C - \{d_{top}\}$
14: **if** $|A_k| = k+1$ **then**
15: $A_k \leftarrow A_k - \{d_{k+1}\}$
16: $C \leftarrow C \cup \{d_{k+1}\}$
17: $t \leftarrow min\{\gamma_{lb}(d) | d \in A_k\}$
18: **else if** $t < T$ **and** $\gamma_{ub}(d_{top}) \ge t$ **then**
19: $C \leftarrow C \cup \{d_{top}\}$ // see line 21 to evaluate $\gamma_{ub}(d_{top})$
20: **for** each data object $d \in C$ **do**
21: $\gamma_{ub}(d) \leftarrow \gamma_{lb}(d) + sum\{l_i | 1 \le i \le m$ such that $\gamma_i(d)$ has not been seen so far$\}$
22: **if** $\gamma_{ub}(d) < t$ **then** $C \leftarrow C - \{d\}$
23: $u \leftarrow max\{\gamma_{ub}(d) | d \in C\}$
24: **if** $t \ge u$ **and** $|A_k| = k$ **then exit** while statement
25: **return** A_k as the top-k set

$SKY(\overline{D} \otimes F_i)$ is first added to H_i. Max heaps $H_1, H_2, \cdots, H_m$ are accessed in a round robin fashion. Whenever the *pop_data_object_with_highest_score* function detailed in Algorithm 2 is called, the data object d_{top} with the highest component score $\gamma_i(d_{top})$ is popped from max heap H_i (line 7). Let l_i $(1 \le i \le m)$ be the last component score that has been seen in H_i and T be a threshold (i.e., an upper bound) for the aggregate score of the data objects that have not been seen in any H_i yet. Then, l_i is set to $\gamma_i(d_{top})$ and T is updated to $sum\{l_i | 1 \le i \le m\}$ (lines 8–9). The lower bound score $\gamma_{lb}(d_{top})$ of d_{top} is also updated with its component score $\gamma_i(d_{top})$ (line 10). Let A_k be the current top-k set and t be the lowest score of the data objects in A_k. If $|A_k| < k$ or $\gamma_{lb}(d_{top}) > t$, then d_{top} is added to A_k. For simplicity, it is assumed in lines 14–16 that $A_k = \{d_1, d_2, \cdots, d_k, d_{k+1}\}$ and $\gamma_{lb}(d_1) \ge \cdots \ge \gamma_{lb}(d_k) \ge \gamma_{lb}(d_{k+1})$. If $|A_k| = k+1$, the data object with the lowest γ_{lb} (i.e., d_{k+1}) is moved from A_k to C, where C is the set of candidate data objects that may be included in the query result. Finally, t is set to the lowest γ_{lb} from the data objects in A_k (lines 11–17). If $t \ge T$, then no newly seen data object can end up in A_k because T stores the upper bound of the aggregate score of unseen data objects in any max heap H_i. Therefore, if $t < T$ and $d_{top} \notin A_k$, then d_{top} is added to C (lines 18–19). For each candidate object $d \in C$, the upper bound score $\gamma_{ub}(d)$ is computed by $\gamma_{ub}(d) \leftarrow \gamma_{lb}(d) + sum$

$\{l_i | 1 \le i \le m$ such that $\gamma_i(d)$ has not been seen so far$\}$. Then, the highest *ub* of all data objects in C is determined and set to u (lines 20-23). If $t \ge u$ and $|A_k| = k$, or if all of the heaps are exhausted, then the algorithm terminates on returning A_k as the query result (lines 24-25).

Algorithm 2 returns data objects $d \in H_i$ individually in descending order based on the component range scores $\gamma_i(d)$. Initially, H_i contains the root node of an R-tree R_i that stores $SKY(\overline{D} \otimes F_i)$. H_i stores entries *e*, each of which takes the form $e = \langle ptr, score \rangle$. Here, *ptr* indicates either a data object or an R-tree node, and *score* denotes the score of either the data object or the highest score of the R-tree node that the ptr points to. If the *ptr* indicates an R-tree node (i.e., $N_{nonleaf}$ or N_{leaf} in lines 3 and 7, respectively), then the score corresponds to the highest score of feature objects enclosed by the R-tree node. Every time the entry *e* at the top of the max heap H_i is popped. If entry *e* refers to an R-tree node, the qualifying entries in the R-tree node are added to H_i. More specifically, if *e* refers to a nonleaf node $N_{nonleaf}$, each entry $w \in N_{nonleaf}$ is examined to verify that $w.mindist \le r$. If so, an entry $\langle w.ptr,\ w.maxscore \rangle$ is added to H_i. If *e* indicates a leaf node N_{leaf}, this denotes that N_{leaf} includes multiple line segments that correspond to $dseg \otimes f$ pairs. Therefore, each data object $d \in dseg$ is examined to verify that $dist(d,f) \le r$. If so, an entry $\langle d, \sigma(f) \rangle$ is added to H_i (lines 10-11). Finally, when data object d_{top} is found at the top of H_i, d_{top} is added to D_i^{rptd} to avoid multiple reports on the same data object, and the top entry $\langle d_{top}, \gamma_i(d_{top}) \rangle$ is returned.

6. PERFORMANCE STUDY

6.1 Experimental settings

In the experiments, we use a real-life roadmap [20] (consisting of 175,813 nodes and 179,179 edges) for the main roads of North America corresponding to a data universe of 2.5×10^7 km^2. According to the American hotel and lodging association [21], there are more than 54,200 hotels in the United States. These hotels correspond to the data objects in this study. The experimental parameter settings are given in Table 4.

The positions of the data and feature objects follow either a uniform or a centroid distribution. The centroid dataset is generated so that it resembles

Algorithm 2 pop_data_object_with_highest_score $(H_i,\ r)$

Input: H_i: a max heap, *r*: range constraint
Output: data object *e* in H_i with the highest component score
1: $e \leftarrow H_i.pop()$ // $e = \langle ptr, score \rangle$ is an entry in H_i
2: **while** $e.ptr \notin D$ **or** $e.ptr \in D_i^{rptd}$ **do**
3: **if** *e* points to a nonleaf node $N_{nonleaf}$ of R_i **then**
4: **for** each entry $w \in N_{nonleaf}$ **do**
5: **if** $w.mindist \le r$ **then**
6: insert an entry $\langle w.ptr,\ w.maxscore \rangle$ to H_i
7: **else** // this means that *e* points to a leaf node N_{leaf} of R_i
8: **for** each entry $w \in N_{leaf}$ **do**
9: **for** each data object $d \in dseg$ **do**
10: **if** $dist(d,f) \le r$ **then**
11: insert an entry $\langle d, \sigma(f) \rangle$ to H_i
12: $e \leftarrow H_i.pop()$
13: $D_i^{rptd} \leftarrow D_i^{rptd} \cup \{d_{top}\}$
14: **return** *e* // note that $e = \langle d_{top},\ \gamma_i(d_{top}) \rangle$

Table 4. Experimental parameter settings

Parameter	Range
Ratio of directed edge sequences to total edge sequences (R_{dir})	**10%**
Number of data objects ($\|D\|$)	**50,000**
Number of feature objects in F_i ($\|F_i\|$)	**50,000**
Number of feature datasets (m)	1, 2, **3**, 4, 5
Distribution of data objects	(U)niform, **(C)entroid**
Distribution of feature objects	**(U)niform**, (C)entroid
Query range (r)	**1**, 2, 3, 4, 5 (km)
Number of data objects to be retrieved (k)	10, 20, **30**, 40, 50

the real world. First, five centroids are chosen where the first centroid is positioned in the middle of the space and the others are positioned randomly. The objects around each centroid follow a Gaussian distribution, in which the mean is set to the centroid and the standard deviation is set to 50 km, which corresponds to 1% of the side length of the data universe. In each experiment, we vary a single parameter within the range that is shown in Table 4 while keeping the other parameters at the bolded default values. Unless otherwise stated, the data objects follow a centroid distribution, whereas the feature objects follow a uniform distribution.

As a benchmark for ALPS$^+$, we use a baseline method that computes the score of every data object using the range network expansion (RNE) algorithm [22] to compute the range scores, respectively. Recall that the baseline method does not use any materialization scheme. We implement and evaluate two versions of ALPS$^+$, referred to as $ALPS^+_{seg}$ and $ALPS^+_{opt}$. $ALPS^+_{seg}$ groups data objects in a segment into a data segment and then generates and stores a skyline set for each data segment. Thus, $ALPS^+_{seg}$ can include redundant pairs of data and feature objects. $ALPS^+_{opt}$ generates and stores a skyline set for each data object and thus includes no redundant pairs of data and feature objects. $ALPS^+_{opt}$ is optimal in terms of query processing time because it does not include any redundant pairs of data and feature objects. We compare $ALPS^+_{seg}$, $ALPS^+_{opt}$, and the baseline method using two measures: query processing time and materialization cost. The three methods are implemented in C++ and run on a desktop PC with a 3.4 GHz processor and 16 GB memory. The datasets are indexed using R-trees with node sizes of 4 KB. The scores of feature objects are randomly generated by 10^{-3} units within the range $[0,1]$. That is, $\sigma(f)=\{10^{-3}\times i|0\leq i\leq 10^3\}$.

6.2 Experimental results

Fig. 8 shows the query processing times for $ALPS^+_{seg}$, $ALPS^+_{opt}$, and the baseline method for the range condition. In summary, $ALPS^+_{opt}$ shows the best performance, the baseline method shows the worst performance, and $ALPS^+_{seg}$ shows comparable performance to ALPS$^+$ opt. Fig. 8(a) shows the query processing time as a function of k, i.e., the number of requested data objects with the highest scores. As shown in this figure, $ALPS^+_{opt}$ outperforms $ALPS^+_{seg}$ slightly because unlike $ALPS^+_{opt}$, $ALPS^+_{seg}$ includes redundant pairs of data and feature objects, which lead to unnecessary search time and storage. Fig. 8(b) shows the query processing time as a function of query radius r between 1 km and 5 km. $ALPS^+_{seg}$ still shows comparable performance to $ALPS^+_{opt}$. Fig. 8(c) shows the query processing time as a function of the number m of feature datasets. The query processing times for all methods increase with the m value. $ALPS^+_{seg}$ shows a similar performance to $ALPS^+_{opt}$ because $ALPS^+_{seg}$ includes a small number of redundant

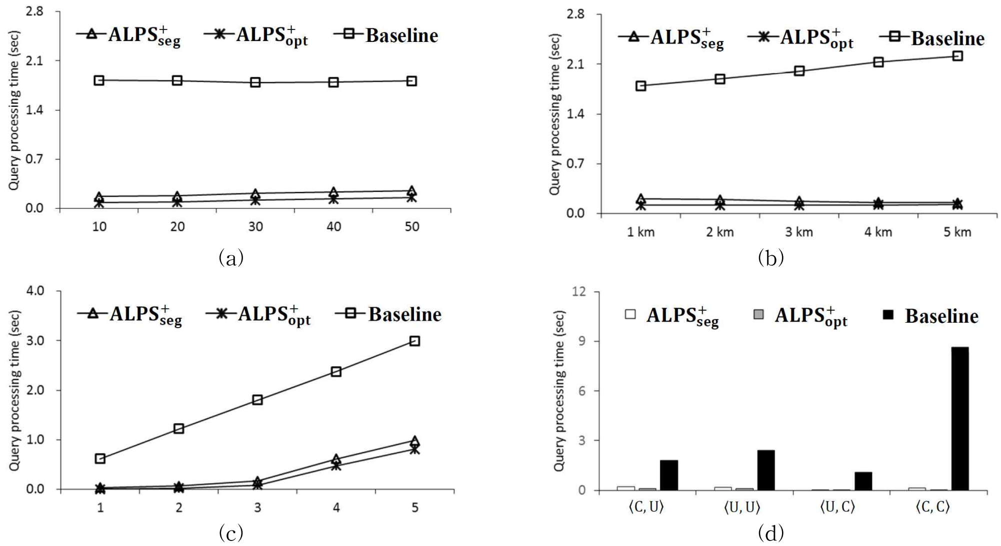


Fig. 8. Comparison of query processing time. (a) Effect of k, (b) Effect of r, (c) Effect of m, and (d) Effect of distributions of objects.

pairs, which are later presented in Fig. 9(a). Fig. 8(d) shows the query processing time for various distributions of data and feature objects. In this figure, each pair (i.e., $\langle C, U\rangle, \langle U, U\rangle, \langle U, C\rangle$, and $\langle C, C\rangle$) indicates a combination of the distributions of data and feature objects where the first and second attributes refer to the distributions of data and feature objects, respectively. $ALPS^+_{seg}$ and $ALPS^+_{opt}$ are not sensitive to the distributions of data and feature objects. However, the baseline method is very sensitive to the distributions. In particular, the query processing time of the baseline method is up to 70 times longer than that of $ALPS^+_{seg}$ for the case of $\langle C, C\rangle$.

Given that the baseline method does not use any materialization scheme, we investigate the materialization costs of $ALPS^+_{seg}$ and $ALPS^+_{opt}$. Fig. 9 shows the comparisons of index size and construction time for $ALPS^+_{seg}$ and $ALPS^+_{opt}$. As shown in Fig. 9(a), the index sizes of $ALPS^+_{opt}$ are smaller than those of $ALPS^+_{seg}$. This is expected because $ALPS^+_{opt}$ has no redundant pairs of data and feature objects, whereas $ALPS^+_{seg}$ has redundant pairs of data and feature objects. The index sizes of $ALPS^+_{seg}$ are sensitive to the distribution of feature objects. Specifically, the index sizes of $ALPS^+_{seg}$ are up to 14 times larger than those of $ALPS^+_{opt}$ when feature objects follow a centroid distribution. As shown in Fig. 9(b), the index construction times

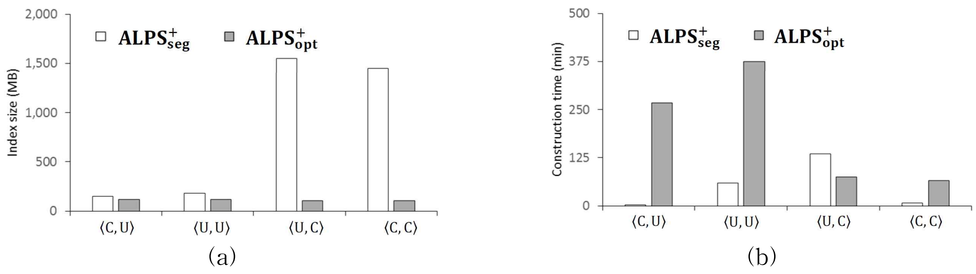


Fig. 9. Index size and construction time. (a) Index size and (b) Construction time.

of $ALPS^{+}_{opt}$ are typically longer than those of $ALPS^{+}_{seg}$ by up to 89 times except for the case $\langle U, C \rangle$. A trade-off exists between $ALPS^{+}_{seg}$ and $ALPS^{+}_{opt}$ with regard to index size and construction time. Typically, the index size of $ALPS^{+}_{opt}$ is smaller than that of $ALPS^{+}_{seg}$, whereas the index construction time of $ALPS^{+}_{opt}$ is longer than that of $ALPS^{+}_{seg}$.

7. CONCLUSIONS

In this study, we proposed a new method called ALPS^{+} for efficient processing of spatial preference queries in spatial network databases. In this method, data objects in a directed segment are grouped and then converted into a data segment. Pairs of data segments and feature objects are mapped to the distance-score space, and a skyline set is generated for each data segment. We implemented and evaluated the two versions of ALPS^{+}, which are referred to as $ALPS^{+}_{seg}$ and $ALPS^{+}_{opt}$, to confirm the superiority and effectiveness of ALPS^{+} in a wide range of problem settings. A trade-off exists between $ALPS^{+}_{seg}$ and $ALPS^{+}_{opt}$ in terms of query processing time and index construction time. $ALPS^{+}_{opt}$ outperforms $ALPS^{+}_{seg}$ in query processing time, whereas $ALPS^{+}_{seg}$ outperforms $ALPS^{+}_{opt}$ in index construction time.

REFERENCES

[1] S.Y. Kim and S.M. Cho, "A Haptic Navigation System for Visually Impaired Persons," *Journal of Korea Multimedia Society*, Vol. 14, No. 1, pp. 133-143, 2011.

[2] W. Park and T. Park, "An Efficient Channel Navigation Scheme Based on Patterns of Watching TV Programs," *Journal of Korea Multimedia Society*, Vol. 13, No. 9, pp. 1357-1364, 2010.

[3] D. Yung, M.L. Yiu, and E. Lo, "A Safe-exit Approach for Efficient Network-based Moving Range Queries," *Data and Knowledge Engineering*, Vol. 72, pp. 126-147, 2012.

[4] T. Abeywickrama, M.A. Cheema, and D. Taniar, "k-Nearest Neighbors on Road Networks: A Journey in Experimentation and In-Memory Implementation," *The Proceedings of the Very Large Data Bases Endowment*, Vol. 9, No. 6, pp. 492-503, 2016.

[5] A.M. Aly, W.G. Aref, and M. Ouzzani, "Spatial Queries with Two kNN Predicates," *The Proceedings of the Very Large Data Bases Endowment*, Vol. 5, No. 11, pp. 1100-1111, 2012.

[6] K. Mouratidis, M.L. Yiu, D. Papadias, and N. Mamoulis, "Continuous Nearest Neighbor Monitoring in Road Networks," *Proceedings of International Conference on Very Large Data Bases*, pp. 43-54, 2006.

[7] S. Yang, M.A. Cheema, X. Lin, and W. Wang, "Reverse k Nearest Neighbors Query Processing: Experiments and Analysis," *The Proceedings of the Very Lare Data Bases Endowment*, Vol. 8, No. 5, pp. 605-616, 2015.

[8] S. Peng and H. Samet, "Analytical Queries on Road Networks: An Experimental Evaluation of Two System Architectures," *Proceedings of International Conference on Advances in Geographic Information Systems*, pp. 1:1-1:10, 2015.

[9] S. Peng, J. Sankaranarayanan, and H. Samet, "SPDO: High-Throughput Road Distance Computations on Spark Using Distance Oracles," *Proceedings of International Conference on Data Engineering*, pp. 1239-1250, 2016.

[10] J.B. Rocha-Junior, A. Vlachou, C. Doulkeridis, and K. Nørvåg, "Efficient Processing of Top-k Spatial Preference Queries," *The Proceedings of the Very Large Data Bases Endowment*, Vol. 4, No. 2, pp. 93-104, 2010.

[11] M.L. Yiu, X. Dai, N. Mamoulis, and M. Vaitis,

"Top-k Spatial Preference Queries," *Proceeding of International Conference on Data Engineering*, pp. 1076-1085, 2007.

[12] M.L. Yiu, H. Lu, N. Mamoulis, and M. Vaitis, "Ranking Spatial Data by Quality Preferences," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 23, No. 3, pp. 433-446, 2011.

[13] H.J. Cho, S.J. Kwon, and T.S. Chung, "ALPS: an Efficient Algorithm for Top-k Spatial Preference Search in Road Networks," *Knowledge and Information Systems*, Vol. 42, No. 3, pp. 599-631, 2015.

[14] L. Chen, J. Xu, X. Lin, C.S. Jensen, and H. Hu, "Answering Why-not Spatial Keyword Top-k Queries via Keyword Adaption," *Proceedings of International Conference on Data Engineering*, pp. 697-708, 2016.

[15] L. Guo, J. Shao, H.H. Aung, and K.L. Tan, "Efficient Continuous Top-k Spatial Keyword Queries on Road Networks," *GeoInformatica*, Vol. 19, No. 1, pp. 29-60, 2015.

[16] J.B. Rocha-Junior and K. Nørvåg, "Top-k Spatial Keyword Queries on Road Networks," *Proceeding of International Conference on Extending Database Technology*, pp. 168-179, 2012.

[17] N. Beckmann, H.P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: an Efficient and Robust Access Method for Points and Rectangles," *Proceeding of International Conference on Management of Data*, pp. 322-331, 1990.

[18] A. Guttman, "R-trees: a Dynamic Index Structure for Spatial Searching," *Proceeding of International Conference on Management of Data*, pp. 47-57, 1984.

[19] R. Fagin, A. Lotem, and M. Naor, "Optimal Aggregation Algorithms for Middleware," *Proceeding of Symposium on Principles of Database Systems*, pp. 102-113, 2001.

[20] Real Datasets for Spatial Databases, https://www.cs.utah.edu/~lifeifei/SpatialDataset.htm (accessed Dec., 20, 2018).

[21] American Hotel and Lodging Association, http://www.ahla.com/ (accessed Dec., 20, 2018).

[22] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query Processing in Spatial Network Databases," *Proceeding of International Conference on Very Large Data Bases*, pp. 802-813, 2003.

Hyung-Ju Cho

is an assistant professor at the department of software, Kyungpook National University, South Korea. His current research interests include moving object databases and query processing in mobile peer-to-peer networks.

Muhammad Attique

is an assistant professor at the department of software, Sejong University, South Korea. His research interests include spatial computing, location-based services, big spatial data and spatial query processing in mobile networks.