World Wide Web
https://doi.org/10.1007/s11280-018-0538-5

Discovery of accessible locations using region-based geo-social data

Yan Wang[1] · Jianmin Li[1] · Ying Zhong[1] · Shunzhi Zhu[1] · Danhuai Guo[2,3] · Shuo Shang[4]

Received: 28 October 2017 / Revised: 5 February 2018 / Accepted: 12 February 2018

Abstract Geo-social data plays a significant role in location discovery and recommendation. In this light, we propose and study a novel problem of discovering accessible locations in spatial networks using region-based geo-social data. Given a set Q of query regions, the top-k accessible location discovery query (kALDQ) finds k locations that have the highest spatial-density correlations to Q. Both the spatial distances between locations and regions and the POI (point of interest) density within the regions are taken into account. We believe

This article belongs to the Topical Collection: *Special Issue on Geo-Social Computing*
Guest Editors: Guandong Xu, Wen-Chih Peng, Hongzhi Yin, Zi (Helen) Huang

✉ Shunzhi Zhu
szzhu@xmut.edu.cn

✉ Shuo Shang
jedi.shang@gmail.com

Yan Wang
wangyan@xmut.edu.cn

Jianmin Li
lijm@xmut.edu.cn

Ying Zhong
yzhong@xmut.edu.cn

Danhuai Guo
guodanhuai@cnic.cn

[1] Xiamen University of Technology, Xiamen, China

[2] Guangdong Province Key Laboratory of Popular High Performance Computers of Shenzhen University, Shenzhen, China

[3] Guangdong Provincial Big Data Collaborative Innovation Center, Shenzhen University CNIC, Chinese Academy of Sciences, Beijing, China

[4] King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

that this type of kALDQ query can bring significant benefit to many applications such as travel planning, facility allocation, and urban planning. Three challenges exist in kALDQ: (1) how to model the spatial-density correlation practically, (2) how to prune the search space effectively, and (3) how to schedule the searches from multiple query regions. To tackle the challenges and process kALDQ effectively and efficiently, we first define a series of spatial and density metrics to model the spatial-density correlation. Then we propose a novel three-phase solution with a pair of upper and lower bounds of the spatial-density correlation and a heuristic scheduling strategy to schedule multiple query regions. Finally, we conduct extensive experiments on real and synthetic spatial data to demonstrate the performance of the developed solutions.

Keywords Location discovery · Region · Recommendation · Geo-social data

1 Introduction

With the rapid development of GPS-equipped smart phones and online-map services (e.g., Google-maps,[1] Baidu-maps,[2] and MapQuest[3]), people can easily get their current geographic locations and post geo-tagged tweets, geo-tagged photos, and geo-tagged checkins to specialized sites (e.g., twitter,[4] flickr,[5] and foursquare[6]). Such massive geo-social data enable many novel application. An emerging one is region-based accessible location discovery. Given a set Q of query regions, the top-k accessible location discovery query (kALDQ) finds k locations that have the highest spatial-density correlations to Q. We believe that this type of kALDQ query can bring significant benefit to many applications such as travel planning, facility allocation, and urban planning.

In most of existing studies [21, 22, 25, 27, 36], location discovery and recommendation is based on point-to-point matching [21, 22, 25] or trajectory-to-point matching [27, 36]. They only take the spatial distances between matched points or matched trajectories and points into account, but no existing study considers the regions of interest and the density of spatial objects (e.g., geo-tagged tweets, geo-tagged photos, and POIs) in the corresponding regions. For example, there exist several regions of interest (e.g., residential area, commercial area, or scenic area), and we want to find an aggregate location for travelers, or we want to find an accessible location to set a new facility (e.g., shopping mall, bank, or petrol station) to maximize its commercial value. The accessible locations are expected spatially close to the regions of interest. In particular, they are expected spatially close to the regions with high spatial-object density.

An example is shown in Figure 1, where R_1, R_2, and R_3 are query regions, p_1 and p_2 are location candidates, and the red points are spatial objects. If we only consider the spatial

[1] http://maps.google.com/

[2] http://map.baidu.com/

[3] http://www.mapquest.com

[4] https://twitter.com/

[5] https://www.flickr.com/

[6] https://foursquare.com/

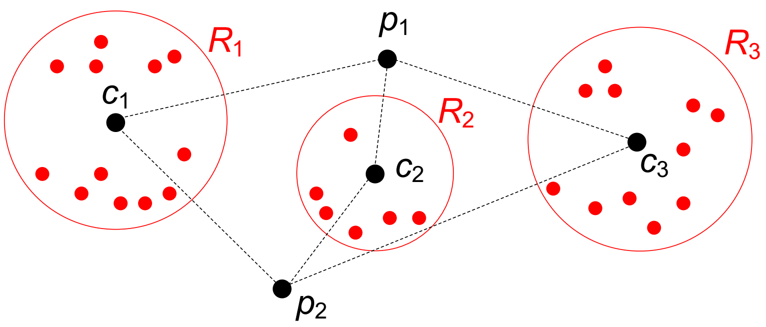


Figure 1 An example of kALDQ

distances between location and query regions (i.e., $d(c_1, p_1) + d(c_2, p_1) + d(c_3, p_1)$, where c_1, c_2, and c_3 are center points of R_1, R_2, and R_3), location p_1 is returned because it is spatially close to R_1, R_2, and R_3. However, if we take both spatial distance and the density of spatial objects in the query regions into account, location p_2 will be returned. Although p_2 is not as good as p_1 in the spatial domain, p_2 is spatially close to the regions with high spatial-object density, so we still consider p_2 as the optimal choice. To the best of our knowledge, this is the first study of location discovery and recommendation that take both spatial distance and spatial-object density into account.

kALDQ is conducted in spatial networks, because in most practical scenarios, objects move in spatial networks (e.g., road networks) rather than in Euclidean space. The existing techniques of location discovery and recommendation cannot compute kALDQ due to three reasons. First, kALDQ considers the matching between regions and locations, rather than the point-to-point matching or trajectory-to-point matching, thus it needs new distance metrics. Second, kALDQ takes the density of spatial objects in query regions into account, so it also needs specific metrics to reflect the density domain. Third, most existing studies of aggregate location query [21, 22] are conducted in Euclidean space, and their optimization techniques cannot be used in spatial networks directly, because most spatial indexes (e.g., R-tree) use Euclidean space as lower bound to prune the search space and such bound is ineffective in spatial networks.

kALDQ faces three challenges. First, it needs a metric to describe spatial-density correlation. Second, it needs a pair of comparable tight upper and lower bounds of the spatial-density correlation to prune the search space. Third, it needs a heuristic scheduling methods to schedule multiple query regions to further enhance the query performance. To overcome the challenges and to process kALDQ efficiently, we propose a novel spatial-density correlation measure. Then we develop a novel three phase algorithm. In the location-search phase, we expand network expansions according to Dijkstra's algorithm [7] to explore the spatial network and to find location candidates. A heuristic scheduling method is defined to schedule the search processes of different query regions. In the filter phase, we define a pair of upper and lower bounds of the spatial-density correlation to prune the search space. Once the global lower bound exceeds the global upper bound, the search terminates and all partly and unscanned locations are pruned. In the refine phase, we compute the exact spatial-density correlations for all full scanned locations and return top-k results.

To sum up, we make the following contributions in this paper.

- We propose a novel type of top-k accessible location discovery query (kALDQ) to find top-k accessible locations in a spatial network. We believe that this type of query can bring significant benefit in many applications such as travel planning, facility allocation, and urban planning.
- We define a series of new metrics to evaluate the spatial and density correlations between regions and locations (Section 2).
- We develop a range-query based baseline algorithm to process kALDQ (Section 3.1).
- We develop a novel three phase algorithm to process kALDQ efficiently with the support of a pair of upper and lower bounds and a heuristic scheduling method (Section 3.2).
- We conduct extensive experiments on real and synthetic spatial data to demonstrate the performance of the developed algorithms (Section 4).

The rest of the paper is organized as follows. The spatial networks and the metrics of the spatial-density correlation as well as problem definitions are introduced in Section 2. The three-phase algorithm is introduced in Section 3, which is followed by extensive experimental results in Section 4. Related work is covered in Section 5 and conclusions and future directions are given in Section 6.

2 Preliminaries

2.1 Spatial networks, locations, spatial objects, and regions

A spatial network $G(V, E)$ is modeled by a connected and undirected graph, where V is a vertex set and $E \subseteq V \times V$ is an edge set. The weight $w(e)$ of an edge $e \in E$ represents its weight (e.g., distance or travel time). We use adjacent list to store spatial networks.

Generally, locations and spatial objects are of the form of $(longitude, latitude)$. We map these locations and spatial objects to the corresponding spatial network by using some existing map-matching method [1]. We assume that all locations and spatial objects are on vertices. It is straightforward to support the scenarios that locations and spatial objects are on edges. Assuming that a location p is on an edge (a, b) and the network distance from p to a and b are known. We can create a new vertex p and split edge (a, b) into (a, p) and (p, b) [26].

We align the definitions of spatial networks, locations, and spatial objects with existing studies [24–26, 29, 37].

Given two vertices a and b in a spatial network, the network shortest path distance between a and b is denoted by $sd(a, b)$. A region R in spatial networks is a circular area (c, r), where c is a center point of R and r is the radius. For all spatial objects whose network distances to c do not exceed r, we say that they are contained by region R.

2.2 Metrics

Given a location p and region R, the spatial-density correlation between p and R is defined as follows.

$$C_{SD}(p, R) = \sum_{o \in R} e^{-sd(p,o)} \tag{1}$$

Here, o is the spatial object in region R, and we consider the aggregate distance between p and all objects in R. Given a set of query regions $Q = \{R_1, R_2, ..., R_n\}$, and a location p, the spatial-density correlation between p and Q is defined as follows.

$$C_{SD}(p, Q) = \sum_{R_i \in Q} \left(\frac{2}{1 + e^{-C_{SD}(p, R_i)}} - 1 \right) \tag{2}$$

Here, we use the Sigmoid function [26, 28] to normalize $C_{SD}(p, R_i)$ to a range [0,1], and each query region plays an equal role in query processing.

2.3 Problem definition

Given a set of query regions Q, and a set of location candidates P, the top-k accessible location discovery query (kALDQ) finds a k-item set A ($|A| = k$) that contains k locations with the highest spatial-density correlations to Q, such that $\forall p' \in P \setminus A(p \in A(C_{SD}(p', Q) \leq C_{SD}(p, Q)))$.

3 Query processing

3.1 Baseline method

Range query is a straightforward baseline method to kALDQ. From each query region R_i, we expand the search from its center c_i at the same speed by increasing rs_i as $rs_i = rs_i + x$, where rs_i is the search radius of query region R_i and x is a user given parameter. An example is shown in Figure 2, where R_1 and R_2 are two query regions, c_1 and c_2 are the centers, r_1 and r_2 are the radiuses of query regions, and rs_1 and rs_2 are the search radiuses.

For a location candidate p, if it is scanned by the search from query region R (e.g., p_1 and R_2), we estimate its spatial-density correlation $C_{SD}(p, R)$ as follows. According to the triangular inequality, for each spatial object $o \in R$, we have that

$$sd(o, p) < sd(o, c) + sd(c, p) \tag{3}$$

$$sd(o, p) > sd(c, p) - sd(o, c) \tag{4}$$

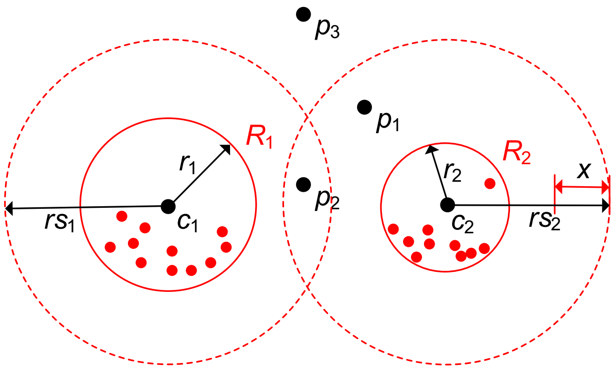


Figure 2 An example of Baseline Algorithm

By substituting (3) into (1), we have the lower bound $C_{SD}(p, R).lb$ of spatial-density correlation as follows.

$$sd(o, p) < sd(o, c) + sd(c, p) \Rightarrow e^{-sd(o,p)} > e^{-(sd(o,c)+sd(c,p))}$$

$$C_{SD}(p, R) = \sum_{o \in R} e^{-sd(p,o)} > \sum_{o \in R} e^{-(sd(c,p)+sd(o,c))} = C_{SD}(p, R).lb \tag{5}$$

If a location candidate is fully scanned (i.e., scanned by expansions from all query regions, e.g., p_2 in Figure 2), we estimate the lower bound of $C_{SD}(p, R)$ by substituting (5) into (2).

$$C_{SD}(p, Q).lb = \sum_{R_i \in Q} \left(\frac{2}{1 + e^{-C_{SD}(p,R_i).lb}} - 1 \right) \tag{6}$$

For unscanned location candidates (e.g., p_3 in Figure 2), we estimate the upper bound of their spatial-density correlations as follows. For an unscanned location candidate p, we have that $sd(c, p) > r$. By substituting it into (4) and (1), we have that

$$sd(o, p) > sd(c, p) - sd(o, c) \Rightarrow sd(o, p) > r - sd(o, c).$$

$$C_{SD}(p, R) = \sum_{o \in R} e^{-sd(p,o)} < \sum_{o \in R} e^{-(r-sd(o,c))} = C_{SD}(p, R).ub \tag{7}$$

By substituting (7) into (2), the upper bound of $C_{SD}(p, Q)$ is defined as follows.

$$C_{SD}(p, Q).ub = \sum_{R_i \in Q} \left(\frac{2}{1 + e^{-C_{SD}(p,R_i).ub}} - 1 \right) \tag{8}$$

The value of $C_{SD}(p, Q).ub$ is the global upper bound of all unscanned location candidates (e.g., p_3 in Figure 2). Among all fully scanned location candidates, we define a global lower bound *LB* as follows.

$$LB = \min_{p \in P_k} \{C_{SD}(p, Q).lb\} \tag{9}$$

Here, $P_k \subseteq P_f \wedge |P_k| = k$, and P_f is a set of all fully scanned location candidates and P_k is a subset of P_f. Set P_k contains k items with the largest values of $C_{SD}(p, Q).lb$, such that $\forall p \in P_k(\forall p' \in P_f \setminus P_k(C_{SD}(p', Q).lb < C_{SD}(p, Q).lb))$. The value of *LB* changes continually during query processing.

Once the value of $C_{SD}(p, Q).ub$ is less than that of *LB*, the searches from all query regions terminate, and all unscanned location candidates are pruned. Then, we continuously refine all fully scanned (e.g., p_2 in Figure 2) and partly scanned location candidates (e.g., p_1 in Figure 2). Among all refined location candidates, we select k location candidates with the largest spatial-density correlation and return them.

The baseline algorithm is detailed in Algorithm 1. The query arguments include a set P of location candidates and a set Q of query regions. The query output is a set A ($|A| = k \wedge A \subseteq Q$) of location candidates with largest spatial-density correlations. Initially, the search radiuses of each query region are set to 0 (line 1). We make the range query from each query region and expand the search as $rs_i = rs_i + x$ (line 2–5). For each fully scanned location candidate p, we compute its lower bound $C_{SD}(p, Q).lb$ of spatial-density correlation (6). If the size of fully scanned set exceeds k, we compute the global lower bound *LB* of fully scanned location candidates (9) and the upper bound $C_{SD}(p, Q).ub$ of unscanned location candidates (8) (lines 8–9). If the value of *LB* exceeds that of $C_{SD}(p, Q).ub$, the searches from all query regions terminate, and all unscanned location candidates are pruned (lines 11–12). Then, we refine all fully scanned and partly scanned location candidates, and find top-k locations in set A with the largest spatial-density correlations (lines 13–14).

Algorithm 1 Baseline Algorithm

```
Data: $P$, $Q$
Result: $A$
$\forall R_i \in Q(rs_i \leftarrow 0)$;
while true do
    for each query location $R_i$ do
        $expand(c_i)$;
        $rs_i \leftarrow rs_i + x$;
    for each fully scanned location candidate $p$ do
        compute $C_{SD}(p, Q).lb$;
    if $|P_f| \geq k$ then
        compute $LB$ and $C_{SD}(p, Q).ub$;
        if $LB > C_{SD}(p, Q).ub$ then
            prune all unscanned location candidates;
            refine fully and partly scanned location candidates;
            return $A$;
```

3.2 Three-phase algorithm

The baseline algorithm includes several weaknesses. First, its search is based on range query, so it is not continuous. It is difficult to set the expansion parameter x. Second, it simply prunes the unscanned location candidates, and it has to refine all fully scanned and partly scanned location candidates. Its pruning power is comparable weak. Third, the expansion speed from each query region is the same, and it lacks an effective scheduling method to schedule the searches from multiple query regions.

To overcome these weaknesses, we propose a novel three phase algorithm to process the kALDQ query efficiently. In the location search phase, we adopt network expansion method [7] to explore the spatial network and to identify locations spatially close to the query regions. A pair of upper and lower bounds of the spatial-density correlation are defined to prune the search space. All partly scanned and unscanned location candidates are pruned. In the scheduling phase, we propose a heuristic scheduling method to schedule multiple query regions to further enhance the query efficiency. In the refinement phase, we refine all fully scanned location candidates and find the query result.

3.2.1 Expansion and bounds

We conduct network expansion from each query region according to Dijkstra's algorithm [7] to explore the spatial network and find location candidates close to the query regions. An example is shown in Figure 3, where R_1 and R_2 are two query regions, c_1 and c_2 are centers, and r_1 and r_2 are radiuses of regions R_1 and R_2. The explored regions are circular areas, and rs_1 and rs_2 are the corresponding radiuses (i.e., the network distance from center to the expansion boundary).

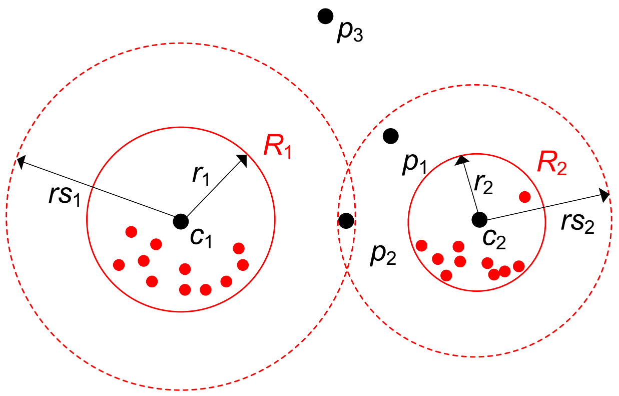


Figure 3 An example of Three-Phase Algorithm

Similar to the baseline algorithm, the location candidates are classified into three categories, fully scanned (e.g., p_2 in Figure 3), partly scanned (e.g., p_1), and unscanned (e.g., p_3). If a location candidate p is fully scanned, we estimate its lower bound of the spatial-density correlation according to (6). Among all fully scanned location candidates, we define a global lower bound according to (9).

For a partly scanned location candidate p, we estimate its upper bound of the spatial-density correlation as follows. If p is scanned by the expansion from R, we have that

$$sd(o,p) > sd(c,p) - sd(o,c)$$
$$\Rightarrow C_{SD}(p,R) = \sum_{o\in R} e^{-sd(p,o)} < \sum_{o\in R} e^{-(sd(c,p)-sd(o,c))} = C_{SD}(p,R).ub \tag{10}$$

If p is unscanned by the expansion from R, we estimate $C_{SD}(p,R).ub$ according to (7).

$$C_{SD}(p,R) = \sum_{o\in R} e^{-sd(p,o)} < \sum_{o\in R} e^{-(r-sd(o,c))} = C_{SD}(p,R).ub$$

Therefore, we have that

$$C_{SD}(p,R).ub = \begin{cases} \sum_{o\in R} e^{-(sd(c,p)-sd(o,c))} & \text{if Case 1} \\ \sum_{o\in R} e^{-(r-sd(o,c))} & \text{if Case 2} \end{cases} \tag{11}$$

Case1: p is scanned by the expansion from R.
Case2: p is unscanned by the expansion from R.
By substituting (11) into (1), we define the upper bound of the spatial-density correlation of a partly scanned location candidate p as follows.

$$C_{SD}(p,Q).ub = \sum_{R_i\in Q}\left(\frac{2}{1+e^{-C_{SD}(p,R_i).ub}} - 1\right) \tag{12}$$

Among all partly scanned location candidates, we define a global upper bound *UB* of spatial-density correlation as follows. Similar to *LB*, the value of *UB* changes dynamically during query processing.

$$UB = \max_{p\in P_p} C_{SD}(p,Q).ub \tag{13}$$

Here, P_p is a set of partly scanned location candidates.

Once the value of *UB* exceeds that of *LB*, the expansion search terminates, and all partly scanned and unscanned location candidates are pruned. We refine the fully scanned location candidates, and find the query results.

Remark There is no need to compute and to maintain the upper bound of unscanned location candidates (different to the baseline algorithm). By comparing (7) and (11), for a partly scanned location candidate p and an unscanned location candidate p', we have that

$$C_{SD}(p, R).ub \geq C_{SD}(p', R).ub \Rightarrow C_{SD}(p, Q).ub > C_{SD}(p', Q).ub.$$

So if $C_{SD}(p, Q).ub < LB$, we have that $C_{SD}(p', Q).ub < LB$. All unscanned location candidates can be pruned safely.

3.2.2 Scheduling

To further enhance the query efficiency, we propose a novel heuristic scheduling strategy to schedule the searches from multiple query regions. Each query region is given a label to describe its priority. At each time, we only search the top-ranked query region, until a new top-ranked query region appears. The priority label is defined as follows.

$$R.L = \sum_{P_p \setminus P_R} \{C_{SD}(p, Q).ub\} \tag{14}$$

Here, P_p is the set of partly scanned location candidates, and P_R is a set of location candidates scanned by the search from R. Our target is to make partly scanned location candidates to fully scanned as soon as possible. We use $P_p \setminus P_R$ to describe the gap of region R. Moreover, we also consider the upper bound of the spatial-density correlation (see (12)) of a location candidate. Obviously, a location candidate with a higher upper bound may have a higher possibility to be the query result. We share the similar idea of existing studies [24, 26, 28, 29].

3.2.3 Algorithm

The detailed procedure of the three-phase algorithm is introduced in Algorithm 2. The query arguments include a set P of location candidates and a set Q of query regions. The query result is a set A ($|A| = k \wedge A \subseteq Q$) of location candidates with largest spatial-density correlations. Initially, the priority labels of all query regions are set to 0, and the values of global upper and lower bounds are set to 0 (lines 1–2). We expand the network expansion according to Dijkstra's algorithm from each query region, and n is a newly scanned vertex (lines 3–5). If n is a location candidate and n is fully scanned, we compute $C_{SD}(n, Q).lb$ (6). Otherwise, we compute $C_{SD}(n, Q).ub$ (12) (lines 6–10). If the size of fully scanned set exceeds k, we compute the global lower bound *LB* (9) and global upper bound *UB* (13) of fully scanned location candidates (lines 11–12). If the value of *LB* exceeds that of *UB*, the searches from all query regions terminate, and all partly scanned and unscanned location candidates are pruned. Then, we refine all fully scanned location candidates, and find top-k locations in set A with the largest spatial-density correlations (lines 13–16).

3.3 Time complexity analysis

Both Algorithms 1 and 2 are based on network expansion (i.e., Dijkstra's algorithm [7]), and their time complexities are $O(|Q|(|V|lg(|V|)+|E|))$, where $|Q|$ is the number of query regions, $|V|$ is the number of vertices and $|E|$ is the number of edges in the corresponding spatial network, and $O(|V|lg(|V|) + |E|)$ is the time complexity of Dijkstra's algorithm. Although Algorithms 1 and 2 have the same time complexity in the worst case, Algorithm 2 has more powerful pruning techniques and it can achieve a higher performance (See the experimental results in Section 4).

Algorithm 2 Three-Phase Algorithm

```
Data: $P$, $Q$
Result: $A$
$\forall R \in Q(R.L \leftarrow 0)$;
$LB \leftarrow 0$; $UB \leftarrow 0$;
while true do
    select $R$ with the highest priority $R.L$;
    $n \leftarrow expand(c)$;
    if n is a location candidate then
        if n is fully scanned then
            compute $C_{SD}(n, Q).lb$;
        else
            compute $C_{SD}(n, Q).ub$;
        if $|P_f| \geq k$ then
            update $LB$ and $UB$;
            if $LB > UB$ then
                prune all partly scanned and unscanned location candidates;
                refine fully scanned location candidates;
                return $A$;
```

4 Experiments

4.1 Settings

We conduct extensive experiments on real and synthetic spatial data to study the performance of the developed algorithms. We use two spatial networks, called the Beijing Road Network (BRN) and the North America Road Network (NRN),[7] which contain 28,342 vertices and 27,690 edges, and 17,813 vertices and 179,179 edges, respectively. For BRN, we use 10,000 real POIs as location candidates and use 30,000 real POIs as spatial objects. For NRN, we generate 100,000 location candidates and 400,000 spatial objects. In the following figures, the baseline algorithm (Algorithm 1) is denoted by "baseline", the three-phase algorithm is denoted by "three-phase", and the three-phase algorithm without heuristic scheduling strategy is denoted by "three-phase w-o-h". The algorithms were implemented in Java and performed on a Windows 8 platform (Intel Core 2.90GHz Processor and 8GB memory). The main performance metrics were runtime.

For the parameter setting, the number of location candidates $|P|$ varies from 7,000 to 10,000 in BRN (default 10,000), and varies from 70,000 to 100,000 in NRN (default

[7] http://www.cs.utah.edu/~lifeifei/SpatialDataset.htm

Table 1 Parameter settings

	NRN	BRN
Number of Location Candidate $\|P\|$	70,000–100,000 (default 100,000)	7,000–10,000 (default 10,000)
Number of Query Regions $\|Q\|$	20–50 (default 20)	10–25 (default 10)
average radius of regions r	20–80 km (default 20 km)	1–4 km (default 1 km)
k	5–20 (default 5)	5–20 (default 5)

100,000). The number of query regions $|Q|$ varies from 10 to 25 in BRN (default 10), and varies from 20 to 50 in NRN (default 20). The average radius of regions varies from 1 km to 4 km in BRN (default 1 km), and varies from 20 km to 80 km in NRN (default 20 km). The value of k varies from 5 to 20 for both BRN and NRN (default 5). The parameter settings are detailed in Table 1.

4.2 Effect of number of location candidates $|P|$

First of all, we study the effect of number of location candidates $|P|$ on the kALDQ query efficiency. Intuitively, a larger number of location candidates $|P|$ may lead to more locations to be processed. Thus, the runtime of all three algorithms are expected to be higher when the number of location candidates $|P|$ increases. From Figure 4, we find that the runtime of the baseline algorithm (Algorithm 1) is about 2–3 times of that of the three-phase algorithm (Algorithm 2). It is clear that the three-phase algorithm has a stronger pruning power so it can achieve a higher query efficiency (the three-phase algorithm prunes partly scanned and unscanned location candidates, while the baseline algorithm only prunes unscanned location candidates). Moreover, we notice that with the help of heuristic scheduling strategy (14), the efficiency of the three-phase algorithm is improved by a factor of 20%–30% in term of runtime, which demonstrates the significant of the scheduling for the scenario of multiple query sources.

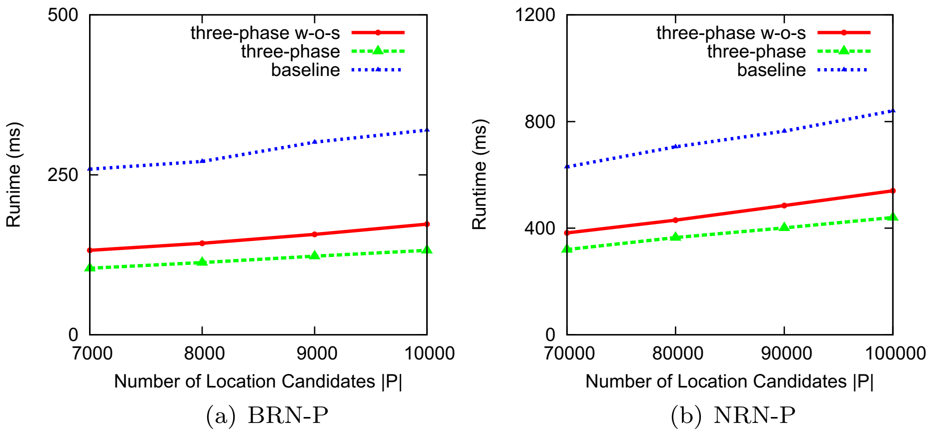


Figure 4 Effect of number of location candidates $|P|$

4.3 Effect of number of query regions $|Q|$

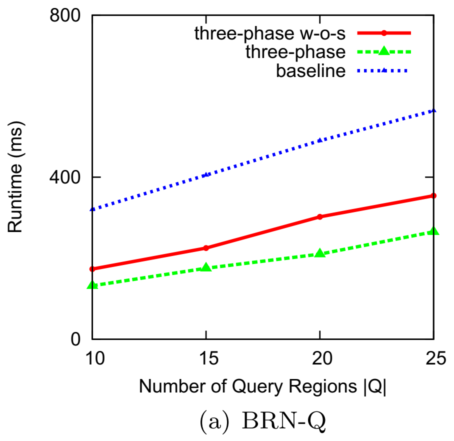
Next, we study the effect of the number of query locations $|Q|$ on the kALDQ query processing (Algorithms 1 and 2) in BRN and NRN. The kALDQ query finds the location with the minimum global travel cost to the query regions. Obviously, a larger number of $|Q|$ means more query regions to be refined. For all three algorithms, the larger the number of query regions is, the more computation effort they may need. Thus, the runtime is expected to be higher when the number of query locations $|Q|$ increases. From Figure 5, it is clear that the three-phase algorithm outperforms the baseline algorithm by a factor of 2–3 times in term of runtime, and outperforms the three-phase algorithm without a heuristic strategy by a factor of 20%–40% in term of runtime.

4.4 Effect of average radius of regions r

Then, we vary the value of the average radius r of query regions. Assuming spatial objects are uniformly distributed. So a larger value of radius means a larger area of a region, and more spatial objects in the region. According to Equations 1 and 2, we see that more spatial objects may need more computation effort. Thus, the runtime of all three algorithm are expected to increase when r increases. Compared to the baseline algorithm, the three-phase algorithm has clear advantage. In Figure 6, we see that when $r = 4$ in BRN, the runtime of baseline is almost three times of that of three-phase algorithm. The three-phase algorithm is able to process query regions with radius of 80 km in NRN within 1 second.

4.5 Effect of k

Finally, we study the effect of k. We see that a larger value of k may lead to a larger search space and more computation efforts are needed. When the value of k increases, the rumtime of all three algorithm will increase. From Figure 7, the three-phase algorithm is able to compute the query within 200 seconds in BRN, and within 500 seconds in NRN. With the help of heuristic scheduling method, the query efficiency is improved notably.



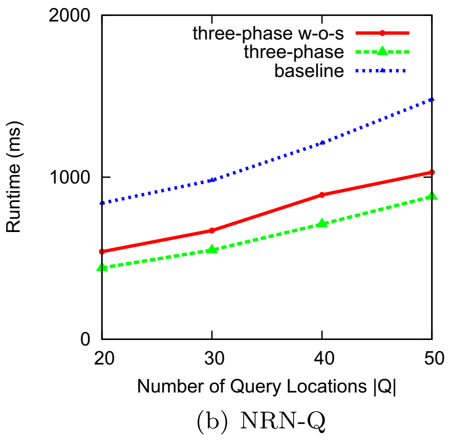


Figure 5 Effect of number of query regions $|Q|$

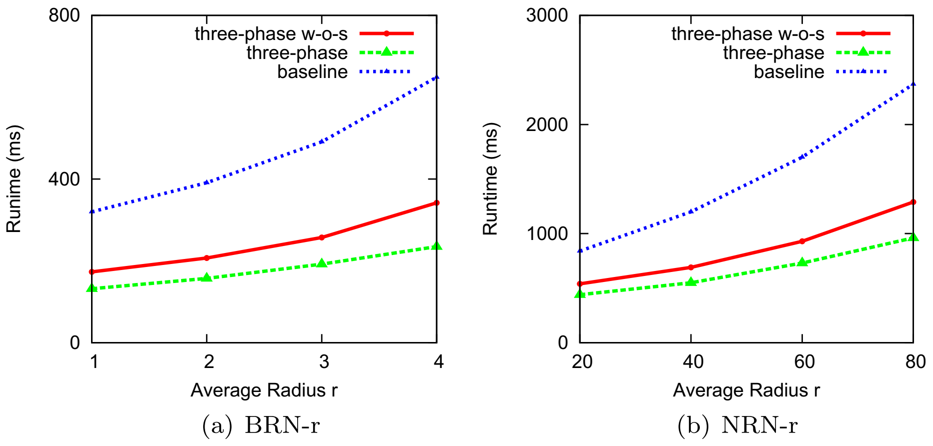


Figure 6 Effect of average radius of regions *r*

5 Related Work

Location search and recommendation are well studied in recent two decades [6, 8–11, 19, 30, 31, 43–48]. Generally, such type of query can be classified into three categories: point-to-point, trajectory-to-point, and region-to-point. In the first point-to-point category, the nearest neighbor (NN) query is a basic spatial query. Given a query location and a set of location candidates, NN finds the location candidate with the minimum travel cost to the query location. Aggregate nearest neighbor query (ANN) [21, 22] and aggregate location recommendation query (ALRQ) [15] further extend NN query. Given a set of travelers' current locations and a set of potential meeting points, the queries find the optimal meeting point with the minimum global travel cost. These queries are useful in many applications such as travel planning. ANN is based on Euclidean space. In contrast to ANN, ALRQ is conducted in dynamic spatial networks, and the instant traffic conditions are taken into account. Collective travel planning (CTP) [25] query further extends the model of ANN,

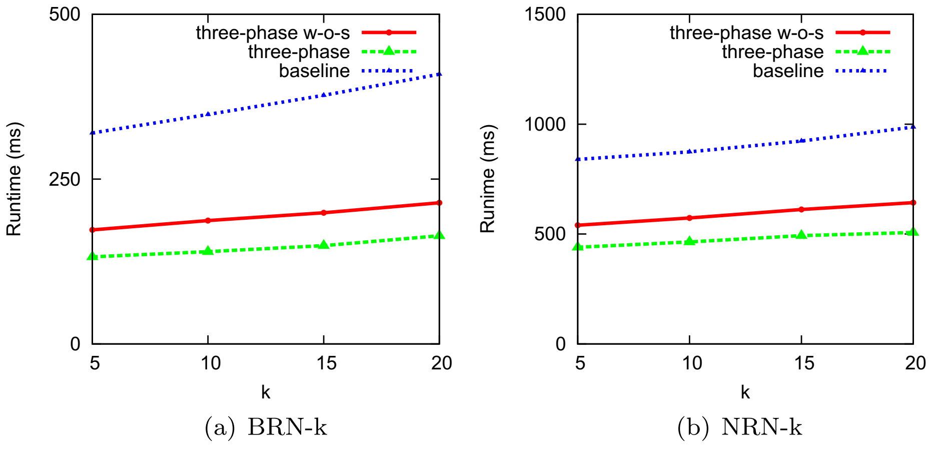


Figure 7 Effect of *k*

which finds the travel route with the minimum travel cost from multiple query source to a destination, via k meeting points.

In the trajectory-to-location category, the query find the optimal location based on one or multiple trajectories. In the path nearest neighbor (PNN) query [5, 27], the query finds the spatial point with the minimum travel cost to the path. Then, Shang et. al extend the PNN query on compressed trajectories [35], and the uncertainty of trajectory data is taken into account. The reverse path nearest neighbor query [36] uses trajectory data to rank locations, which is useful in finding most accessible location in spatial networks. For example, if a location is the path nearest neighbor of k trajectories, we define its influence factor as k.

To the best of our knowledgeless, the top-k accessible location discovery query (kALDQ) is the first study in the region-to-location category. No existing study considers the regions of interest and the density of spatial objects (e.g., geo-tagged tweets, geo-tagged photos, and POIs) in the corresponding regions. The existing techniques of location discovery and recommendation cannot compute kALDQ due to three reasons. First, kALDQ considers the matching between regions and locations, rather than the point-to-point matching or trajectory-to-point matching, thus it needs new distance metrics. Second, kALDQ takes the density of spatial objects in query regions into account, so it also needs specific metrics to reflect the density domain. Third, most existing studies of aggregate location query [21, 22] are conducted in Euclidean space, and their optimization techniques cannot be used in spatial networks directly, because most spatial indexes (e.g., R-tree) use Euclidean space as lower bound to prune the search space and such bound is ineffective in spatial networks.

How to use social media data to achieve a better recommendation is an interesting direction. The following related papers [2–4, 12–14, 16, 17, 23, 40–42] provide some inspirations. In the future, it is also of interest to study routing problem with geo-social data [18, 20, 32–34, 38, 39, 49].

6 Conclusion and future studies

In this paper, we proposed and studied a novel top-k accessible location discovery query (kALDQ) in spatial networks. This type of query is useful in many mobile applications, such as travel planning, facility allocation, and urban planning. We develop a range-query based algorithm, and a novel three-phase algorithm with upper and lower bounds and a heuristic scheduling strategy to process the kALDQ query efficiently. The performance of kALDQ were verified by extensive experiments on real and synthetic spatial data.

There exist two interesting future directions. First, it is of interest to take into account the significance of query regions. Users can specify significant query regions when conducting the query. So the upper and lower bounds should be reworked. Second, it is of interest to take into account the temporal information (e.g., departure time from a query region) and to make the query a spatiotemporal query.

Acknowledgments This paper is partly supported by Natural Science Foundation of P.R.China (No. 61373147, and No. 61672442), Fujian Province Science and Technology Plan Project (No. 2016Y0079, No. 2017J01783), the Education and Scientific Research Project for Youth and Middle-aged Teachers in Fujian (No. JA15365), the Open Research Fund Program of Guangdong Province Key Laboratory of Popular High Performance Computers of Shenzhen University, and the Open Research Fund Program of Guangdong Provincial Big Data Collaborative Innovation Center, Shenzhen University.

References

1. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On map-matching vehicle tracking data. In: VLDB, pp. 853–864 (2005)
2. Chen, Z., Cafarella, M., Chen, J., Prevo, D., Zhuang, J.: Senbazuru: A prototype spreadsheet database management system. PVLDB **6**(12), 1202–1205 (2013)
3. Chen, Z., Cafarella, M.J.: Integrating spreadsheet data via accurate and low-effort extraction. In: SIGKDD, pp. 1126–1135 (2014)
4. Chen, Z., Cafarella, M.J., Jagadish, H.V.: Long-tail vocabulary dictionary extraction from the Web. In: WSDM, pp. 625–634 (2016)
5. Chen, Z., Shen, H., Zhou, X., Zheng, Y., Xie, X.: Searching trajectories by locations: an efficiency study. In: SIGMOD, pp. 255–266. ACM (2010)
6. Dai, J., Yang, B., Guo, C., Jensen, C.S., Hu, J.: Path cost distribution estimation using trajectory data. PVLDB **10**(3), 85–96 (2017)
7. Dijkstra, E.W.: A note on two problems in connection with graphs. Numerische Math **1**, 269–271 (1959)
8. Ding, Z., Yang, B., Chi, Y., Guo, L.: Enabling smart transportation systems: A parallel spatio-temporal database approach. IEEE Trans. Comput. **65**(5), 1377–1391 (2016)
9. Ding, Z., Yang, B., Güting, R.H., Li, Y.: Network-matched trajectory-based moving-object database: Models and applications. IEEE Trans. Intell. Transp. Syst. **16**(4), 1918–1928 (2015)
10. Guo, C., Jensen, C.S., Yang, B.: Towards total traffic awareness. SIGMOD Record **43**(3), 18–23 (2014)
11. Guo, C., Yang, B., Andersen, O., Jensen, C.S., Torp, K.: Ecosky: Reducing vehicular environmental impact through eco-routing. In: ICDE, pp. 1412–1415 (2015)
12. Guo, D., Zhu, Y., Xu, W., Shang, S., Ding, Z.: How to find appropriate automobile exhibition halls: Towards a personalized recommendation service for auto show. Neurocomputing **213**, 95–101 (2016)
13. Han, J., Zheng, K., Sun, A., Shang, S., Wen, J.: Discovering neighborhood pattern queries by sample answers in knowledge base. In: ICDE, pp. 1014–1025 (2016)
14. Hu, S., Wen, J., Dou, Z., Shang, S.: Following the dynamic block on the Web. World Wide Web **19**(6), 1077–1101 (2016)
15. Li, J., Wang, Y., Zhong, Y., Guo, D., Zhu, S.: Aggregate location recommendation in dynamic transportation networks. World Wide Web Journal, online first:1–18 (2017)
16. Liu, A., Wang, W., Shang, S., Li, Q., Zhang, X.: Efficient task assignment in spatial crowdsourcing with worker and task privacy protection. online first:1–26 (2017)
17. Liu, J., Shang, S., Zheng, K., Wen, J.: Multi-view ensemble learning for dementia diagnosis from neuroimaging: An artificial neural network approach. Neurocomputing **195**, 112–116 (2016)
18. Liu, J., Zhao, K., Sommer, P., Shang, S., Kusy, B., Lee, J., Jurdak, R.: A novel framework for online amnesic trajectory compression in resource-constrained environments. IEEE Trans. Knowl. Data Eng. **28**(11), 2827–2841 (2016)
19. Liu, K., Li, Y., Ding, Z., Shang, S., Zheng, K.: Benchmarking big data for trip recommendation. In: ICCCN, pp. 1–6 (2014)
20. Liu, K., Yang, B., Shang, S., Li, Y., Ding, Z.: MOIR/UOTS: trip recommendation with user oriented trajectory search. In: MDM, pp. 335–337 (2013)
21. Papadias, D., Shen, Q., Tao, Y., Mouratidis, K.: Group nearest neighbor queries. In: ICDE, pp. 301–312 (2004)
22. Papadias, D., Tao, Y., Mouratidis, K., Hui, C.K.: Aggregate nearest neighbor queries in spatial databases. TODS **30**(2), 529–576 (2005)
23. Rong, X., Chen, Z., Mei, Q., Egoset, E.Adar.: Exploiting word ego-networks and user-generated ontology for multifaceted set expansion. In: WSDM, pp. 645–654 (2016)
24. Shang, S., Chen, L., Jensen, C.S., Wen, J., Kalnis, P.: Searching trajectories by regions of interest. IEEE Trans. Knowl. Data Eng. **29**(7), 1549–1562 (2017)
25. Shang, S., Chen, L., Wei, Z., Jensen, C.S., Wen, J., Kalnis, P.: Collective travel planning in spatial networks. IEEE Trans. Knowl. Data Eng. **28**(5), 1132–1146 (2016)
26. Shang, S., Chen, L., Wei, Z., Jensen, C.S., Zheng, K., Kalnis, P.: Trajectory similarity join in spatial networks. PVLDB **10**(11), 1178–1189 (2017)
27. Shang, S., Deng, K., Xie, K.: Best point detour query in road networks. In: ACM GIS, pp. 71–80 (2010)
28. Shang, S., Ding, R., Yuan, B., Xie, K., Zheng, K., Kalnis, P.: User oriented trajectory search for trip recommendation. In: EDBT, pp. 156–167 (2012)
29. Shang, S., Ding, R., Zheng, K., Jensen, C.S., Kalnis, P., Zhou, X.: Personalized trajectory matching in spatial networks. VLDB J. **23**(3), 449–468 (2014)

30. Shang, S., Guo, D., Liu, J., Liu, K.: Human mobility prediction and unobstructed route planning in public transport networks. In: MDM(2), pp. 43–48 (2014)
31. Shang, S., Guo, D., Liu, J., Zheng, K., Wen, J.: Finding regions of interest using location based social media. Neurocomputing **173**, 118–123 (2016)
32. Shang, S., Liu, J., Zheng, K., Lu, H., Pedersen, T.B., Wen, J.: Planning unobstructed paths in traffic-aware spatial networks. GeoInformatica **19**(4), 723–746 (2015)
33. Shang, S., Lu, H., Pedersen, T.B., Xie, X.: Finding traffic-aware fastest paths in spatial networks. In: SSTD, pp. 128–145 (2013)
34. Shang, S., Lu, H., Pedersen, T.B., Xie, X.: Modeling of traffic-aware travel time in spatial networks. In: MDM (1), pp. 247–250 (2013)
35. Shang, S., Yuan, B., Deng, K., Xie, K., Zheng, K., Zhou, X.: Pnn query processing on compressed trajectories. GeoInformatica **16**(3), 467–496 (2012)
36. Shang, S., Yuan, B., Deng, K., Xie, K., Zhou, X.: Finding the most accessible locations: reverse path nearest neighbor query in road networks. In: ACM SIGSPATIAL, pp. 181–190 (2011)
37. Shang, S., Zheng, K., Jensen, C.S., Yang, B., Kalnis, P., Li, G., Wen, J.: Discovery of path nearby clusters in spatial networks. IEEE Trans. Knowl. Data Eng. **27**(6), 1505–1518 (2015)
38. Shang, S., Zhu, S., Guo, D., Lu, M.: Discovery of probabilistic nearest neighbors in traffic-aware spatial networks. World Wide Web **20**(5), 1135–1151 (2017)
39. Xie, K., Deng, K., Shang, S., Zhou, X., Zheng, K.: Finding alternative shortest paths in spatial networks. ACM Trans. Database Syst. **37**(4), 29:1–29:31 (2012)
40. Xie, Q., Shang, S., Yuan, B., Pang, C., Zhang, X.: Local correlation detection with linearity enhancement in streaming data. In: CIKM, pp. 309–318 (2013)
41. Xie, X., Lu, H., Chen, J., Shang, S.: Top-k neighborhood dominating query. In: DASFAA, pp. 131–145 (2013)
42. Xu, Y., Chen, L., Yao, B., Shang, S., Zhu, S., Zheng, K., Li, F.: Location-based top-k term querying over sliding window. In: WISE, pp. 299–314 (2017)
43. Yang, B., Dai, J., Guo, C., Jensen, C.S.: Pace: A PAth-CEntric paradigm for stochastic path finding. VLDB Journal, online first (2015)
44. Yang, B., Guo, C., Jensen, C.S., Kaul, M., Shang, S.: Stochastic skyline route planning under time-varying uncertainty. In: ICDE, pp. 136–147 (2014)
45. Zheng, K., Shang, S., Yuan, N.J., Yang, Y.: Towards efficient search for activity trajectories. In: ICDE, pp. 230–241 (2013)
46. Zheng, K., Su, H., Zheng, B., Shang, S., Xu, J., Liu, J., Zhou, X.: Interactive top-k spatial keyword queries. In: ICDE, pp. 423–434 (2015)
47. Zheng, K., Zheng, Y., Yuan, N.J., Shang, S.: On discovery of gathering patterns from trajectories. In: ICDE, pp. 242–253 (2013)
48. Zheng, K., Zheng, Y., Yuan, N.J., Shang, S., Zhou, X.: Online discovery of gathering patterns over trajectories. IEEE Trans. Knowl. Data Eng. **26**(8), 1974–1988 (2014)
49. Zhu, S., Wang, Y., Shang, S., Zhao, G., Wang, J.: Probabilistic routing using multimodal data. Neurocomputing **253**, 49–55 (2017)