Top-k Spatial Keyword Queries on Road Networks

Information Retrieval

12/21/2012

Wesley Mathew
Introduction

• A top-k spatial keyword query returns the k best objects ranked in terms of both the distance to the query location and textual relevance to the query keywords.

• In top-K spatial keyword queries over a road network, the distance between the objects and query location is constrained by a road network.

• The basic indexing architecture is the combination of a spatio-textual index such as IR – tree and the road network architecture.
Example

when road networks are considered, the top-1 hotel is p4 hotel is p9
Example 2

Query Bar & cafe
Outlines

• Top K spatial keyword queries on road networks
• Indexing architecture
• Basic query processing
• Enhanced approach
• Overlay network
• Results
Top K spatial keyword queries on road networks

The road network is a graph $G(V, E, W)$

$V$ is the set of vertices - The vertex $v \in V$ represents the road intersection or end of the road network.

$E$ is the set of edges - Edge $(v, v') \in E$ represents a road segment.

$W$ is the set of weights (network distances) – A weight $w \in W$ is associated with each edge $(v, v')$

A bidirectional graph is considered
Spatio-textual object (p)

- Set of spatio-textual objects \( p \in P \) associated to the edges \( E \) of the road network.
- \( p.l = \) spatial location
- \( p.d = \) textual description
- \( ||p,p'|| = \) The shortest path between two objects \( p \) and \( p' \) on a road network graph \( G \)
Problem statements

• Query, Qn= (ql, q.d, q.k)
• q.l is the query location, it lies on the edge of the road network
• q.d is the set of query keywords
• q.k is the number of results of interest.
• We assume that the query location q.l lies on an edge of the road network. This assumption can be bypassed by finding the nearest edge of a given query location.
• A query function will return q.k spatio-textual objects in descending order of score a T

Top-K Spatial Keyword Queries on Road Networks
The score of an object $T(p)$

$$T(p) = \frac{\theta(p.d, q.d)}{1 + \alpha \cdot \delta(p.l, q.l)}$$

- $\delta(p.l, q.l)$ is the network proximity between query location and object location.
- $\theta(p.d, q.d)$ is the textual relevance of $p.d$ according to the query keyword $q.d$.
- $\alpha$ is the positive real number and defines the importance of one measure over others.
  - If it is 0 then only consider the text relevancy
  - If it is greater than 1 then increase the importance of location proximity over textual relevance.
Textual relevancy

\[ \theta(p,d,q,d) = \frac{\sum_{t \in q,d} w_{t,p,d} \cdot w_{t,q,d}}{\sqrt{\sum_{t \in p,d} (w_{t,p,d})^2 \cdot \sum_{t \in q,d} (w_{t,p,d})^2}} \]

\[ w_{t,p,d} = 1 + \ln (f_t \cdot p_d) \]

\[ w_{t,q,d} = \ln \left( 1 + \frac{|p|}{d_{f_t}} \right) \]

- \( f_t \) is the number of occurrence of term t in p.d.
- \( |p| \) is the number of object in the collection
- \( d_{f_t} \) is the number of objects contains term t in their description
Network proximity

\[ \delta(p, l, q, l) = ||p, l, q, l|| \]

- The shortest path between p and q
Indexing architecture

- The basic indexing architecture is the combination of a spatio-textual index such as IR–tree, and the road network architecture.
- This framework supports starting query at any location of the road network
Indexing architecture

Spatial Component

Map B-tree

Adjacency Component

Mapping Component

Spatial textual Component
Spatial component

- MBR of the polyline is stored in the network R-tree (R*-tree). The detailed polyline is stored in the *polyline file*
- Locating the edge in which the query q.l lies is executed in two steps
  1. A point location query is performed in the network R tree for finding all MBRs covering q.l
  2. Filtering process is executed for selecting the exact polyline.
Adjacency component

- It points to the adjacent vertices of a given vertex, permitting traversing the network from vertex to vertex.
- The adjacent B-tree points to the block in the adjacency file where the adjacent vertices of a given vertex \( V_i \) are.
- Adjacency file stores the id of the edge (e.g., \( (V, V_i) \)), and length of the edge (e.g., \( |V,V_i| \))
Mapping component

- Map B-tree that maps an edge id to the MBR of the edge and it points to the polyline of the edge.
Spatial textual component

- It stores spatio and textual information of the object
- Spatio and textual index is accessed to retrieve all spatial objects inside the MBR enclosing the edge
Algorithm: Basic Query Processing (Query QN)

1. input $Q_N = (q.l, q.d, q.k)$
2. MaxHeap $H_{q,k} \leftarrow \emptyset$ // $q.k$ best objects in decreasing order of
3. $\varepsilon < 0$ // $k$-th score in $H_{q,k}$; While $|H_{q,k}| < q.k$, $\varepsilon = 0$
4. MinHeap $N \leftarrow \emptyset$ // vertices $v$ in increasing order of $|v, q.l|$ 
5. $(v, v') \leftarrow$ network edge in which $q.l$ lies
6. Compute $|v, q.l|$ and $|v', q.l|$ using the polyline of $(v, v')$
7. Insert $v$ and $v'$ into $N$, mark $(v, v')$ are visited
8. $C \leftarrow$ FindCandidates (ID($v, v'$), $p.d, \varepsilon$)
9. Update $H_{q,k}$ (and $\varepsilon$ ) with $P \in C$
10. $V \leftarrow$ N.pop()
11. while $v \neq \emptyset$ and $\varepsilon < \frac{1}{1 + \alpha \cdot \delta(u.l, q.l)}$ do
12. For each non visited adjacent edge $(v, v')$ of $v$ do
13. $C \leftarrow$ FindCandidates (ID($v, v'$), $p.d, \varepsilon$)
14. Update $H_{q,k}$ and $P \in C$
15. Insert $v'$ into $N$ mark $(v, v')$ as visited
16. End for
17. $V \leftarrow$ N.pop()
18. End while
19. Return $H_{q,k}$
Example

Find candidates procedure is used to get the object lying on the (V11, V12) with the keywords in q.d, whose score is higher than $\varepsilon$. Procedure return p0 that is added into $H_{q.k}$, and $\varepsilon$ is updated ($\varepsilon=1/4$).

$q.d=\text{“cafe”}$

$|v11, q.l| = 2$

$|v12, q.l| = 8$

Aggregate score $T(V11) = 1/3$

Aggregate score $T(V12) = 1/9$
Enhancement approach

- An enhanced approach that indexes the objects lying on the edges of the road network for improving the query processing performance
Mapping component

- It connects between the network and the objects through the edge of the network
- The mapping component that maps a key composed by the pair \( \text{edge id} \) and \( \text{term id} \) to the inverted list that contains the objects lying on the edge with term \( t \) in their description
- The mapping component contains also the \( \text{maximum impact} \ t \) of a given term \( t \) among the description of the objects lying on a given edge.
Inverted file component

- It replaces the spatio-textual component
- It composed by inverted lists and vocabulary

1. The network distance between the object and the reference vertex of the edge
2. The impact of the term $t_i$, in the description of the object

- The Vocabulary file stores the document frequency of each term.
Enhanced query process

Only change lies on the findCandidates procedure

• The mapping component is accessed to compute an upper-bound score using the maximum impact $\delta t$ of a given term $t \in q.d$ and the minimum network distance between the edge and the query location.

• If the upper-bound score is higher than $\varepsilon$, the inverted lists are accessed.
Benefits of enhanced query

1. Only the lists that can produce relevant objects are accessed.
2. Retrieving the inverted lists is faster than processing a query location for retrieving all objects inside a given MBR.
3. It does not require a filtering process to remove the objects that are inside the MBR of the edge.
4. It does not require computing the network distance between the objects and the end vertices of the edges.
The aggregation cost takes into account the textual similarity between the region and the candidate edge, and the number of border vertices of the region.
Example

Query location is on the edge \((v_{11}, v_{12})\)
Query keyword = \{pub\}. 
The documents with similar content are stored in the same region, which increases the probability of pruning more regions during the query processing. (Solsiden in R3 Store in R1, hotel in R2 Cafe in R4)
Multiple layers

• Each layer of multiple layers corresponds to an abstract representation of the subsequent layer.
• Neighbouring regions with similar content are grouped, which permits pruning even larger regions of the road network.
Adjacency hierarchy

• A hierarchy model to represent the adjacencies of the border vertices.
• The adjacency hierarchy helps to avoid regions that do not contain objects relevant for the query in terms of textual relevance.
Algorithm : Overlay Query processing

1. INPUT: Top-k spatial keyword query on road networks QN
2. Lines 2-9 of Algorithm 1 (Basic Query Processing Algorithm)
3. \( V\leftarrow N\).pop()
4. while \( v \neq \emptyset \) and \( \epsilon < \frac{1}{1 + \alpha \cdot \delta(u, l, q.l)} \) do
5.   For each nonvisited adjacent vertex \( u \) of \( v \) do
6.     if \( u \) is an intermediated vertex then
7.       \( R\leftarrow \) region associated with then intermediary vertex \( u \)
8.       if \( \epsilon < \frac{\theta(R, d, q.d)}{1 + \alpha \cdot \delta(u, l, q.l)} \) then
9.         insert \( u \) into \( N \)
10.     end if
11.     mark \( u \) as visited
12.   else // \( u \) is a regular vertex
13.     Lines 13-15 of Algorithm 1
14.   end if
15. end for
16. \( V\leftarrow N\).pop()
17. end while
18. Return \( H^{q.k} \)
Example

the vertex p2 is added in the top K
Results: Varying the number of Results (K)

Graphs showing the response time and edges expanded for different values of K. The graphs indicate how the time and edges processed change as the number of results increases.
Results: Varying the number of keywords

Response time

Edges Expanded
Results: Varying the query preference parameter
References

Thanks for your attention

Thank you

Questions?