IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 12, December 2015

Turn Around Group Query Optimization System Analysis

M. Venkatasalam[1], V. Bakyalakshmi[2]

Research Scholar, Department of Computer Science,

Sri Jayendra Saraswathy Maha Vidyalaya College of Arts & Science, Coimbatore.[1]

Head, Department of Computer Applications,

Sri Jayendra Saraswathy Maha Vidyalaya, College of Arts & Science ,Coimbatore.[2]

Abstract: The group nearest neighbor query uses group points to provide the optimal solution for the nearest group point in the dataset. The novel type of spatial keyword query called Group Nearest Group (GNG) query will be used to optimize the query. Given a data point set D, a query point set Q and an integer k, the Group Nearest Group query finds a subset of points from D, ω ($|\omega| \leq k$), such that the total distance from all points in Q to the nearest point in ω is no greater than any other subset of points in D. Each nearest point obtained matches at least one of the query keywords. For processing this query several algorithms are proposed. The processing of GNG query consists of Exhaustive Hierarchical Combination algorithm and Subset Hierarchical Refinement algorithm. A group nearest neighbor (GNN) query returns the location of a meeting place that minimizes the aggregate distance from a spread out group of users; for example, a group of users can ask for a restaurant that minimizes the total travel distance from them. The duplicates in the dataset can be identified to improve the search query from the given data. The dataset can be analyzed to find out the duplicates in the data set. The applications of group query come from location-based services, e.g., finding a meeting venue for a group of people such that the traveling distances is minimized. In order to prune large portion of query objects reducing the number of node accesses, this extensive experiments on spatial databases is effective in reducing group query response time which exhibits good scalability with the query objects and the number of query keywords.

Keywords: Boolean spatial keyword query, reverse k Boolean spatial keyword query, road network, query processing.

I. INTRODUCTION

RKNN retrieval has received lots of attention from the database research community in the past decade, due to its importance in a wide spectrum of applications such as decision support, profile - based marketing, and resource allocation. Given a set P of data points and a query point q in a euclidean space, a reverse k nearest neighbor (RkNN) query finds the points in P that have q as one of their k nearest neighbors (NNs). The queries of two RNN (k ¼ 1) are issued at q1 and q2 in the euclidean space. The RNN of q1 is Ø, as none of the objects takes q1 as its nearest neighbor; and the RNN of q2 is A as A's NN is q2.

The traditional RkNN is retrieved from two aspects,

i) Different from existing RkNN search that assumes a euclidean space(road network)

ii)The textual characteristics of the objects are considered in addition to spatial properties.The combination of spatial and textual properties offers greater flexibility to its users when looking for interesting objects.

A new query named reverse top-k Boolean spatial keyword (RkBSK) query is used which assumes objects on the road network, and returns the objects having a specified query point q as one of the answer objects for the top-k Boolean spatial keyword (TkBSK) query. The significance of RkBSK queries and the lack of efficient search algorithm for processing RkBSK retrieval, efficient algorithms based on filter – and - refinement framework to support RkBSK search is defined. Both spatial and textual information to prune the search space significantly is utilised. It can tackle exact RkBSK retrieval with an arbitrary k, without any pre-computation.

ABOUT SPATIAL MINING

Spatial data mining is the process of discovering interesting and previously un- known, but potentially useful patterns from large spatial datasets. Extracting interesting and useful patterns from spatial datasets is more difficult than extracting the corresponding patterns from traditional numeric and categorical data due to the complexity of spatial data types, spatial relationships, and spatial autocorrelation. Spatial data is about instances located in a physical space. When spatial information becomes dominant interest, spatial data mining should be applied. Spatial data structures can facilitate spatial mining. Standard data mining algorithms can be modified for spatial data mining, with a substantial part of preprocessing to take into account of spatial information. The main difference between data mining in relational data base system and in spatial database system is that attributes of the neighbors of some object of interest may have an influence on the object and therefore have to be considered as well.

The explicit location and extension of spatial objects define implicit relations of spatial neighborhood (such as

topological, distance and direction relations) which are used by spatial data mining algorithms. Therefore, new techniques are required for effective and efficient data mining. Spatial data is stored in spatial databases. Multidimensional trees are used, in order to build indices for these data (e.g. quad trees, k-d trees, R-trees, R*-trees). Often attributes of spatial objects are still one-dimensional, so that this non-spatial part can be stored in relational databases with references to the spatial data. Spatial operations like spatial join and map overlay are the most expensive.

Spatial analysis or spatial statistics includes any of the formal techniques which study entities using their topological, geometric, or geographic properties. The phrase properly refers to a variety of techniques, many still in their early development, using different analytic approaches and applied in fields as diverse as astronomy, with its studies of the placement of galaxies in the cosmos, to chip fabrication engineering, with its use of 'place and route' algorithms to build complex wiring structures. The phrase is often used in a more restricted sense to describe techniques applied to structures at the human scale, most notably in the analysis of geographic data. The phrase is even sometimes used to refer to a specific technique in a single area of research, for example, to describe geo statistics.

Complex issues arise in spatial analysis, many of which are neither clearly defined nor completely resolved, but form the basis for current research. The most fundamental of these is the problem of defining the spatial location of the entities being studied. For example, a study on human health could describe the spatial position of humans with a point placed where they live, or with a point located where they work, or by using a line to describe their weekly trips; each choice has dramatic effects on the techniques which can be used for the analysis and on the conclusions which can be obtained.

Other issues in spatial analysis include the limitations of mathematical knowledge, the assumptions required by existing statistical techniques, and problems in computer based calculations. Spatial analysis confronts many fundamental issues in the definition of its objects of study, in the construction of the analytic operations to be used, in the use of computers for analysis, in the limitations and particularities of the analyses which are known, and in the presentation of analytic results. Many of these issues are active subjects of modern research. Common errors often arise in spatial analysis, some due to the mathematics of space, some due to the particular ways data are presented spatially, some due to the tools which are available. Census data, because it protects individual privacy by aggregating data into local units, raises a number of statistical issues.

ABOUT NEAREST KEYWORD SEARCH

Nearest neighbor search (NNS), also known as proximity search, similarity search or closest point search, is an optimization problem for finding closest points in metric spaces. The problem is: given a set S of points in a metric space M and a query point $q \in M$, find the closest point in S to q. In metric space, there is a valid concept of distance between points. The distance between two features is calculated by calculating the points distance between them. The lesser the distance between them, the more similar they are in appearance.

Various solutions to the Nearest Keyword Search problem have been proposed. The quality and usefulness of the algorithms are determined by the time complexity of queries as well as the space complexity of any search data structures that must be maintained. The informal observation usually referred to as the curse of dimensionality states that there is no general-purpose exact solution for NNS in high-dimensional Euclidean space using polynomial preprocessing and poly logarithmic search time.

The simplest solution to the Nearest Keyword Search problem is to compute the distance from the query point to every other point in the database, keeping track of the "best so far". This algorithm, sometimes referred to as the naive approach, has a running time of O(Nd) where N is the cardinality of S and d is the dimensionality of M. There are no search data structures to maintain, so linear search has no space complexity beyond the storage of the database. Naive search can, on average, outperform space partitioning approaches on higher dimensional spaces.

Algorithms for searching virtual spaces are used in constraint satisfaction problem, where the goal is to find a set of value assignments to certain variables that will satisfy specific mathematical equations . They are also used when the goal is to find a variable assignment that will maximize or minimize a certain function of those variables. Algorithms for these problems include the basic brute-force search (also called "naïve" or "uninformed" search), and a variety of heuristics that try to exploit partial knowledge about structure of the space, such as linear relaxation, constraint generation, and constraint propagation.

An important subclass are the local search methods, that view the elements of the search space as the vertices of a graph, with edges defined by a set of heuristics applicable to the case; and scan the space by moving from item to item along the edges, for example according to the steepest descent or best-first criterion, or in a stochastic search. This category includes a great variety of general heuristic methods, such as simulated annealing, A-teams, and genetic programming that combine arbitrary heuristics in specific ways. This class also includes various tree search algorithms, that view the elements as vertices of a tree and traverse that tree in some special order. Examples of the latter include the exhaustive methods such as depth-first search and breadth-first search, as well as various heuristic-based search tree pruning methods such as backtracking and branch and bound.

Unlike general heuristics, which at best work only in a probabilistic sense, many of these tree-search methods are guaranteed to find the exact or optimal solution, if given

DOI 10.17148/IJARCCE.2015.41208

enough time. Another important sub-class consists of algorithms for exploring the game tree of multiple-player games, such as chess or backgammon, whose nodes consist of all possible game situations that could result from the current situation.

The goal in these problems is to find the move that provides the best chance of a win, taking into account all possible moves of the opponent(s). Similar problems occur when humans or machines have to make successive decisions whose outcomes are not entirely under one's control, such as in robot guidance or in marketing, financial or military strategy planning. This kind of problem - combinatorial search - has been extensively studied in the context of artificial intelligence. Examples of algorithms for this class are the minimax algorithm, alpha-beta pruning, and the A* algorithm.

An increasing number of applications require the efficient execution of nearest neighbor (NN) queries constrained by the properties of the spatial objects. Due to the popularity of keyword search, particularly on the Internet, many of these applications allow the user to provide a list of keywords that the spatial objects (henceforth referred to simply as objects) should contain, in their description or other attribute. For example, online yellow pages allow users to specify their address and a set of keywords, and return businesses whose description contains these keywords, ordered by their distance to the specified address location. As another example, real estate web sites allow users to search for properties with specific keywords in their description and rank them according to their distance from a specified location.

A spatial keyword query consists of a query area and a set of keywords. The answer is a list of objects ranked according to a combination of their distance to the query area and the relevance of their text description to the query keywords. A simple yet popular variant, which is used in our running example, is the distance-first spatial keyword query, where objects are ranked by distance and keywords are applied as a conjunctive filter to eliminate objects that do not contain them.

OBJECTIVE

The objective of "Efficient Turn Around Group Query optimization in Spatial Data "is to provide the finest solution for the group point in the dataset. In nearest neighbor queries, an optimization problem is evaluated for finding the closest points in metric spaces. Given a data point set D, a query point set Q and an integer k, the Group Nearest Group query finds a subset of points from D, ω ($|\omega| \leq k$), such that the total distance from all points in Q to the nearest point in ω is no greater than any other subset of points in D.

The processing focus of our approaches is on minimizing the access and evaluation of subsets of cardinality k in D since the number of such subsets is exponentially greater than the dataset. To do that, the hierarchical blocks of data points at high level are used to find an intermediate solution and then refined by following the guided search direction at low level so as to prune irrelevant subsets.

Reverse top-k Boolean Spatial Keyword (RkBSK) retrieval, which assumes objects are on the road network and considers both spatial and textual information. Given a data set P on a road network and a query point q with a set of keywords, an RkBSK query retrieves the points in P that have q as one of answer points for their top-k Boolean spatial keyword queries. We formalize the RkBSK query and then propose filter-and-refinement framework based algorithms for answering RkBSK search with arbitrary k and no any pre-computation. To accelerate the query process, several novel pruning heuristics that utilize both spatial and textual information are employed to shrink the search space efficiently.

II. REVIEW LITERATURE RANKED REVERSE NEAREST-NEIGHBOR SEARCH

A range nearest-neighbor (RNN) query retrieves the nearest neighbor (NN) for every point in a range. Consider the ranges as (hyper) rectangles and propose efficient in-memory processing and secondary memory pruning techniques for RNN queries in both 2D and high-dimensional spaces. These techniques are generalized for kRNN queries, which return the k nearest neighbors for every point in the range [1]. In general, processing an NN query on a spatial index involves two interleaving phases:

- secondary memory pruning of distant index nodes
- In-memory computation of the nearest neighbors.

LOCATION-BASED INSTANT SEARCH

Location-based instant search that combines location based keyword search with instant search is formulated. Initially the filtering-effective hybrid index (FEH) is evaluated. Then development of indexing and search techniques are utilized for the FEH index and store prefix information to efficiently answer instant queries.

First present an index structure called "filtering-effective hybrid" (FEH) index. It judiciously uses two types of keyword filters in a node of a spatial tree based on the selectiveness of each keyword. One filter called child filter, maps keywords and their corresponding children nodes. Another Filter called "object filter", maps keywords to their corresponding records in the sub tree of the nodes. During a traversal of the FEH index tree, the object filter at each node allows to directly retrieve records for these keywords in the filter, thus bypassing those intermediate nodes in the sub tree. Next is to find answers to a query as the user is typing the keywords character by character. Existing index techniques are utilized and queries are answered using .

EFFICIENT RETRIEVAL OF THE TOP-k MOST RELEVENT SPATIAL WEB OBJECTS

Location-based instant search that combines location based keyword search with instant search is formulated. Nearest neighbor (NN) queries on a spatial database is a classical problem. The k-NN algorithm for R-trees traverses an R-tree while maintaining a list ofk potential nearest neighbors in a priority queue in a Depth-First (DF) manner. The DF algorithm is sub-optimal, i.e., it accesses

more nodes than necessary. The Best-First (BF) algorithm achieves the optimal I/O performance by maintaining a heap with the entries visited so far, sorted by their mindist. DF can be more I/O consuming than BF. However, DF requires only bounded memory and at most a single tree path resides in memory during search.

The closest pair queries (CPQ) are a combination of spatial join and nearest neighbor queries, which find the pair with the minimum distance among all pairs from two data sets. The difference between nearest neighbor queries and closest pair queries is that the algorithms of the latter access two index structures (one for each data set) and utilize the distance function of the two intermediate nodes to prune the pairs.NNK specifies only one query location specifies a set of query locations .

KEYWORD SEARCH ON SPATIAL DATABASES

The spatial data search on k nearest neighbor queries is based on the Revived R*-tree index structure. The incremental methods for search have the following drawbacks:

- They cannot support objects in multidimensional space
- Their methods are low efficient for incremental query.

To solve such search problem efficiently, the novel incremental search on spatial data is applied to multidimensional spatial databases. The counter for every entry of RR*-tree index structure, which marks the number of nearest neighbor and thus offers the information about the influences of a query point.

CONTINUOUS SKYLINE QUERY

A continuous skyline query involves both static and dynamic dimensions. The spatio - temporal coherence of the problem and the continuous skyline query processing are based on the moving query points. Distinguish the data points that are permanently in the skyline and use them to derive a search bound. Investigate the connection between the spatial positions of data points and their dominance relationship, which provides an indication of where to find changes in the skyline and how to maintain the skyline continuously. The analyze finds the space and time of skyline query processing requires more cost for query to process and to find the result.

A dynamic skyline query retrieves the moving data objects that are not spatially dominated by any other object with respect to a given query point. Existing efforts on supporting such queries, however, supports location as a single dynamic attribute and one or more static dimensions.

AGGREGATE NEAREST NEIGHBOR QUERIES

Aggregate nearest neighbor queries return the object that minimizes an aggregate distance function with respect to a set of query points. The processing of such queries for the query where the position and accessibility of spatial objects are constrained by spatial (e.g., road) networks. Consider alternative aggregate functions and techniques that utilize Euclidean distance bounds, spatial access methods, and/or network distance materialization structures. The results show that their relative performance depends on the problem characteristics. In a realistic location-based application environment, a user can be indecisive about committing to a particular detour option. The user may wish to browse multiple (k) MDOs before making a decision. Furthermore, when a user moves, the kMDO results at one location may become obsolete. The continuous detour query (CDQ) processing based on incremental construction of a shortest path tree.

K-MEDIAN LOCAL SEARCH HEURISTICS

Local search algorithms move from solution to solution in the space of candidate solutions (the search space) by applying local changes, until a solution deemed optimal is found or a time bound is elapsed. Local search heuristics for the metric k-median and facility location problems define the locality gap of a local search procedure for a minimization problem as the maximum ratio of a locally optimum solution (obtained using this procedure) to the global optimum. Local search is a Meta heuristic method for solving computationally and optimization problems. Local search can be used on problems that can be formulated as finding a solution maximizing a criterion among a number of candidate solutions. For k-median, local search swaps the locality gap with the lower space found in the search. This is the first analysis of a local search for k-median that provides a bounded performance guarantee with only k medians.

DISTANCE BROWSING IN SPATIAL DATABASES

The process of finding the nearest k neighbors relies on estimating the network distance of the objects from the dataset, objects cannot be inserted into the middle until their exact distances are known. The distance browsing in spatial database is presented for finding the k nearest neighbors in a spatial network in a best-first manner using network distance. The distance browsing is based on pre computing the shortest paths between all possible vertices in the network and then making use of an encoding that takes advantage of the fact that the shortest paths from vertex to all of the remaining vertices can be decomposed into subsets based on the nearest edges on the shortest paths to them from the given point.

Thus, the amount of work depends on the number of objects that are examined and the number of links on the shortest paths to them from the given point, rather than depending on the number of vertices in the network. The amount of storage required to keep track of the subsets is reduced by taking advantage of their spatial coherence which is captured by the aid of a shortest path quad tree. The pre computation of the shortest paths along the network essentially decouples the process of computing shortest paths along the network from that of finding the neighbors, and thereby also decouples the domain S of the query objects and that of the objects from which the neighbors are drawn from the domain V of the vertices of the spatial network.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 12, December 2015

HIGH DIMENSIONAL SPACES: INDEX STRUCTURES

Query processing in high-dimensional spaces provides searching data in a relational database, a content based retrieval requires the search of similar objects as a basic functionality of the database system. The hyper planes are defined by a split dimension (the normal vector of the hyper plane) and a split value (defining the actual location of the hyper plane).Space partitioning is done by an algorithm that is similar to the well-known Quick sort algorithm although operating on secondary storage. This technique is invariant against a specific split strategy i.e., gives the freedom to partition the space according to arbitrary split dimensions and split values. To create an optimized space partitioning that is unbalanced and therefore cannot be achieved by a dynamic index structure.

Although partitioning is unbalanced, the algorithm guarantees that the resulting index structure is balanced. The results show that our bulk-load operation can be done in average O(n log n) time. Hilbert R-tree is created by externally sorting all the data vectors according to their Hilbert value and assigning equally sized, sub sequential portions of the sorted data to data pages. Finally, the bounding boxes of the data pages are stored in directory pages clustering these directory pages recursively until it reach a single root node. The costs for bulk loading a Hilbert R-tree are obviously in O(nlogn) due to external sorting. Partitioning of the data space can be done in a top-down fashion which means that hierarchically divide the d-dimensional space using (d-1)-dimensional hyper planes as borderlines between the partitions.

III. EXPERIMENTAL RESULTS

MODULE DESCRIPTION

The project is divided into four modules:

- Google View
- Data Group
- Group Combination
- Subset Refinement

GOOGLE VIEW

There is always a requirement to use a map in a web application, so by considering the preceding requirement I decided to write this article. So let us learn step-by-step how to integrate Google Maps with an ASP.Net web application.

Key points for integrating Google Maps:

1. Google provides the free API to integrate Google Maps in our application.
2. You can customize Google Maps depending on your requirements.

DATA GROUP

A real data set of points are collected which consists of the place with the longitude and latitude of the metropolitan city. The synthetic data points were obtained containing the uniformly distributed points around the city. These data sets are unified into a unit region. Q is distributed in an area whose Minimum Bound Rectangle is a percentage of the whole data space, denoted as M. All the data sets are indexed by R-trees for EHC and SHR.

GROUP COMBINATION

Exhaustive Hierarchical Combination algorithm minimizes the access and evaluation of potential subsets. The data points in EHC are hierarchically represented by data blocks, e.g., using R-tree. The algorithm process GNG query by treating the blocks as points to find an intermediate solution in higher hierarchical level. To refine the solution, the search space in lower hierarchical level is minimized by following the guided search direction.

SUBSET REFINEMENT

Subset Hierarchical Algorithm is a local search heuristic with support of the database techniques. In higher hierarchical level, each block is treated as a point by SHR to replace every element in the subset, and the resultant subset with the current best value is refined by visiting the children of the block. The solution of SHR is usually close to the global optimum and guaranteed to be within a factor of at most close to the global optimum.

EXPERIMENTAL SETUP

Deploy both real and synthetic data sets. As we employ three real road networks CA, NA, and SF as our data sets. For these datasets, POIs with real keyword sets are randomly generated in a. We also generate two synthetic datasets. The first data set is to study the impact of the keyword set size per data point on the search performance. We preserve the road network and the data points of NA but change the keyword settings to generate five sets, viz., NA-K2, NA-K4, NA-K6, The average distinct number of keywords per data point in each dataset is roughly 2, 4, 6, 8, and 10, respectively. The second data set is to explore the impact of data point density on the search performance. We preserve the road network of NA but change the number of data points per edge to generate five datasets, NA-C4, NA-C6, NA-C8, NA-C10, and NA-C12. For every dataset NA-Ci, the average number of data points per edge is approximately set to i. The experiments investigate the performance of the proposed algorithms under a variety of parameters

(i)The response time

(ii) The number of page accesses by various algorithms during the search; and

(iii)The number of data points pruned by each pruning heuristic.

All data sets are indexed by our modified CCAM model with the disk page size fixed to 4,096 bytes. In each experiment, we vary only one parameter and fix other parameters at their defaults. Hundred random queries are evaluated in every experiment and their average performance is reported. To be more specific, the query is randomly located in one edge of the road network, and the query keywords are randomly extracted from the vocabulary of the data set. The server maintains a buffer of 200 pages with LRU as the cache replacement policy.

Effectiveness of Pruning Heuristics The first set of experiments is to verify the effectiveness of our presented pruning heuristics based on the number of data points pruned. As different lemmas/heuristics are introduced for different purposes, the points they try to prune are different. For Lemma 1 and Lemma 2, they are integrated simultaneously in RkBSK algorithm, and hence we report their joint pruning power based on the number of data points they, but not BM, can discard. For Heuristic 1 and Heuristic 2, we refer to the data points they, but not newly developed lemmas, can prune. For Heuristic 3, we measure those candidate points in the candidate set Sc that can be discarded without invoking BV algorithm. We change jq.keyj from 1 to 9 and depict the results in Fig. 9. We also vary k and show the results in Evidently, all the lemmas and heuristics have excellent pruning power. Take Heuristic 1 for NA as an example. It saves the detailed examination of about 33,026 points when jq.keyj¼ 5. Based on our experiments, Heuristic 3 is not as effective as others. This is because Heuristic 3 is only applied to the candidate points in Sc. Since the majority of data points have already been pruned away by Lemmas, Heuristic 1, and Heuristic 2, the candidate set is not big, which leaves

IV. CONCLUSIONAND FUTURE WORK

The Efficient Turn around Group Query retrieves number of objects from Query keyword Q with minimum sum of distances to its nearest Data points with integration with Google map view. Reverse top-k Boolean spatial keyword and Filter and Refinement framework algorithm, prunes the query objects and finally the minimized summed distance is calculated. The number of node accesses is also reduced which reduces the query response time, which exhibits good scalability with the query objects and the number of query keywords.

Reverse top-k Boolean spatial keyword and Filter and Refinement framework algorithm provide good scalability and accuracy which is implemented by geographical data sets in Google map view. In future work the time constraint will be introduced to improve the accuracy level, response time and further the data sets based on textual data also can be integrated on their geographical location with its graphical map points of the scale (x, y axis) will be updated to prune the search time of the algorithm.

REFERENCES

[1]. X. Cao, L. Chen, G. Cong, C. S. Jensen, Q. Qu, A. Skovsgaard, D. Wu, and M. L. Yiu, "Spatial keyword querying," in Proc. Int. Conf. Conceptual Model., 2012, pp. 16–29.

[2]. X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi, "Collective spatial keyword querying," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2011, pp. 373–384.

[3]. A. Cary, O. Wolfson, and N. Rishe, "Efficient and scalable method for processing top-k spatial Boolean queries," in Proc. Int. Conf. Sci. Statistical Database Manage., 2010, pp. 87–95.

[4]. M. A. Cheema, W. Zhang, X. Lin, Y. Zhang, and X. Li, "Continuous reverse k nearest neighbors queries in euclidean space and in spatial networks," VLDB J., vol. 21, no. 1, pp. 69–95, Feb. 2012.

[5]. L. Chen, G. Cong, C. S. Jensen, and D. Wu, "Spatial keyword query processing: An experimental evaluation," Proc. VLDB Endowment, vol. 6, no. 3, pp. 217–228, Jan. 2013.

[6]. G. Cong, C. S. Jensen, and D. Wu, "Efficient retrieval of the top-k most relevant spatial web objects," Proc. VLDB Endowment, vol. 2, no. 1, pp. 337–348, Aug. 2009.

[7]. I. D. Felipe, V. Hristidis, and N. Rishe, "Keyword search on spatial databases," in Proc. Int. Conf. Data Eng., 2008, pp. 656–665.

[8]. Y. Gao, B. Zheng, G. Chen, W.-C. Lee, K. C. Lee, and Q. Li, "Visible reverse k-nearest neighbor query processing in spatial databases," IEEE Trans. Knowl. Data Eng., vol. 21, no. 9, pp. 1314– 1327, Sep. 2009.

[9]. F. Korn and S. Muthukrishnan, "Influence sets based on reverse nearest neighbor queries," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2000, pp. 201–212.

[10]. K. C. Lee, B. Zheng, and W.-C. Lee, "Ranked reverse nearest neighbor search," IEEE Trans. Knowl. Data Eng., vol. 20, no. 7, pp. 894–910, Jul. 2008.

[11]. G. Li, J. Feng, and J. Xu, "Desks: Direction-aware spatial keyword search," in Proc. Int. Conf. Data Eng., 2012, pp. 474–485.

[12]. Z. Li, K. C. Lee, B. Zheng, W.-C. Lee, D. Lee, and X. Wang, "IRtree: An efficient index for geographic document search," IEEE Trans. Knowl. Data Eng., vol. 23, no. 4, pp. 585–599, Apr. 2011.

[13] .G. Li, Y. Li, J. Li, L. Shu, and F. Yang, "Continuous reverse k nearest neighbor monitoring on moving objects in road networks," Inf. Syst., vol. 35, no. 8, pp. 860–883, Dec. 2010.

[14]. J. Lu, Y. Lu, and G. Cong, "Reverse spatial and textual k nearest neighbor search," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2011, pp. 349–360.

[15]. S. Luo, Y. Luo, S. Zhou, G. Cong, and J. Guan, "Distributed spatial keyword querying on road networks," in Proc. Int. Conf. Extending Database Technol., 2014, pp. 235–246.

[16]. D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query processing in spatial network databases," in Proc. Int. Conf. Very Large Data Bases, 2003, pp. 802–813.

[17]. J. B. Rocha-Junior, O. Gkorgkas, S. Jonassen, and K. Norvag, "Efficient processing of top-k spatial keyword queries," in Proc. Int. Symp. Advances Spatial Temporal Databases, 2011, pp. 205–222.

[18].J. B. Rocha-Junior and K. Norvag, "Top-k spatial keyword queries on road networks," in Proc. Int. Conf. Extending Database Technol., 2012, pp. 168–179.

[19]. M. Safar, D. Ibrahimi, and D. Taniar, "Voronoi-based reverse nearest neighbor query processing on spatial networks," Multimedia Syst., vol. 15, no. 5, pp. 295–308, Oct. 2009.

[20]. S. Shekhar and D.-R. Liu, "CCAM: A connectivity-clustered access method for networks and network computations," IEEE Trans. Knowl. Data Eng., vol. 9, no. 1, pp. 102–119, Jan. 1997.