Extensible Location Dependent Spatial Object Search on Road Networks
P Karthigeyan¹, M Senthil Raja², B Prabhu³, R Gnanaselvam⁴
¹PG scholar, Pondicherry Engineering College, Pondicherry-605014, India
²³Assistant Professor, VI institute of Technology, Chennai, India
⁴PG Scholar, SRM University, Chennai, India

Abstract—Extensible Location-Dependent Spatial Object that searches for spatial object on road networks. Here it provides Distance Indexing technique and providing a dynamic object mapping mechanism of spatial object search. It is very efficient and flexible for various types of queries, namely, range search and nearest neighbor search, on objects over large-scale networks. In spatial object search, a large road network is organized as a hierarchy of interconnected regional sub-networks augmented with 1) shortcuts for accelerating network traversals; and 2) object abstracts for guiding traversals. It also exploits search space pruning, a powerful technique for efficient object search. Upon the framework, efficient search algorithms for single source and multisource LDSQs are devised. Via a comprehensive performance evaluation on real road networks, The LDSO are (i) the design and implementation of the Single and multisource Indexing technique, (ii) efficient object search algorithms, and we conducted extensive experiments with real spatial object search on road networks. The experiment result shows the superiority of road over the state-of-the-art approaches.

Keywords—LDSQ, LDSO, SQL

Introduction
The purpose of spatial databases is to correlate data in space and they provide answers to questions such as how far has waste product extended from the spill location? How many miles away is the closest hospital of his house? Most spatial databases do not stand on their own, but instead are just an extension to relational databases. They use a dialect of SQL called Spatial Feature Structured Query Language—which simply adds spatial functions to SQL such as distance, touches, centroid, inside, area, and extent. Table 1.1 shows the usability of Spatial Database Management Systems (SDBMS).

Table 1.1: Usability of SDBMS

<table>
<thead>
<tr>
<th>User</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile phone user</td>
<td>Where is the nearest gas station?</td>
</tr>
<tr>
<td>Army field commander</td>
<td>Has there been any significant enemy troop movement</td>
</tr>
<tr>
<td>Medical doctor</td>
<td>Has there been any significant enemy troop movement since last night?</td>
</tr>
</tbody>
</table>

Farmer How can I minimize the use of pesticide on my farm?
Emergency service Where is the person calling for help located?
Transport specialist How should the road network be expanded to minimize traffic congestion?

At the time of writing, there are many query languages for spatial database management systems found in literature. They can be classified into two categories extended SQL style query languages and visual spatial query languages. Extended SQL style query languages are SQL/OGIS, PSQL, QL/G, Spatial SQL, Geo SQL, CSQL, GEOQL, and SQL/SDA and so on. Visual spatial query languages are Spatial-Query-by-Sketch, Query-by-Visual Authors in argues that the extended SQL approach is the more natural interface language to query a spatial database since the SQL query language is now widely accepted to query a relational database. Therefore, in this chapter, we will discuss four different extended SQL style spatial query languages—SQL/OGIS, QL/G, SQL/SDA, and PSQL, and compare them with ParaSQL.

**SQL/OGIS**

1.1.2.1 SQL/OGIS Data Model

The OGIS consortium was formed by major software vendors to formulate an industry wide standard related to GIS interoperability. The OGIS spatial data model can be embedded in SQL. The OGIS Consortium has standardized spatial feature geometry and spatial operations. The OGIS specification defines a standard for SQL which supports the storage and query of spatial data. The spatial data is based on the OGIS Geometry Object Model. The non-instantiable class Geometry serves as the base class with subclasses for Point, Curve (line) and Surface (Polygon). Conceptually, spatial entities are stored as tables with geometry valued columns. Instances of the entities are stored as rows in a table.

Data types of spatial attributes are drawn from the Geometry Model while those of non-spatial attributes are from SQL. Implementation of a spatially-enabled table called the feature table, are described for two target environments: SQL3 and SQL3 with Geometry Types. In the SQL3 environment, a geometry-valued column is implementing as a Foreign Key reference into a geometry table. A geometric value is stored using one or more rows in the geometry table. The geometry table may be implemented using either standard SQL numeric types or SQL binary types. In SQL3 with Geometry Types, a
A geometry-valued column is implemented as a column whose SQL type is drawn from the set of Geometry Types.

**Proposed System**

The search space pruning technique is expected to be very effective in road networks because spatial objects are often clustered and concentrated in some areas, e.g., hotels and resorts are likely to be in business and scenic areas, respectively. Thus, many subspaces do not contain objects of interest and can be pruned. Though well received in various database searches, to the best of our knowledge, the idea of search space pruning has not been exploited in the context of object search on road networks. In this system following seven fold of the significant contribution as

1. A novel system framework to support spatial object searches on road networks. Its cleanly separates the road network and objects, exploits the idea of search space pruning, and supports searches with different distance metrics.
2. Formulate region hierarchy and explore several properties to reduce indexing overhead and improve query and update performance.
3. Efficient search algorithms for single source range queries and (k) NN queries, i.e., classical types of LDSQs, upon the ROAD Networks.
4. Efficient search algorithms for multisource range queries and (k) NN queries to illustrate the extensibility of ROAD for different LDSQs.
5. Developed efficient update techniques for ROAD maintenance to handle object and layer.
6. Providing a theoretical analysis on the space and time efficiency of object search.
7. To evaluate the performance of object search and compare it with the state-of-the-art approaches.

**2.4 REQUIREMENT SPECIFICATION**

The purpose of the Software Requirement Specification is to produce the specification of the analysis task and also to establish complete information about the requirement, behavior and other constraints such as functional performance and so on. The goal of Software Requirement Specification is to completely specify the technical requirements for the software product in a concise and unambiguous manner.

**2.4.1 Hardware Requirements**

Processor: Pentium IV  
RAM: 1 GB  
Hard disk: 160 GB

**2.4.2 Software Requirements**

Operating System: Windows XP  
Framework: ASP.NET 4.0  
Frontend: MS Visual studio 2010  
Backend: PostGreSQL 8.0  
GIS: OpenGIS

**MODULE DESCRIPTION**

Extracting the specific map from the world map with the help of shape file and using Open GIS (Geographical Information system) to select the specific map details. The Open GIS plays vital role of the spatial databases and providing different module as follows.

1. Authentication Module  
2. Layers Module  
3. Map Stream Module  
4. Distance Index Module  
5. Search Module

**SOFTWARE FEATURES**

**3.6.1 ASP.NET 4.0**

**3.6.1. Features:**

ASP.NET 4.0 is a web development technology based on the Microsoft.NET framework. The .NET framework, in turn is based on the CLR. Importing all the CLR benefits to ASP .Net applications. Some of these benefits are automatic memory management, cross-language integration, interoperability with existing code and system and simplified deploy

**OpenGIS**

Interfaces for GeoSpatial concepts, often defined by the OGC or ISO standards bodies. The interfaces in this module serve as a great reference if you do not have the time to purchase and read the official standards documents. Approach the standards using an environment you are comfortable with - Java!

**PostgreSQL**

An object-relational database management system (ORDBMS) based on POSTGRES, Version 4.2, developed at the University of California at Berkeley Computer Science Department. POSTGRES pioneered many concepts that only became available in some commercial database systems much later. PostgreSQL is an open-source descendant of this original Berkeley code. It supports a large part of the SQL standard and offers many modern features: complex queries, foreign keys, triggers, views etc

**Search Algorithm**

There are two type of algorithm are used in this search module that is SingleSourcekNNSearch and Multi Source kNN Search (as outlined in exploits all the above discussed techniques for multisource kNN query. The algorithm maintains a priority queue P to sort pending entries
in a nondecreasing distance order from respective query nodes. In this module SingleSource ranges search and MultiSource ranges search also to be done. The following algorithm for kNNSearch. With the logic of network expansion as the basis, our Algorithm kNNSearch incorporates shortcuts in Route Overlay and object abstracts in Association Directory to speed up the search. In general, it iteratively expands the search in a network from nq by visiting the closest unexplored node. This gradual expansion guarantees the first k objects satisfying search condition to be the kNN objects to the query point. We maintain a priority queue P to sort pending entries in the non-decreasing distance order from nq.

RESULTS AND PERFORMANCE ANALYSIS

4.1 PERFORMANCE ANALYSIS

This section evaluates our proposed spatial object searches on framework in terms of both indexing overhead and query performance. We applied Spatial object on four real road networks, namely, CA, NA, SF, and PRS.5 CA and NA consist of highways in California and North America, respectively. SF and PRS correspond to streets and roads in San Francisco and Paris, respectively. We generated 100 to 100,000 objects following either uniform distribution or clustered distribution over these road networks. To simulate clustered distribution, we select a set of nodes as cluster centroids and distribute equal numbers of objects within 10 nodes around them. Four performance metrics are measured in this evaluation, 1. index construction time: the time to construct an index; 2. index size: storage used to store an index; 3. query processing time: time duration from the time a query is initiated to the time a complete result is obtained; 4. index update time: the time spent in maintaining the underlying indices when an update (either object update or network update) is processed.

4.1.1 Index Construction

The first experiment set evaluates the index construction time and index sizes for all the approaches with various numbers of objects and networks. Here, we use the default p and l for ROAD and leave the evaluation of their impacts in the final set of experiments. Fig. 4.1 shows the index construction times and index sizes for different networks with the number of objects fixed at 10,000 and the number of clusters fixed at 100, i.e., 100 objects per cluster. As shown in the figure, NetExp and Euclidean incur the shortest index construction times and smallest storage overhead. However, they both are not query efficient as will be discussed next. On the other hand, DistIdx, DistBrws and ROAD incur different index construction time and index size as networks change but ROAD always outperforms the other two. For example, when the largest network PRS is evaluated, DistIdx takes over 1,000 hours to build the index and 10 GB to store it; while DistBrws takes over a month to build the index and more than 15 GB to store it. Differently, ROAD incurs significantly shorter construction time (less than 1 hour) and consumes less storage (about 18 MB). Compared with DistIdx, ROAD only requires around 0.6 percent of its index construction time and 0.03 percent of its index size.

4.1.2 Query Performance

The second set of experiments evaluate the search performance of ROAD and other approaches in answering singlesource LDSQs and multisource LDSQs on the following factors:

- 1. networks,
- 2. numbers of objects,
- 3. object distributions,
- 4. query parameters, and
- 5. the number of sources for multiple-source LDSQs.

In the experiments, we generate 100 random queries and report the average query processing time. Experiments on Single-Source kNN Query First, we conduct evaluations for single-source kNN queries. As depicted in Fig. 4.2a, euclidean performs the worst because of exhaustive shortest path searches for a possibly large number of candidate objects, consistent with the observations made in Further, both DistIdx and DistBrws perform worse than NetExp and ROAD due to the excessive accesses to distance signatures and shortest path quad-trees and slow node-by-node network traversals. As expected, ROAD consistently performs the best. For clustered objects, ROAD can effectively bypass those Rnets with no object of interest.
first four experiments, we fix m at two (i.e., two-source kNN queries), as shown in Fig. 4.2a, 2b, 2c, and 2d, and the last experiment studies the impact of m with k set to one (i.e., multisource NN queries). As observed from the results, ROAD consistently performs better than NetExp and euclidean. This is because NetExp has to explore all the subnet works (i.e., edges and nodes) around query points; while euclidean has to invoke multiple network traversals to determine the network distances of candidate objects. Differently, ROAD can effectively prune away some search spaces that have no result objects.

![Fig 4.3: Multisource kNN query performance. (a) Networks. (b) No. of objects (NA). (c) Distributions (NA). (d) k (NA)](image)

CONCLUSION

The on-going trend of web-based LBSs demands a system that can be extended to accommodate diverse objects, provide efficient processing of various location dependent spatial objects, and support different distance metrics. In response to these needs, we propose a new system Extensible Location dependent spatial object search on road networks for LDSO processing, in this paper. The design of Location dependent achieves a clear separation between objects and network for better system extensibility. It also exploits search space pruning, a powerful technique for efficient object search. Upon the framework, efficient search algorithms for single source and multisource LDSQs are devised. Via a comprehensive performance evaluation on real road networks, the new system LDSO is shown to significantly outperform the state-of-the-art techniques.

REFERENCES:


