Query Processing in Spatial Databases

Antonio Corral  
University of Almeria, Spain

Michael Vassilakopoulos  
TEI of Thessaloniki, Greece

INTRODUCTION

Spatial data management has been an active area of intensive research for more than two decades. In order to support spatial objects in a database system, several important issues must be taken into account such as spatial data models, indexing mechanisms, and efficient query processing. A spatial database system (SDBS) is a database system that offers spatial data types in its data model and query language and supports spatial data types in its implementation, providing at least spatial indexing and efficient spatial query processing (Güting, 1994).

The main reason that has caused the active study of spatial database management systems (SDBMS) comes from the needs of the existing applications, such as geographical information systems (GIS), computer-aided design (CAD), very large-scale integration design (VLSI), multimedia information systems (MIS), data warehousing, and so forth.

Some of the most important companies in the commercial database industry (Oracle, Informix, Autodesk, etc.) have products specifically designed to manage spatial data. Moreover, research prototypes as Postgres and Paradise offer the possibility to handle spatial data. The main functionality provided by these products includes a set of spatial data types such as the point, line, polygon, and region, and a set of spatial operations, including intersection, enclosure, and distance. The performance enhancement provided by these operations includes spatial access methods and query algorithms over such indexes (e.g., spatial range queries, spatial joins, etc.). We must also cite the Open Geographic Information Systems (OGIS) consortium (http://www.opengis.org/), which has developed a standard set of spatial data types and operators and SQL3/SQ199, which is an object-relational query language that provides the use of spatial types and operations.

In a spatial database system, the queries are usually expressed in a high-level declarative language such as SQL; therefore, specialized database software has to map the query in a sequence of spatial operations supported by spatial access methods (Shekhar & Chawla, 2003).

Spatial query processing refers to the sequence of steps that a SDBMS will initiate to execute a given spatial query. The main target of query processing in the database field is to process the query accurately and quickly (-consuming the minimum amount of time and resources on the computer), by using both efficient representations and efficient search algorithms (Graefe, 1993). Query processing in a spatial environment focuses on the design of efficient algorithms for spatial operators (e.g., selection operations, spatial joins, distance-based queries, etc.). These algorithms are both CPU and I/O intensive, despite common assumptions of traditional databases that the I/O cost will dominate CPU cost (except expensive distance-based queries), and therefore an efficient algorithm is one that minimizes the number of disk accesses.

BACKGROUND IN SPATIAL QUERIES AND PROCESSING

From the query processing point of view, the following three properties characterize the differences between spatial and relational databases (Brinkhoff, Kriegel & Seeger, 1993): (1) unlike relational databases (Elmasri & Navathe, 2000), spatial databases do not have a fixed set of operators that serve as building blocks for query evaluation; (2) spatial databases deal with extremely large volumes of complex objects, which have spatial extensions and cannot be sorted in a one-dimensional array; (3) computationally expensive algorithms are required to test the spatial operators, and the assumption that I/O costs dominate CPU costs is no longer valid.

We generally assume that the given spatial objects are embedded in d-dimensional Euclidean space ($E^d$). An object $obj$ in a spatial database is usually defined by several non-spatial attributes and one attribute of some spatial data type (point, line, polygon, region, etc.). This spatial attribute describes the geometry of the object $obj.G \subseteq E^d$, that is, the location, shape, orientation, and size of the object. The most representative spatial operations, which are the basis for the query processing in spatial databases, are (1) update operations; (2) selection operations (point and range queries); (3) spatial join; and
(4) spatial aggregate queries (Gaede & Günther, 1998; Shekhar & Chawla, 2003).

- **Update Operations**: Standard database operations such as modify, create, and so forth.
- **Point Query (PQ)**: Given a query point \( p \in E^d \), find all spatial objects \( O \) that contain it.
- **Range Query (RQ)**: Given a spatial polygon \( P \), find all spatial objects \( O \) that intersect \( P \). When the query polygon is a rectangle, this is called a window query.
- **Spatial Join Query (SJQ)**: Given two collections \( R \) and \( S \) of spatial objects and a spatial predicate \( \theta \), find all pairs of objects \( (O, O') \in R \times S \) such that \( \theta(O, O') \) evaluates to true. Some examples of the spatial predicate \( \theta \) are intersects, contains, is_enclosed_by, distance, northwest, adjacent, meets, and so on. For spatial predicates such as contains, encloses, or adjacent, for example, the intersection join is an efficient filter that yields a set of candidate solutions typically much smaller than the Cartesian product \( R \times S \). An extension of the intersection join is the multiway spatial join, which involves an arbitrary number of spatial inputs (Mamoulis & Papadias, 2001). Very interesting distance join queries are actually being studied, for example, closest pairs query (Corral et al., 2000), buffer query (Chan, 2003), nearest neighbors join (Böhm & Krebs, 2002), iceberg queries (Shou et al., 2003), distance join queries of multiple inputs (Corral et al., 2003), and so on.
- **Spatial Aggregate Queries (SAQ)**: This kind of spatial query involves specifying a region of space and asking for the value of some aggregate function (e.g., count, sum, min, max, average) for which we have measurements for this given region (Papadias et al., 2001). Spatial aggregates are usually variants of the nearest neighbor problem (Shekhar & Chawla, 2003). The Nearest Neighbor Query (NNQ) has the form: given a spatial object \( O' \), find all spatial objects \( O \) having a minimum distance from \( O' \). The distance between extended spatial data objects is usually defined as the distance between their closest points (common distance functions for points include the Euclidean and the Manhattan distance). An interesting variant of NNQ is the reverse nearest neighbor query (RNNQ), which reports the points that have the query point as their nearest neighbor (Korn & Muthukrishnan, 2000).

The spatial queries are often processed using filter and refine techniques to minimize both the CPU and I/O cost (Brinkhoff et al., 1994). Approximate geometry such as the minimal orthogonal bounding rectangle (MBR) of an extended spatial object is first used to filter out many irrelevant objects quickly. An MBR is characterized by min and max points of hyper-rectangles with faces parallel to the coordinate axes. Using the MBR instead of the exact geometrical representation of the spatial object, its representational complexity is reduced to two points, where the most important object features (position and extension) are maintained. The R-tree (Guttman, 1984) is a spatial access method that represents the spatial objects by their MBR, and it is a height-balanced tree. Therefore, in the filter step, many candidates are eliminated using the spatial predicate and the MBRs of the spatial objects. In the refinement step, the exact geometry of each spatial object from the candidate set (result of the filter step) and the exact spatial predicate are examined. This step usually requires the use of CPU-intensive algorithms like computational geometry algorithms for spatial operations (Right, Scholl & Voisard, 2001). Strategies for range-queries include a scan and index-search in conjunction with the plane-sweep algorithm (Brinkhoff et al., 1993). Strategies for the spatial join include the nested loop, tree matching (Brinkhoff et al., 1993; Huang, Jing & Rundensteiner, 1997), when indices are present on all participating inputs and space partitioning (Arge et al., 1998; Lo & Ravishankar, 1996; Patel & De Witte, 1996) in absence of indexes. For the case when one spatial input is indexed, the most representative join strategies have been proposed by Lo & Ravishankar (1994) and Mamoulis & Papadias (2003).

Nearest neighbor queries (NNQ) are common in many applications, for example, GIS, pattern recognition, document retrieval, and learning theory. As the previous spatial queries, NNQ algorithms are also two-step algorithms (filter-refine paradigm), and they follow branch-and-bound techniques, using distance functions and pruning heuristics in order to reduce the search space. The most representative algorithms to perform NNQ over spatial data have been proposed by Roussopoulos, Kelley, and Vincent (1995) and Hjaltason and Samet (1999) on R-trees. The first query algorithm follows a depth-first traversal, whereas the second one is an incremental algorithm following a best-first search on the R-tree. These algorithms can be extended to find K-nearest neighbors by slight modification of the pruning rules to retain the K best candidates.

**PERSPECTIVE AND NEW IMPORTANT ISSUES**

We have reviewed the most representative spatial queries, using mainly the overlap predicate for range queries and spatial join queries. However, there is a need to develop and evaluate query strategies for many other frequent spatial queries that can be demanded by the
Related Content

**INDUSTRY AND PRACTICE: New Direction for Database Management Research**
[www.igi-global.com/article/industry-practice-new-direction-database/51179?camid=4v1a](www.igi-global.com/article/industry-practice-new-direction-database/51179?camid=4v1a)

**Toward a Unified Model of Information Systems Development Success**
Keng Siau, Yoanna Long and Min Ling (2012). *Cross-Disciplinary Models and Applications of Database Management: Advancing Approaches* (pp. 80-102).
[www.igi-global.com/chapter/toward-unified-model-information-systems/63663?camid=4v1a](www.igi-global.com/chapter/toward-unified-model-information-systems/63663?camid=4v1a)

**Integrating Hypermedia Functionality into Database Applications**
[www.igi-global.com/chapter/integrating-hypermedia-functionality-into-database/8272?camid=4v1a](www.igi-global.com/chapter/integrating-hypermedia-functionality-into-database/8272?camid=4v1a)

**Transaction-Relationship Oriented Log Division for Data Recovery from Information Attacks**
[www.igi-global.com/article/transaction-relationship-oriented-log-division/3293?camid=4v1a](www.igi-global.com/article/transaction-relationship-oriented-log-division/3293?camid=4v1a)