

## SPATIAL DATABASES

### INTRODUCTION

Spatial database management systems (1–6) aim at the effective and efficient management of data related to

- space in the physical world (geography, urban planning, astronomy, human anatomy, fluid flow, or an electromagnetic field),
- biometrics (fingerprints, palm measurements, and facial patterns),
- engineering design (very large-scale integrated circuits, layout of a building, or the molecular structure of a pharmaceutical drug), and
- conceptual information space (virtual reality environments and multidimensional decision-support systems).

A spatial database management system (SDBMS) can be characterized as follows:

- A SDBMS is a software module that can work with an underlying database management system, for example, an object-relational database management system or object-oriented database management system.
- SDBMSs support multiple spatial data models, commensurate spatial abstract data types (ADTs), and a query language from which these ADTs are callable.
- SDBMSs support spatial indexing, efficient algorithms for spatial operations, and domain-specific rules for query optimization.

Spatial database research has been an active area for several decades. The results of this research are being used several areas. To cite a few examples, the filter-and-refine technique used in spatial query processing has been applied to subsequence mining; multidimensional-index structures such as R-tree and Quad-tree used in accessing spatial data are applied in the field of computer graphics and image processing; and space-filling curves used in spatial query processing and data storage are applied in dimension-reduction problems. The field of spatial databases can be defined by its accomplishments; current research is aimed at improving its functionality, extensibility, and performance. The impetus for improving functionality comes from the needs of existing applications such as geographic information systems (GIS), location-based services (LBS) (7), sensor networks (8), ecology and environmental management (9), public safety, transportation (10), earth science, epidemiology (11), crime analysis (12), and climatology.

Commercial examples of spatial database management include ESRI's ArcGIS Geodatabase (13), Oracle Spatial

(14), IBM's DB2 Spatial Extender and Spatial Datablade, and future systems such as Microsoft's SQL Server 2008 (code-named Katmai) (15). Spatial databases have played a major role in the commercial industry such as Google Earth (16) and Microsoft's Virtual Earth (17). Research prototype examples of spatial database management systems include spatial datablades with PostGIS (18), MySQL's Spatial Extensions (19), Sky Server (20), and spatial extensions. The functionalities provided by these systems include a set of spatial data types such as a points, line segments and polygons, and a set of spatial operations such as inside, intersection, and distance. The spatial types and operations may be made a part of a query language such as SQL, which allows spatial querying when combined with an object-relational database management system (21,22). The performance enhancement provided by these systems includes a multidimensional spatial index and algorithms for spatial database modeling such as OGC (23) and 3-D topological modeling; spatial query processing including point, regional, range, and nearest-neighbor queries; and spatial data methods that use a variety of indexes such as quad trees and grid cells.

### Related Work and Our Contributions

Published work related to spatial databases can be classified broadly as follows:

- Textbooks (3,4,6,24), which explain in detail various topics in spatial databases such as logical data models for spatial data, algorithms for spatial operations, and spatial data access methods. Recent textbooks (6,25) deal with research trends in spatial databases such as spatio-temporal databases and moving objects databases.
- Reference books (26,27), which are useful for studying areas related to spatial databases, for example, multidimensional data structures and geographic information systems (GIS).
- Journals and conference proceedings (28–37), which are a source of in-depth technical knowledge of specific problem areas in spatial databases.
- Research surveys (1,38,39), which summarize key accomplishments and identify research needs in various areas of spatial databases at that time.

Spatial database research has continued to advance greatly since the last survey papers in this area were published (1,38,39). Our contribution in this chapter is to summarize the most recent accomplishments in spatial database research, a number of which were identified as research needs in earlier surveys. For instance, bulk loading techniques and spatial join strategies are rereferenced here as well as other advances in spatial data mining and conceptual modeling of spatial data. In addition, this chapter provides an extensive updated list of research needs in

such areas as management of 3-D spatial data, visibility queries, and many others. The bibliography section at the end of this chapter contains a list of over 100 references, updated with the latest achievements in spatial databases.

**Scope and Outline**

The goal of this chapter is to provide the reader with a broad introduction to spatial database systems. Spatial databases are discussed in the context of object-relational databases (21,22,40), which provide extensibility to many components of traditional databases to support the spatial domain. Three major areas that receive attention in the database context—conceptual, logical, and physical data models—are discussed (see Table 1). In addition, applications of spatial data for spatial data mining are also explored.

Emerging needs for spatial database systems include the handling of 3-D spatial data, spatial data with temporal dimension, and effective visualization of spatial data. The emergence of hardware technology such as storage area networks and the availability of multicore processors are two additional fields likely to have an impact on spatial databases. Such topics of research interest are introduced at the end of each section. References are provided for more exploration. Because of the size constraints of this chapter, several other overlapping research needs such as spatio-temporal databases and uncertainty are not included in this chapter.

The rest of this chapter is organized as follows: Fundamental concepts helpful to understand spatial databases are presented. Spatial database modeling is described at the conceptual and logical levels; techniques for spatial query processing are discussed; file organizations and index data structures are presented; and spatial data mining patterns and techniques are explored.

**MATHEMATICAL FRAMEWORK**

**Accomplishments**

Spatial data are relatively more complex compared with traditional business data. Specific features of spatial data include: (1) rich data types (e.g., extended spatial objects), (2) implicit spatial relationships among the variables, (3) observations that are not independent, and (4) spatial autocorrelation among the features.

Spatial data can be considered to have two types of attributes: nonspatial attributes and spatial attributes. nonspatial attributes are used to characterize nonspatial features of objects, such as name, population, and unemployment rate for a city. Spatial attributes are used to define the spatial location and extent of spatial objects

(41). The spatial attributes of a spatial object most often include information related to spatial locations, for example, longitude, latitude, elevation, and shape. Relationships among nonspatial objects are explicit in data inputs, e.g., arithmetic relation, ordering, instance of, subclass of, and membership of. In contrast, relationships among spatial objects are often implicit, such as overlap, intersect, and behind.

Space is a framework to formalize specific relationships among a set of objects. Depending on the relationships of interest, different models of space such as set-based space, topological space, Euclidean space, metric space, and network space can be used (6). Set-based space uses the basic notion of elements, element-equality, sets, and membership to formalize the set relationships such as set-equality, subset, union, cardinality, relation, function, and convexity. Relational and object-relational databases use this model of space.

Topological space uses the basic notion of a neighborhood and points to formalize the extended object relations such as boundary, interior, open, closed, within, connected, and overlaps, which are invariant under elastic deformation. Combinatorial topological space formalizes relationships such as Euler’s formula (number of faces + number of vertices – number of edges = 2 for planar configuration). Network space is a form of topological space in which the connectivity property among nodes formalizes graph properties such as connectivity, isomorphism, shortest-path, and planarity.

Euclidean coordinatized space uses the notion of a coordinate system to transform spatial properties and relationships to properties of tuples of real numbers. Metric spaces formalize the distance relationships using positive symmetric functions that obey the triangle inequality. Many multidimensional applications use Euclidean coordinatized space with metrics such as distance.

**Research Needs**

Many spatial applications manipulate continuous spaces of different scales and with different levels of discretization. A sequence of operations on discretized data can lead to growing errors similar to the ones introduced by finite-precision arithmetic on numbers. Preliminary results (1) are available on the use of discrete basis and bounding errors with peg-board semantics. Another related problem concerns interpolation to estimate the continuous field from a discretization. Negative spatial autocorrelation makes interpolation error-prone. More work is needed on a framework to formalize the discretization process and its associated errors, and on interpolation.

**Table 1. Spatial database topics**

Mathematical Framework	
Conceptual Data Model	
Logical Data Model	Trends: Spatial Data Mining
Query Languages	
Query Processing	
File Organizations and Indices	

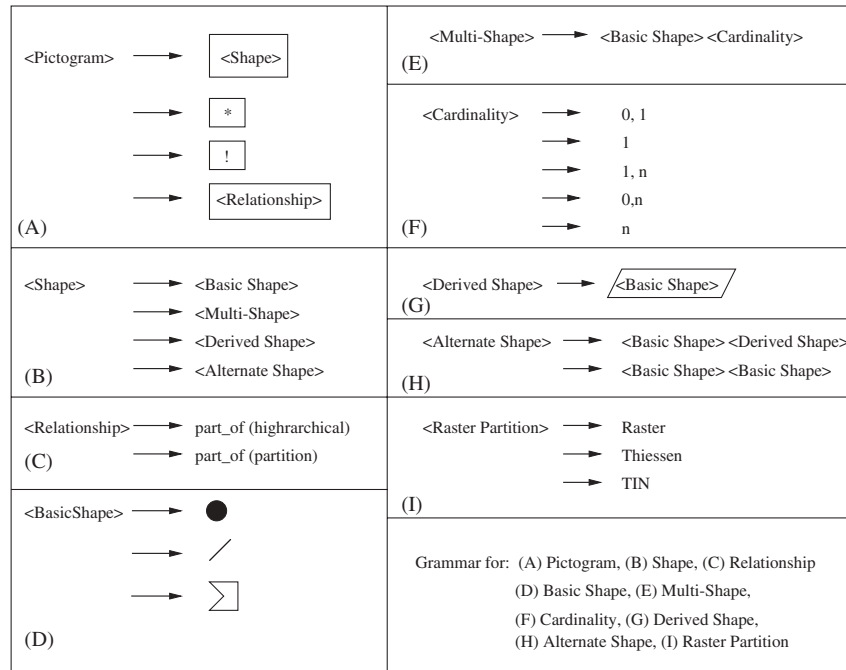


Figure 1. Pictograms.

## SPATIAL-DATABASE CONCEPTUAL MODELING

### Accomplishments

Entity relationship (ER) diagrams are commonly used in designing the conceptual model of a database. Many extensions (42) have been proposed to extend ER to make the conceptual modeling of spatial applications easier and more intuitive. One such extension is the use of pictograms (43). A pictogram is a graphical icon that can represent a spatial entity or a spatial relationship between spatial entities. The idea is to provide constructs to capture the semantics of spatial applications and at the same time to keep the graphical representation simple. Figure 2 provides different types of pictograms for spatial entities and relationships. In the following text we define pictograms to represent spatial entities and relationships and their grammar in graphical form.

*Pictogram:* A pictogram is a representation of the object inserted inside of a box. These iconic representations are used to extend ER diagrams and are inserted at appropriate places inside the entity boxes. An entity pictogram can be of a basic shape or a user-defined shape.

*Shape:* Shape is the basic graphical element of a pictogram that represents the geometric types in the spatial data model. It can be a basic shape, a multi-shape, a derived shape, or an alternate shape. Most objects have simple (basic) shapes [Fig. 1 (b)].

*Basic Shape:* In a vector model, the basic elements are point, line, and polygon. In a forestry example, the

user may want to represent a facility as a point (0-D), a river or road network as lines (1-D), and forest areas as polygons (2-D) [Fig. 1(d)].

*Multishape:* To deal with objects that cannot be represented by the basic shapes, we can use a set of aggregate shapes. Cardinality is used to quantify multishapes. For example, a river network that is represented as a line pictogram scale will have cardinality 0 [Fig. 1(b) and (e)].

*Derived shape:* If the shape of an object is derived from the shapes of other objects, its pictogram is italicized. For example, we can derive a forest boundary (polygon) from its “forest-type” boundaries (polygon), or a country boundary from the constituent-state boundaries [Fig. 1(c) and (g)].

*Alternate shape:* Alternate shapes can be used for the same object depending on certain conditions; for example, objects of size less than  $x$  units are represented as points, whereas those greater than  $x$  units are represented as polygons. Alternate shapes are represented as a concatenation of possible pictograms. Similarly, multiple shapes are needed to represent objects at different scales; for example, at higher scales lakes may be represented as points and at lower scales as polygons [Fig. 1(d) and (h)].

*Any possible shape:* A combination of shapes is represented by a wild card \* symbol inside a box, which implies that any geometry is possible (Fig. 1(e)).

*User-defined shape:* Apart from the basic shapes of point, line, and polygon, user-defined shapes are possible. User-defined shapes are represented by an exclamation symbol (!) inside a box [Fig. 1(a)].

*Relationship pictograms:* Relationship pictograms are used to model the relationship between entities. For example, *part\_of* is used to model the relationship between a route and a network, or it can be used to model the partition of a forest into forest stands [Fig. 1(c)].

The popularity of object-oriented languages such as C++ and Java has encouraged the growth of object-oriented database systems (OODBMS). The motivation behind this growth in OODBMS is that the direct mapping of the conceptual database schema into an object-oriented language leads to a reduction of impedance mismatch encountered when a model on one level is converted into a model on another level.

UML is one of the standards for conceptual-level modeling for object-oriented software design. It may also be applied to an OODBMS to capture the design of the system conceptually. A UML design consists of the following building blocks:

*Class:* A class is the encapsulation of all objects that share common properties in the context of the application. It is the equivalent of the entity in the ER model. The class diagrams in a UML design can be further extended by adding pictograms. In a forestry example, classes can be forest, facility, forest stand, and so forth.

*Attributes:* Attributes characterize the objects of the class. The difference between an attribute ER and a UML model design is that no notion of a key attribute exists in UML, in an object-oriented system, each object has an implicit system-generated unique identification. In UML, attributes also have a scope that restricts the attribute’s access by other classes. Three levels of scope exist, and each has a special symbol: + Public: This symbol allows the attribute to be accessed and manipulated from any class.

- Private: Only the class that owns the attribute is allowed to access the attribute. # Protected: Other than the class that owns the attribute, classes derived from the class that owns can access the attribute.

*Methods:* Methods are functions and a part of class definition. They are responsible for modifying the behavior or state of the class. The state of the class is embodied in the current values of the attributes. In object-oriented design, attributes should only be accessed through methods.

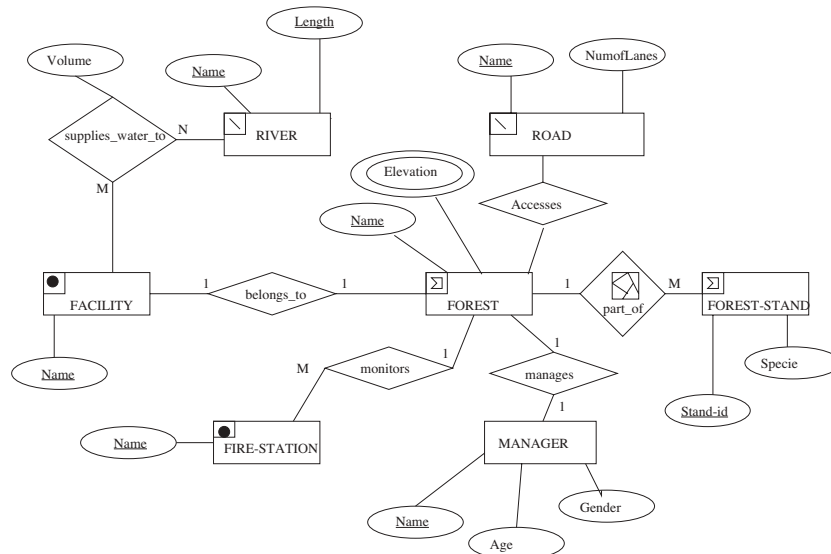
*Relationships:* Relationships relate one class to another or to itself. This concept is similar to the concept of relationship in the ER model. Three important categories of relationships are as follows:

- *Aggregation:* This category is a specific construct to capture the part–whole relationship. For instance, a group of forest–stand classes may be aggregated into a forest class.
- *Generalization:* This category describes a relationship in which a child class can be generalized to a parent class. For example, classes such as point, line, and polygon can be generalized to a geometry class.
- *Association:* This category shows how objects of different classes are related. An association is binary if it connects two classes or ternary if it connects three classes. An example of a binary association is *supplieswater to* between the classes river and facility.

Figures 2 and 3 provide an example for modeling a *State-Park* using ER and UML with pictograms, respectively.

**Research Needs**

Conceptual modeling for spatio–temporal and moving-object data needs to be researched. Pictograms as introduced in this section may be extended to handle such data.



**Figure 2.** Example ER diagram with pictograms.

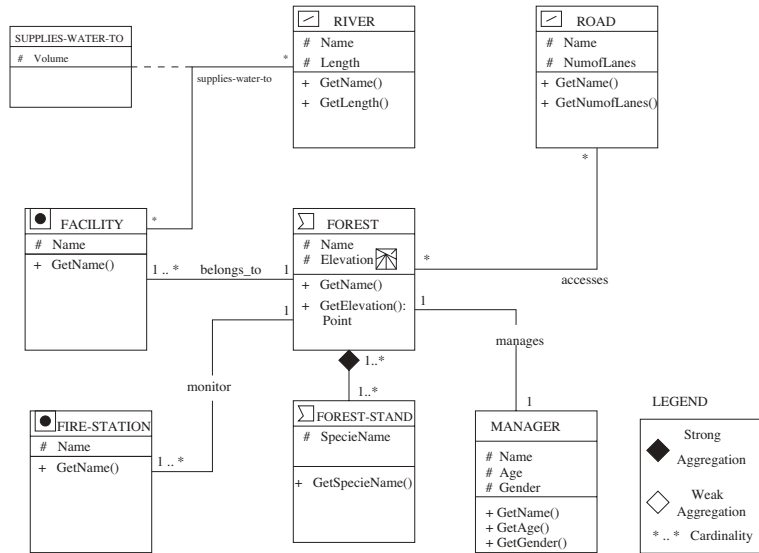


Figure 3. Example UML class diagram with pictograms.

Models used in the spatial representation of data can be extended to consider the time dimension. For instance, the nine-intersection matrix used to represent topology can be differentiated to consider the change in topology over a period of time. Similarly, other spatial properties such as position, orientation, and shape can be differentiated to consider effects over time such as motion, rotation, and deformation of a spatial object. Similarly, series of points can be accumulated to represent time-varying spatial data and properties.

Another area of research is the use of ontology for knowledge management. An ontology defines a common vocabulary that allows knowledge to be shared and reused across different applications. Ontologies provide a shared and common understanding of some domain that can be communicated across people and computers.

*Geospatial ontology* (44) is specific to the geospatial domain. Research in geospatial ontology is needed to provide interoperability between geospatial data and software. Developing geospatial ontologies is one of the long-term research challenges for the University Consortium for Geographic Information Systems (UCGIS) (37). Research in this area is also being carried out by companies such as CYC for geospatial ontology.

Geospatial ontology can be extended to include the temporal dimension. The ontology of time has been researched in the domain of artificial intelligence as situation calculus. OWL-Time (45) is an ontology developed to represent time.

Semantic web (46) is widely known as an efficient way to represent data on the Web. The wealth of geographic information currently available on the Web has prompted research in the area of GeoSpatial Semantic Web (47,48). In this context, it becomes necessary to create representations of the geographic information resources. This necessity must lead to a framework for information retrieval based on the semantics of spatial ontologies. Developing the geo-

spatial ontology that is required in a geo-spatial semantic web is challenging because the defining properties of geographic entities are very closely related to space (i.e., multi-dimensional space). In addition, each entity may have several subentities resulting in a complex object (48). One popular data model used in representing semantic web is the resource description framework (RDF) (49).

RDF is being extended (GeoRDF) (50) to include spatial dimensions and hence to provide the necessary support for geographica data on the web.

### SPATIAL DATA MODELS AND QUERY LANGUAGES

#### Accomplishments

**Data Models.** A spatial data model provides the data abstraction necessary to hide the details of data storage. The two commonly used models are the field-based model and the object-based model. Whereas the field-based model adopts a functional viewpoint, the object-based model treats the information space as a collection of discrete, identifiable, spatially referenced entities. Based on the type of data model used, the spatial operations may change. Table 2 lists the operations specific to the field-based and object-based models. In the context of object-relational databases, a spatial data model is implemented using a set of spatial data types and operations. Over the last two decades, an enormous amount of work has been done in the design and development of spatial abstract data types and their embedding in a query language. Serious efforts are being made to arrive at a consensus on standards through the OGC (51).

OGC proposed the general feature model (51) where features are considered to occur at two levels, namely, feature instances and feature types. A geographic feature is represented as a discrete phenomenon characterized by its geographic and temporal coordinates at the instance

**Table 2. Data model and operations**

Data Model	Operator Group	Operation
Vector Object	Set-Oriented	equals, is a member of, is empty, is a subset of, is disjoint from, intersection, union, difference, cardinality
	Topological	boundary, interior, closure, meets, overlaps, is inside, covers, connected, components, extremes, is within
	metric	distance, bearing/angle, length, area, perimeter
	Direction	east, north, left, above, between
	Network	successors, ancestors, connected, shortest-path
Raster Field	Dynamic	translate, rotate, scale, shear, split, merge
	Local	point-wise sums, differences, maximums, means, etc.
	Focal	slop, aspect, weighted average of neighborhood
	Zonal	sum or mean or maximum of field values in each zone

level, and the instances with common characteristics are grouped into classes called feature types. Direction is another important feature used in spatial applications. A direction feature can be modeled as a spatial object (52). Research has also been done to efficiently compute the cardinal direction relations between regions that are composed of sets of spatial objects (53).

**Query Languages.** When it comes to database systems, spatial database researchers prefer object-based models because the data types provided by object-based database systems can be extended to spatial data types by creating abstract data types (ADT). OGC provides a framework for object-based models. Figure 4 shows the OpenGIS approach to modeling geographic features. This framework provides conceptual schemas to define abstract feature types and provides facilities to develop application schemas that can capture data about feature instances. Geographic phenomena fall into two broad categories, discrete and continuous. Discrete phenomena are objects that have well-defined boundaries or spatial extent, for example, buildings and streams. Continuous phenomena vary over space and have no specific extent (e.g., temperature and elevation). A continuous phenomenon is described in terms of its value at a specific position in space (and possibly time). OGC represents discrete phenomena (also called vector data) by a set of one or more geometric primitives (points, curves, surfaces, or solids). A continuous phenomenon is represented through a set of values, each associated with one of the elements in an array of points. OGC uses the term “coverage” to refer to any data representation that assigns values directly to spatial position. A coverage is a function from a spatio-temporal domain to an attribute domain. OGC provides standardized representations for spatial characteristics through geometry and topology. Geometry

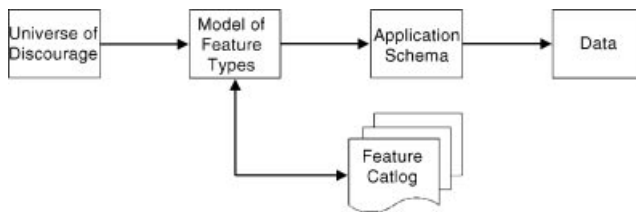
provides the means for the quantitative description of the spatial characteristics including dimension, position, size, shape, and orientation. Topology deals with the characteristics of geometric figures that remain invariant if the space is deformed elastically and continuously. Figure 5 shows the hierarchy of geometry data types. Objects under primitive (e.g., points and curves) will be open (i.e., they will not contain their boundary points), and the objects under complex (e.g., disjoint objects) will be closed.

In addition to defining the spatial data types, OGC also defines spatial operations. Table 3 lists basic operations operative on all spatial data types. The topological operations are based on the ubiquitous nine-intersection model. Using the OGC specification, common spatial queries can be posed intuitively in SQL. For example, the query *Find all lakes which have an area greater than 20 sq. km. and are within 50 km. from the campgrounds* can be posed as shown in Table 4 and Fig. 6. Other GIS and LBS example queries are provided in Table 5. The OGC specification is confined to topological and metric operations on vector data types. Also, several spatio-temporal query languages have been studied that are trigger-based for relational-oriented models (54), moving objects (55), future temporal languages (56), and constraint-based query languages (57).

For spatial networks, commonly used spatial data types include objects such as node, edge, and graph. They may be constructed as an ADT in a database system. Query languages based on relational algebra are unable to express certain important graph queries without making certain assumptions about the graphs. For example, the transitive closure of a graph may not be determined using relational algebra. In the SQL3, a recursion operation RECURSIVE has been proposed to handle the transitive closure operation.

### Research Needs

**Map Algebra.** Map Algebra (58) is a framework for raster analysis that has now evolved to become a preeminent language for dealing with field-based models. Multiple operations can be performed that take multiple data layers that are overlaid upon each other to create a new layer. Some common groups of operations include local, focal, and zonal. However, research is needed to account for the generalization of temporal or higher dimensional data sets (e.g., 3-D data).

**Figure 4.** Modeling geographic information [source: (51)].

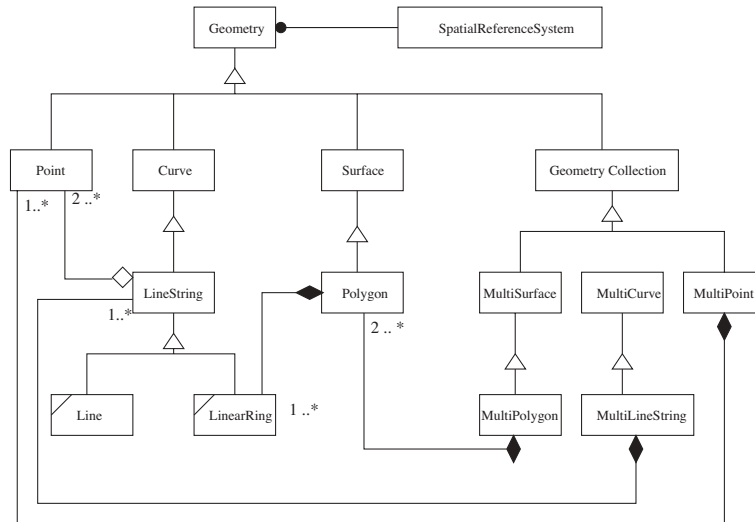


Figure 5. Hierarchy of data types.

**Modeling 3-D Data.** The representation of volumetric data is another field to be researched. Geographic attributes such as clouds, emissions, vegetation, and so forth, are best described as point fields on volumetric bounds. Sensor data from sensor technologies such as LADAR (Laser Detection and Ranging), 3-D SAR (Synthetic Arper-

ture Radar), and EM collect data volumetrically. Because volumetric data is huge, current convention is to translate the data into lower-dimensional representations such as B-reps, Point clouds, NURBS, and so on. This action results in loss of intrinsic 3-D information. Efforts (59) have been made to develop 3-D data models that emphasize the

Table 3. A sample of operations listed in the OGC standard for SQL

Basic Functions	
SpatialReference()	Returns the underlying coordinate system of the geometry
Envelope()	Returns the minimum orthogonal bounding rectangle of the geometry
Export ()	Returns the geometry in a different representation
IsEmpty()	Returns true if the geometry is an empty set.
IsSimple()	Returns true if the geometry is simple (no self-intersection)
Boundary ()	Returns the boundary of the geometry
Topological/ Set Operators	
Equal	Returns true if the interior and boundary of the two geometries are spatially equal
Disjoint	Returns true if the boundaries and interior do not intersect.
Intersect	Returns true if the interiors of the geometries intersect
Touch	Returns true if the boundaries intersect but the interiors do not.
Cross	Returns true if the interior of the geometries intersect but the boundaries do not
Within	Returns true if the interior of the given geometry does not intersect with the exterior of another geometry.
Contains	Tests if the given geometry contains another given geometry
Overlap	Returns true if the interiors of two geometries have non-empty intersection
Spatial Analysis	
Distance	Returns the shortest distance between two geometries
Buffer	Returns a geometry that consists of all points whose distance from the given geometry is less than or equal to the specified distance
ConvexHull	Returns the smallest convex set enclosing the geometry
Intersection	Returns the geometric intersection of two geometries
Union	Returns the geometric union of two geometries
Difference	Returns the portion of a geometry which does not intersect with another given geometry
SymmDiff	Returns the portions of two geometries which do not intersect with each other

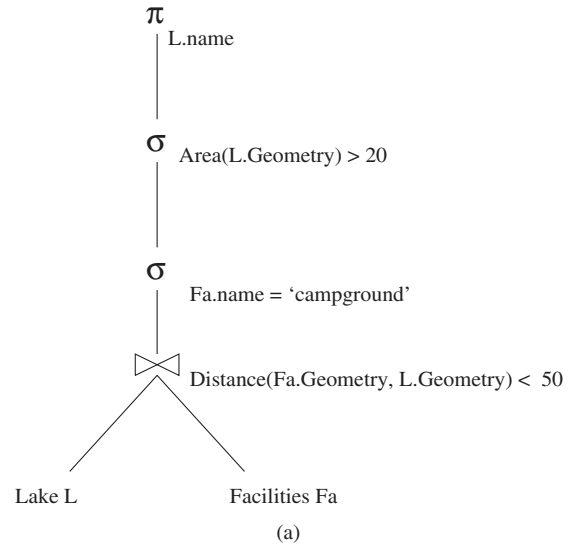
**Table 4. SQL query with spatial operators**

```
SELECT L.name
FROM Lake L, Facilities Fa
WHERE Area(L.Geometry) > 20 AND
Fa.name = 'campground' AND
Distance(Fa.Geometry, L.Geometry) < 50
```

significance of the volumetric shapes of physical world objects. This topological 3-D data model relies on Poincare algebra. The internal structure is based on a network of simplexes, and the internal data structure used is a tetrahedronized irregular network (TIN) (59,60), which is the three-dimensional variant of the well-known triangulated irregular network (TIN).

**Modeling Spatial Temporal Networks.** Graphs have been used extensively to represent spatial networks. Considering the time-dependence of the network parameters and their topology, it has become critically important to incorporate the temporal nature of these networks into their models to make them more accurate and effective. For example, in a transportation network, the travel times on road segments are often dependent on the time of the day, and there can be intervals when certain road segments are not available for service. In such, time-dependent networks modeling the time variance becomes very important. Time-expanded graphs (61) and time-aggregated graphs (62) have been used to model time-varying spatial networks. In the time-expanded representation, a copy of the entire network is maintained for every time instant, whereas the time-aggregated graphs maintain a time series of attributes, associated to every node and edge.

Network modeling can be extended to consider 3-D spatial data. Standard road network features do not represent 3-D structure and material properties. For instance, while modeling a road tunnel, we might want to represent its overpass clearance as a spatial property. Such properties will help take spatial constraints into account while selecting routes.



**Figure 6.** SQL query tree.

**Modeling Moving Objects.** A moving object database is considered to be a spatio-temporal database in which the spatial objects may change their position and extent over a period of time. To cite a few examples, the movement of taxi cabs, the path of a hurricane over a period of time, and the geographic profiling of serial criminals are a few examples in which a moving-objects database may be considered. References 25 and 63 have provided a data model to support the design of such databases.

**Markup Languages.** The goals of markup languages, such as geography markup language (GML) (64), are to provide a standard for modeling language and data exchange formats for geographic data. GML is an XML-based markup language to represent geographic entities and the relationships between them. Entities associated with geospatial data such as geometry, coordinate systems, attributes, and

**Table 5. Typical spatial queries from GIS and LBS**

GIS Queries	
Grouping	Recode all land with silty soil to silt-loadm soil
Isolate	Select all land owned by Steve Steiner
Classify	If the population density is less than 100 people / sq. mi., land is acceptable
Scale	Change all measurement's' to the metric system
Rank	If the road is an Interstate, assign it code 1; if the road is a state or US highway, assign it code 2; otherwise assign it code 3
Evaluate	If the road code is 1, then assign it Interstate; if the road code is 2, then assign it Main Artery; if the road code is 3, assign it Local Road
Rescale	Apply a function to the population density
Attribute Join	Join the Forest layer with the layer containing forest-cover codes
Zonal	Produce a new map showing state populations given county population
Registration	Align two layers to a common grid reference
Spatial Join	Overlay the land-use and vegetation layers to produce a new layer
LBS Queries	
Nearest Neighbor	List the nearest gas stations
Directions	Display directions from a source to a destination (e.g. Google Maps, Map Quest)
Local Search	Search for restaurants in the neighborhood (e.g. Microsoft Live Local, Google Local)

so forth, can be represented in a standard way using GML. Several computational challenges exist with GML, such as spatial query processing and indexing (65). CityGML (66) is a subclass of GML useful for representing 3-D urban objects, such as buildings, bridges, tunnels, and so on. CityGML allows modeling of spatial data at different levels of detail regarding both geometry and thematic differentiation. It can be used to model 2.5-D data (e.g., digital terrain model), and 3-D data (walkable architecture model). Keyhole markup language (KML) (67) is another XML-based markup language popular with commercial spatial software from Google. Based on a structure similar to GML, KML allows representation of points, polygons, 3-D objects, attributes, and so forth.

**SPATIAL QUERY PROCESSING**

**Accomplishments**

The efficient processing of spatial queries requires both efficient representation and efficient algorithms. Common representations of spatial data in an object model include spaghetti, the node-arc-node (NAA) model, the doubly connected-edge-list (DCEL), and boundary representation, some of which are shown in Fig. 7 using entity-relationship diagrams. The NAA model differentiates between the topological concepts (node, arc, and areas) and the embedding space (points, lines, and areas). The spaghetti-ring and DCEL focus on the topological concepts. The representation of the field data model includes a regular tessellation (triangular, square, and hexagonal grid) and triangular irregular networks (TIN).

Query processing in spatial databases differs from that of relational databases because of the following three major issues:

- Unlike relational databases, spatial databases have no fixed set of operators that serve as building blocks for query evaluation.
- Spatial databases deal with extremely large volumes of complex objects. These objects have spatial extensions

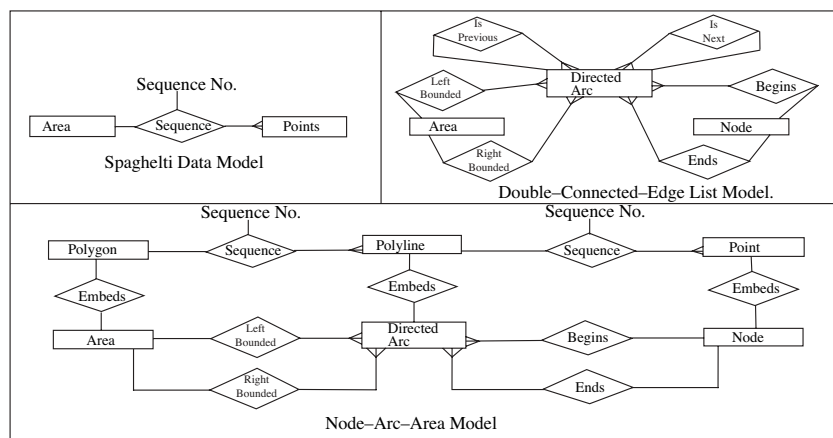
and cannot be naturally sorted in a one-dimensional array.

- Computationally expensive algorithms are required to test for spatial predicates, and the assumption that I/O costs dominate processing costs in the CPU is no longer valid.

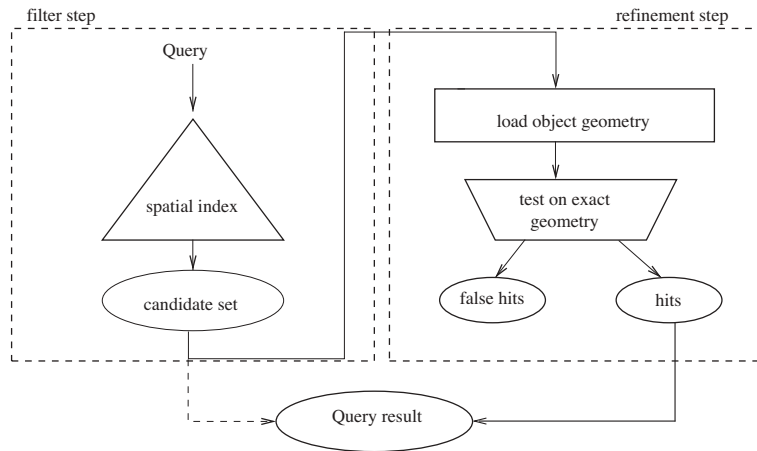
In this section, we describe the processing techniques for evaluating queries on spatial databases and discuss open problems in spatial query processing and query optimization.

**Spatial Query Operations.** Spatial query operations can be classified into four groups (68).

- **Update Operations:** These include standard database operations such as modify, create, and delete.
- **Spatial Selection:** These can be of two types:
  - **Point Query:** Given a query point, find all spatial objects that contain it. An example is the following query, “Find all river flood-plains which contain the SHRINE.”
  - **Regional Query:** Given a query polygon, find all spatial objects that intersect the query polygon. When the query polygon is a rectangle, this query is called a window query. These queries are sometimes also referred to as range queries. An example query could be “Identify the names of all forest stands that intersect a given window.”
  - **Spatial Join:** Like the join operator in relational databases, the spatial join is one of the more important operators. When two tables are joined on a spatial attribute, the join is called a spatial join. A variant of the spatial join and an important operator in GIS is the *map overlay*. This operation combines two sets of spatial objects to form new ones. The “boundaries” of a set of these new objects are determined by the nonspatial attributes assigned by the overlay operation. For example, if the operation assigns the same value of the nonspatial attribute



**Figure 7.** Entity relationship diagrams for common representations of spatial data.



**Figure 8.** Two-step processing.

to two neighboring objects, then the objects are “merged.” Some examples of spatial join predicates are *intersect*, *contains*, *is\_enclosed\_by*, *distance*, *northwest*, *adjacent*, *meets*, and *overlap*. A query example of a spatial join is “Find all forest-stands and river flood-plains which overlap.”

- **Spatial Aggregate:** An example of a spatial aggregate is “Find the river closest to a campground.” Spatial aggregates are usually variants of the *Nearest Neighbor* (69–71) search problem: Given a query object, find the object having minimum distance from the query object. A *Reverse Nearest Neighbor* (RNN) (72–76) query is another example of a spatial aggregate. Given a query object, a RNN query finds objects for which the query object is the nearest neighbor. Applications of RNN include army strategic planning where a medical unit, *A*, in the battlefield is always in search of a wounded soldier for whom *A* is the nearest medical unit.

**Visibility Queries.** Visibility has been widely studied in computer graphics. Visibility may be defined as the parts of objects and the environment that are visible from a point in space. A visibility query can be thought of as a query that returns the objects and part of the environment visible at the querying point. For example, within a city, if the coverage area of a wireless antenna is considered to be the visible area, then the union of coverage areas of all the antennas in the city will provide an idea about the area that is not covered. Such information may be used to place a new antenna strategically at an optimal location. In a visibility query, if the point in space moves, then the area of visibility changes. Such a query may be called a continuous visibility query. For example, security for the president’s motorcade involves cordoning off the buildings that have route visibility. In such a case, the visibility query may be thought of as a query that returns the buildings visible at different points on the route.

**Visual Queryings.** Many spatial applications present results visually, in the form of maps that consist of graphic images, 3-D displays, and animations. These applications allow users to query the visual representation by pointing to the visual representation using pointing devices such as a mouse or a pen. Such graphical interfaces are needed to query spatial data without the need by users to write any SQL statements. In recent years, map services, such as Google Earth and Microsoft Earth, have become very popular. more work is needed to explore the impact of querying by pointing and visual presentation of results on database performance.

**Two-Step Query Processing of Spatial Operations.** Because spatial query processing involves complex data types, a lake boundary might need a thousand vertices for exact representation. Spatial operations typically follow a two-step algorithm (*filter* and *refinement*) as shown in Fig. 8 to process complex spatial objects efficiently (77). Approximate geometry, such as the minimal orthogonal bounding rectangle of an extended spatial object, is first used to filter out many irrelevant objects quickly. Exact geometry then is used for the remaining spatial objects to complete the processing.

- **Filter step:** In this step, the spatial objects are represented by simpler approximations like the minimum bounding rectangle (MBR). For example, consider the following point query, “Find all rivers whose flood-plains overlap the SHRINE.” In SQL this query will be:

```
SELECT  river.name
FROM    river
WHERE   overlap (river.flood-plain, :
SHRINE)
```

If we approximate the flood-plains of all rivers with MBRs, then it is less expensive to determine whether the point is in a MBR than to check whether a point is in an irregular polygon, that is, in the exact shape of the flood-plain. The answer from this approximate test is a superset of the real answer set. This superset is sometimes called the candidate set. Even the spatial predicate

may be replaced by an approximation to simplify a query optimizer. For example, touch (river.flood-plain, :SHRINE) may be replaced by overlap(MBR(river.flood-plain, :SHRINE), and MBR(:SHRINE)) in the filter step. Many spatial operators, for example, inside, north-of, and buffer, can be approximated using the overlap relationship among corresponding MBRs. Such a transformation guarantees that no tuple from the final answer using exact geometry is eliminated in the filter step.

- **Refinement step:** Here, the exact geometry of each element from the candidate set and the exact spatial predicate is examined. This examination usually requires the use of a CPU-intensive algorithm. This step may sometimes be processed outside the spatial database in an application program such as GIS, using the candidate set produced by the spatial database in the filter step.

**Techniques for Spatial Operations.** This section presents several common operations between spatial objects: selection, spatial join, aggregates, and bulk loading.

**Selection Operation.** Similar to traditional database systems, the selection operation can be performed on indexed or non indexed spatial data. The difference is in the technique used to evaluate the predicate and the type of index. As discussed in the previous section, a two-step approach, where the geometry of a spatial object is approximated by a rectangle, is commonly used to evaluate a predicate. Popular indexing techniques for spatial data are R-tree, and space-filling curves. An R-tree is a height-balanced tree that is a natural extension of a B-tree for  $k$ -dimensions. It allows a point search to be processed in  $O(\log n)$  time. Hash filling curves provide one-to-one continuous mappings that map points of multidimensional space into one-dimensional space. This mapping allows the user to impose order on higher-dimensional spaces. Common examples of space-filling curves are row-order Peano, Z-order, and Hilbert curves. Once the data has been ordered by a space-filling curve, a B-tree index can be imposed on the ordered entries to enhance the search. Point search operations can be performed in  $O(\log n)$  time.

**Spatial Join Operation.** Conceptually, a join is defined as a cross product followed by a selection condition. In practice, this viewpoint can be very expensive because it involves materializing the cross product before applying the selection criterion. This finding is especially true for spatial databases. Many ingenious algorithms have been proposed to preempt the need to perform the cross product. The two-step query-processing technique described in the previous section is the most commonly used. With such methods, the spatial join operation can be reduced to a rectangle-rectangle intersection, the cost of which is relatively modest compared with the I/O cost of retrieving pages from secondary memory for processing.

A number of strategies have been proposed for processing spatial joins. Interested readers are encouraged to refer to Refs. 78 through 82.

**Aggregate Operation: Nearest Neighbor, Reverse Nearest Neighbor.** Nearest Neighbor queries are common in many applications. For example, a person driving on the road may want to find the nearest gas station from his current location. Various algorithms exist for nearest neighbor queries (69–71,83,84). Techniques based on Voronoi diagrams, Quad-tree indexing, and Kd-trees have been discussed in Ref. 27.

Reverse Nearest Neighbor queries were introduced in Ref. 72 in the context of decision support systems. For example, a RNN query can be used to find a set of customers who can be influenced by the opening of a new store-outlet location.

**Bulk Loading.** Bulk operations affect potentially a large set of tuples, unlike other database operations, such as insert into a relation, which affects possibly one tuple at a time. Bulk loading refers to the creation of an index from scratch on a potentially large set of data. Bulk loading has its advantages because the properties of the data set may be known in advance. These properties may be used to design efficiently the space-partitioning index structures commonly used for spatial data. An evaluation of generic bulk loading techniques is provided in Ref. 85.

**Parallel GIS.** A high-performance geographic information system (HPGIS) is a central component of many interactive applications like real-time terrain visualization, situation assessment, and spatial decision-making. The geographic information system (GIS) often contains large amounts of geometric and feature data (e.g., location, elevation, and soil type) represented as large sets of points, chains of line segments, and polygons. This data is often accessed via range queries. The existing sequential methods for supporting GIS operations do not meet the real-time requirements imposed by many interactive applications.

Hence, parallelization of GIS is essential for meeting the high performance requirements of several realtime applications. A GIS operation can be parallelized either by function partitioning (86–88) or by data partitioning (89–97). Function-partitioning uses specialized data structures (e.g., distributed data structures) and algorithms that may be different from their sequential counterparts. Data partitioning techniques divide the data among different processors and independently execute the sequential algorithm on each processor. Data partitioning in turn is achieved by declustering (98,99) the spatial data. If the static declustering methods fail to distribute the load equally among different processors, the load balance may be improved by redistributing parts of the data to idle processors using dynamic load-balancing (DLB) techniques.

## Research Needs

This section presents the research needs for spatial query processing and query optimization.

**Query Processing.** Many open research areas exist at the logical level of query processing, including query-cost modeling and queries related to fields and networks. Cost

**Table 6. Difficult spatial queries from GIS**

Voronoi	Classify households as to which supermarket they are closest to
Network	Find the shortest path from the warehouse to all delivery stops
Time-dependent network	Find the shortest path where the road network is dynamic
Allocation	Where is the best place to build a new restaurant
Transformation	Triangulate a layer based on elevation
BulkLoad	Load a spatial data file into the database
Raster $\leftrightarrow$ Vector	Convert between raster and vector representations
Visibility	Find all points of objects and environment visible from a point
EvacuationRoute	Find evacuation routes based on capacity and availability constraints
PredictLocation	Predict the location of a mobile person based on personal route patterns

models are used to rank and select the promising processing strategies, given a spatial query and a spatial data set. However, traditional cost models may not be accurate in estimating the cost of strategies for spatial operations, because of the distance metric and the semantic gap between relational operators and spatial operation. Comparison of the execution costs of such strategies required that new cost models be developed to estimate the selectivity of spatial search and join operations. Preliminary work in the context of the R-tree, tree-matching join, and fractal-model is promising (100,101), but more work is needed.

Many processing strategies using the overlap predicate have been developed for range queries and spatial join queries. However, a need exists to develop and evaluate strategies for many other frequent queries such as those listed in Table 6. These include queries on objects using predicates other than overlap, queries on fields such as slope analysis, and queries on networks such as the shortest path to a set of destinations.

Depending on the type of spatial data and the nature of the query, other research areas also need to be investigated. A *moving objects query* involves spatial objects that are mobile. Examples of such queries include “Which is the nearest taxi cab to the customer?”, “Where is the hurricane expected to hit next?”, and “What is a possible location of a serial criminal?” With the increasing availability of streaming data from GPS devices, continuous queries has become an active area of research. Several techniques (25,102,103) have been proposed to execute such queries.

A skyline query (104) is a query to retrieve a set of interesting points (records) from a potentially huge collection of points (records) based on certain attributes. For example, considering a set of hotels to be points, the skyline query may return a set of interesting hotels based on a user’s preferences. The set of hotels returned for a user who prefers a cheap hotel may be different from the set of hotels returned for a user who prefers hotels that are closer to the coast. Research needed for skyline query operation includes computation of algorithms and processing for higher dimensions (attributes). Other query processing techniques in which research is required are querying on *3-D spatial data* and *spatio-temporal data*.

**Query Optimization.** The *query optimizer*, a module in database software, generates different evaluation plans and determines the appropriate execution strategy. Before the query optimizer can operate on the query, the high-level

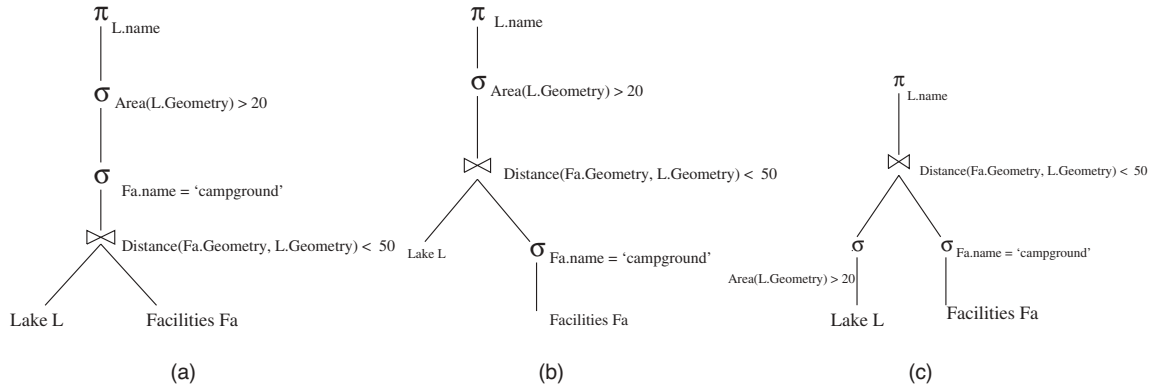
declarative statement must be scanned through a *parser*. The parser checks the syntax and transforms the statement into a *query tree*. In traditional databases, the data types and functions are fixed and the parser is relatively simple. Spatial databases are examples of an extensible database system and have provisions for user-defined types and methods. Therefore, compared with traditional databases, the parser for spatial databases has to be considerably more sophisticated to identify and manage user-defined data types and map them into syntactically correct query trees. In the query tree, the leaf nodes correspond to the relations involved and the internal nodes correspond to the basic operations that constitute the query. Query processing starts at the leaf nodes and proceeds up the tree until the operation at the root node has been performed.

Consider the query, “Find all lakes which have an area greater than 20 sq. km. and are within 50 km. from the campground.” Let us assume that the *Area()* function is not precomputed and that its value is computed afresh every time it is invoked. A query tree generated for the query is shown in Fig. 9(a). In the classic situation, the rule “select before join” would dictate that the *Area* function be computed before the join predicate function, *Distance()* [Fig. 9(b)], the underlying assumption being that the computational cost of executing the select and join predicate is equivalent and negligible compared with the I/O cost of the operations. In the spatial situation, the relative cost per tuple of *Area()* and *Distance()* is an important factor in deciding the order of the operations (105). Depending the implementation of these two functions, the optimal strategy may be to process the join before the select operation [Fig. 9(c)]. This approach thus violates the main heuristic rule for relational databases, which states “Apply select and project before the join and binary operations” are no longer unconditional. A cost-based optimization technique exists to determine the optimal execution strategy from a set of execution plans. A quantitative analysis of spatial index structures is used to calculate the expected number of disk accesses that are required to perform a spatial query (106). Nevertheless, in spite of these advances, query optimization techniques for spatial data need more study.

## SPATIAL FILE ORGANIZATION AND INDICES

### Accomplishments

**Space-Filling Curves.** The physical design of a spatial database optimizes the instructions to storage devices for



**Figure 9.** (a) Query tree, (b) “pushing down”: select operation, and (c) “pushing down” may not help.

performing common operations on spatial data files. File designs for secondary storage include clustering methods and spatial hashing methods. Spatial clustering techniques are more difficult to design than traditional clustering techniques because no natural order exists in multidimensional space where spatial data resides. This situation is only complicated by the fact that the storage disk is a logical one-dimensional device. Thus, what is needed is a mapping from a higher-dimensional space to a one-dimensional space that is distance-preserving: This mapping ensures that elements that are close in space are mapped onto nearby points on the line and that no two points in the space are mapped onto the same point on the line (107). Several mappings, none of them ideal, have been proposed to accomplish this feat. The most prominent ones include row-order, Z-order, and the Hilbert-curve (Fig. 10).

Metric clustering techniques use the notion of distance to group nearest neighbors together in a metric space. Topological clustering methods like connectivity-clustered access methods (108) use the min-cut partitioning of a graph representation to support graph traversal operations efficiently. The physical organization of files can be supplemented with indices, which are data structures to improve the performance of search operations.

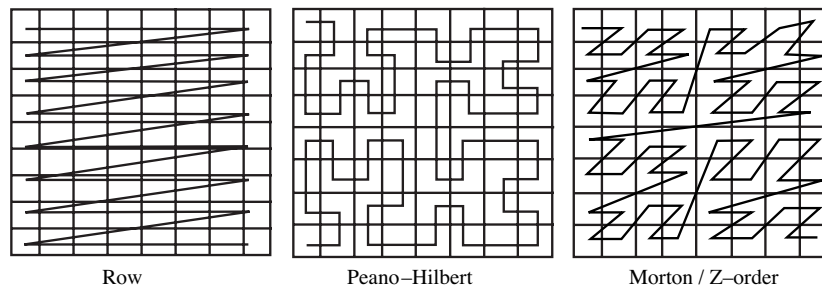
Classical one-dimensional indices such as the B<sup>+</sup>-tree can be used for spatial data by linearizing a multidimensional space using a space-filling curve such as the Z-order. Many spatial indices (27) have been explored for multidimensional Euclidean space. Representative indices for point objects include grid files, multidimensional grid files

(109), Point-Quad-Trees, and Kd-trees. Representative indices for extended objects include the R-tree family, the Field-tree, Cell-tree, BSP-tree, and Balanced and Nested grid files.

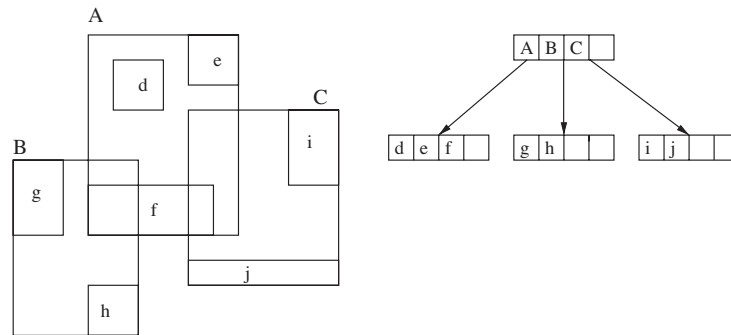
**Grid Files.** Grid files were introduced by Nievergelt (110). A grid file divides the space into *n*-dimensional spaces that can fit into equal-size buckets. The structures are not hierarchical and can be used to index static uniformly distributed data. However, because of its structure, the directory of a grid file can be so sparse and large that a large main memory is required. Several variations of grid files, exist to index data efficiently and to overcome these limitations (111,112). An overview of grid files is given in Ref. 27.

**Tree indexes.** *R-tree* aims to index objects in a hierarchical index structure (113). The R-tree is a height-balanced tree that is the natural extension of the B-tree for *k*-dimensions. Spatial objects are represented in the R-tree by their minimum bounding rectangle (MBR). Figure 11 illustrates spatial objects organized as an R-tree index. R-trees can be used to process both point and range queries.

Several variants of R-trees exist for better performance of queries and storage use. The R<sup>+</sup>-tree (114) is used to store objects by avoiding overlaps among the MBRs, which increases the performance of the searching. R<sup>-</sup>-trees (115) rely on the combined optimization of the area, margin, and overlap of each MBR in the intermediate nodes of the tree, which results in better storage use.



**Figure 10.** Space-filling curves to linearize a multidimensional space.



**Figure 11.** Spatial objects (d, e, f, g, h, and i) arranged in an R-tree hierarchy.

Many R-tree-based index structures (116–119,120,121) have been proposed to index spatio-temporal objects. A survey of spatio-temporal access methods has been provided in Ref. 122.

*Quad tree* (123) is a space-partitioning index structure in which the space is divided recursively into quads. This recursive process is implemented until each quad is homogeneous. Several variations of quad trees are available to store point data, raster data, and object data. Also, other quad tree structures exist to index spatio-temporal data sets, such as overlapping linear quad trees (24) and multiple overlapping features (MOF) trees (125).

The *Generalized Search Tree (GiST)* (126) provides a framework to build almost any kind of tree index on any kind of data. Tree index structures, such as B<sup>+</sup>-tree and R-tree, can be built using GiST. A spatial-partitioning generalized search tree (SP-GiST) (127) is an extensible index structure for space-partitioning trees. Index trees such as quad tree and kd-tree can be built using SP-GiST.

**Graph Indexes.** Most spatial access methods provide methods and operators for point and range queries over collections of spatial points, line segments, and polygons. However, it is not clear if spatial access methods can efficiently support network computations that traverse line segments in a spatial network based on connectivity rather than geographic proximity. A *connectivity-clustered access method for spatial network (CCAM)* is proposed to index spatial networks based on graph partitioning (108) by supporting network operations. An auxiliary secondary index, such as B<sup>+</sup>-tree, R-tree, and Grid File, is used to support network operations such as *Find()*, *get-a-Successor()*, and *get-Successors()*.

### Research Needs

**Concurrency Control.** The R-link tree (128) is among the few approaches available for concurrency control on the R-tree. New approaches for concurrency-control techniques are needed for other spatial indices. Concurrency is provided during operations such as search, insert, and delete. The R-link tree is also recoverable in a write-ahead logging environment. Reference 129 provides general algorithms for concurrency control for GiST that can also be applied to tree-based indexes. Research is required for concurrency control on other useful spatial data structures.

## TRENDS: SPATIAL DATA MINING

### Accomplishments

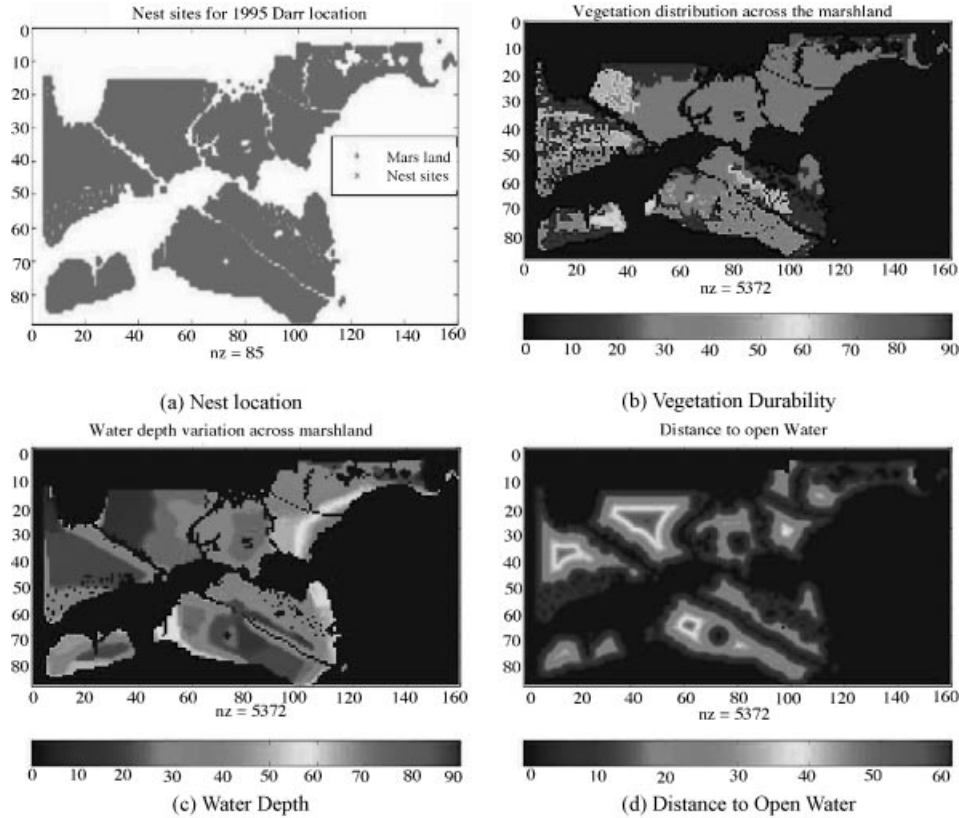
The explosive growth of spatial data and widespread use of spatial databases emphasize the need for the automated discovery of spatial knowledge. Spatial data mining is the process of discovering interesting and previously unknown, but potentially useful, patterns from spatial databases. Some applications are location-based services, studying the effects of climate, land-use classification, predicting the spread of disease, creating high-resolution three-dimensional maps from satellite imagery, finding crime hot spots, and detecting local instability in traffic. A detailed review of spatial data mining can be found in Ref. 130.

The requirements of mining spatial databases are different from those of mining classic relational databases. The difference between classic and spatial data mining parallels the difference between classic and spatial statistics. One fundamental assumption that guides statistical analysis is that the data samples are generated independently, as with successive tosses of a coin or the rolling of a die. When it comes to the analysis of spatial data, the assumption about the independence of samples is generally false. In fact, spatial data tends to be highly self-correlated. For example, changes in natural resources, wildlife, and temperature vary gradually over space. The notion of *spatial autocorrelation*, the idea that similar objects tend to cluster in geographic space, is unique to spatial data mining.

For detailed discussion of spatial analysis, readers are encouraged to refer to Refs. 131 and 132.

**Spatial Patterns.** This section presents several spatial patterns, specifically those related to location prediction, Markov random fields, spatial clustering, spatial outliers, and spatial colocation.

**Location Prediction.** Location prediction is concerned with the discovery of a model to infer locations of a spatial phenomenon from the maps of other spatial features. For example, ecologists build models to predict habitats for endangered species using maps of vegetation, water bodies, climate, and other related species. Figure 12 shows the learning data set used in building a location-prediction



**Figure 12.** (a) Learning data set: The geometry of the Darr wetland and the locations of the nests, (b) the spatial distribution of *vegetation durability* over the marshland, (c) the spatial distribution of *water depth*, and (d) the spatial distribution of *distance to open water*.

model for red-winged blackbirds in the Darr and Stubble wetlands on the shores of Lake Erie in Ohio. The data set consists of nest location, vegetation durability, distance to open water, and water depth maps. Spatial data mining techniques that capture the spatial autocorrelation (133,134) of nest location such as the spatial autoregression model (SAR) and markov random fields (MRF) are used for location-prediction modeling.

**Spatial Autoregression Model.** Linear regression models are used to estimate the conditional expected value of a dependent variable  $y$  given the values of other variables  $X$ . Such a model assumes that the variables are independent. The spatial autoregression model (131,135–137) is an extension of the linear regression model that takes spatial autocorrelation into consideration. If the dependent values  $y$  and  $X$  are related to each other, then the regression equation (138) can be modified as

$$y = \rho W y + X \beta + \varepsilon \quad (1)$$

Here,  $W$  is the neighborhood relationship contiguity matrix, and  $\rho$  is a parameter that reflects the strength of the spatial dependencies between the elements of the dependent variable. Notice that when  $\rho = 0$ , this equation

collapses to the linear regression model. If the spatial autocorrelation coefficient is statistically significant, then SAR will quantify the presence of spatial autocorrelation. In such a case, the spatial autocorrelation coefficient will indicate the extent to which variations in the dependent variable ( $y$ ) are explained by the average of neighboring observation values.

**Markov Random Field.** Markov random field-based (139) Bayesian classifiers estimate the classification model,  $\hat{f}_c$ , using MRF and Bayes' rule. A set of random variables whose interdependency relationship is represented by an undirected graph (i.e., a symmetric neighborhood matrix) is called a Markov random field. The Markov property specifies that a variable depends only on its neighbors and is independent of all other variables. The location prediction problem can be modeled in this framework by assuming that the class label,  $l_i = f_c(s_i)$ , of different locations,  $s_i$ , constitutes an MRF. In other words, random variable  $l_i$  is independent of  $l_j$  if  $W(s_i, s_j) = 0$ .

The Bayesian rule can be used to predict  $l_i$  from feature value vector  $X$  and neighborhood class label vector  $L_i$  as follows:

$$Pr(l_i | X, L_i) = \frac{Pr(X | l_i, L_i) Pr(l_i | L_i)}{Pr(X)} \quad (2)$$

The solution procedure can estimate  $Pr(l_i | L_i)$  from the training data, where  $L_i$  denotes a set of labels in the neighborhood of  $s_i$  excluding the label at  $s_i$ . It makes this estimate by examining the ratios of the frequencies of class labels to the total number of locations in the spatial framework.  $Pr(X | l_i, L_i)$  can be estimated using kernel functions from the observed values in the training data set.

A more detailed theoretical and experimental comparison of these methods can be found in Ref. 140. Although MRF and SAR classification have different formulations, they share a common goal of estimating the posterior probability distribution. However, the posterior probability for the two models is computed differently with different assumptions. For MRF, the posterior is computed using Bayes' rule, whereas in SAR, the posterior distribution is fitted directly to the data.

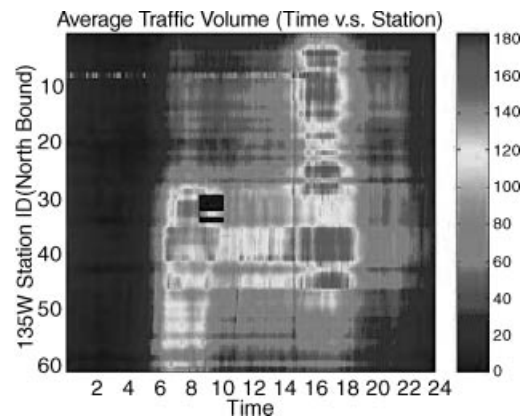
**Spatial Clustering.** Spatial clustering is a process of grouping a set of spatial objects into clusters so that objects within a cluster have high similarity in comparison with one another but are dissimilar to objects in other clusters.

For example, clustering is used to determine the "hot spots" in crime analysis and disease tracking. Many criminal justice agencies are exploring the benefits provided by computer technologies to identify crime hot spots to take preventive strategies such as deploying saturation patrols in hot-spot areas.

Spatial clustering can be applied to group similar spatial objects together; the implicit assumption is that patterns in space tend to be grouped rather than randomly located. However, the statistical significance of spatial clusters should be measured by testing the assumption in the data. One method to compute this measure is based on quadrats (i.e., well-defined areas, often rectangular in shape). Usually quadrats of random location and orientations in the quadrats are counted, and statistics derived from the counters are computed. Another type of statistics is based on distances between patterns; one such type is Ripley's  $K$ -function (141). After the verification of the statistical significance of the spatial clustering, classic clustering algorithms (142) can be used to discover interesting clusters.

**Spatial Outliers.** A spatial outlier (143) is a spatially referenced object whose nonspatial attribute values differ significantly from those of other spatially referenced objects in its spatial neighborhood. Figure 13 gives an example of detecting spatial outliers in traffic measurements for sensors on highway I-35W (North bound) for a 24-hour time period. Station 9 seems to be a spatial outlier as it exhibits inconsistent traffic flow as compared with its neighboring stations. The reason could be that the sensor at station 9 is malfunctioning. Detecting spatial outliers is useful in many applications of geographic information systems and spatial databases, including transportation, ecology, public safety, public health, climatology, and location-based services.

Spatial attributes are used to characterize location, neighborhood, and distance. Nonspatial attribute dimensions are used to compare a spatially referenced object with its neighbors. Spatial statistics literature provides two kinds of bipartite multidimensional tests, namely



**Figure 13.** Spatial outlier (station ID 9) in traffic volume data.

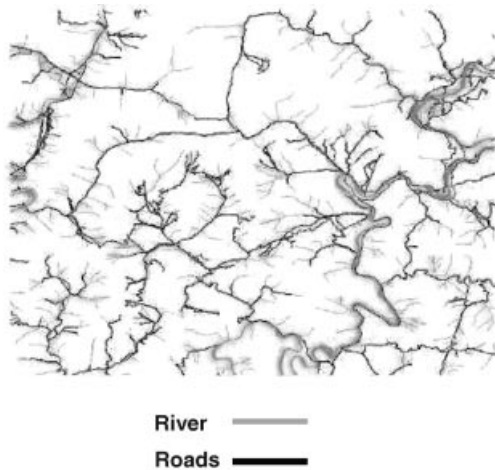
graphical tests and quantitative tests. Graphical tests, which are based on the visualization of spatial data, highlight spatial outliers, for example, variogram clouds (141) and Moran scatterplots (144). Quantitative methods provide a precise test to distinguish spatial outliers from the remainder of data. A unified approach to detect spatial outliers efficiently is discussed in Ref. 145., Reference 146 provides algorithms for multiple spatial-outlier detection.

**Spatial Colocation** The colocation pattern discovery process finds frequently collocated subsets of spatial event types given a map of their locations. For example, the analysis of the habitats of animals and plants may identify the colocations of predator-prey species, symbiotic species, or fire events with fuel, ignition sources and so forth. Figure 14 gives an example of the colocation between roads and rivers in a geographic region.

Approaches to discovering colocation rules can be categorized into two classes, namely spatial statistics and data-mining approaches. Spatial statistics-based approaches use measures of spatial correlation to characterize the relationship between different types of spatial features. Measures of spatial correlation include the cross  $K$ -function with Monte Carlo simulation, mean nearest-neighbor distance, and spatial regression models.

Data-mining approaches can be further divided into transaction-based approaches and distance-based approaches. Transaction-based approaches focus on defining transactions over space so that an a priori-like algorithm can be used. Transactions over space can be defined by a reference-feature centric model. Under this model, transactions are created around instances of one user-specified spatial feature. The association rules are derived using the a priori (147) algorithm. The rules formed are related to the reference feature. However, it is nontrivial to generalize the paradigm of forming rules related to a reference feature to the case in which no reference feature is specified. Also, defining transactions around locations of instances of all features may yield duplicate counts for many candidate associations.

In a distance-based approach (148–150), instances of objects are grouped together based on their Euclidean distance from each other. This approach can be considered to be an event-centric model that finds subsets of spatial



**Figure 14.** Colocation between roads and rivers in a hilly terrain (Courtesy: Architecture Technology Corporation).

features likely to occur in a neighborhood around instances of given subsets of event types.

### Research Needs

This section presents several research needs in the area of spatio-temporal data mining and spatial-temporal network mining.

**Spatio-Temporal Data Mining.** Spatio-temporal (ST) data mining aims to develop models and objective functions and to discover patterns that are more suited to Spatio-temporal databases and their unique properties (15). An extensive survey of Spatio-temporal databases, models, languages, and access methods can be found in Ref. 152. A bibliography of Spatio-temporal data mining can be found in Ref. 153.

Spatio-temporal pattern mining focuses on discovering knowledge that frequently is located together in space and time. References 154, 155, and 156 defined the problems of discovering mixed-drove and sustained emerging Spatio-temporal co-occurrence patterns and proposed interest measures and algorithms to mine such patterns. Other research needs include conflation, in which a single feature is obtained from several sources or representations. The goal is to determine the optimal or best representation based on a set of rules. Problems tend to occur during maintenance operations and cases of vertical obstruction.

In several application domains, such as sensor networks, mobile networks, moving object analysis, and image analysis, the need for spatio-temporal data mining is increasing drastically. It is vital to develop new models and techniques, to define new spatio-temporal patterns, and to formulize monotonic interest measures to mine these patterns (157).

**Spatio-Temporal Network Mining.** In the post-9/11 world of asymmetric warfare in urban area, many human activities are centered about ST infrastructure networks, such

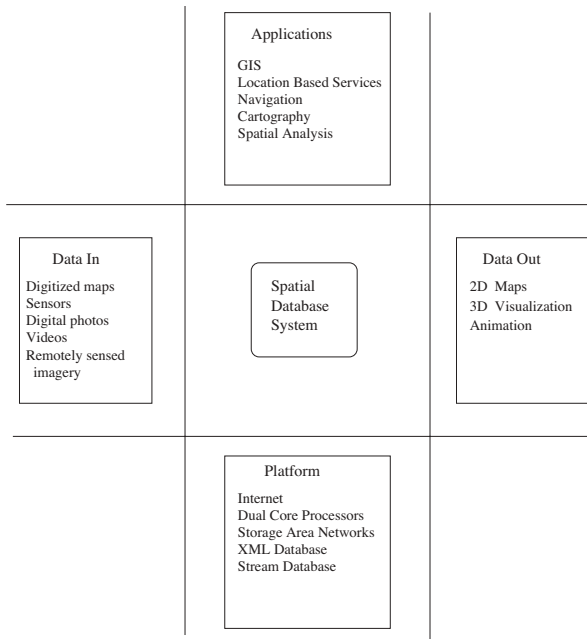
as transportation, oil/gas-pipelines, and utilities (e.g., water, electricity, and telephone). Thus, activity reports, for example, crime/insurgency reports, may often use network-based location references, for example, street address such as “200 Quiet Street, Scaryville, RQ 91101.” In addition, spatial interaction among activities at nearby locations may be constrained by network connectivity and network distances (e.g., shortest path along roads or train networks) rather than geometric distances (e.g., Euclidean or Manhattan distances) used in traditional spatial analysis. Crime prevention may focus on identifying subsets of ST networks with high activity levels, understanding underlying causes in terms of ST-network properties, and designing ST-network-control policies.

Existing spatial analysis methods face several challenges (e.g., see Ref. 158). First, these methods do not model the effect of explanatory variables to determine the locations of network hot spots. Second, existing methods for network pattern analysis are computationally expensive. Third, these methods do not consider the temporal aspects of the activity in the discovery of network patterns. For example, the routes used by criminals during the day and night may differ. The periodicity of bus/train schedules can have an impact on the routes traveled. Incorporating the time-dependency of transportation networks can improve the accuracy of the patterns.

### SUMMARY

In this chapter we presented the major research accomplishments and techniques that have emerged from the area of spatial databases in the past decade. These accomplishments and techniques include spatial database modeling, spatial query processing, and spatial access methods. We have also identified areas in which more research is needed, such as spatio-temporal databases, spatial data mining, and spatial networks.

Figure 15 provides a summary of topics that continue to drive the research needs of spatial database systems. Increasingly available spatial data in the form of digitized maps, remotely sensed images, Spatio-temporal data (for example, from videos), and streaming data from sensors have to be managed and processed efficiently. New ways of querying techniques to visualize spatial data in more than one dimension are needed. Several advances have been made in computer hardware over the last few years, but many have yet to be fully exploited, including increases in main memory, more effective storage using storage area networks, greater availability of multicore processors, and powerful graphic processors. A huge impetus for these advances has been spatial data applications such as land navigation systems and location-based services. To measure the quality of spatial database systems, new benchmarks have to be established. Some benchmarks (159,160) established earlier have become dated. Newer benchmarks are needed to characterize the spatial data management needs of other systems and applications such as Spatio-temporal databases, moving-objects databases, and location-based services.



**Figure 15.** Topics driving future research needs in spatial database systems.

## ACKNOWLEDGMENTS

We thank the professional organizations that have funded the research on spatial databases, in particular, the National Science Foundation, Army Research Laboratory, Topographic Engineering Center, Oak Ridge National Laboratory, Minnesota Department of Transportation, and Microsoft Corporation. We thank members of the spatial database and spatial data mining research group at the University of Minnesota for refining the content of this chapter. We also thank Kim Koffolt for improving the readability of this chapter.

## BIBLIOGRAPHY

- R. H. Gutting, An introduction to spatial database systems, *VLDB Journal, Special Issue on Spatial Database Systems*, 3(4): 357–399, 1994.
- W. Kim, J. Garza, and A. Kesin, Spatial data management in database systems, in *Advances in Spatial Databases, 3rd International Symposium, SSD'93*, Vol. 652. Springer, 1993.
- Y. Manolopoulos, A. N. Papadopoulos, and M. G. Vassilakopoulos, *Spatial Databases: Technologies, Techniques and Trends*. Idea Group Publishing, 2004.
- S. Shekhar and S. Chawla, *Spatial Databases: A Tour*. Prentice Hall, 2002.
- Wikipedia. Available: [http://en.wikipedia.org/wiki/Spatial\\_Database](http://en.wikipedia.org/wiki/Spatial_Database), 2007.
- M. Worboys and M. Duckham, *GIS: A Computing Perspective*, 2nd ed. CRC, 2004.
- J. Schiller, *Location-Based Services*. Morgan Kaufmann, 2004.
- A. Stefanidis and S. Nittel, *GeoSensor Networks*. CRC, 2004.
- R. Scally, *GIS for Environmental Management*. ESRI Press, 2006.
- L. Lang, *Transportation GIS*, ESRI Press, 1999.
- P. Elliott, J. C. Wakefield, N. G. Best, and D. J. Briggs, *Spatial Epidemiology: Methods and Applications*. Oxford University Press, 2000.
- M. R. Leipnik and D. P. Albert, *GIS in Law Enforcement: Implementation Issues and Case Studies*, CRC, 2002.
- D. K. Arctur and M. Zeiler, *Designing Geodatabases*, ESRI Press, 2004.
- E. Beinat, A. Godfrind, and R. V. Kothuri, *Pro Oracle Spatial*, Apress, 2004.
- SQL Server 2008 (Code-name Katmai). Available: <http://www.microsoft.com/sql/prodinfo/futureversion/default.aspx>, 2007.
- Google Earth. Available: <http://earth.google.com>, 2006.
- Microsoft Virtual Earth. Available: <http://www.microsoft.com/virtualearth>, 2006.
- PostGIS. Available: <http://postgis.refractor.net/>, 2007.
- MySQL Spatial Extensions. Available: <http://dev.mysql.com/doc/refman/5.0/en/spatial-extensions.html>, 2007.
- Sky Server, 2007. Available: <http://skyserver.sdss.org/>.
- D. Chamberlin, Using the New DB2: IBM's Object Relational System, *Morgan Kaufmann*, 1997.
- M. Stonebraker and D. Moore, Object Relational DBMSs: The Next Great Wave. Morgan Kaufmann, 1997.
- OGC. Available: <http://www.opengeospatial.org/standards>, 2007.
- P. Rigaux, M. Scholl, and A. Voisard, *Spatial Databases: With Application to GIS*, Morgan Kaufmann Series in Data Management Systems, 2000.
- R. H. Gutting and M. Schneider, *Moving Objects Databases (The Morgan Kaufmann Series in Data Management Systems)*. San Francisco, CA: Morgan Kaufmann Publishers Inc., 2005.
- S. Shekhar and H. Xiong, *Encyclopedia of GIS*, Springer, 2008, forthcoming.
- H. Samet, *Foundations of Multidimensional and Metric Data Structures*, Morgan Kaufmann Publishers, 2006.
- ACM Geographical Information Science Conference. Available: <http://www.acm.org>.
- ACM Special Interest Group on Management Of Data. Available: <http://www.sigmod.org/>.
- Geographic Information Science Center Summer and Winter Assembly. Available: <http://www.gisc.berkeley.edu/>.
- Geoinformatica. Available: <http://www.springerlink.com/content/100268/>.
- IEEE International Conference on Data Engineering. Available: <http://www.icde2007.org/icde/>.
- IEEE Transactions on Knowledge and Data Engineering (TKDE). Available: <http://www.computer.org/tkde/>.
- International Journal of Geographical Information Science. Available: <http://www.tandf.co.uk/journals/tf/13658816.html>.
- International Symposium on Spatial and Temporal Databases. Available: <http://www.cs.ust.hk/sstd07/>.
- Very Large Data Bases Conference. Available: <http://www.vldb2007.org/>.
- UCGIS, 1998.
- S. Shekhar, R. R. Vatsavai, S. Chawla, and T. E. Burke, Spatial Pictogram Enhanced Conceptual Data Models and Their Translations to Logical Data Models, *Integrated*

*Spatial Databases: Digital Images and GIS, Lecture Notes in Computer Science*, **1737**: 77–104, 1999.

39. N. R. Adam and A. Gangopadhyay, *Database Issues in Geographic Information Systems*, Norwell, MA: Kluwer Academic Publishers, 1997.
40. M. Stonebraker and G. Kemnitz, The Postgres Next Generation Database Management System, *Commun. ACM*, **34**(10): 78–92, 1991.
41. P. Bolstad, *GIS Fundamentals: A First Text on Geographic Information Systems*, 2nd ed. Eider Press, 2005.
42. T. Hadzilacos and N. Tryfona, An Extended Entity-Relationship Model for Geographic Applications, *ACM SIGMOD Record*, **26**(3): 24–29, 1997.
43. S. Shekhar, R. R. Vatsavai, S. Chawla, and T. E. Burk, Spatial pictogram enhanced conceptual data models and their translation to logical data models, *Lecture Notes in Computer Science*, **1737**: 77–104, 2000.
44. F. T. Fonseca and M. J. Egenhofer, Ontology-driven geographic information systems, in Claudia Bauzer Medeiros (ed.), *ACM-GIS '99, Proc. of the 7th International Symposium on Advances in Geographic Information Systems*, Kansas City, US 1999, pp. 14–19. ACM, 1999.
45. Time ontology in owl, *Electronic*, September 2005.
46. H. J. Berners-Lee, T. Lassila, and O. Lassila, The semantic web, *The Scientific American*, 2001, pp. 34–43.
47. M. J. Egenhofer, Toward the semantic geospatial web, *Proc. Tenth ACM International Symposium on Advances in Geographic Information Systems*, 2002.
48. F. Fonseca and M. A. Rodriguez, From geo-pragmatics to derivation ontologies: New directions for the geospatial semantic web, *Transactions in GIS*, **11**(3), 2007.
49. W3C. Resource Description Framework. Available: <http://www.w3.org/RDF/>, 2004.
50. G. Subbiah, A. Alam, L. Khan, B. Thuraisingham, An integrated platform for secure geospatial information exchange through the semantic web, *Proc. ACM Workshop on Secure Web Services (SWS)*, 2006.
51. Open Geospatial Consortium Inc. OpenGIS Reference Model. Available: <http://orm.opengeospatial.org/>, 2006.
52. S. Shekhar and X. Liu, Direction as a Spatial Object: A Summary of Results, in R. Laurini, K. Makki, and N. Pissinou, (eds.), *ACM-GIS '98, Proc. 6th international symposium on Advances in Geographic Information Systems*. ACM, 1998, pp. 69–75.
53. S. Skiadopoulos, C. Giannoukos, N. Sarkas, P. Vassiliadis, T. Sellis, and M. Koubarakis, Computing and managing cardinal direction relations, *IEEE Trans. Knowledge and Data Engineering*, **17**(12): 1610–1623, 2005.
54. *A Spatiotemporal Model and Language for Moving Objects on Road Networks*, 2001.
55. *Modeling and querying moving objects in networks*, volume **15**, 2006.
56. *Modeling and Querying Moving Objects*, 1997.
57. *On Moving Object Queries*, 2002.
58. K. K. L. Chan and C. D. Tomlin, Map Algebra as a Spatial Language, in D. M. Mark and A. U. Frank, (eds.), *Cognitive and Linguistic Aspects of Geographic Space*. Dordrecht, Netherlands: Kluwer Academic Publishers, 1991, pp. 351–360.
59. W. Kainz, A. Riedl, and G. Elmes, eds, *A Tetrahedronized Irregular Network Based DBMS Approach for 3D Topographic Data*, Springer Berlin Heidelberg, September 2006.
60. C. Arens, J. Stoter, and P. Oosterom, Modeling 3D Spatial Objects in a Geo-DBMS Using a 3D Primitive, *Computers and Geosciences*, **31**(2): 165–177, act 2005.
61. E. Köhler, K. Langkau, and M. Skutella, Time-expanded graphs for flow-dependent transit times, in *ESA '02: Proceedings of the 10th Annual European Symposium on Algorithms*. London, UK: Springer-Verlag, 2002, pp. 599–611.
62. B. George and S. Shekhar, Time-aggregated graphs for modeling spatio-temporal networks, in *ER (Workshops)*, 2006, pp. 85–99.,
63. K. Eickhorst, P. Agouris, and A. Stefanidis, Modeling and Comparing Spatiotemporal Events, in *Proc. 2004 annual national conference on Digital government research*. Digital Government Research Center, 2004, pp. 1–10.
64. Geographic Markup Language. Available: <http://www.open-gis.net/gml/>, 2007.
65. S. Shekhar, R. R. Vatsavai, N. Sahay, T. E. Burk, and S. Lime, WMS and GML based Interoperable Web Mapping System, in *9th ACM International Symposium on Advances in Geographic Information Systems, ACMGIS01*. ACM, November 2001.
66. CityGML, 2007. Available: <http://www.citygml.org/>.
67. Keyhole Markup Language. Available: <http://code.google.com/apis/kml/documentation/>, 2007.
68. V. Gaede and O. Gunther, Multidimensional access methods, *ACM Computing Surveys*, **30**, 1998.
69. G. R. Hjaltason and H. Samet, Ranking in spatial databases, in *Symposium on Large Spatial Databases*, 1995, pp. 83–95.
70. N. Roussopoulos, S. Kelley, and F. Vincent, Nearest neighbor queries, in *SIGMOD '95: Proc. 1995 ACM SIGMOD international conference on Management of data*, New York: ACM Press, 1995, pp. 71–79.
71. D. Papadias, Y. Tao, K. Mouratidis, and C. K. Hui, Aggregate nearest neighbor queries in spatial databases, *ACM Trans. Database Systems*, **30**(2): 529–576, 2005.
72. F. Korn and S. Muthukrishnan, Influence Sets Based on Reverse Nearest Neighbor Queries, in *Proc. ACM International Conference on Management of Data, SIGMOD*, 2000, pp. 201–212.
73. J. M. Kang, M. Mokbel, S. Shekhar, T. Xia, and D. Zhang, Continuous evaluation of monochromatic and bichromatic reverse nearest neighbors, in *Proc. IEEE 23rd International Conference on Data Engineering (ICDE)*, 2007.
74. I. Stanoi, D. Agrawal, and A. ElAbbadi, Reverse Nearest Neighbor Queries for Dynamic Databases, in *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2000, pp. 44–53.
75. A. Nanopoulos, Y. Theodoridis, and Y. Manolopoulos, C2P: Clustering based on Closest Pairs, in *Proc. International Conference on Very Large Data Bases, VLDB*, 2001, pp. 331–340.
76. T. Xia and D. Zhang, Continuous Reverse Nearest Neighbor Monitoring, in *Proc. International Conference on Data Engineering, ICDE*, 2006.
77. T. Brinkhoff, H.-P. Kriegel, R. Schneider, and B. Seeger, Multi-step processing of spatial joins, In *Proc. ACM International Conference on Management of Data, SIGMOD*, 1994, pp. 197–208.
78. L. Arge, O. Procopiuc, S. Ramaswamy, T. Suel, and J.S. Vitter, Scalable sweeping-based spatial join, in *Proc. Very Large Data Bases (VLDB)*, 1998, pp. 570–581.

79. N. Mamoulis and D. Papadias, Slot index spatial join, *IEEE Trans. Knowledge and Data Engineering*, **15**(1): 211–231, 2003.
80. M.-J. Lee, K.-Y. Whang, W.-S. Han, and I.-Y. Song, Transform-space view: Performing spatial join in the transform space using original-space indexes, *IEEE Trans. Knowledge and Data Engineering*, **18**(2): 245–260, 2006.
81. S. Shekhar, C.-T. Lu, S. Chawla, and S. Ravada, Efficient join-index-based spatial-join processing: A clustering approach, in *IEEE Trans. Knowledge and Data Engineering*, **14**(6): 1400–1421, 2002.
82. M. Zhu, D. Papadias, J. Zhang, and D. L. Lee, Top-k spatial joins, *IEEE Trans. Knowledge and Data Engineering*, **17**(4): 567–579, 2005.
83. H. Hu and D. L. Lee, Range nearest-neighbor query, *IEEE Trans. Knowledge and Data Engineering*, **18**(1): 78–91, 2006.
84. M. L. Yiu, N. Mamoulis, and D. Papadias, Aggregate nearest neighbor queries in road networks, *IEEE Trans. Knowledge and Data Engineering*, **17**(6): 820–833, 2005.
85. J. van den Bercken and B. Seeger, An evaluation of generic bulk loading techniques, in *VLDB '01: Proc. 27th International Conference on Very Large Data Bases*. San Francisco, CA: Morgan Kaufmann Publishers Inc., 2001, pp. 461–470.
86. A. Aggarwal, B. Chazelle, L. Guibas, C. O'Dunlaing, and C. Yap, Parallel computational geometry. *Proc. 25th IEEE Symposium on Foundations of Computer Science*, 1985, pp. 468–477.
87. S. G. Akl and K. A. Lyons, *Parallel Computational Geometry*, Englewood Cliffs, NJ: Prentice Hall, 1993.
88. R. Sridhar, S. S. Iyengar, and S. Rajanarayanan, Range search in parallel using distributed data structures, *International Conference on Databases, Parallel Architectures, and Their Applications*, 1990, pp. 14–19.
89. M. P. Armstrong, C. E. Pavlik, and R. Marciano, Experiments in the measurement of spatial association using a parallel supercomputer, *Geographical Systems*, **1**: 267–288, 1994.
90. G. Brunetti, A. Clematis, B. Falcidieno, A. Sanguineti, and M. Spagnuolo, Parallel processing of spatial data for terrain characterization, *Proc. ACM Geographic Information Systems*, 1994.
91. W. R. Franklin, C. Narayanaswami, M. Kankanahalli, D. Sun, M. Zhou, and P. Y. F. Wu, Uniform grids: A technique for intersection detection on serial and parallel machines, *Proc. 9th Automated Cartography*, 1989, pp. 100–109.
92. E. G. Hoel and H. Samet, Data Parallel RTree Algorithms, *Proc. International Conference on Parallel Processing*, 1993.
93. E. G. Hoel and H. Samet, Performance of dataparallel spatial operations, *Proc. of the 20th International Conference on Very Large Data Bases*, 1994, pp. 156–167.
94. V. Kumar, A. Grama, and V. N. Rao, Scalable load balancing techniques for parallel computers, *J. Parallel and Distributed Computing*, **22**(1): 60–69, July 1994.
95. F. Wang, A parallel intersection algorithm for vector polygon overlay, *IEEE Computer Graphics and Applications*, **13**(2): 74–81, 1993.
96. Y. Zhou, S. Shekhar, and M. Coyle, Disk allocation methods for parallelizing grid files, *Proc. of the Tenth International Conference on Data Engineering, IEEE*, 1994, pp. 243–252.
97. S. Shekhar, S. Ravada, V. Kumar, D. Chubband, and G. Turner, Declustering and load-balancing methods for parallelizing spatial databases, *IEEE Trans. Knowledge and Data Engineering*, **10**(4): 632–655, 1998.
98. M. T. Fang, R. C. T. Lee, and C. C. Chang, The idea of de-clustering and its applications, *Proc. of the International Conference on Very Large Data Bases*, 1986, pp. 181–188.
99. D. R. Liu and S. Shekhar, A similarity graph-based approach to declustering problem and its applications, *Proc. of the Eleventh International Conference on Data Engineering, IEEE*, 1995.
100. A. Belussi and C. Faloutsos. Estimating the selectivity of spatial queries using the 'correlation' fractal dimension, in *Proc. 21st International Conference on Very Large Data Bases, VLDB*, 1995, pp. 299–310.
101. Y. Theodoridis, E. Stefanakis, and T. Sellis, Cost models for join queries in spatial databases, in *Proceedings of the IEEE 14th International Conference on Data Engineering*, 1998, pp. 476–483.
102. M. Erwig, R. Hartmut Guting, M. Schneider, and M. Vazirgiannis, Spatio-temporal data types: An approach to modeling and querying moving objects in databases, *GeoInformatica*, **3**(3): 269–296, 1999.
103. R. H. Girtling, M. H. Bohlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, and M. Vazirgiannis, A foundation for representing and querying moving objects, *ACM Transactions on Database Systems*, **25**(1): 1–42, 2000.
104. S. Borzsonyi, D. Kossmann, and K. Stocker, The skyline operator, In *Proc. the International Conference on Data Engineering*, Heidelberg, Germany, 2001, pp. 421–430.
105. J. M. Hellerstein and M. Stonebraker, Predicate migration: Optimizing queries with expensive predicates, In *Proc. ACM-SIGMOD International Conference on Management of Data*, 1993, pp. 267–276.
106. Y. Theodoridis and T. Sellis, A model for the prediction of r-tree performance, in *Proceedings of the 15th ACM Symposium on Principles of Database Systems PODS Symposium*, ACM, 1996, pp. 161–171.
107. T. Asano, D. Ranjan, T. Roos, E. Wiezl, and P. Widmayer, Space-filling curves and their use in the design of geometric data structures, *Theoretical Computer Science*, **181**(1): 3–15, July 1997.
108. S. Shekhar and D.R. Liu, A connectivity-clustered access method for networks and network computation, *IEEE Trans. Knowledge and Data Engineering*, **9**(1): 102–119, 1997.
109. J. Lee, Y. Lee, K. Whang, and I. Song, A physical database design method for multidimensional file organization, *Information Sciences*, **120**(1): 31–65(35), November 1997.
110. J. Nievergelt, H. Hinterberger, and K. C. Sevcik, The grid file: An adaptable, symmetric multikey file structure, *ACM Transactions on Database Systems*, **9**(1): 38–71, 1984.
111. M. Ouksel, The interpolation-based grid file, *Proc. of Fourth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, 1985, pp. 20–27.
112. K. Y. Whang and R. Krishnamurthy, *Multilevel grid files*, IBM Research Laboratory Yorktown, Heights, NY, 1985.
113. A. Guttman, R-trees: A Dynamic Index Structure for Spatial Searching, *Proc. of SIGMOD International Conference on Management of Data*, 1984, pp. 47–57.
114. T. Sellis, N. Roussopoulos, and C. Faloutsos, The R<sup>+</sup>-tree: A dynamic index for multidimensional objects, *Proc. 13th International Conference on Very Large Data Bases*, September 1987, pp. 507–518.
115. N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, The R<sup>\*</sup>-tree: An Efficient and Robust Access Method for Points and Rectangles, *Proc. ACM SIGMOD International Conference on Management of Data*, 1990, pp. 322–331.

116. Y. Theodoridis, M. Vazirgiannis, and T. Sellis, Spatio-temporal indexing for large multimedia applications, *International Conference on Multimedia Computing and Systems*, 1996, pp. 441–448.
117. S. Saltenis and C.S. Jensen, R-tree based indexing of general Spatio-temporal data, *Technical Report TR-45 and Chorochronos CH-99-18, TimeCenter*, 1999.
118. M. Vazirgiannis, Y. Theodoridis, and T. Sellis, Spatio-temporal composition and indexing large multimedia applications, *Multimedia Systems*, **6**(4): 284–298, 1998.
119. M. Nascimento, R. Jefferson, J. Silva, and Y. Theodoridis, Evaluation of access structures for discretely moving points, *Proc. International Workshop on Spatio-temporal Database Management*, 1999, pp. 171–188.
120. S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez, Indexing the positions of continuously moving objects, in *SIGMOD Conference*, 2000, pp. 331–342.
121. Y. Tao, D. Papadias, and J. Sun, The TPR<sup>+</sup>-Tree: An Optimized Spatio-temporal Access Method for Predictive Queries, in *VLDB*, 2003, pp. 790–801.
122. M. F. Mokbel, T. M. Ghanem, and W. G. Aref, Spatio-temporal access methods, *IEEE Data Engineering Bulletin*, **26**(2): 40–49, 2003.
123. R. A. Finkel and J. L. Bentley, Quad trees: A data structure for retrieval on composite keys, *Acta Informatica*, **4**:1–9, 1974.
124. T. Tzouramanis, M. Vassilakopoulos, and Y. Manolopoulos, Overlapping linear quadtrees: A Spatio-temporal access method, *ACM-Geographic Information Systems*, 1998, pp. 1–7.
125. Y. Manolopoulos, E. Nardelli, A. Papadopoulos, and G. Proietti, MOF-Tree: A Spatial Access Method to Manipulate Multiple Overlapping Features, *Information Systems*, **22**(9): 465–481, 1997.
126. J. M. Hellerstein, J. F. Naughton, and A. Pfeffer, Generalized Search Trees for Database System, *Proc. 21th International Conference on Very Large Database Systems*, September 11–15 1995.
127. W. G. Aref and I. F. Ilyas, SP-GiST: An Extensible Database Index for Supporting Space Partitioning Trees, *J. Intell. Inf. Sys.*, **17**(2–3): 215–240, 2001.
128. M. Kornacker and D. Banks, High-Concurrency Locking in R-Trees, 1995.
129. M. Kornacker, C. Mohan, and Joseph M. Hellerstein, Concurrency and recovery in generalized search trees, in *SIGMOD '97: Proc. 1997 ACM SIGMOD international conference on Management of data*. New York, ACM Press, 1997, pp. 62–72.
130. S. Shekhar, P. Zhang, Y. Huang, and R. R. Vatsavai, Spatial data mining, in Hillol Kargupta and A. Joshi, (eds.), *Book Chapter in Data Mining: Next Generation Challenges and Future Directions*.
131. N. A. C. Cressie, *Statistics for Spatial Data*. New York: Wiley-Interscience, 1993.
132. R. Haining, *Spatial Data Analysis : Theory and Practice*, Cambridge University Press, 2003.
133. Y. Jhung and P. H. Swain, Bayesian Contextual Classification Based on Modified M-Estimates and Markov Random Fields, *IEEE Trans. Pattern Analysis and Machine Intelligence*, **34**(1): 67–75, 1996.
134. A. H. Solberg, T. Taxt, and A. K. Jain, A Markov Random Field Model for Classification of Multisource Satellite Imagery, *IEEE Trans. Geoscience and Remote Sensing*, **34**(1): 100–113, 1996.
135. D.A. Griffith, *Advanced Spatial Statistics*. Kluwer Academic Publishers, 1998.
136. J. LeSage, *Spatial Econometrics*. Available: <http://www.spatial-econometrics.com/>, 1998.
137. S. Shekhar, P. Schrater, R. Raju, and W. Wu, Spatial contextual classification and prediction models for mining geospatial data, *IEEE Trans. Multimedia*, **4**(2): 174–188, 2002.
138. L. Anselin, *Spatial Econometrics: methods and models*, Dordrecht, Netherlands: Kluwer, 1988.
139. S.Z. Li, A Markov Random Field Modeling, *Computer Vision*. Springer Verlag, 1995.
140. S. Shekhar et al. Spatial Contextual Classification and Prediction Models for Mining Geospatial Data, *IEEE Transaction on Multimedia*, **4**(2), 2002.
141. N. A. Cressie, *Statistics for Spatial Data (revised edition)*. New York: Wiley, 1993.
142. J. Han, M. Kamber, and A. Tung, *Spatial Clustering Methods in Data Mining: A Survey, Geographic Data Mining and Knowledge Discovery*. Taylor and Francis, 2001.
143. V. Barnett and T. Lewis, *Outliers in Statistical Data*, 3rd ed. New York: John Wiley, 1994.
144. A. Luc, Local Indicators of Spatial Association: LISA, *Geographical Analysis*, **27**(2): 93–115, 1995.
145. S. Shekhar, C.-T. Lu, and P. Zhang, A unified approach to detecting spatial outliers, *GeoInformatica*, **7**(2), 2003.
146. C.-T. Lu, D. Chen, and Y. Kou, Algorithms for Spatial Outlier Detection, *IEEE International Conference on Data Mining*, 2003.
147. R. Agrawal and R. Srikant, Fast algorithms for mining association rules, in J. B. Bocca, M. Jarke, and C. Zaniolo (eds.), *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, Morgan Kaufmann, 1994, pp. 487–499.
148. Y. Morimoto, Mining Frequent Neighboring Class Sets in Spatial Databases, in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001.
149. S. Shekhar and Y. Huang, Co-location Rules Mining: A Summary of Results, *Proc. Symposium on Spatial and Spatio-temporal Databases*, 2001.
150. Y. Huang, S. Shekhar, and H. Xiong, Discovering Co-location Patterns from Spatial Datasets: A General Approach, *IEEE Trans. Knowledge and Data Engineering*, **16**(12): 1472–1485, December 2005.
151. J. F. Roddick and B. G. Lees, *Paradigms for spatial and Spatio-temporal data mining*. Taylor and Frances, 2001.
152. M. Koubarakis, T. K. Sellis, A. U. Frank, S. Grumbach, R. Hartmut Guting, C. S. Jensen, N. A. Lorentzos, Y. Manolopoulos, E. Nardelli, B. Pernici, H.-J. Schek, M. Scholl, B. Theodoulidis, and N. Tryfona, eds, *Spatio-temporal Databases: The CHOROCHRONOS Approach*, Vol. **2520** of *Lecture Notes in Computer Science*. Springer, 2003.
153. J. F. Roddick, K. Hornsby, and M. Spiliopoulou, An updated bibliography of temporal, spatial, and Spatio-temporal data mining research, *Proc. First International Workshop on Temporal, Spatial and Spatio-temporal Data Mining*, 2001, pp. 147–164.
154. M. Celik, S. Shekhar, J. P. Rogers, and J. A. Shine, Sustained emerging Spatio-temporal co-occurrence pattern mining: A summary of results, *18th IEEE International Conference on Tools with Artificial Intelligence*, 2006, pp. 106–115.
155. M. Celik, S. Shekhar, J. P. Rogers, J. A. Shine, and J. S. Yoo, Mixed-drove Spatio-temporal co-occurrence pattern mining: A summary of results, *Sixth International Conference on Data Mining, IEEE*, 2006, pp. 119–128.

156. M. Celik, S. Shekhar, J. P. Rogers, J. A. Shine, and J. M. Kang, Mining at most top-k mixed-drove Spatio-temporal co-occurrence patterns: A summary of results, in *Proc. of the Workshop on Spatio-temporal Data Mining (In conjunction with ICDE 2007)*, 2008, forthcoming.
157. J. F. Roddick, E. Hoel, M. J. Egenhofer, D. Papadias, and B. Salzberg, Spatial, temporal and Spatio-temporal databases - hot issues and directions for phd research, *SIGMOD record*, **33**(2), 2004.
158. O. Schabenberger and C. A. Gotway, *Statistical Methods for Spatial Data Analysis*. Chapman & Hall/CRC, 2004.
159. M. Stonebraker, J. Frew, K. Gardels, and J. Meredith, The Sequoia 2000 Benchmark, in Peter Buneman and Sushil Jajodia (eds.), *Proc. 1993 ACM SIGMOD International Conference on Management of Data*. ACM Press, 1993. pp. 2–11.
160. J. M. Patel, J.-B. Yu, N. Kabra, K. Tufte, B. Nag, J. Burger, N. E. Hall, K. Ramasamy, R. Lueder, C. Ellmann, J. Kupsch, S. Guo, D. J. DeWitt, and J. F. Naughton, Building a scaleable geo-spatial dbms: Technology, implementation, and evaluation, in *ACM SIGMOD Conference*, 1997, pp. 336–347.

VIJAY GANDHI  
JAMES M. KANG  
SHASHI SHEKHAR  
University of Minnesota  
Minneapolis, Minnesota