Research Issues in Spatial Databases

O. Guenther
FAW Ulm
Postfach 2060
D7900 Ulm, Germany

A. Buchmann
GTE Laboratories, Incorporated
40 Sylvan Road
Waltham, MA 02254, USA

1. Introduction

With a recent emergence of applications that rely heavily on spatial data, the database research community is currently devoting considerable attention to spatial database issues. While the main thrust came initially from the geosciences and mechanical CAD, the range of possible applications has expanded to areas such as robotics, visual perception and autonomous navigation, tracking, environmental protection, and medical imaging.

Just as broad as the range of applications is the range of interpretations given to the term "spatial data management." One interpretation comes from mechanical CAD and its need for processing 3-dimensional solids, whereas VLSI CAD and cartography are typical for applications that rely mostly on 2D- or layered 2D-data. Other applications emphasize the processing of unanalyzed images, such as X-rays and satellite images from which features are extracted.

The terms spatial database and image database are often used interchangeably. Strictly speaking, however, spatial databases contain n-dimensional data with explicit knowledge about objects, their extent and position in space. The relative position of objects may be implicit (derived from the internal representation of their absolute positions) or explicit. Image databases, on the other hand, often place less emphasis on the need for analyzing the data and provide storage and retrieval for unanalyzed pictorial data. Techniques developed for the storage and manipulation of image data can be applied to more general signals as well, such as infrared sensor signals and sound. In the remainder of this paper we will assume that the goal is to manipulate analyzed spatial data, and that unanalyzed images are only handled as the source from which spatial data can be derived. The operations performed are often domain-specific, and some of the features we have come to associate with a full-fledged DBMS, such as transaction management, may be less important in some applications. However, the handling of multiple spatial representations and data models, spatial access methods, pictorial/spatial query languages, and optimization form a significant subset of today's database research.

The challenge for the developers of DBMSs with spatial capabilities lies not so much in providing yet another special-purpose data structure that is marginally faster when used in a particular application, but in defining abstractions and architectures to implement systems that offer generic spatial data management capabilities and that can be tailored to the requirements of a particular domain.

In this paper we do not attempt a comprehensive review. Instead, we will summarize application requirements, identify areas in which considerable progress has been made, and discuss several problems that require a stronger emphasis.

2. Application Requirements

The spatial data management needs differ widely between applications. Mechanical CAD is perhaps the application with the most demanding geometrical requirements. Early representations consisted of line drawings of the three views of a solid. Cell decompositions, in which solids are decomposed into simpler polyhedric cells (such as convex polytopes or simplices), are best suited

for finite element methods. Constructive solid geometry (CSG), which represents solids as compositions of primitive components and regularized set operations, is best for composing regular objects and deriving other properties. Boundary representations, which map a solid's surface into patches that are represented via edges and vertices, are best suited for modelling irregular surfaces. Because no single approach satisfies all requirements, the same object is often represented in multiple forms. In addition to the representation and manipulation of the spatial data, mechanical CAD applications often require its association with descriptive properties, such as material and weight (which may itself be derived from volume and density data). Database retrievals and updates are frequent, but often performed in individual workspaces.

VLSI CAD has received most attention in the database community, both because it is easily accessible to computer scientists and because its geometric requirements are relatively simple and tractable. Most objects are represented sufficiently well by rectilinear polylines or polygons whose edges are all parallel to the axes of the same coordinate system. This makes for regular boundaries and clean intersection of the geometric shapes, all properties that help when indexing on an object's spatial properties.

Geographic/cartographic applications are characterized by massive volumes of data, both spatial and non-spatial, as well as the need to record their evolution in time. Spatial data are mostly two-dimensional with points, lines, and polygons as the basic primitives, and are characterized by extreme irregularity. The third dimension is usually reduced through well-established conventions for the representation of elevation. Cartographic applications require the ability to overlay maps, extract common features and make independent data sources compatible. Feature extraction from aerial photographs and spatial images is common. The operations most often performed are overlay, containment, distance operations, and spatial searches. The sheer volume of geographic data prompted early application of database technology. However, the integration of spatial and non-spatial data in a common database approach, although desirable, has been limited.

Vision systems have been evolving in the areas of robotics and manufacturing, as well as autonomous navigation [Good89, Brol89]. The general problem is one of identifying and determining the position of a 3D object based on 2D images. Vision systems are driven by tight time constraints, and generally depend on the extraction of relevant features, such as descriptions of junctions, that are stored in structures used for fast matching. Features are often extracted from the underlying CAD databases to produce models of the objects off-line.

3. Research Areas in Spatial Data Management

The spatial properties of objects have to be captured both at the conceptual level, and at the physical data structure level. How tightly these two are coupled will determine the flexibility (or lack thereof) in a spatial database system. We will analyze the issues by looking first at the representations of spatial data and working from there outward to the conceptual level, query languages, optimization, and problems derived from intrinsic spatial properties, such as precision.

3.1 Representations

The goal of a spatial data structure is to provide a scheme for partitioning the entire space into cells, and to provide a two-way mapping between these cells and the region in space occupied by an object [Niev89]. A variety of representations have been proposed; for a review see [Nagy79, Requ80, Gunt88, Same89]. The simplest approach is a raster representation in which the spatial extent of an object is represented by pixels. 2D objects can also be represented by polygons made up of lines (so-called vector representation). The equivalent for 3D is the representation of a solid through surface patches. Solids can also be composed of primitive objects and a specification of their relationships, which is the basic representation in CSG systems. Each of these representations supports a certain class of spatial operators, such as intersection, containment, or the set

operations. However, some operations are not defined for certain representations. For example, outline and endpoints of a line are not defined for a pixel representation. Operations are also performed with varying efficiency in different representations. For example, raster representations support intersection well but vector representations are more efficient for distance operations. When choosing a data structure it is therefore important to take into account what operations are to be supported primarily.

However, no matter how carefully one selects a representation, operations across representations will be required. This is even more so as systems with different underlying representations will be integrated. A variety of open research problems have hindered data exchange and integration of spatial databases, and has confined spatial databases to the use of a single representation.

For most of the representations the specification of the spatial operators is incomplete. No comprehensive list of operators exists, no minimal sets of operators have been identified, and the semantics of many operators are ill-defined. Presently, no known generalizations exist for spatial operations across representations. For each of the representations an algebra is required and a "superalgebra" for cross-representation operations has to be defined if the user is to be shielded from the physical representation and if any kind of optimization across representations is to be performed. The need for representation-independent abstractions is discussed below.

3.2 Spatial Access Methods

A typical spatial database query may ask for all objects that contain a given point (point search) or that overlap a given search space (range search). In all of these cases, it is necessary to retrieve from the database those objects that occupy a given location in space. To support such search operations, one needs to use spatial index structures that enable the user to access efficiently the objects in a certain spatial neighbourhood.

A large variety of such structures have been proposed, both hash-based and tree-based; see [Gunt88] or [Ooi88] for a survey. However, due to a lack of comparative studies it is not clear yet which structure has the best expected performance for a given application, and how robust each is. It is an important issue for future research to classify spatial search applications according to a small set of parameters (such as database size, average object size, or object distribution), and to determine the most promising structures for a given set of parameters. For initial experimental results see [Ooi88, Free90, Krie90, Gunt90, Smit90]; no analytical results are known to us.

Unfortunately, many spatial access methods are rather inefficient if the database contains objects of varying shapes and sizes [Six88]. In geographic applications, for example, where a spatial database may contain geometric objects representing geographic entities such as states, roads, and buildings, this problem may have a significant impact on average performance. The reasons for these inefficiencies depend on the particular properties of the various index structures. Most of these structures have originally been designed to manage point data only, that is, to provide efficient spatial access to large sets of points. Usually, the points in the database are organized in a number of buckets, each of which corresponds to some partition of the data space. The buckets are then accessed by means of a search tree or some hashing scheme.

Problems arise when these point access structures are modified to manage extended spatial objects such as polygons. As discussed in [Seeg88], there are three basic methods: clipping, where objects are divided along the partitioning lines of the underlying access structure; overlapping regions, where each partition of the access structure may contain any object it overlaps (rather than just those objects it encloses); and transformation, where extended spatial objects are mapped into higher-dimensional points.

Transformation only works for relatively simple objects, such as rectangles. More complicated entities, such as arbitrary polygons (as they typically occur in geographic applications), can only be represented by means of their bounding boxes [Niev84]. However, this principle has led to the development of spatial filters, which return approximate (conservative) results which can then be refined by more precise searches based on the actual geometry [Oren86].

Overlapping region schemes tend to experience major performance problems when the spatial database contains objects whose size is large relative to the total size of the data space. Each insertion of a new data object may increase the overlap. Eventually, the overlap between regions may become large enough to render the index inefficient: one ends up searching a significant portion of the whole index for a single point query. A well-known example where this behavior has been observed is the R-tree [Gutt84; Gree89].

In the case of clipping schemes, the problems with large objects are of a somewhat different nature. During insertion, each data object is divided along the partitioning hyperplanes of the access structure. This fragmentation effect gets worse as the spatial database and its index continue to become more populated. The net result of this fragmentation effect is not only an increase in the average search time but also an increase in the frequency of node overflows, which in turn leads to an increase in fragmentation, and so on [Gunt89]. Such an endless loop may cause the whole index to collapse and therefore has to be avoided at all cost.

It is evident that ever more specialized access methods can be devised. While these may perform well in a narrow application with "well-behaved" data, it is more important in a database environment to develop a sound understanding of the behavior of a few robust access methods under varying conditions, and to develop performance metrics that can be employed by an optimizer. It is important to realize that spatial indexes and filters can be constructed on most underlying representations, and can be substituted at a moderate cost. However, changing the representation of the data is a major undertaking and is often coupled with a loss of information or the introduction of imprecisions or inconsistencies.

3.3 Conceptual Models and Architectural Considerations

The objects handled in spatial database systems are usually complex. Some objects may be atomic (e.g., a building in a city map) while others may have a molecular structure (e.g., a city that is made up of districts, which consist of streets and buildings). For the representation of structured objects, one may resort to semantic networks [Luge89, Ch. 9], or object-oriented techniques [Luge89, Ch. 14; Ditt88; Oost89]. Depending on the scale at which one views such an object, its components may or may not be visible. Associated with each object is a geometric entity representing its spatial extension. For the lowest level objects this is usually a point, a line, or a polygon. The spatial extent of more complex objects may be explicit or derived from that of the component objects, for example, by inferring their geometry using cartographic generalization techniques [McMa87; McMa88]. In many cases, however, this is not a practical solution, since derivation algorithms do not consistently produce "nice" geometric representations. The utilization of object-oriented techniques for spatial data management is an active research topic, but the application of inheritance has not been researched thoroughly and deserves further attention.

The fundamental question is how to embed the spatial aspects in a data model and the underlying DBMS such that acceptable interfaces (optimizable query languages and pictorial interfaces) can be defined.

Early attempts decomposed the spatial data and forced them into relations, which resulted in unacceptable performance [Kemp87]. Other systems implement the spatial operations in a "hard-wired" form using conventional DBMS technology for non-spatial data and special-purpose file

structures for the spatial data [Moor85]. A more general approach consists in providing abstract data types (ADTs) to customize the DBMS. Examples are INGRES [Ston84] and PSQL [Rous88]. Both approaches, hardwiring and ADTs, leave the spatial semantics outside the DBMS's query processor. It is desirable for the DBMS to perform set-oriented queries on abstract objects that are representation independent, leaving the detailed spatial operations to be performed by specialized object classes that understand a specific representation.

To formulate abstract operations on spatial objects it is desirable to define a structure that is independent of any underlying representation. The notion of point sets [Mant83, Mano86] serves this purpose. It is a specialization of type "object" that acts as a placeholder and introduces a collection of spatial operators, such as union, intersection, and difference, as well as spatial predicates, such as overlap, containment, and proximity. It is similar to the abstraction of integers and floating point numbers on which operations can be defined independently of the particular representation of these numbers in a computer. The type Point-set can be further specialized, for example, creating subtypes for 2D and 3D. Each of these can, in turn, be specialized by defining representation-specific point-sets. The operations (or methods) defined on these lower level types are the implementations of operations such as intersection or overlap for a particular representation, for example, boundary representation. Operations performed on spatial objects of different representations are defined as methods of more general types.

While point sets are a good abstraction for expressing relationships of equality, inclusion, and intersection, they lack the power for describing relationships such as *neighborhood* in the topological sense. This has led to the definition of data structures for topological relationships based on simplices, and the formulation of theories of topological relationships [Egen89].

Further research is required in the specification of representation-independent spatial abstractions and their operations as a basis for seamless integration of spatial and non-spatial data in query languages. These issues will be addressed further in Section 3.5.

3.4 Spatial Database Languages

The goal of a spatial query language is to allow the easy formulation of queries that involve both spatial and non-spatial predicates without loss of spatial semantics. The query should be optimizable by a DBMS query optimizer. While the former has been accomplished in several ways, the latter, optimizability, has been elusive.

Most spatial query languages are extensions of a relational query language, either QUEL (e.g.GEO-QUEL [Go75], QBE (e.g. Query-by-Pictorial-Example [Chan80] and GEOBASE [Barr81] which also incorporates inheritance and aggregation), or SQL (e.g. PSQL [Rous88] and Extended SQL [Herr88]). A good review and comparison can be found in [Egen89]. The widespread use of SQL for non-spatial data has made SQL extensions popular. Recently, a variety of object-oriented languages have been proposed (e.g. [Oost89], [Moha88]). They are able to represent the semantics of complex objects better and encapsulate specialized operations. A fully pictorial approach to constraint definition is taken in [Piza89].

As pointed out in [Egen89], all existing languages use a diverse set of spatial relationships (containment, adjacency, distance, overlay, etc.). In most cases the semantics of these spatial relationships have not been rigorously defined, and would benefit from a mathematical grounding.

Depending on the method chosen for embedding spatial characteristics in a data model and the underlying DBMS, the query languages may be of varying complexity. For example, if one chooses ADTs to represent the objects, then the internal structure of the object representation is hidden to the database system (encapsulation). In this case, it is sufficient to enhance the query

language by some simple constructs that allow the embedding of user-defined data types and operators. However, the performance characteristics of the ADTs are unknown to the optimizer.

On the other hand, it is possible to use object representations that allow the user to access the structure of an object explicitly, for example, by embedding in non-first normal form relations [Sche82]. In this case, the database contains information about the hierarchical structure of a given object and the query language may have to be extended significantly in order to access this information efficiently [Kemp87].

In systems that use an object-oriented approach, user-defined object classes and encapsulation of their behavior make it difficult to optimize queries. The optimization of object-oriented query languages and the definition of optimizable object algebras is an active research topic. Once the generic problem is solved, it will be possible to apply those solutions to spatial query languages.

3.5 Optimization of Spatial Operations

Queries that contain both spatial and non-spatial attributes may require optimization algorithms that are significantly different from those that have been used for conventional business databases. Existing spatial data management systems (many of them prototypes) have basically ignored the optimization issues, not only because they often have only one underlying representation and one spatial index, but also because little is known about the cost of executing alternate sequences of spatial operations.

Research is required to establish cost estimates for spatial operations. Many optimal algorithms for spatial operations that are derived from computational geometry have not matured enough in practice; they often optimize for the worst case and require very complex data structures. Suboptimal algorithms that perform adequately in the majority of cases, and for which detailed cost functions are known, may be preferable.

The use of spatial filters appears to be a reasonable optimization strategy. Filters that are based on space-filling curves, such as Z-order encoding, allow for a two-phased optimization strategy. The filtering phase returns an approximate answer that is conservative and includes all the objects to be retrieved, and possibly more. Filtering reduces the set of candidate objects on which detailed operations have to be performed. The second phase then uses the specific spatial operations discussed above.

Much more research is required in this direction; for a more detailed discussion of related problems see [Oren89, Kemp87, Ooi88].

3.6 Scale and Precision

In spatial applications the spatial extent of an object has a limited precision. Handling spatial objects with varying precision is difficult at best. Errors in spatial data representations can propagate leading to topological inconsistencies (known also as wandering points). To avoid the problem of points contained within a region at one scale to appear outside the region at another scale, and false intersections, it is necessary to introduce additional topological structures. This property of spatial objects affects also the algebraic operations across representations that were discussed before. A good discussion of operation robustness can be found in [Hoff89, Ch. 4]. Furthermore, the management of uncertain information is a major issue in current artificial intelligence research [Bouc88]. At this point it is not clear which of these techniques could be applied to spatial data as well.

There has been a trend in spatial database research towards so-called scaleless maps, that is, digital maps where each entity is represented only at the largest available scale to minimize redundancy.

Given a query, one first selects the qualifying objects via some access method before scaling them down to the specified target scale. In order to evaluate such queries more efficiently, it would be advantageous to have access methods that take the target scale into account from the beginning. Ideally, there should be a positive correlation between the target scale on the one hand and the response time on the other hand for a given query, that is, one could receive a coarse (small-scale) answer to a query rather quickly, whereas it would take longer to receive a more accurate response. This approach has an effect similar to filters, and has only recently been studied [Beck90].

4. Conclusions

There is a steadily increasing demand for spatial data management capabilities, and a growing volume of spatial data. The techniques developed for spatial data management have been driven by the applications and are, therefore, often case-specific. The database community can contribute to the generalization of these techniques. So far it has concentrated on spatial access methods and on providing a spatial veneer on top of existing query languages. For database technology to have more impact on spatial data management, we need to address the problems of representation-independent spatial operations, their mapping into representation-specific operations, cross-representational algebras, and optimization of spatial queries. These issues have to be addressed if we want to exploit spatial data repositories fully, and eventually integrate existing spatial repositories.

Acknowledgements

We gratefully acknowledge interesting discussions with Renato Barrera, Frank Manola, Jack Orenstein, and Peter Widmayer.

References

[Barr81] R.Barrera, A.Buchmann, "Schema Definition and Query Language for a Geographical Database System", IEEE Trans. on Computer Architecture: Pattern Analysis and Image Database Management, Nov. 1981.

[Beck90] B. Becker, P. Widmayer, "Spatial Priority Search: An Access Technique for Scaleless Maps," Manuscript, University of Freiburg, 1990.

[Bouc88] B. Bouchon, L. Saitta, R. Yager (eds.), "Uncertainty and Intelligent Systems," Springer Lecture Notes in Computer Science, 1988.

[Brol89] J. Brolio, B.A.Draper, J.R.Beveridge, A.R.Hanson, "ISR: A Database for Symbolic Processing in Computer Vision", IEEE Computer, 22, 12, Dec. 1989.

[Buch90] A. Buchmann, O. Gunther, T. R. Smith, Y.-F. Wang (eds.), "Design and Implementation of Large Spatial Databases," Proceedings SSD'89, Lecture Notes in Computer Science No. 409, Springer-Verlag, 1990.

[Chan80] N.S.Chang, K.S.Fu, "Query-by-Pictorial-Example", IEEE Trans. on Software Engineering, 6, 6, Nov. 1980.

[Ditt88] K. R. Dittrich (ed.), "Advances in object-oriented database systems," Lecture Notes in Computer Science No. 334, Springer-Verlag, 1988.

[Egen89] M.Egenhofer, "Spatial Query Languages", PhD Thesis, U. of Maine, Orono, April 1989.

[Free90] M. Freeston, "A well-behaved structure for the storage of geometric objects," in [Buch90].

[Go75] A.Go et al, "An Approach to implementing a Geo-Data System", TR ERL-M529, University of California, Berk;ey, June 1975.

[Good89] A.M.Goodman, R.M.Haralick, L.G.Shapiro, "Knowledge-Based Computer Vision," IEEE Computer, 22, 12, Dec. 1989.

[Gree89] D. Greene, "An implementation and performance analysis of spatial data access methods," Proc. IEEE 5th Int. Conf. on Data Engineering (Los Angeles), 606-615, 1989.

[Gunt88] O. Gunther, "Efficient Structures for Geometric Data Management," Lecture Notes in Computer Science No. 337, Springer-Verlag, 1988.

[Gunt89] O. Gunther, "The design of the cell tree: an object-oriented index structure for geo-metric databases," Proc. IEEE 5th Int. Conf. on Data Engineering (Los Angeles), 598-605, 1989.

[Gunt90] O. Gunther and J. Bilmes, "Tree-based access methods for spatial databases: implementation and performance evaluation." To appear in IEEE Transactions on Knowledge and Data Engineering, 1990.

[Gutt84] A. Guttman, "R-trees: Dynamic index structure for spatial searching," Proc. of ACM SIGMOD Conference on Management of Data, Boston, Ma., 1984.
[Herr88] J. Herring et al, "Extensions to the SQL Language to Support Spatial Analysis in a Topological Data Base", in GIS/LIS'88, San Antonio, TX, Nov. 1988.
[Hoff89] C.M. Hoffmann, "Geometric and Solid Modelling: An Introduction", Chap 4., Morgan Kaufmann Publishers, Inc., 1989.
[Iyen88] S.S.Iyengar, R.I.Kashyap, (eds.) special issue on Image Databases, IEEE Trans. on Software Engineering, 14, 5, May 1988.
[Kemp87] A. Kemper and M. Wallrath, "An analysis of geometric modelling in database systems," ACM Comp. Surveys 19, 1, pp. 47-91, 1987.
[Krie90] H. P. Kriegel et al., "A performance comparison of point and spatial access methods," in [Buch90].
[Luge89] G. F. Luger, and W. A. Stubblefield, "Artificial intelligence and the design of expert systems," Benjamin/Cummings, 1989.
[Mano86] F. Manola, J.A.Orenstein, "Toward a General Spatial Data Model for an Object-Orienetd DBMS", Proc. 12th Intl. Conf. on Very Large Data Bases, Kyoto, 1986.
[Mant83] M.Mantyla, M.Tamminen, "Localized set operations for solid modeling", Computer Graphics, 17, 3, 1983, (279-288).
[McMa87] R. B. McMaster, "Automated Line Generalization," Cartographica 24, 2, pp. 74-111, 1987.
[McMa88] R. B. McMaster, "Cartographic Generalization in a Digital Environment: A Framework for Implementation in Geographic Information System," Proc. GIS/LIS Conference, pp. 240-255, 1988.
[Moha88] L.Mohan, R.L.Kashyap, "An Object-Oriented Knowledge Representation for Spatial Information", in [Iyen88]
[Moor85] S.Moorehouse, "ARC/INFO: A Geo-Relational Model for Spatial Information", Proc. Seventh Intl. Symp. on Computer Assisted Cartography, ACSM.
[Nagy79] G.Nagy, S.Wagle, "Geographic Data Processing", ACM Computing Surveys, 11,2, June 1979.
[Niev84] J. Nievergelt, H. Hinterberger, and K. C. Sevcik, "The grid file: an adaptable, symmetric multikey file structure," ACM Trans. on Database Systems 9, 1, pp. 38-71, 1984.
[Niev89] J. Nievergelt, "7+-2 criteria for assessing and comparing spatial data structures", in [Buch90].
[NSF90] M. Brodie et al., "Database Systems: Achievements and Opportunities," Report of the NSF Invitational Workshop on Future Directions in DBMS Research, June 1990.
[Ooi88] B. C. Ooi, "Efficient query processing in a geographic information system," Ph. D. Dissertation, Dep. of Computer Science, Monash University, 1988.
[Oost89] P. v. Oosterom and J. v. d. Bos, "An object-oriented approach to the design of geographic information systems," Computers and Graphics, Vol. 13, No. 4, 1989.
[Oren86] J. A. Orenstein, "Query processing in an object-oriented database system," Proc. of ACM SIGMOD Conference on Management of Data, 1986.
[Oren89] J. A. Orenstein, "Redundancy in spatial databases," Proc. of ACM SIGMOD Conference on Management of Data, 1989.
[Piza89] A.Pizano, A.Klinger, A.Cardenas, "Specification of Spatial Integrity Constraints in Pictorial Databases", IEEE Computer, 22, 12, Dec. 1989.
[Requ80] A. A. G. Requicha, "Representations for rigid solids: theory, methods, and systems," ACM Computing Surveys 12,4, 1980.
[Rous88] N.Roussopoulos, C.Faloutsos, T.Sellis, "An Efficient Pictorial Database System for PSQL", in [Iyen88]
[Same89] H.Samet, "The design and analysis of spatial data structures," Addison-Wesley, 1989.
[Sche82] H.J. Schek, P. Pistor, "Data Structures for an Integrated Data Base Management and Information Retrieval System", Proc. 8th Intl. Conf. on Very Large Data Bases, Mexico City, 1982.
[Seeg88] B. Seeger and H. P. Kriegel, "Techniques for design and implementation of efficient spatial access methods," Proc 14th Int. Conf. on Very Large Databases (Los Angeles), 1988.
[Six88] H.-W. Six and P. Widmayer, "Spatial searching in geometric databases," Proc. IEEE 4th Int. Conf. on Data Engineering (Los Angeles), 496-503, 1988.
[Smit87] T. R. Smith, D. Peuquet, S. Menon, and P. Agarwal, "KBGIS-II - A Knowledge-Based Geographical Information System," Int. J. Geographical Information Systems, 1, 2, 1987.
[Smit90] T. R. Smith and P. Gao, "Experimental Performance Evaluations on Spatial Access Methods," Proc. 4th Int. Symp. on Spatial Data Handling, Zurich 1990.
[Ston84] M. Stonebraker, E. Anderson, E. Hanson, and B. Rubenstein, QUEL as a Data Type, Proc. of ACM SIGMOD Conference on Management of Data, June 1984.