



## A Review on Spatial Access Methods

Arun Yadav<sup>1</sup>, Divakar Yadav<sup>2</sup>

<sup>1</sup>Department of Computer Science and Engineering  
Ajay Kumar Garg Engineering College  
Ghaziabad(U.P) INDIA

<sup>2</sup>Faculty of Computer Science and Engineering  
Jaypee Institute of Information Technology  
Gautam Budh Nagar, Noida(U.P) INDIA

**Abstract:** Geographical information system and information retrieval has been very interesting research field since last two decades. Geographical Information Retrieval (GIR) is new direction of these two fields. A better search method provides better result to the end user. In this paper, we present a survey of different aspects of spatial access methods on the basis of query type and region type for spatial search. At the end of this paper, we conclude the relative performance of different access methods with their pros and cons.

**Keywords:** spatial access; Geographical information retrieval; information retrieval.

### I. Introduction

Geographical Information Retrieval (GIR) is a very important field of information retrieval, which deals with representation, storage, organization, and access of geographical object. As we know that world is totally unstructured in terms of geography and to store geographical references in the structural form is very complex task. Therefore, we require suitable spatial access method for accessing spatial object. SPIRIT [1,2,3] provide very important and useful background for querying and indexing of spatial data.

Different access methods for different types of queries are defined in [4,5]. According to [1], the queries can be divided into four categories:

- (1) Distance
- (2) Topological
- (3) Directional
- (4) Impression Regions

We require better spatial access method to solve above queries. To identify better special access method for a good GIR system, we require comparative analysis of spatial access methods according to their pros and cons.

According to [2], a common request for spatial database system is to meet the following user demand.

- (i) What is present (attribute data(text))?
- (ii) Where it is (spatial data)?

To make these demands fulfilled specific conceptual and technological models are required and are known as spatial access methods. A query for spatial search is represented as:

$Q = T \sqcup S = \{T, T_2, T_m\} \sqcup \{S_1, S_2\}$  where  $Q$ =query,  $T$ =textual part of query and  $S$ =spatial part of query.

The paper is organized as follows: In Sec. 2, we discuss different types of multidimensional access method for spatial search. In Sec. 3, we discuss space-filling curve for point data. In Sec. 4, we discuss study of spatial access method with respect to query window and geographical structure for particular geographic region. Finally, in Sec. 5 conclusion along with future aspects are discussed.

### II. Spatial Data Structure

Spatial access methods are generally suited under the category of the multidimensional access method. Our main focus is to study the different access methods and define which one is best for designing spatial index for spatial search. In [4,5], good survey for multidimensional access methods are being presented. According to [4], SpatialAccess Method (SAM) is divided into two groups:

**(i) Space-driven structure:** This type of SAM is based on the partition of the 2-D spaces into rectangular cells. Objects are mapped to the cells according to some geometric criterion and then a linear structure or ordering is established and used to index geometric objects.

**(ii) Data-driven structure:** These structures are organized by partitioning the set of objects as opposite to space, then partitioning adapts to the objects distribution in embedding space.

Spatial data can be divided into two parts:

- **Point Access Method**

These types of methods are used for the point query and region query. Point access method cannot solve window query and enter section query because they cannot support for spatial extend [4]. Where window query

is a query in which search operation is performed in certain range of geography and enter section query is a query in which query is performed by combining two geographical spaces. Using point access method, we cannot define the spatial relationship between objects. According to [6], any spatial access method cannot be mapped into one-dimensional because it is very tough to compare data in one-dimensional object. In this section, we discuss some point access methods. These point access methods are mapped to spatial access method (SAM). Our aim in the current work is to discuss point access methods and their comparative analysis, so that we can decide which one is useful for our requirement.

Paper[7] define that point access method is to store points in buckets in rectangular linear order and then they can indexed either flat or hierarchical data structure.

Point access method is further divided into two sub methods: multidimensional hashing access method and hierarchical access method as shown in Fig 1:

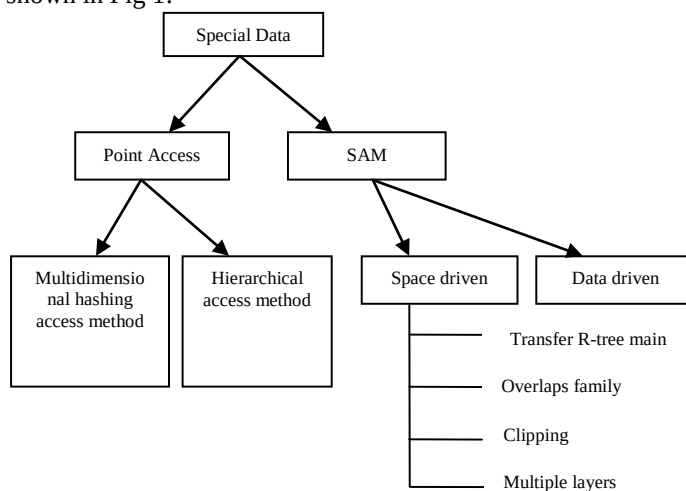


Fig. 1: Structure of Spatial Access Method

#### • Multidimensional hashing access method

Multidimensional hashing method for spatial data is defined in [6,8,12]. Paper [6] explains that point data has not any spatial relation between each other. The goal of these approaches is to put all related points together as they are close in original space. In this section, first we discuss hashing techniques, their uses and in the second part we discuss grid file approach to make hashing with closed objects together. Hashing is divided two parts:

1. Static Hashing
2. Dynamic hashing (Extended hashing)

In [7], limitation of static hashing, which suffers from bucket overflow, has been discussed. To reduce overflow, we must split the bucket and rearrange the data in buckets. But using this approach related data may be scattered in different buckets, which take more time to search.

According to [9], dynamic hashing solve the overflow problem and create different hash table for collide keys, but it takes more time to search. In the next part, we discuss dynamic hashing using grid file and its variants.

**Grid File:** Grid file superimposes an n-dimensional orthogonal grid on the universe [10,11]. Grid may be of same or different size. Each cell is associated with one bucket, but a bucket may contain more than one cell. Fig. 2 shows a grid file with cell indexing and its bucket.

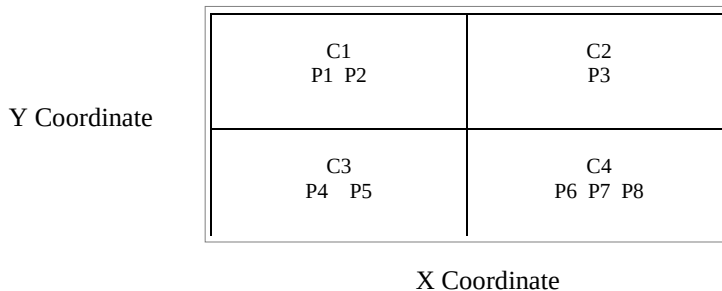


Fig. 2(a): Grid Structure

Values of cell in primary index can be defined using hash function. We are not going to discuss detail working of hash table. Our main goal is how to indexed grid cell and how we can use hash table and their importance. Cell  $C_1$  belongs to bucket A with point  $P_1$  and  $P_2$  same as cell  $C_2$ ,  $C_3$  and  $C_4$  belongs to corresponding bucket.

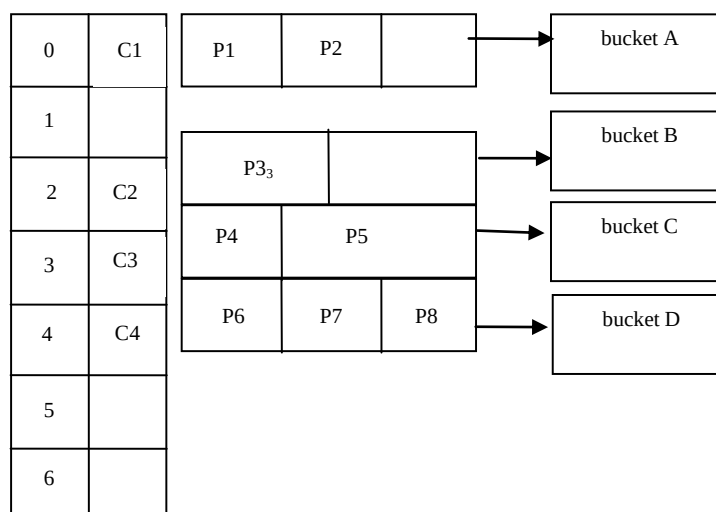


Fig. 2(b): Hashing of Grid Structure

#### • Hierarchical access method

Hierarchical access methods are used to store spatial data in the form of tree structure. In next few paragraphs, we are going to discuss some prominent hierarchical structure for spatial data.

**Quad tree:** Quad tree is very important and useful data structure for point access. It was first developed in 1974 by Finkel. Details about it can be found in [13,14,15,16].

Quad tree is more or less same as K-d-tree. Quad tree is rooted tree with four children for every node. All children are divided on the basis of regions. Regions are rectangular shape of universe and referred as NW, SW, NE, SE quadrants as shown in Fig. 3(a). The decomposition of objects goes until objects in each partition are below a given threshold. Quad tree is not necessarily balanced. Searching in Quad tree is same as binary tree. At each level, one has to decide which of the four sub tree need to be included for next search. In range queries, several paths are required to search.

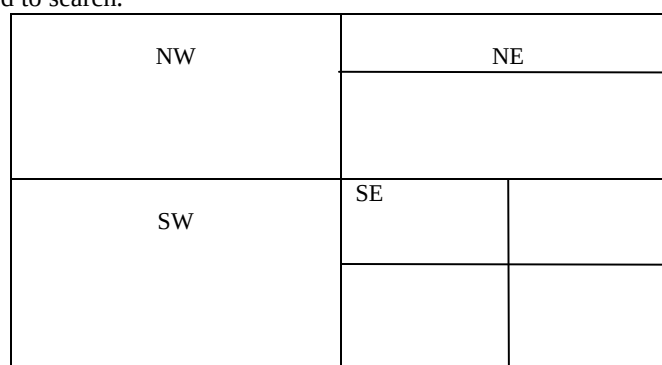


Fig. 3(a): Quad Tree structure

Region quad tree is very useful for region based point Query. The main disadvantage of quad tree is that it inserts a point at end of leaf where the search ends. Deletion of a point makes reconstruction of the tree which is very complex. In Fig. 3 (a), Quad tree is divided into four regions and every region can be divided into four other regions as shown in Fig. 3(b). Circle represents the internal nodes and rectangle represents the leaf of nodes.

**K-d-tree:** K-d-tree is multi-dimensional data structure known as k-d-tree[17]. It is a binary search tree that store points of the k-dimensional space. The direction of the hyper plane i.e. the dimension based on which the division is made, alternates between the possibilities from one tree level to the next. Each splitted hyper planes contains at least one point.

Searching and insertion is straight forward but deletion in k-d tree is complex and creates reorganization of tree. The main disadvantage of K-d tree is that it is totally unbalanced because it cannot divide the plane in equal number of points.

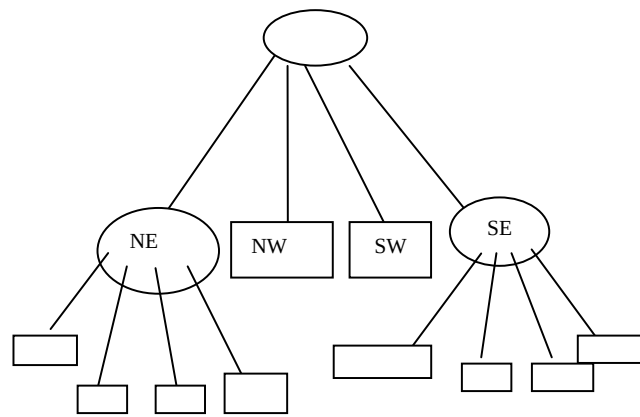


Fig. 3(b): Quad Tree

It works when data is known prior and there are rare updates in the tree. It is not suitable for spatial tree. Its next variant explained by Robinson is known as k-d-b-tree (Robinson, 1981). He has represented k-d-b tree as k-d tree + B tree. It defines equal number of points distribution in hyper plane but problem remains same for spatial data.

**BSP-Tree:** Fuchs proposed and implemented BSP tree in [18,19]. In this tree, authors define that universe is divided into number of parts according to the requirements independently. BSP-tree is same as k-d binary tree except that it is a deep search tree. Its main advantage is that it is binary tree and supports independent division of universe. The main disadvantage of BSP tree is that it allows the universe decomposition till the object becomes lower than threshold in particular decomposition. All the objects in BSP tree are defined in the leaf node. Due to large decomposition, it is converted in a high depth tree which reduces its performance.

Searching in BSP tree is time consuming because all the objects belong to leaf and it is a high depth tree. Again deletion in tree also takes long time because to delete we must search path and delete from leaf. It is an unbalanced tree. For point query, we must have to insert and delete points at leaf of nodes. Fig. 4 defines the example of BSP tree. In 4(a), universe is divided into two parts IND 1 and IND 2. IND 1 is again divided into two parts IND 1.1 (CNB, LUCK) and IND 1.2 (HPR, GZE). IND 2 is divided into IND 2.1 (DL) and IND 2.2 (PUNE, BANGL).

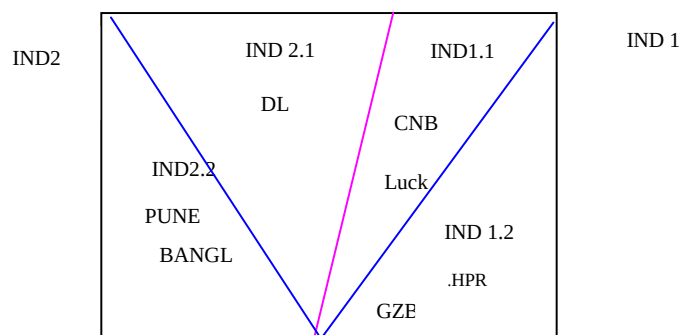


Fig. 4(a): Universe

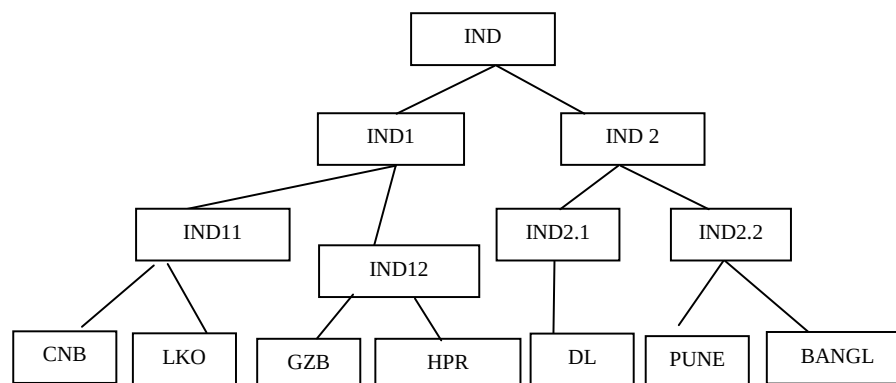


Fig. 4(b): Tree Structure of 4(a)

**Buddy Tree:** Buddy tree is one of the Hierarchical hybrid tree structure proposed by Seeger, B. and H.P. Kriegel [20]. Buddy tree is combination of dynamic hashing and k-d tree. It uses the universe decomposition using k-d-tree method and stores each level into the dynamic hashing which reduces the total tree traversal time. In Buddy tree, each directory node contains at least two entries. For dynamic hashing, there is exactly one pointer to each directory page.

The main advantage of buddy file is that it removes the deadlock problem of grid file. According to (Fuchs, 1980), buddy file is superior to other Point Access Methods (PAMS), Balanced and Nested Grid ILE (BANG) file and two level grid file. In this section, we are not going to discuss BANG file because according to (Seeger, 1990), buddy is superior to other.

**R-Tree:** In [21], an R-tree, a height balance tree and similar to B-Tree with index records in its leaf nodes has been discussed. Each leaf node contains pointers to data object nodes that belongs to disk pages. According to [21], R-trees are used to store minimum boundary rectangle (MBR) instead of storing original objects. MBR minimum boundary rectangle, which stores the spatial objects in r-dimensional rectangle. R-trees does not answer the actual query because it stores the object of the data known as topple identifier, which refer to the original position of the query answering in database.

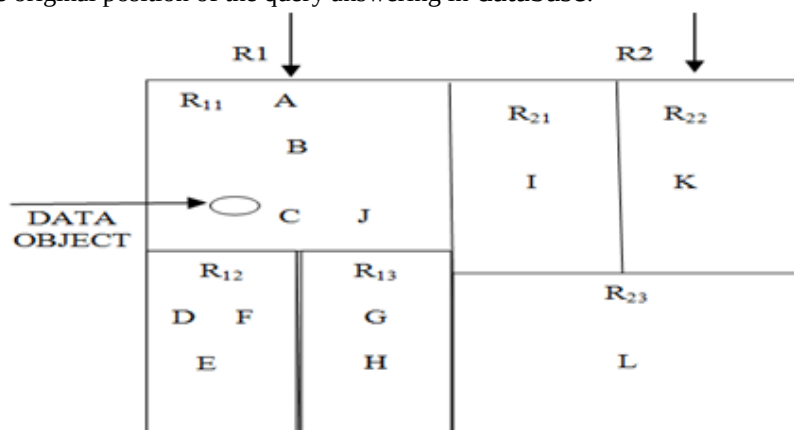


Fig. 5 (a): R-Tree Structure

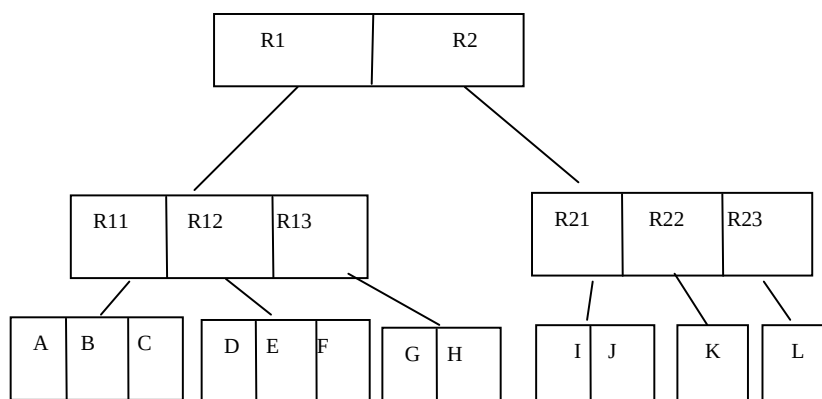


Fig. 5(b): R-Tree of Fig. 5(a)

In the Fig. 5(a), line of rectangle defines the intermediate nodes of R-tree and alphabets A-B-C define points in the object shape that belongs to leaf node. Every non leaf node contains entries of the form (I, child pointer), where I cover rectangles in the lower nodes entries e.g. (R1 covers I1 for A, I2 for B, I3 for C) and child pointer contains address of lower node in R-tree (i.e. address of A in R11). For leaf node: every node contain entries between m and M (e.g.  $M=4$  so  $m \leq 4/2=2$  (at least)) index record for each index record (I, topple-identifier) in a leaf node. I is the smallest rectangle that spatially contains n-dimensional data object (i.e. A, B, C in Fig 5(b)). Every data object is represented by a topple identifier which is unique for every object. Topple is used to store spatial object in the database. Insertion in R-tree is based on search at leaf node and find match interval. Once we reached at leaf, insert the object at leaf. Deletion is straightforward based on exact match. In case of underflow, merge the nodes and create a new node.

According to [21], R-tree provides good storage utilization for index creation and efficient searching at index level. Main drawback in R-tree is overlapping of objects in intermediate nodes so that it provides redundant data to the end users. Other variants of R-tree have been developed to reduce node overlapping. One of them is

Hebert R-tree, which store the largest Hilbert value of largest rectangle in sub tree in addition to MBR and pointer.

**R\*-Tree:** Several weaknesses of the original R-tree insertion algorithms was found on the basis of deep sticky and analysis by [22]. They observed that best insertion policy affect the searching also.

According to [22], R\*-tree has following objectives.

1. Minimize the overlap region between sibling nodes of tree.
2. Most preferable rectangle is square and storage utilization should be minimized.

The main advantage of these objectives is to minimize the paths that are traversed at an object search.

A modification of R-tree called X-tree, which is suitable for high dimensional data, has been proposed in [23]. X-tree reduces overlap by using new technique known as super nodes. To find suitable split X-tree maintain the history of previous splits.

### III. Space filling Curve for Point Data

As discussed earlier multidimensional is so difficult compared with one dimensional case. The idea is that if two objects are located closed together in original space, then there should be at least high probability that they are closed together in the total order. For organization of this total order one could then use one dimensional access method, which may provide good performance at least for some spatial query.

A good survey on space filling curve is provided in [24] which provides good relevancy for spatial searching. Conclusion of these proposals is that they first partition the universe into grid. Each of the grids is labeled with unique number in total order. Points in the given data set are then sorted and indexed according to the grid all they are contained.

Based on several assumption, experiments and analysis [25], it is concluded that Z-ordering and Hilbert curve are most suitable for multidimensional access method. Some researchers prefer Hilbert curve between these two. Advantage of space-filling curve is that they transform number of dimensions to one dimension.

The main disadvantage of space filling curve is that complex index portions cannot be joined without restricting one of the index.

### IV. Spatial Access Method

In the previous section we discussed designing and implementing point data and handling spatial query using them. As discussed earlier that they cannot support spatial relationship. To handle objects with spatial relation, point access methods have been modified using the follows approaches. According to [20] extended objects can be classified as follows.

- i. Transformation
- ii. Overlapping Regions
- iii. Clipping
- iv. Multiple Layers

**Transformation:** In this technique, we have to transform spatial objects using one dimensional point access method. According to [26], either transforms it into higher dimensions or one dimensional using space filling curves. There are its own advantages and disadvantages at different level.

As discussed in Sec. 3 space filling curve is one of the most prominent factors which is used to transform spatial object to one dimensional indexing. For higher dimensions, we can use any tree structure as discussed in Sec. 2.

**Overlapping Regions:** Overlapping region is one of the representations of spatial object extends. In the overlapping regions, we divide universe into the overlapped rectangle and allow overlapping regions at same level.

Region overlapping increases the complexity of searching objects because same query may take multiple paths for particular point data.

In Sec. 2, we discussed about R-tree and its variant which prove solution of overlapping regions.

R\*-tree and X-tree is best solution for overlapping reduction and optimized solution.

**Clipping:** Clipping is the best solution for overlapping problem because it does not allow any overlap between the nodes at the same level. Paper [27] defines R+-tree that does not allow any overlap between the nodes.

The main disadvantage in clipping method is insertion and deletion of the data objects. For insertion many data object belongs to more than one bucket region. Bucket should be split to insert objects, resulting more than one bucket entry can be created for the same object. Next problem in clipping based access method is that it does not partition the complete data space. One more problem with clipping is that, it is difficult to know that at which level bucket would be split. It may be bucket split causes no data point at hyper place. To solve these problems, paper in [28] stores large objects in spatial buckets called Oversize shelves.

**Multiple Layers:** Multilayer grid file is extension of the grid file for extended object. Multilayer grid file is also hierarchal implementation of universe. According to [29], every grid in multilayer is independent to other at each level of hierarchy. It removes overlapping problem in R-tree. To insert new object in grid file, it does not required clipping of object.

Multilayer best perform with exact match query, point and range query because it determine the sequence to hold the search interval. Multilayer grid file, superior to conventional grid file, is examined in [29]. Main

disadvantage is low storages utilization and expansive directory maintenance. In [30], a new structure which overcomes the problem of the multilayer grid file has been defined. To avoid low storage R-file uses single directory, the universe is partitioned into bang file and use z-ordering to encode the bucket regions. R-file is competitive to the R-tree and gives better result. Main drawback in R-file is that it partitions the entire space, whereas R-tree indexes the object containing search-points. Z-ordering with space filling curve provide good performance in transformation and to transform multidimensional object to one dimensional. The variant of R-tree, Hilbert R-Tree and R\*-tree perform good result in terms of spatial object searching for overlapping regions because both (R-tree and R\*-tree) reduce the problem of overlapping of objects at same level. Cell-tree with oversize shelves is better than R+-tree for clipping region. R-file performs better than other structure in terms of multilayer grid file which reduce the overlapping grid file and perform good search result.

## V. Conclusion and Future Work

In this paper, we focused to provide an overview of spatial access methods and comments of researchers on particular access method. In this survey, we found that no spatial access method is good for all conditions. Researchers gave their comments on the basic of particular parameter and compared their results with the work of other researchers. They concluded that their work is better than others in particular condition. But no one provide that which access method is best for all conditions. In addition, in the whole paper we explained the examples, advantage and disadvantage on the basis of researchers' comments. In future, our aim is to design spatial index for relevant and optimized result to remove spatial dead links.

## VI. References

- [1] Jones CB, Purves R, Ruas A, Sanderson M, Sester M, van Kreveld M, Weibel R Spatial information retrieval and geographical ontologies an overview of the SPIRIT project. In: Proceedings of the 25th ACM SIGIR conference, pp 387–388, 2002.
- [2] Jones CB, Abdelmoty AI, Fu G Maintaining ontologies for geographical information retrieval on the web. In: Proceedings of on the move to meaningful internet systems 2003: ODBASE 03, LNCS, vol 2888, 2003.
- [3] Jones CB, Abdelmoty AI, Fu G, Vaid S, The SPIRIT spatial search engine: architecture, ontologies and spatial indexing. In: Proceedings of the 3rd int. conf. on geogr. Inform. Science, LNCS, vol. 3234, pp. 125–139, 2004.
- [4] Rigaux, P., Scholl, M., and Voisard, A. *Spatial Databases: With Application to GIS*. Morgan Kaufmann, 2001.
- [5] Robinson, J. T., "The K-D-B-tree: A Search Structure For Large Multidimensional Dynamic Indexes," In *Proc. ACM SIGMOD International Conference on Management of Data*, pp. 10-18, 1981.
- [6] Gaede, V. and O. Günther, "Survey on Multidimensional Access Method," *Technical Report ISS-16*, Department of Economics and Business Administration, Humboldt University Berlin, revised version, 1997.
- [7] Larson, P. A. "Linear hashing with partial expansions." In *Proc. 6th Int. Conf. on Very Large Data Bases*, pp. 224-232, 1980.
- [8] Samet, H., "The Design and Analysis of Spatial Data Structures," Addison Wesley, Reading, Mass., on Very Large Data Bases, pp. 224-232, 1990
- [9] Fagin, R., J. Nievergelt, N. Pippenger, and R. Strong, "Extendible hashing: A fast access method for dynamic files," *ACM Transaction on Database Systems*, 4(3), pp. 315-344, 1979.
- [10] Blanken, H., A. Ijbema, P. Meek, and B van den Akker, "The generalized grid file: Description and performance aspects," In *Proceeding of 6th IEEE International Conference on Data Engineering*, pp. 380-388, 1990.
- [11] Hinrichs, K., "The grid file system: implementation and case studies of applications," PhD.dissertation, Institute fur Informatik, ETH, Zurich, Switzerland, 1985.
- [12] Hutflesz, A., H.W. Six, and P. Widmayer "Globally order preserving multidimensional linear hashing." In *Proc. 4th IEEE Int. Conf. on Data Eng.*, pp.572-579, 1988.
- [13] Samet, H., "The quadtree and related hierarchical data structures," *ACM Computing Surveys*, 16, June 1984.
- [14] [Samet, H., "An overview of quadtrees, octrees, and related hierarchical data structures," In R. A.Earnshaw, editor, *Theoretical Foundations of Computer Graphics and CAD*. NATO ASI Series F, vol.40, pp. 51-68. Springer-Verlag, 1988.
- [15] Samet, H., "Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS," Addison-Wesley, Reading, MA, 1990.
- [16] Samet, H., "The Design and Analysis of Spatial Data Structures," Addison Wesley, Reading, Mass., 1990.
- [17] Bentley, J. L., "Multidimensional Binary Search Trees Used For Associative Searching," *Communications of the ACM*, 18(9), 509-517, 1975.
- [18] Fuchs, H., G. D. Abram, and E. D. Grant, near real-time shaded display of rigid objects. *Computer Graphics* 17 (3), 65-72, 1983.
- [19] Fuchs, H., Z. Kedem, and B. Naylor, on visible surface generation by a priori tree structures. *Computer Graphics* 14 (3), 1980.
- [20] Seeger, B. and H.-P.Kriegel, "The buddy-tree: an efficient and robust access method for spatial data base systems," In *Proc. 16th Int. Conference on Very Large Data Bases*, pp. 590-601, 1990.
- [21] Guttman, A., "R-trees: A Dynamic Index Structure for Spatial Searching," In *Proc. ACM SIGMOD International Conference on Management of Data*, pp. 47-57, 1984
- [22] Beckmann, N., H.-P.Kriegel, R. Schneider, and B. Seeger, "The R\*-tree: An Efficient and Robust Access Method for Points and Rectangles," In *Proc. ACM SIGMOD International Conference on Management of Data*, pp. 322-331, 1990.
- [23] Berchtold, S., D. Keim, and H.-P. Kriegel . The X-tree: An index structure for high-dimensional data. In *Proc. 22nd Int. Conf. on Very Large Data Bases*, Bombay, India, pp. 28-39, 1996.
- [24] Samet, H. , The design and analysis of spatial data structures. Reading, MA: Addison Wesley, 1989.
- [25] Morton, G. , A computer oriented geodetic data base and a new technique in le sequencing. IBM Ltd. 1966.
- [26] Hinrichs, K. , Implementation of the grid le: Design concepts and experience. BIT 25, 569{592 le sequencing. IBM Ltd. 1985.
- [27] Stonebraker, M., T. Sellis, and E. Hanson .An analysis of rule indexing implementations.in data base systems.In *Proc. 1st Int. Conf. on Expert Data Base Systems*, 1986.