

On Spatial Information Retrieval and Database Generalization

Xiaofang Zhou¹ Yanchun Zhang², Sanglu Lu³ and Guihai Chen³

¹School of Computer Science & Electrical Engineering, University of Queensland, Australia

²Department of Computing & Mathematics, University of Southern Queensland, Australia

³Department of Computer Science and Technology, Nanjing University, China

Abstract

Spatial data are often stored in a database with the finest level of details. The problem of cartographic generalization of spatial data concerns deriving spatial data with proper level of details when the data are used for an application. The generalization process is not simply a process of selecting objects or parts of objects suitable for an application. It should maintain the semantic information carried with the spatial data, such as topological relationships among objects and the overall “look and feel” of the data after generalization. Most difficulties of map generalization come from the fact that the importance of objects, which determines whether an object should be included and how it should be visualized in the generalized data set, is decided not only by their sizes, but also the “meaning” of these objects to the application. The latter is application-dependent, and often subjective. As a classic problem in cartography, maps at different scales are pre-derived for different applications, with intensive human involvement. The performance of spatial data generalization is a secondary issue to the quality of the maps generalized. With more and more computerized applications making use of shared spatial data sources, and the increasing number of applications emerging from Internet-based spatial applications (such as map-based navigation), efficient spatial data generalization becomes a critical issue. At the same time, it becomes less and less likely to have human intervention or have pre-materialized spatial data generalization for all applications in the Internet environment. In this paper, we investigate issues in efficient spatial data generalization, with a unique focus on spatial database design and spatial operations processing. We propose several types of extensions and novel applications of z-values, a commonly used spatial indexing mechanism, to achieve our goals.

1. Introduction

Spatial data types such as points, lines and polygons are used to describe the “location” aspect of data objects such as roads, rivers, administrative regions, land parcels and building footprints, telephone poles and trenches. Government agencies and large organizations have invested huge amounts of effort in collecting spatial data in the past. As an important corporate asset, it is a trend to migrate spatial data from proprietary file systems to modern

Database Management Systems (DBMS) for multiple users to share. While spatial objects are often stored in a database representing the finest level of details available to support some applications, such details may not be necessary for other applications. Excessive details translate to higher processing costs for retrieving data from the database, transferring data to applications (often over a computer network), and processing data in applications (for example, in a Java Applet running on the client’s machine). Large portions of spatial applications use spatial data to draw, directly or indirectly, a map for a human user to read, to interact with for locating objects or regions of interest, or as a backdrop for displaying other information. One example is map-based Internet navigation [10,13]. For this type of applications, one simple idea to reduce the total processing cost is “not to process what you cannot see”. That is, any objects or parts of objects that are not discernible due to limited display area should be discarded as early as possible. Retrieval of data from a database is usually the first task in the processing chain, and in many cases also among the most time-consuming ones. Thus, maximum benefit can be achieved if those spatial objects or parts of spatial objects that cannot be appreciated by an application are not retrieved from the database at all. A challenge is to find intelligent spatial query processing strategies and necessary supporting data structures such that a spatial DBMS can return a “right” level of details for an application. Such strategies should reduce the contents and complexity of a spatial data set in such a way that the structural characteristics of the original dataset are maintained for the purpose of visualization.

The process of reducing the size and complexity of a spatial dataset with visual quality preserved or enhanced is known as *generalization* in cartography [3]. The generalization process is not simply a process of selecting objects or parts of objects suitable for an application. It should maintain the semantic information carried with the spatial data, such as topological relationships among objects and the overall “look and feel” of the data after generalization [2, 3, 6, 8, 9]. In other words, the goal is to retrieve spatial information, as versus spatial data. Most difficulties of map generalization come from the fact that the importance of objects, which determines whether an object should be included and how it should be visualized in the generalized data set, is decided not only by their sizes, but also the “meaning” of these objects to the application. The latter is application-dependent, and often subjective. For example, parks and picnic areas might be important when

preparing a map for cyclists, but may be of little interest for truck drivers. As a classic problem in cartography, maps at different scales are pre-derived, with intensive human involvement, for different applications. The performance of spatial data generalization is a secondary issue to the quality of the maps generalized. With more and more computerized applications making use of shared spatial data sources, and the increasing number of applications emerging from Internet-based spatial applications, efficient spatial data generalization becomes a critical issue [13]. At the same time, it becomes less and less likely to have human intervention or have pre-materialized spatial data generalization for all applications in the Internet environment. In this paper, we investigate issues in efficient spatial data generalization, with a unique focus on spatial database design and spatial operations processing. We propose several types of extensions and novel applications of z-values, a commonly used spatial indexing mechanism, to achieve our goals of reducing as much as possible excessive spatial data in the earliest possible stage of spatial information retrieval while keeping the quality of spatial data visualization acceptable for an application.

The rest of this paper is organized as follows. In section 2 we give an introduction to the problems of spatial data generalization. Then, a database-oriented approach is proposed in section 3 to support spatial data generalization within a spatial database system. In section 4 we show how use and extend the z-value based spatial indexing mechanism to support some of the fundamental spatial operations. We conclude this paper in section 5.

2. Spatial Data Generalization

In this section, we introduce the problem of spatial data generation, in the context of cartography and also in the context of spatial database processing, which adds a new dimension to the data generalization problem.

Let us look at a motivational example here. Toowoomba is a city in Australia. It has its famous Spring Carnival every September, featuring public shows of award-winning gardens and a beauty queen parade. During the festival week, many people driving to the city from other places for a day tour. For those tourists a standard map of city is not very useful. What they need is a customized map for the event, with details of the route of the parade, and locations of open gardens, with the roads leading to those gardens. Many inter-city tourists seek such information from the Toowoomba City Council website. Clearly, for such a map there is no need to show the same level of details for the entire city. Rather, those places of interest for the festival and surrounding areas need more details, including some street numbers and on which side the gardens are. Unnecessary details on the map can result in many difficulties for web users such as longer waiting time and difficulties in reading maps. The City Council website display maps generated from a spatial database. Obviously, some generalization process is needed to produce such a map. It is not an attractive solution, and sometimes it is not even feasible, to pre-materialize maps to meet the needs of

all tourists, who may have different interests such as visiting local museums or craft shops. While many object selections can be simply done by the users specifying the types of objects they are interested (i.e., choosing object layers [10]), other types of generalizations are more difficult. Examples are showing different level of details according to the approximation to the places of interest [6], or simplifying the route of a road while maintaining road connectivity and not putting houses on the wrong side [8].

Now we give a closer look at the generalization problems. Generalization has been the subject of intensive research in the area of cartography and computer science [3]. Besides proposing numerous algorithms for simplifying spatial objects in a way that they bear ‘maximum similarity’ with the source objects, this research is also characterized by a strong emphasis on correctness aspects [9]. What makes generalization so difficult is that there is no unique solution, but numerous constraints have to be taken into account during the process of generalization. Following [8], the following four constraints have to hold for spatial data that is to be displayed to humans:

- *Metric constraints*: This class of constraints tries to ensure that all details of spatial data are perceptible for a human observer. Examples of metric constraints are minimal separability, minimal size and minimal width.
- *Topological constraints*: Existing topological relations between source data objects should be preserved by the generalization process. In the above example, road connectivity and directional relationship between road and houses are examples of topological constraints.
- *Semantic constraints*: During the process of generalization, the semantics of objects may need to be considered. For example, a street parallel to another road should not bear the name of that road after generalization.
- *Gestalt constraints*: This type of constraint aims at capturing the overall impression of a scenario and is concerned with perceptual criteria such as maintaining the distribution and the arrangement of features. We call this type of constraints as visual quality of the map generalized. Visual quality is a constraint that is very hard to translate them in terms of operations. They can only be checked after all other types of constraints have been enforced, and often this can only be checked by application users.

The generalization process consists of some basic operations [3,8]. These operations are classified into the following three categories [13]:

- *Selection*: This operation is concerned with the question of identifying a subset of representative objects from a given set for the purpose of generalization.
- *Simplification*: This category of operations addresses the issue of which part of an object should be displayed by the application. Certain conditions, such as preserving the similarity between generalized and source data and maintaining their topological relationship are important in this category.

- *Tokenization and amalgamation*: Objects that are important enough to be displayed but too small to be recognized have to be replaced by some visible token. Furthermore, it might be necessary to amalgamate several objects to form a new one. It is also necessary to displace some objects to preserve their topological relationship.

Object selection is the first step of generalization. The operations for simplification, tokenization and amalgamation are applied to the objects selected from the database. Based on this observation, we will focus on the selection operation within a spatial database, and the use of the same data structures to support simplification, tokenization and amalgamation problems.

3. A Database Oriented Approach

Performance has long been a major barrier for making better use of spatial data. It is difficult for any attempt that requires retrieving *all* spatial objects out from the database to solve this problem, as for a large amount of complex spatial objects data retrieval itself is often the bottleneck. The other side of this diploma is that which objects or which parts of objects are required after generalization are often not known until these objects and the objects nearby are considered (that is, being fetched from the database and processed using some generalization algorithms such as those discussed in [3,8]).

In this paper we propose to approach this problem through innovative use and extension of spatial indexing techniques used in the past mainly for fast spatial data access in spatial DBMSs through establishing a mapping between data space and object approximation [1]. If the importance of spatial objects to an application can be determined from using their index entries, which are usually much simpler than spatial data themselves, spatial data generalization can be done *within* spatial DBMS. Apart from much higher efficiency, another major benefit can also be expected from doing so. That is, applications choosing to take advantage of properly simplified spatial dataset should incur minimum extra effort (i.e., only display parameters need to be provided). This is in contrast to previous approaches that require writing complex code in application programs to do spatial data generalization. We will discuss details of such a database-oriented approach for generalization with examples in the next section.

3.1 Internet Spatial Applications

Figure 1 shows a typical architecture for Internet-based spatial applications. A spatial database is used to store spatial and non-spatial data. The database can be browsed by means of a Web-browser capable of running Java applets. Generating requests and displaying the result data is handled by the applet that runs on the user's machine. On the server site, a database server translates the user's request into queries against the spatial database to fetch qualifying objects. These objects are then encoded according to some spatial data transfer protocol, and transferred to the applet over the network. After decoding the data on the client side,

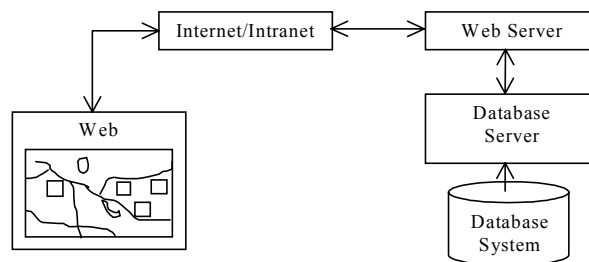


Figure 1: A schematic architecture for Web-based spatial applications

they are drawn on the screen. That is, the following major steps are involved: data retrieval (handled by the database system), data encoding (handled by the database server), data transferring over the network, data decoding and drawing (handled by the applet). Generalization can improve the performance on both the database side (e.g., less data to encode, smaller amount of data to transfer) and the application side (e.g., fewer spatial objects to draw, simpler shapes). Obviously, once the objects are retrieved from the database, they can be simplified (typically before the data-encoding step) on the server site by removing parts of an individual spatial object or finding a suitable abstraction that preserves its characteristics. We call this type of simplification object generalization. Most of previous studies in generalization are focused on *object generalization* [8]. There is a complementary type of generalization, *database generalization*, which deals with removing objects from data sets and addresses the issue of spatially significant objects (e.g., objects that are visually recognizable when being displayed) [13]. It approaches the problem of generalization from a database perspective, treating generalization as an integral part of database query processing. Database generalization can be done inside a database system as well as in the database server. A database query generated from a user's request is modified by the database server taking into account of how these data are to be used. If some objects cannot be perceived when being displayed in the web browser, database generalization tries, with the support from the database system, not to retrieve them from the database at all. The benefit of a smaller amount of data fetched from the database flows on to other steps. However, generalization can degrade the quality of map if the characteristics of the original map are not preserved during generalization. In addition, the database generalization step may also increase the overall processing costs. Thus, it is crucial to find an efficient data generalization method that can significantly reduce the amount of data, and at the same maintain the key characteristics of the original data set, at a as little as possible overhead.

One simple way to use spatial data with different levels of details is to physically store multiple views of a dataset in the database, with one view for one type of applications. That is, each view consists of a generalized dataset of the original database, and the views for all the expected applications are pre-computed and stored in the database. This approach, known as the multi-resolution database, may require a very large (and sometimes prohibitive) amount of extra disk storage since spatial data sets are typically very

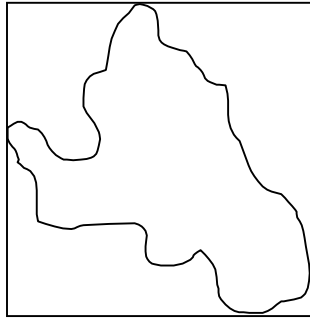


Figure 2: Minimum bounding rectangle

large (from gigabytes to terabytes) and the number of possible combinations of queries and display devices can be large. It can also be costly and problematic for maintaining consistency between multiple representations of the same object after the database is updated [6]. Another way is to simplify data sets outside the database. That is, all data objects satisfying a query condition are fetched from the database into a special program to process. Most generalization algorithms used by cartographers in producing high quality maps are in this category [3,8]. This approach can reduce a data set to a desired level of complexity. However, as mentioned before, this approach, designed for producing high-quality paper-based maps, does not address the performance problem as it does nothing to reduce the major component of the total processing time - the cost of fetching spatial objects from databases.

3.2 Spatial Indexing and Z-Values

Before we discuss different algorithms that can be used for database generalization inside the spatial database, it is necessary to introduce briefly object approximation and spatial indexing, and spatial data processing strategies based on them.

A polygon can be arbitrarily complex. It is a common practice to approximate a spatial object using its minimum bounding rectangle (MBR), which is defined as the smallest rectangle that completely encloses the object, see Figure 2. In such a way, an object with any number of points can be approximated using a rectangle that can be represented by only two points (i.e., the lower-left and upper-right corners). The cost of processing spatial data can be reduced significantly by processing their approximations first. For example, two polygons cannot overlap if their MBRs do not overlap; thus, a simple operation on MBRs may eliminate the need to fetch and process full geometry for some polygons, reducing both I/O and CPU cost.

An object often needs to be approximated at a much finer level than its MBR. For example, the MBR of a long road crossing the data space is obviously an inappropriate approximation. The z-ordering technique is based on space filling curves. It has been discovered by several researchers independently under different names (see [1] for a survey), and is used in several commercial spatial database systems, such as Oracle [5], as a spatial indexing mechanism. It

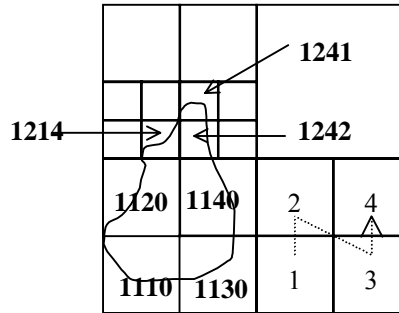


Figure 3: Polygon approximation using z-values

approximates a given object's shape by recursively decomposing the embedding data space into smaller subspaces known as the Peano cells. The z-ordering decomposition works as follows. The whole data space is divided into four smaller rectangles of the same size. The four quadrants are numbered as 1 to 4 following certain order (e.g., the z-order in Figure 3). These quadrants can be further divided and numbered recursively. The z-value of a Peano cell can be determined using the following steps:

1. The z-value of the initial space is 1;
2. The z-values of the four quadrants of a Peano cell whose z-value is $z = z_1 \dots z_n$, $1 \leq z_i \leq 4$, $1 \leq i \leq n$, are $z_1 \dots z_n 1$, $z_1 \dots z_n 2$, $z_1 \dots z_n 3$ and $z_1 \dots z_n 4$ respectively following the z-order;

A spatial object can be approximated using a set of Peano cells it overlaps with. The z-values can have different length, as a result of different Peano cell sizes. Clearly, the larger the Peano cells the shorter is the z-value sequence. The maximum number of decomposition level, also called resolution, determines the maximum length of z-values. In order to simplify processing, a number of '0's are often appended at the end of shorter z-values to make the length of all z-values identical (i.e., the maximum length). (However, we define the length of a z-value as the number of non-zero digits it has.) In Figure 3, the polygon has been approximated by 7 polygons, where the resolution is 4. By transforming a two-dimensional object into a set of one-dimensional points (i.e., the z-values), spatial objects can be represented as numbers and therefore can be maintained by a ubiquitous one-dimensional access method such as the B+-tree.

4. Novel Applications of Z-Values

In this section, we discuss how to use the z-values to support the fundamental generalization operations, namely, selection, simplification, tokenization and amalgamation.

4.1 Object Selection

A sound generalization method should be able to identify *significant objects* and *significant parts of the objects* regarding to the visual quality of the map. We distinguish two types of "significance" here. The first one is determined by the semantics of a query request. For example, users searching for crop data in a particular region typically have

no interest in data irrelevant for their request. Therefore, crop data is considered as significant. For this type of significance, it is possible for the user to explicitly specify as attribute constraints in the query, which can be readily used by the underlying DBMS for object retrieval. The second type of significance, which deals with implicit constraints inferred from the settings of the application and the request. An example of the second type of significance is that for a given application request it is possible to infer constraints that objects smaller than a certain size should not be displayed because they are too small to be discernible. Interactive spatial applications which do not need to display all the data items meeting the explicit constraints have to derive automatically implicit conditions allowing the identification of visually significant objects. We say that a spatial object is *visually significant* if it is in one of the following cases:

- *Large objects*: Objects larger than certain size are important and therefore, they should be selected. Here, size refers not only to area, but also to other criteria such as extension. For example, a long road with a comparatively small area should be considered as large in terms of visual significance.
- *Objects in sparse region*: Small objects in a sparsely populated region can also be significant. A small town in the desert of central Australia can be more significant than a suburb in a metropolitan area, even when the latter is considerably larger.
- *Representative objects*: For a large number of objects which all qualify the query condition and cannot be discriminated by the previous two criteria, it might be necessary to select a subset of objects as representatives for the complete set. For example, while it may not be possible to display all the land parcels in an area, some land parcels need to be displayed to indicate the land use.

The first criterion above is a metric constraint, and the last two attempts to capture the overall “look and feel” of the original dataset.

Next we will look at how to use z-values to find visually significant objects. Recall that our goal is to identify the objects that are *likely* to be visually important from a very large dataset *quickly*. This is in contrast (and complementary) to previous work which are more interested in simplifying a single complex object to for maximum similarity. The z-value indexing method has several properties that can be used to determine visual significance:

- 1) Peano cells can vary in sizes. The z-value for a larger Peano cell is shorter than that for a smaller cell. Thus, large objects, which are more likely to be approximated by larger Peano cells, can be quickly identified using z-values. There is no need to fetch all the objects and then compare their sizes to find larger objects.
- 2) For those objects with small area but large footprint, such as roads and rivers, they are likely to be approximated with a large number of z-values. Therefore, they are also easily recognizable by z-values.

- 3) Peano cells can be nested with each other. A Peano cell z inside another cell z' can be either recognized as z must have z' as prefix. By comparing objects by their first k digits (k can be derived from the spatial extent of the data space and the size of display area), objects in a sparse region can be identified visually significant objects; and the objects in dense regions can also be identified for further processing.
- 4) There is an underlying grid for z-values, and the size and location of a grid is encoded in a z-value. Thus, representative objects can be selected following the underlying grid (at a level determined by visual parameters) using their z-values. This is better than random filtering, and cheaper than calculating relative positions of these objects.

Comparing our definition of visual significance, the observations above suggest that the z-values, although much simpler than spatial objects they represent, carry sufficient information for determining visual significance. Note that most of operations above can be easily expressed as SQL queries, and can be executed by a DBMS efficiently.

The above techniques are proposed for identifying visually significant objects. However, for a visually significant object, not every part of the object is necessarily visually significant. How visually significant parts can be identified by using z-values is less well understood at this stage. We believe that some new database storage structures need to be designed. When the part of an object inside a Peano cell which is below a pre-defined threshold, that part can be returned as a point, ignoring its actual shape. This will also be useful to tokenization, as we will discuss later.

4.2 Simplification

The problem of spatial data simplification concerns reducing the level of details of a spatial object with the goal of maximizing the similarity of the data before and after simplification. There are many simplifications algorithms (see [2] for a survey). This problem for a single polyline has been largely solved, with several algorithms which are both efficient and of high quality in terms of maintaining data similarity. When this problem is putting into the context of a set of objects, however, it becomes more complex as the topological relations after simplification should be the same as those relationships before simplification. After simplification, for example, two roads should still connect to each other at or near where they intersect, and should not intersect at where they do not intersect. And those objects on one side of the road should not appear on the other side of the road after simplification.

To ensure that all topological constraints are met after the generalization process, many objects need to be processed at least three times. Typically, the first pass aims at identifying topological relations among the objects. The second pass then applies the generalization operations. A third pass is necessary to check the constraints. If the constraints are not met, steps two and three have to be repeated using a different parameter set.

Z-values can be thought of a rasterization of vector spatial data with variable sizes of pixels. Pixel size is determined by the need to approximate objects. Such a perception of the z-values can lead to more efficient way to process spatial operations, because only those “pixels” (in this case, those objects approximated using these “pixels”) identical or close to each other need to be processed. This can effectively reduce the need of more costly spatial operations. For example, Z-values can be used to check topological relationships quickly, though with less accuracy. Assume that two lines L and L' are approximated by two sets of z-values Z and Z' . One can imagine that the z-values of a line as an envelope fully enclosing a line. When L and L' intersect, there should have a pair of z-values in Z and Z' such that they are identical or one is nested inside another (both relationships can be easily detected using z-values, as explained before). For the z-values of the simplified lines (these z-values may be different from the original z-values), there should still have a pair of z-values which are identical or one is nested in another. This is a necessary, though not sufficient, condition for the simplification process of the two lines to be correct with regard to a topological constraint. Manipulation of z-values has a much lower cost, in particular when the lines are complex. Thus, incorrect simplifications can be detected at a lower cost. This new type of use of z-values is in principle similar to the idea of the filter-and-refine approach that is a standard way of processing spatial operations [8,11]. The process of topological relationship check after simplification can be “localized” to certain parts of lines as indicated by z-values. Similar optimization techniques are applicable to other types of spatial relationships, being topological or directional.

Recently a concept called “perfect generalization” is proposed for simplifying the process of object simplification [2]. The idea is simple. Excessive details of spatial data are reduced in a way as if all the details are fetched and rendered to that device (i.e., objects and parts of objects are removed only if all the pixels they are mapped to have already been occupied by other objects). Thus, removal of objects or simplification of objects will not change the visual perception of the resultant data at all. In such as case, all those topological constraint problems do not matter any more, as the map obtained from perfect generalization will be visually the same as the original map. This is particularly useful when the display device has a much lower resolution than the data stored in the database, such as WAP (wireless access protocol) applications.

4.3 Tokenization

An object can be simplified to a point, all completely disappear. Tokenization is a process to display an object which is too small but very important to an application (for example, hydrants in a safety map) as a token on the map. For those objects which need to be tokenized, its object approximation (i.e., z-values) is sufficient as all one needs to know in the case of tokenization is where those objects are. Other spatial details such as its geometry will not be useful.

Therefore, if the z-value for an object (or part of an object) represents an area which is smaller than the area which will be mapped to a single pixel in the display device, the details of this object (or parts of object) can be safely skipped (i.e., not being fetched from the database), and a token will be placed on where the z-value indicates if the object is important (as determined by its attributes).

The above discussions in simplification, perfect generalization and tokenization suggest that if spatial data (i.e., the geometry of spatial data) are stored in groups, instead of as an entire block as implemented in many spatial databases, following their z-value decomposition structures, some major performance gain can be expected for applications such as data generalization. In other words, with z-values, it is possible to find which parts of an object needs to be retrieved, thus only those parts of data will be fetched if the underlying data structures in a database support that.

4.4 Polygon Amalgamation

The polygon amalgamation operation merges a set of polygons into a new polygon which is the union of the source polygons. This is an important operation for spatial online analytical processing, spatial data mining and spatial data visualisation, where polygons representing different spatial objects often need to be amalgamated by varying criteria or display device limitations. A straightforward way of amalgamating polygons is to fetch all the source polygons from the database, and merge them by eliminating those parts of the boundary shared by more than one polygon using computational geometry algorithms. The processing cost of this operation can be very high when the number of source polygons is large.

Based on the observation that not all source polygons contribute to the boundary of the target polygon, an occupancy-based polygon amalgamation algorithm is proposed in [12]. It uses z-values to identify internal polygons. Each z-value index entry is of the form (z, pid) , stating that object pid overlaps with Peano cell z . We extend each index entry to the form of (z, pid, α) where α is the *occupancy ratio* (we record not only which polygons overlap with a Peano cell, but also the percentage of area that each polygon occupies). Those cells completely inside an amalgamated polygon can be identified simply by adding up the occupancy ratios of the polygons for each Peano cell. Those cells whose aggregate occupancy ratio is 100 % are internal cells. Therefore, only those polygons that overlap with non-internal cells need to be retrieved from the database and to be processed. So object selection here are done again using z-values. Furthermore, a less than 100% threshold can be used such holes inside the amalgamated polygon (i.e., those objects which do not belong to the amalgamated polygon) can be ignored if they are too small to be seen after being displayed at a particular device.

This occupancy-based extension of z-values is also useful to efficient processing of spatial joins [7].

5. Conclusions

Spatial data generalization is an integral step of spatial information retrieval. Spatial data, unlike many other types of data, are retrieved mainly for visualization. When not all spatial objects or not all parts of a spatial object can be discernable for human eyes, some objects need to be filtered out and transformed into some simpler forms according to their importance to the applications they are used for. Efficient spatial data generalization is a problem that becomes increasingly important with the popularity of Internet-based spatial applications. Database operations are typically among the most time-consuming operations in Internet-based spatial applications. The benefit from reduced spatial data sets resulted from optimized spatial data access strategies often flow to other steps such as data simplification and data transfer. In this paper, we have proposed a database-oriented processing framework that optimizes the performance of spatial data generalization from inside a spatial database, rather than treating such operations as post-database processing. This new approach addresses a significant step of the entire generalization process, and is complimentary to the research of cartographic generalization algorithms, which so far are largely focused on in-memory computation cost or correctness issues only. We have demonstrated how to use z-value based spatial indexes to reduce the amount of spatial data which otherwise can only be eliminated by costly post-database processing. We have shown that such improvement can be made for a wide range of spatial operations.

References

- [1] V. Gaede and O. Günther. "Multidimensional Access Methods". *ACM Computing Surveys*, 30(2):170-231, 1998.
- [2] C. Jackson, "Spatial Data Simplifications", Hons. Thesis, University of Queensland, 2000.
- [3] R. B. McMaster and K. S. Shea. *Generalization in Cartography*. Association of American Geographers, Washington, D. C., 1992.
- [4] M. E. Orłowska and X. Zhou, "A Spatial Database as a Component of Integrated Database System", *Proc. Of the 1999 Int'l Symposium on Database Applications in Non-Traditional Environments*, pages 203-209, November 1999, Kyoto. IEEE CS Press.
- [5] Oracle Inc. "Oracle 8i Spatial: Experiences with Extensible Database, an Oracle Technical White Paper", May 1999
- [6] S. Spaccapietra, C. Parent and E. Zimanyi, "MurMur: A Research Agenda on Multiple Representations", *Proc. Of the 1999 Int'l Symposium on Database Applications in Non-Traditional Environments*, pages 373-384, November 1999, Kyoto. IEEE CS Press.
- [7] D. Truffet and M. E. Orłowska. "Two Phase Query Processing with Fuzzy Approximations". *Proc. 4th ACM Int. Workshop on Advances in Geographic Information Systems (ACM-GIS'96)*, Rockville, USA, 1996.
- [8] R. Weibel. "Generalization of Spatial Data". In *CISM Advanced School on Algorithmic Foundations of Geographical Information Systems*, pages 346-367, 1996.
- [9] R. Weibel. "A Topology of Constraints to Line Simplification". In *Proc. of the 7th Int'l Symposium on Spatial Data Handling (SDH'96)*, pages 9A.1-9A.14, 1996.
- [10] J. D. Yates and X. Zhou. "Search the Web Using a Map", *Proc. of the 1st Int'l Conf. on Web Information Systems Engineering*, pages 222-229, June 2000, Hong Kong. IEEE CS Press.
- [11] X. Zhou, Y. Zhang, X. Lin and C. Liu, "On the Optimization of Complex Spatial Queries", *Proc. of the 2nd Int'l Symposium on Cooperative Databases Systems for Advanced Applications (CODAS'99)*, March 1999, Wollongong Australia. Springer-Verlag.
- [12] X. Zhou, D. Truffet and J. Han, "Efficient Polygon Amalgamation Methods for Spatial OLAP and Spatial Data Mining", *Proc. of the 6th Int'l Symposium on Large Spatial Databases (SSD'99)*, July 1999, Hong Kong. Lecture Notes in Computer Science, Springer-Verlag.
- [13] X. Zhou, A. Krumm-Heller and V. Gaede, "Generalization of Spatial Data for Web Presentation", *Proc. the 2nd Asia Pacific Web Conf. (APWeb'99)*, September 1999, Hong Kong.