Efficient Managing Large Scale Species Range Maps in a Spatial Database Environment

Jianting Zhang
Department of Computer Science
City College of the City University of New York
New York, USA
jzhang@cs.ccny.cuny.edu

***Abstract*—Species distribution data are becoming increasingly available over the past few years and the availability is likely to increase significantly in the near future due to technological advances. While traditionally GIS are used to visualize the distributions of a limited number of species and to generate biodiversity indices in predefined regions in an offline mode, it is desirable to manage such data in a spatial database environment and allow customer applications to efficiently query the database with arbitrary dynamically defined regions. In this study, we have developed a Variable-Fanout Space Partition (VF-SP) tree structure to represent species distribution maps by extending the classic quad-tree data structures to accommodate user-defined raster tessellations. Subsequently we have developed an approach to import multiple VF-SP trees representing a large number of species distribution maps into a spatial database for efficient query processing. Experimental results using NatureServe 4000+ bird species distribution data demonstrate that the proposed approach can be 30-300 times faster than the baseline approach that manages the same data as polygons in the same spatial database with respect to the average query response time using a query window size of 0.1 degree to 1 degree at a global scale. The average response times for such queries are less than 1 second when querying more than 15 million boxes in a PostgreSQL database. The results are encouraging with respect to using state-of-the-art spatial database technologies to manage large-scale species distribution data and answer dynamic queries in generating indices that are important to biodiversity research**

Keywords: Spaital Databases, Space-Partition Tree, Variable Fanout, Species Distribution

I. INTRODUCTION

More than a million of species have been recorded in several repositories, such as the repositories provided by the "Catalogue of Life" project (www.catalogueoflife.org) and the uBio project (/www.ubio.org). The range maps of more and more species are also becoming available which allows researchers to explore the relationships between species distribution and the environments. Synergizing geoinformatics and bioinformatics technologies for species distributions and biodiversities research will contribute to our understanding of the ecological consequences of climate change and human impacts. Recently, NatureServe (www.natureserve.org) published Digital Distribution Maps of the Birds of the Western Hemisphere Version 3.0 which covers 4,273 bird species. Similar range maps for 1737 mammal species of the west hemisphere and more than 6000 world's amphibians are also available from NatureServe's website. We envision that the availability of species range maps will be significantly increased over the next few years, which will be important not only to environmental modeling but also to phylogeography, phylogenetics and other branches of bioinformatics. It is imperative to develop efficient data management techniques, including novel data structures and algorithms, to effectively support global biodiversity research and subsequent species conservation practices.

Currently all known species range maps from USGS and NatureServe are distributed as ESRI Shapefiles. While the Shapefile format is well accepted by a number of GIS, including ESRI's ArcGIS, these range maps are not readily usable for analysis other than simple visualization. The reason behind is that species distribution analysis, especially biodiversity related research, often involve queries across a large number of layers. While sophisticated GIS, such as ArcGIS, is empowered by spatial indexing techniques (such as R-Tree or Quad-tree, see Gaede and Gunther 1998 for a review) and is very efficient in performing intra-layer spatial queries on geometric objects with no or little overlapping, they are usually not good at querying tens of thousands of layers in a single query. A procedure language would be needed to loop through all the layers and a temporary layer would also be needed to keep the intermediate results. This is neither convenient nor efficient. An alternative solution might be to combine all layers into a single layer and then to query against the combined layer. However, while the geometric objects within a single layer has no or very little overlap, they can be highly overlapped in the combined layer which makes spatial indexing much less effective. Probably more importantly, queries that compute list of species and their area sizes in an arbitrarily defined region often require on-the-fly geometric clipping between the complex polygons of the distribution data and the user defined query region. The computation is usually quite expensive which makes the approach not scalable to process queries that require fast response, for example, interactive visual explorations.

While spatial databases have provided rich query functionality for vector data, they are not readily usable to support such species distribution analysis (see more details in the next section). Compared to spatial databases, GIS are more frequently used for species distribution analysis for small groups of species based on a raster tessellation. Since polygons used to delineate species ranges are usually less accurate (or fuzzy, uncertain) at finer spatial resolutions, a raster tessellation makes more sense in species distribution

Authorized licensed use limited to: City College of New York. Downloaded on April 23,2010 at 16:12:23 UTC from IEEE Xplore. Restrictions apply.

analysis in many cases. Unfortunately, raster data are much less well supported in spatial databases. Spatial databases provide many desirable features for developing domain applications, such as structured query language, efficient indexing support and client/server architecture. It is imperative to assemble core spatial database functionality and develop new modules when necessary to support large-scale species distribution analysis and biodiversity studies.

In this study, we have developed a Variable Fanout Space Partition (VF-SP) tree data structure to represent species range maps, which is an application-oriented extension to quad-tree, and an algorithm to generate a VF-SP tree directly from a polygonal species range map. The data structure and the algorithm convert complex polygonal datasets of species distribution data into simple rectangular boxes. They are more suitable for spatial queries to derive species distribution indices with respect to computation costs and query response time. Our technical contributions also include system integration of multiple open source packages, including GDAL (www.gdal.org), PostgreSQL (www.postgresql.org), PostGIS (postgis.refractions.net) and the newly developed module to provide an end-to-end solution for managing large-scale species distribution data in an effective way.

II. Background and Related Works

Exploring the relationships among species distributions and the environment is central to basic ecology and biogeography research and biological conservation practices. A variety of biodiversity indices have been developed for analytical purposes. We refer readers to (Ricotta 2005) for a brief overview. Among them, alpha diversity, i.e., the number of species and their abundances in a region, and beta diversity, i.e., the dissimilarities of species compositions in multiple regions, are two most frequently used ones. Overviews of alpha diversity and beta diversity indices can be found in (Moreno et al, 2006) and (Koleff et al 2003), respectively. As discussed in (Zhang and Gruenwlad 2008), biodiversity indices can be computed based on either a vector or a raster geographical tessellation.

Most of existing biodiversity studies use one or more predefined geographical regions and a predefined group of species. While this is a viable solution for traditional, individual-based and small-scale research, considering the possible combinations of taxonomic groups, ecosystem types and geographical configurations, it is highly desirable to allow users to define taxonomic, geographical and ecosystem scopes of interests and generate biodiversity indices on the fly through a query interface. This is especially true for the situations when domain-knowledge is lacking and visual exploration is necessary as a first step of investigation. In addition, while taxonomic hierarchy (e.g., family-genus-species) was the primary source for grouping species and computing biodiversity indices, there are increasing interests in generating biodiversity indices across geographical regions based on phylogentic groupings of species which are generally much more dynamic than taxonomic grouping. This further warrants developing a database based solution that allow efficient dynamic queries. The query interface may significantly reduce or even eliminate the computing skills (e.g., GIS) needed for sophisticated species distribution data preprocessing. It is desirable to allow users to dynamically define a group of species based on a taxonomic hierarchy or some phylogenetic criteria, dynamically define a region of interests in a particular ecosystem and compute the biodiversity indices accordingly.

In our previous work (Zhang and Gruenwald 2008), we have developed the LEEASP prototype system to facility users visually exploring integrated taxonomic, geographical and environmental data. USGS Little tree species distribution data in ESRI's shapefile format was rasterized at a half degree spatial resolution. While the prototype system met its design goals, the main-memory data structure to associate taxonomic data with raster cells, i.e., a two-dimensional array of bit vectors, is not scalable. Replacing the main-memory data structure with a database backend query interface and providing efficient query support is the primary motivation of this research.

In parallel effort to our work on using a spatial database to manage the NatureServe's animal species distribution data, the Social-Economic Data and Application Center (SEDAC) at Columbia University has rasterized the original ESRI datasets in shapefile format into GEOTIFF images at 30 arc seconds resolution and make them ready to be downloaded through a Web-interface (http://sedac.ciesin.columbia.edu/species/) for individual species. They have also derived a family richness grid dataset by counting the number of species for all the species families. It is unclear about the storage requirements for the intermediate and final datasets. Our work goes a step further by allowing users to define regions and taxonomic groups dynamically in computing biodiversity indices. In addition, we strive to minimize query response time at server side to allow a variety of customer applications to utilize the database backend, e.g., the works reported in (Zhang et al 2007, Zhang and Gruenwald 2008, He and Zhang 2009).

Polygonal data rasterization and hierarchical spatial representation using tree structures are two related and well-studied areas in spatial databases. Optimal quad-tree construction algorithms from raster datasets have been discussed extensively in (Shaffer and Samet 1987). Generating quad-trees from polygons directly was addressed in Hunter's Ph.D. dissertation (1978), Mark (1985) and Lattanzi and Shaffer (1991). However, there are two problems for us to apply these algorithms directly in managing species distribution data. First of all, most of existing algorithms are specifically designed for quad-trees, i.e., a parent node has exactly 2*2 =4 child nodes. It is unclear whether they can be easily adapted for spatial partition trees with arbitrary user-specified number of children along both x and y directions, i.e., user-defined hierarchical raster tessellation. Second, while the algorithms may work well for simple polygons, it is unclear whether they can be extended to handle complex polygons, e.g., polygon with holes, which is typical in species distribution data. Finally, we are not aware of any existing

Authorized licensed use limited to: City College of New York. Downloaded on April 23,2010 at 16:12:23 UTC from IEEE Xplore. Restrictions apply.

open source software packages that implement these algorithms and are ready to use.

III. VF-SP TREE

The VF-SP tree is an extension to traditional quad-tree structures with the following features. VF-SP tree allows users to specify the levels of granularity that they consider important. For example, users may want to perform analysis at the 1 degree, 10 minutes and 1 minute scales. We thus can make the tree five levels as shown in Fig. 1. Each tree node has a pointer pointing to its parent node, an array of pointers pointing to its children nodes and a vector data structure to store the y values of line segments that are completely fall under the tree node along the x direction. The utilization of the vector data structure will be clearly shortly.

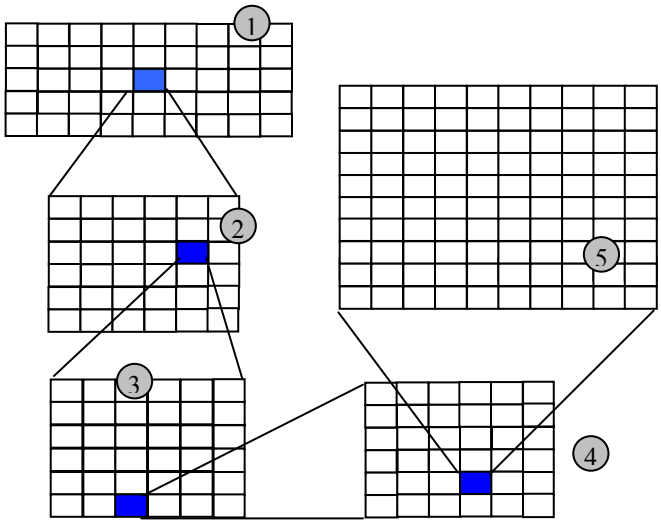

Fig. 1 An example of Variable-Fanout Space Partition Tree

To use the unified grid system to represent all the polygons of species range maps, we first adopt the scan-line algorithm implemented in the GDAL package to rasterize the boundary of each ring of the polygons. For each pair of boundary cells (left x1 and right x2) at scan line y, we fill the cells in between at the line by hierarchically divide the cells into segments aligning with the space partition tree hierarchy An example is shown in Fig. 2. The key idea for aligning the cells with the space partition tree is to divide cells between x1 and x2 at line y into three parts recursively, starting from the root of the tree. The middle part is the cells that align with the grid boundaries at the current level whose indices can range from zero to the number of children along the x (usually longitude) direction (e.g., 0-5 for level 2 in Fig. 1). Also in this step, y value will be pushed into the vectors associated with the child nodes covered by the middle part. The left and right parts are the remainder cells that fall within the scope of a single child node at the current level and thus need to be processed at the next level. This process is performed recursively until the finest scale is reached. Note that any one or two parts out of the three parts may be missing. This step generates a pseudo tree for a range map. We call it pseudo because many cells are stored in the vectors, not the tree nodes.

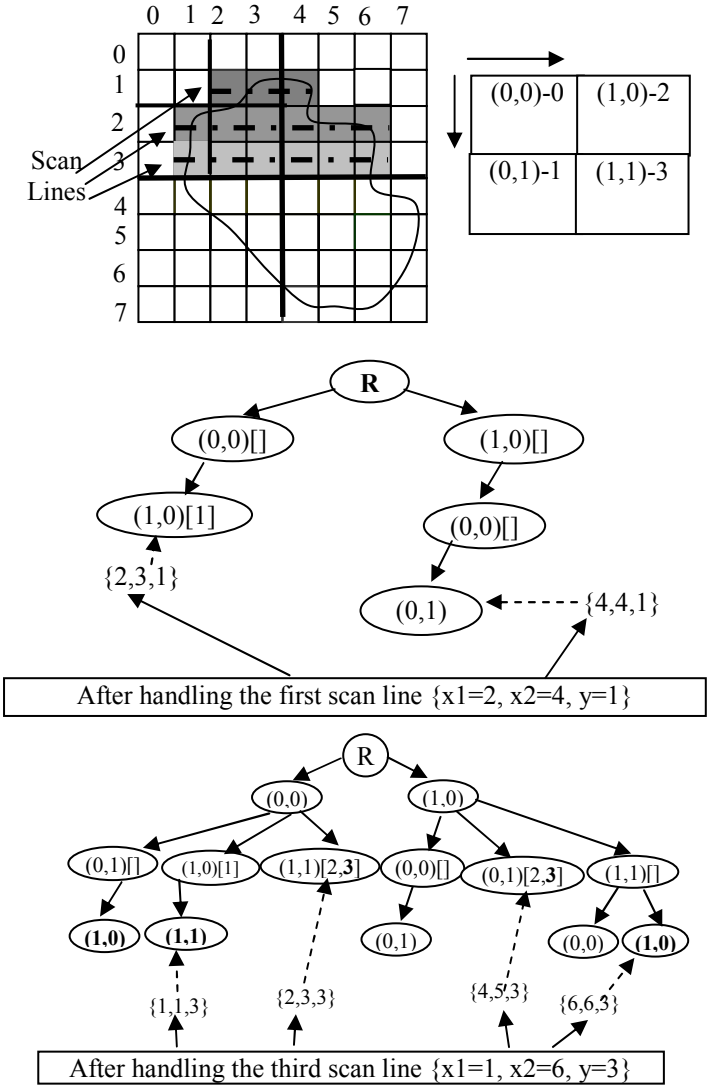

Fig. 2 Illustration of the Process of Building a Pseudo VF-SP Tree

After all the scan lines of all polygons in a layer are processed, we start the tree trimming process to eliminate the vectors associated with the tree nodes and build a real space partition tree. The tree trimming process is as follows. For a VF-SP tree node, first, we examine the size of the vector. If the size is the same as the number of cells (at the finest scale) along the y direction, then we know that the node is full. We thus can delete all its children nodes and mark the node as a leaf node in representing the range map polygons (e.g., node N2 in Fig. 3). Otherwise we push down the elements in the vector to its corresponding child nodes (e.g., node N1 in Fig. 3). This process starts from the root of a VF-SP tree and is carried out recursively until a leaf node is reached. For a non-leaf node, after the pushing down process is finished, we also

Authorized licensed use limited to: City College of New York. Downloaded on April 23,2010 at 16:12:23 UTC from IEEE Xplore. Restrictions apply.

check whether all its children are full. If so, we delete all its children nodes and mark it as a leaf node.

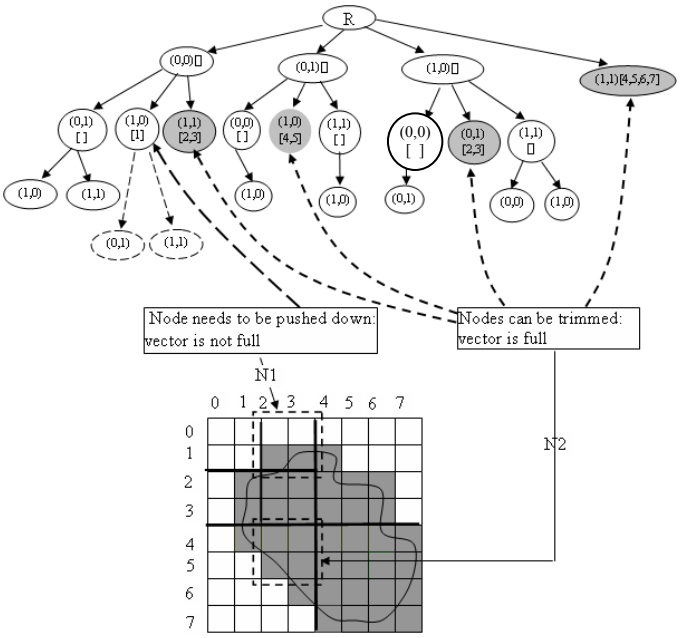

Fig. 3 Illustration of Trimming and Pushing Down Nodes during VF-SP Tree Construction

IV. From VF-SP Trees to Spatial Databases

While we can derive quite some biodiversity statistics for predefined ecoregion or administrative regions (see Zhang et al 2007 and references thereafter for more information) in an offline mode, as discussed previously, the capability of providing fast responses for dynamic queries in an online mode makes spatial databases a more preferable choice in real applications. We have decided to dump the VF-SP trees into a spatial database and use it to manage the species distribution data for the following reasons. First of all, The VF-SP is a main-memory data structure. While VF-SP trees, similar to quad-trees, are efficient in representing individual species spatial distributions, it is unlikely that a large number of VF-SP trees can be held in memory simultaneously. As such, using a disk-resident spatial database to make VF-SP trees persistent is desirable. Second, our ultimate goal is to allow customer applications, e.g., those reported in (Zhang et al 2007, Zhang et al 2008, He and Zhang 2009), to connect to a server program and query biodiversity indices dynamically to ease the burden of managing large-scale species distribution data inside the applications. Spatial databases, such as PostgreSQL/PostGIS, provide sophisticated indexing and structured query functionality which match our goals very well.

A straightforward approach to dump the VF-SP trees into a spatial database is to traverse each VF-SP tree sequentially. For each of the leaf nodes in a VF-SP tree, we compute the bounding box of the cell. The bounding box together with the species identifier consist a database tuple. We term this approach as "fully clipped" as the distribution polygons of each individual species are clipped into boxes according to its VF-SP tree construction. We can index on the bounding box and/or the species identifier column to expedite query processing. One problem with such approach is that if the geographical region corresponding to a box has N species distributed in, the box will be duplicated N times in the spatial database. Subsequently, the storage, indexing time and query processing time will be increased significantly, if N is big. In addition, it is conceivable that when we index on the box column using GIST (Generalized Search Tree) in PostgreSQL database, the leaf node of the index tree will contain multiple tuples that have the same geometric values. These tuples could span multiple database pages which may reduce the selection power of the spatial index tree at the bottom levels.

The second approach, which is more sophisticated from development perspective, is to combine the traversal outputs of multiple VF-SP trees and associate each unique leaf node of the VF-SP trees with a species list. We term this approach as "Geometrically Clipped", as the species with a same box derived from their respective VF-SP trees are combined. The approach, as shown in the experiment section, is what we found most suitable for making large scale species distribution data persistent in a spatial database after rasterization based on the VF-SP tree structure.

In the "Geometrically Clipped" approach, the whole study area that has at least one species distributed, i.e., the geometric union of all polygons of species distribution data, is clipped into non-overlapping boxes at each hierarchy level. Rather than associating species identifiers with only leaf nodes, we have decided to allow intermediate nodes to associate with species identifiers. While the decision is likely to increase the number of records corresponding to the boxes moderately, the number of records corresponding to the combinations of box identifiers and species identifiers can be significantly reduced. This is due to the reason that a VF-SP tree can have a large number of fanouts.

For comparison purposes, the shapefiles of the species distribution datasets is imported into PostgreSQL databases through a tool provided by PostGIS. In this approach, each polygon in the distribution dataset of a species is stored in PostgreSQL as a tuple. The baseline approach is termed as the "non-clipping", or "original" approach.

V. Performance Evaluations

As discussed previously, the motivation of using a spatial database to manage rasterized species distribution data is to support deriving biodiversity indices by processing dynamic queries with compositions of different spatial, taxonomic, phylogenetic and ecological criteria. We assume taxonomic, phylogenetic and ecological criteria can be fulfilled by relational queries which will generate a set of species identifiers to refine spatial queries and can be easily realized in a database. Due to the reason that no phylogenetic and ecosystem data come with the NatureServe species distribution data and the difficulty of identifying representative queries based on taxonomic criteria, in this

Authorized licensed use limited to: City College of New York. Downloaded on April 23,2010 at 16:12:23 UTC from IEEE Xplore. Restrictions apply.

study, the performance evaluations are mostly on spatial queries. However, we would like to point out that the database design is ready to include these criteria in query processing.

A. Experiment Setups

All experiments are performed on a Dell Precision T5400 workstation with 8G memory and 1.5T disk space using PostgreSQL 8.3.5 and PostGIS 1.3.6. In the experiments, we use the hierarchical raster tessellation for the globe as illustrated in Fig. 1. The hierarchy has five levels, the first level has 10*5 cells with a resolution of 36 degrees, each first level cell has 6*6 second level cell with a resolution of 6 degrees, each second level cell has 6*6 third level cells with a resolution of 1 degrees, the third level cell has 6*6 fourth level cells with a resolution of 10 minutes and finally each fourth level cell has 10*10 fifth level cells with a resolution of 1 minutes. Thus the whole globe is divided into (10*6*6*6*10)*(5*6*6*6*10) = 233,280,000 1 minute resolution cells.

We download the species range maps from NatureServe's Website (http://www.natureserve.org/getData/animalData.jsp). We discard species that have only point data and species whose ranges are less than the size of the cell of the most detailed resolution (1 minute). The numbers of species tested are as the follows: mammals (1693 species), birds (4148 species) and amphibian (5816 species). Due to space limit, we only report the results derived from the bird data whose distribution maps are significantly more complex than the other two. The numbers of records in the tables derived by using the three approaches are as the following: original (708,326), the "geometrically clipped" (15,821,307) and the "fully clipped" (240,414,497).

In all experiments, we use four window sizes, 0.1, 0.5, 1 and 5 degrees for both width and height. For each experiment, we first randomly generate the center and then compute the bounding box of the query window. Assuming the query window is given in the form of (x1 y1, x2 y2), a typical query in PostgreSQL looks like the following.

```
Select sp_id, sum(area(intersection_geom)) from
(select sp_id, ST_Intersection(bk_loc, 'BOX(x1 y1, x2
y2)'::BOX2D) AS intersection_geom from TB where
geometry_column && 'BOX( x1 y1, x2 y2)'::BOX2D ) as b
group by sp_id having sum(area(intersection_geom))>0;
```

Here TB is the table name and the geometry_column is the name of the column storing geometric data, which. is the original polygons in the baseline approach and boxes in the two clipped approaches. The **WHERE** clause in the query string is used to eliminate empty intersections that are undesired but returned by PostgreSQL/PostGIS by default. As we can see from the query string, the non-spatial query criteria (e.g., taxonomic and phylogenetic) in generating biodiversity indices can be easily incorporated in to the **WHERE** clause and the **HAVING** clause if necessary.

In the experiments, if any portion of the query window is outside of the study area or the query window does not result in at least two species, the experiment will be discarded. For each query window, the experiments are repeated 100 times and only the experiments with valid query results are recorded for further analysis. The numbers of valid tests for the four query windows are 30, 35, 27 and 35, respectively. Note that percentages of the valid tests agree with the percentages of the land on the Earth's surface in general.

B. Results

Fig. 4 is the scatter plot for the total 127 tests (by combining all four test window sizes) with the x axis representing the numbers of species in the query results and the y axis representing the query response times in seconds. Each test is performed against the three tables derived from the three approaches, respectively. As we can see from Fig. 4, the geometrically clipped table overall has the best performance. The fully clipped approach performs the worst except for the tests result in small numbers of species.

Table 1 lists the averages and maximums of the query response times for the tests derived from the three approaches using the four window sizes. As we can see from the table, the geometrically clipped approach has the best performances for both average and maximum response time under all the four query window sizes. More importantly, the average query response times for query window sizes 0.1, 0.5 and 1 degree are below one second. The results show that it is feasible to use PostgreSQL as a database backend for various customer applications with acceptable response time. Although the geometrically clipped approach is still better than the baseline non-clipping approach under query window size 5 degrees with respect to both average and maximum query response times, it is not realistic to use any of them for interactive applications that involve large query windows. A solution might be to pre-compute biodiversity statistics for large grids, e.g., 5 by 5 degree. Large query windows can be decomposed into the predefined grids plus some smaller query windows. The pre-computed results and the dynamic query results are then merged to generate final results. More efforts are needed to evaluate this hybrid approach.

For small query window sizes (0.1 degree to 1 degree), the results presented in Table 1 suggest that the proposed geometrically clipped approach can be up to 300 times better (0.12s vs. 36.88s for window size 0.1 degree) than the baseline non-clipping approach with respect to average query response time. The experiments have demonstrated clearly the superiority of the proposed approach

VI. Summary and Future Work Directions

In this study we have addressed the issues related to scalable management of large-scale species distribution data in a spatial database environment. Motivated by the inadequacies of layer-based data management in GIS and insufficient support for raster-tessellated distribution data in spatial databases, we have developed a VF-SP tree structure to represent species distribution maps using a user-defined hierarchical raster tessellation. We have also developed an algorithm to convert multiple VF-SP trees representing a large

Authorized licensed use limited to: City College of New York. Downloaded on April 23,2010 at 16:12:23 UTC from IEEE Xplore. Restrictions apply.

number of species distribution maps into a spatial database for efficient query processing. Experimental results using more than four thousands bird species data have demonstrated that the proposed approach can be 30-300 times faster than the baseline approach for the average query response time using a query window size of 0.1 degree to 1 degree. The average query response time is below 1 second for query window size up to 1 degree which makes it possible to use the spatial database backend in a variety of customer applications.

For future work, first of all, we would like to develop a cost model for the "geometrically clipped" approach. The cost model will help to choose the best approach under different query window sizes. Second, as discussed in the experimental results section, we would like to explore the possibility of combing pre-computation and dynamic query processing to efficiently answer queries with large window sizes. Our goal is to reduce average query response time to below one second for any query window size. Finally, the work reported in this paper is part of a larger effort in building a visual exploration system for large scale species distribution data as reported in (Zhang and Gruenwald 2008). Instead of the main-memory data structures used in the previous study, we plan to use the spatial database backend for building a scalable system that can ultimately handle all known (more than a million) species.

REFERENCES

[1] Gaede, V. and O. Gunther (1998). "Multidimensional access methods." ACM Computing Surveys 30(2): 170-231.

[2] He, K. and Zhang, J., 2009. Testing the correlation between beta diversity and differences in productivity among global ecoregions, biomes, and biogeographical realms. Ecological Informatics 4(2), pp.93-98.

[3] Hunter, G. M. and K. Steiglitz (1979). "Operations on Images Using Quad Trees." Ieee Transactions on Pattern Analysis and Machine Intelligence 1(2): 145-153.

[4] Koleff, P., K. J. Gaston, et al. (2003). "Measuring beta diversity for presence-absence data." Journal of Animal Ecology 72(3): 367-382.

[5] Lattanzi, M. R. and C. A. Shaffer (1991). "An optimal boundary to quadtree conversion algorithm." CVGIP: Image Understanding 53(3): 303-312.You

[6] Mark, D. M. and D. J. Abel (1985). "Linear Quadtrees from Vector Representations of Polygons." Ieee Transactions on Pattern Analysis and Machine Intelligence 7(3): 344-349.

[7] Moreno, C., I. Zuria, et al. (2006). "Trends in the measurement of alpha diversity in the last two decades." Interciencia 31(1): 67-71.

[8] Ricotta, C. (2005). "Through the Jungle of Biological Diversity." Acta Biotheoretica 53(1): 29-38.

[9] Shaffer, C. A. and Samet, H., 1987. Optimal Quadtree Construction Algorithms. Computer Vision Graphics and Image Processing 37, 402-419.

[10] Wilson, M. V. and A. Shmida (1984). "Measuring Beta Diversity with Presence Absence Data." Journal of Ecology 72(3): 1055-1064.

[11] Zhang, J. and L. Gruenwald (2008). Embedding and extending GIS for exploratory analysis of large-scale species distribution data. Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems. Irvine, California, ACM.

[12] Zhang, J., D. D. Pennington, et al. (2007). "GBD-Explorer: Extending open source java GIS for exploring ecoregion-based biodiversity data." Ecological Informatics 2(2): 94-102.

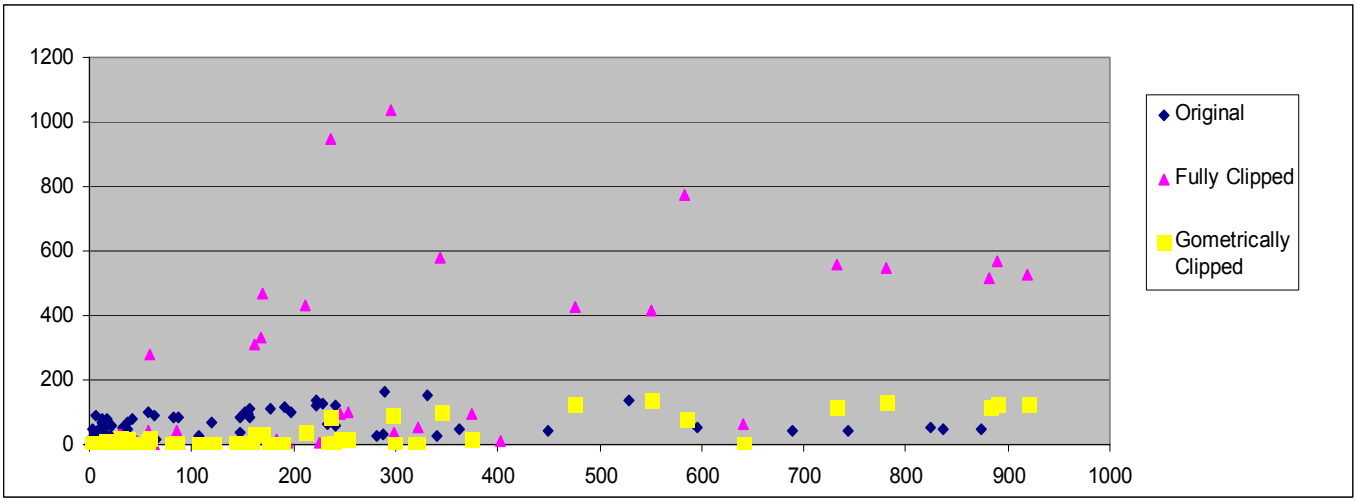

Fig. 4 Scatter Plot of Query Response Time versus Resulting Numbers of Species in All 127 Tests Using the Three Approaches

Table 1 Average and Maximum Response Times under Four Query Windows for the Three Approaches

Window Size	Original (Baseline)		Fully Clipped		Geometrically Clipped	
	AVG	MAX	AVG	MAX	AVG	MAX
0.1	36.88	127.28	1.40	7.90	0.12	0.58
0.5	27.75	117.66	4.48	61.78	0.39	1.90
1	28.14	121.11	6.25	53.25	0.92	5.11
5	51.82	164.78	263.32	1034.35	44.10	136.73

Authorized licensed use limited to: City College of New York. Downloaded on April 23,2010 at 16:12:23 UTC from IEEE Xplore. Restrictions apply.