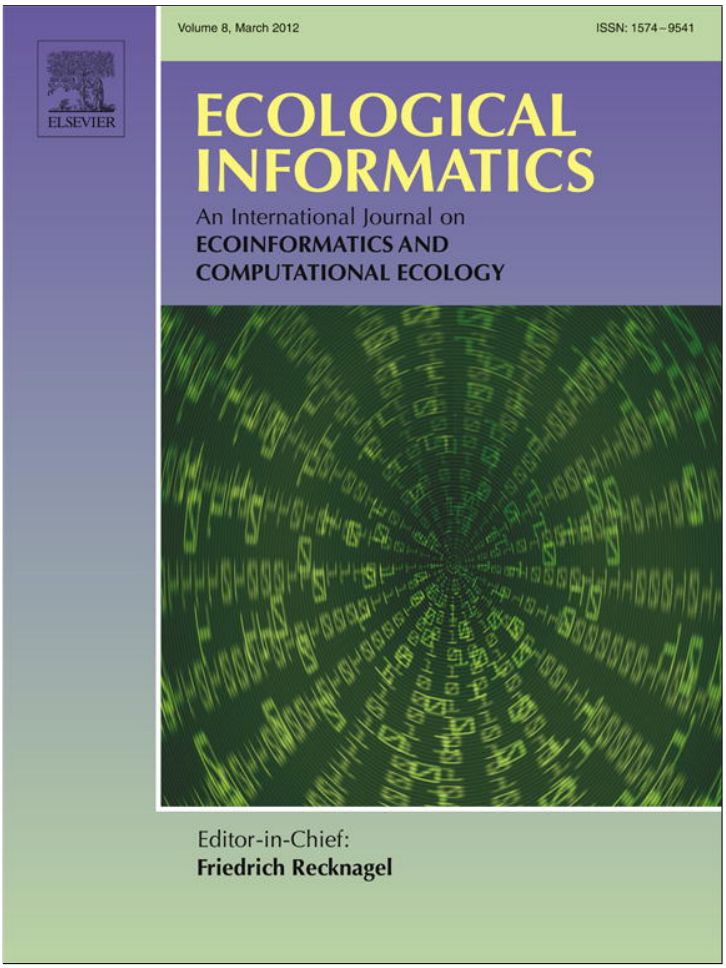


Ecological Informatics 8 (2012) 68–77

Contents lists available at SciVerse ScienceDirect

Ecological Informatics

journal homepage: www.elsevier.com/locate/ecolinf

A high-performance web-based information system for publishing large-scale species range maps in support of biodiversity studies

Jianting Zhang *

Department of Computer Science, The City College of the City University of New York, New York City, NY, 10031, United States

ARTICLE INFO

Article history:
Received 25 August 2011
Received in revised form 17 January 2012
Accepted 22 January 2012
Available online 31 January 2012

Keywords:
High-performance
Species range map
Quadtree
Web-based
Spatial databases
GIS

ABSTRACT

Functionality, performance and scalability are critical to Web-based information systems for publishing and disseminating large-scale species distribution data. Existing systems do not support dynamic spatial window queries on large-scale species range maps that are important to compute alpha and beta diversities for biodiversity analysis and modeling. In this study, we have developed a main-memory based novel quadtree data structure to represent large-scale species range maps and support dynamic spatial window queries to retrieve a list of species and their area sizes within a query window efficiently. Using the NatureServe's 400**0**+ bird species range maps, experiment results have shown that the memory footprint of the proposed quadtree data structure representing the range maps of all the species is about 1/6 of the quadtree derived by combining individual quadtrees each representing a species range map. The experiment results have also demonstrated that the query response times of our main-memory spatial database are well below a fraction of a second for query windows as large as $10 \times 10°$, which are 2–3 orders better than using a typical disk-resident spatial database system.

1. Introduction

More than a million of species have been recorded in several repositories, such as the repositories provided by the "Catalogue of Life" project (COL, 2010) and the uBio project (uBio, 2010). The range maps of more and more species are also becoming available which allows researchers to explore the relationships among species distributions 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. Last decade saw massive developments of biodiversity related information systems on the Internet and the available species distribution data has been increased dramatically (Bisby, 2000). As an example, at the time of writing, the Global Biodiversity Facility (GBIF) data portal has hosted 322, 366, 001 species occurrence records (GBIF, 2012).

While the point locations of species occurrences themselves are useful to model species distributions at individual species level (Guisan and Zimmermann, 2000), range maps compiled from museum records and scientific literature play an important role in global and regional biodiversity studies, especially when correlating biodiversities with environmental variables (Field et al, 2009, Mittelbach et al, 2001; Qian, 2010; Waide and Willig, 1999). USGS Digital Representations of Tree Species Range Maps dataset from "Atlas of United States Trees" by Elbert L. Little, Jr. consists of 679 tree species (USGS, 2006) and has played important roles in USGS's vegetation-climate modeling. More recently, NatureServe published Digital Distribution Maps of the Birds of the Western Hemisphere Version 3.0 which covers 4,273 bird species (NatureServe, 2010b). Similar range maps for 1737 mammal species of the Western Hemisphere and more than 6000 world's amphibians are also available from NatureServe's website (NatureServe, 2010b). 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, query processing algorithms and information system architectures, to effectively support global biodiversity research and subsequent species conservation practices.

Web-based information systems provide an ideal means to publish and disseminate species distribution data. Quite a few operational systems are available, such as the World Wild Fund (WWF) WildFinder (WWF, 2010), the NatureServe Explorer (NatureServe, 2010a) and the USDA PLANTS database Web interface (USDA, 2010). A few research systems also have been developed, such as WebGRMS (Greene et al, 2007), GBIF-MAPA (Flemons et al, 2007), SFMN-GeoSearch (Gonzales et al, 2009) and OBIS-SEAMAP (Halpin et al, 2009). However, most of them only allow query/visualize the distributions of a single species at a time and/or only support point locations or predefined administrative or ecological regions. While the functions are well supported by the state-of-the-art Web-GIS and database technologies, it is technically challenging to support querying species range maps dynamically with

* Tel.: +1 212 650 6175.
E-mail address: jzhang@cs.ccny.cuny.edu.

1574-9541/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.ecoinf.2012.01.004

arbitrarily defined spatial windows. Indeed, large-scale species range maps in vector polygon format can have very large data volumes and generally are computationally expensive to process. Our experiments on building such a system for publishing the NatureServe bird range maps (NatureServe, 2010b) that includes three quarters of a million complex polygons have indicated that the performance is too poor to be practically useful (Zhang, 2009).

Among many types of visual, analytical and modeling tasks on species distribution data, a basic operation is to retrieve a list of species in a spatial window so that alpha and beta diversities based analysis can be further performed. As discussed in detail in the next section, while quite a few species distribution datasets have already associated a list of species with predefined ecological and administrative regions which makes the query trivial, the availability of species range maps allows using arbitrary spatial windows to query against species distribution data for fine-grained studies. 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 Geographical Information Systems (GIS), these range maps are not readily usable for the purposes beyond simple display. The reason behind is that species distribution analysis, especially biodiversity related research, often requires cross-layer queries (see more details in the next section) which is poorly supported by current GIS and spatial databases (Samet, 2004). While sophisticated GIS and spatial databases, such as ArcGIS, are empowered by spatial indexing techniques (such as R-Tree or Quadtree, see Gaede and Gunther, 1998; Samet, 2005) and are very efficient in performing intra-layer spatial queries on geometric objects with no or have little overlap, they are usually not good at querying thousands of layers in a single query. This is largely due to the fundamental limitations of layer based vector data model in supporting co-location type query as discussed by Samet (2004) and Zhang and Gruenwald (2008). Co-location type queries need to be transformed to N-way spatial joins before they can be processed in current spatial databases. In GIS, a procedure language would be needed to loop through all the layers and temporary tables would also be needed to keep the intermediate results for processing a query. They are 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 have no or very little overlap, they can be highly overlapped in the combined layer which makes spatial indexing much less effective. More importantly, queries to compute a 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 range maps and user defined query regions. The computation is usually quite expensive which makes the approach not scalable in applications that require fast responses, for example, interactive visual explorations and online modeling.

In this study, we aim at developing data structures, query processing algorithms and information systems to overcome the scalability and efficiency related issues in publishing and disseminating large-scale species range maps and supporting visual explorations, analysis and modeling. Built on top of our previous works on rasterizing species range maps to derive quadtrees (Zhang, 2009) and using an extensible database to manage quadtrees (Zhang et al, 2009a), we have developed a Main-Memory Spatial Database (MMSDB) based solution to speed up spatial window queries on large-scale species range maps. Two middleware modules also have been developed to provide essential functionality for both interactive visual explorations and biodiversity analysis/modeling through Web services. Our experiments using NatureServer's 400**0**+ bird species data (NatureServe, 2010b) have shown that the main memory query processing engine we have developed can be 2–3 orders faster than regular disk-resident databases and can answer spatial window queries in a fraction of a second even for large query windows. The Web-based information system is efficient and effective in publishing large-scale species range maps.

The reset of the paper is arranged as the following. Section 2 introduces the background and related works of the proposed research and development efforts. Section 3 presents the system architecture and the implementations of key components. Section 4 reports the experiment results on 400**0**+ NatureServe bird species range maps. Finally Section 5 is the summary and future work directions.

2. 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. Several enabling technologies have made biodiversity data available at much finer scales in the past decade. First, progresses in molecular systematics have made species identifications much easier. Second, GPS technology has been widely used in modern field survey and geo-referring technology has been successfully applied to transforming descriptive museum records to geographical coordinates. Third, database technologies in general and spatial databases in particular have been widely used to effectively manage not only species presence locations, but also related taxonomic and environmental data. Fourth, GIS have been extensively used for visualization and spatial analysis of biodiversity data. Some of the spatial analytical functions have directly contributed to species distribution data modeling and analysis Finally, the newly emerging Cyberinfrastructure technologies (e.g., metadata, ontology, Web Services and scientific workflow) have made exchanging and sharing species distribution data over the Web much easier. The increasingly available species distribution data can potentially leverage our understanding of global and regional biodiversity patterns to a new level through interactive visual explorations, analysis and modeling (Bisby, 2000; Guralnick and Hill, 2009; Soberon and Peterson, 2004).

A variety of biodiversity indices have been developed for analysis and modeling purposes. We refer 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 assemblages, 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. Biodiversity indices can be computed based on either a vector or a raster geographical tessellation (Zhang and Gruenwald, 2008). In both cases, biodiversity indices are associated with a basic geographical unit, i.e., a polygon representing a predefined region in the vector tessellation or a grid cell in the raster tessellation. When species distribution maps are provided as individual GIS layers, as shown in Fig. 1, cross layer queries are needed to generate biodiversity indices. Most of existing biodiversity studies use one or more predefined geographical regions (such as eco-regions or administrative regions) and predefined groups of species (such as specific families or genus). While this is a viable solution for individual-based and small-scale research, considering the possible combinations of taxonomic groups, ecosystem types and geographical configurations (as illustrated in Fig. 1), 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. The query interface may significantly reduce or even eliminate the computing skills (e.g., GIS) needed for sophisticated data preprocessing. In the example shown in Fig. 1, 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.

Most existing biodiversity studies on a large number of species use very coarse spatial resolutions. For example, the World Wild Fund (WWF) WildFinder database (WWF, 2006) uses the world's top-level ecoregion as the basic geographical units to associate ecoregions

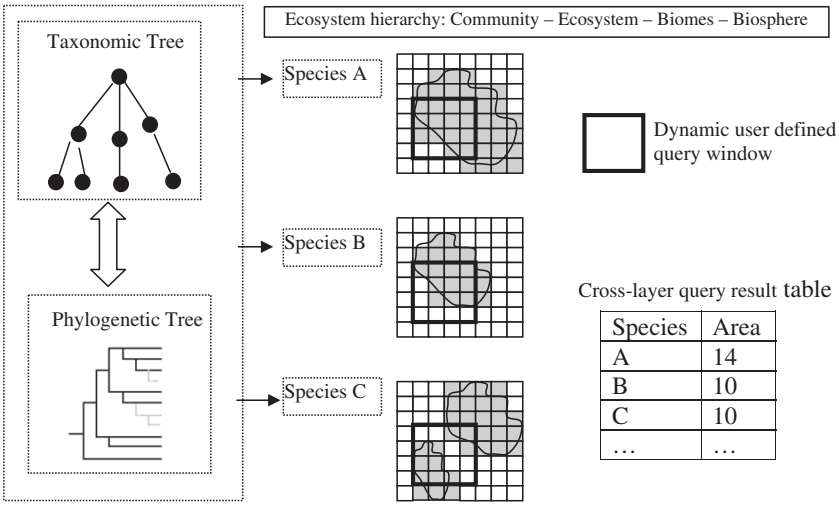


Fig. 1. Illustration of generating biodiversity indices through cross-layer query.

with species lists. The ecoregion can be as large as millions of square kilometers which make it very hard for seeking possible relationships between biodiversities and the environment variables. The GBIF data portal (GBIF, 2012) utilized a pre-computing approach to improving query performances by allowing only one predefined query window size at a zoom level. While the technique may work well for overview purposes over the Web, they are not suitable for precise queries that involve arbitrary sizes of query windows. For the studies based on raster tessellation, at the global scale, very few studies use a grid cell finer than 1° and some studies use even 15° resolution (Proches, 2005) despite both species distribution data and environmental data is available at finer resolutions. Technical difficulties in handling large-scale datasets might be one of the important factors in the mismatches between the available and utilized high resolution data. The Social-Economic Data and Application Center (SEDAC) at Columbia University has rasterized the NatureServe bird species range maps into GEOTIFF images at 30 arc-seconds resolution and made them ready to be downloaded through a Web-interface for individual species (SEDAC, 2010). SEDAC has also derived a family richness grid dataset by counting the number of species for all the species families. Unfortunately, no dynamic queries are supported.

The Service-Oriented Architecture (SOA) has received considerable interests in environmental sciences, e.g., Jaeger et al., 2005, Goodall et al, 2008; Berrick et al, 2009; Kooistra et al, 2009; Zhang et al, 2009b; Yang et al, 2010, in general and biodiversity data processing in particular, e.g., Zhang et al., 2005; Frehner and Brandli, 2006; Best et al, 2007; Fook et al, 2009, due to its potential in inter-linking information systems in interoperable ways. The performance of SOA applications heavily rely on the efficiency of the query operations provided by the underlying GIS and/or databases. Tailoring data structures and query processing algorithms for publishing large scale species range maps is likely to improve the performance of SOA-based applications significantly. We would like to note that GIS tools have been extensively used in managing biodiversity and environmental data and performing biodiversity analysis and modeling in many aspects. We refer to (Foody, 2008), (Steiniger and Hay, 2009) and (Boyd and Foody, 2011) for overviews and more details. However, to the best of our knowledge, we are not aware of previous works aimed at supporting dynamic queries efficiently on large-scale species range maps.

In our previous work (Zhang and Gruenwald, 2008), we have developed the LEEASP (Linked Environment for Exploratory Analysis of Large-Scale Species Distribution Data) prototype system to facilitate 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, it uses a simple data structure to associate taxonomic data with raster cells, i.e., a two-dimensional array of bit vectors, which can be very sparse. The simple data structure may require excessive memory for high-resolution rasters and/or a large number of species and thus is not scalable from a data management perspective. We have also built a specialized quadtree by rasterizing each species distribution polygons using a scan-line fill algorithm (Zhang, 2009) and associate species identifies with different levels of nodes of the quadtree. Using the classic linear quadtree techniques (Samet, 2005), the quadtree nodes can be stored as tree paths and the associated species identifiers can be stored as arrays in the PostgreSQL database (PostgreSQL, 2010). The tree paths can then be indexed by PostgreSQL to speed up query processing using a query transformation technique as reported in (Zhang et al, 2009a). Both the average and maximum query response times are in the order of a few seconds for query window up to $10\times10^{\circ}$ for the 400**0**+ NatureServe West Hemisphere bird species data using a 14 level quadtree spatial tessellation (16,384*16,384 cells). However, it is desirable to further improve query performances and reduce response times to sub-second level for lager numbers of species at the finer spatial scales and allow a variety of customer applications to utilize the database backend, e.g., our works reported in (He and Zhang, 2009; He et al, 2009; Zhang and Gruenwald, 2008; Zhang et al, 2007).

Studies have shown that the differences between CPU, memory and disk I/O speeds have increased significantly over the past decade (Hennessy and Patterson, 2006). Currently the bandwidth of hard drives of commodity servers is in the order of 100 megabytes per second whereas the main-memory bandwidth of the same servers can be tens of gigabytes per second, i.e., two orders higher. The significant performance differences have motivated many in-memory applications that require high performance, including main-memory databases (Grund et al., 2010). As memory capacities of typical computer systems are getting larger while the price per gigabyte have dropped to under $50, it is natural to explore the main-memory option in managing large-scale species distribution data for fast data processing. It is crucial to develop domain-specific novel data structures, query processing algorithms and information system architectures to realize the potential performance gains offered by main-memory based information systems.

Alternative to the proposed approach that utilizes efficient and main-memory-based data structures and algorithms to achieve high performance, the Map of Life project (MOL 2012) utilize parallel processing power of Cloud computing to achieve a similar purpose. Indeed, the inherent parallelisms of the biodiversity data and targeted

queries allow easy parallelization across geographical regions and taxonomic groups. The state-of-the-art Cloud computing technologies and facilities have provided an economic means to address the same technical challenges. However, we consider the two approaches complementary rather than rival. The simple reason is that improving the computing efficiency of a single computing node will immediately improve the overall performance of a Cloud-based information system. For information systems (including their sub-systems) that can not scale up linearly, it becomes more important to increase the efficiency of individual computing nodes in a parallel/distributed system. Along the direction of parallel computing, we have also achieved 20× speedups on polygon rasterization using an Nvidia General Purpose Graphics Processing Unit (GPGPU) device with 448 GPGPU cores (Zhang, 2011). We plan to implement the quadtree indexing pipeline described in Section 3.1 on GPGPUs and integrate it with this system to speed up data preprocessing.

3. System architecture and components

We aim at developing an information system prototype that allow users to efficiently query large-scale species range maps and retrieve a list of species in an arbitrarily defined rectangular region, i.e., window queries in spatial databases. Although developing novel data structure and query processing algorithms for species distribution data is the key to high-performance, the system also includes components that utilize existing Web technologies to deliver base maps and visualize query results. The system architecture is illustrated in Fig. 2. In the following, the two most important components of the main-memory spatial database (represented by the dark gray box in Fig. 2), i.e., indexing and query processing, are presented in Sections 3.1 and 3.2, respectively, and, Section 3.3 provides technical details on the components that connect Web clients and Web services consumers with the database backend (represented by the light gray boxes in Fig. 2).

3.1. Quadtree indexing of species range maps

Before presenting the quadtree indexing approach to managing large-scale species range maps, we first explain why existing commercial and open source spatial databases, such as Oracle Spatial, Microsoft SQL Server Spatial and PostgreSQL/PostGIS, are inadequate in managing large-scale range maps. Given that existing species range maps are available in ESRI's Shapefile format and most of the existing spatial databases have provided tools to import shapefiles, it is straightforward to store the combined species range maps in polygonal vector format in the databases, index the polygons (e.g., using R-Trees) and support spatial window queries. As an example, the SQL syntax for querying all species in window (x1, y1, x2, y2) against a PostgreSQL database is as follows.

```
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 name of the table that stores the combined layers, including geometric information (geometry_column) and its associated relational information, such as species identifiers (sp_id), scientific names, taxonomic and phylogenetic hierarchies, etc. As we will be focusing on spatial window queries, we assume non-spatial criteria can be combined and integrated in the *WHERE* condition in the query. The problem with this straightforward solution is that the *ST_INTERSECTION* operation used in the query involves complex geometrical computation on polygons and is usually very expensive.

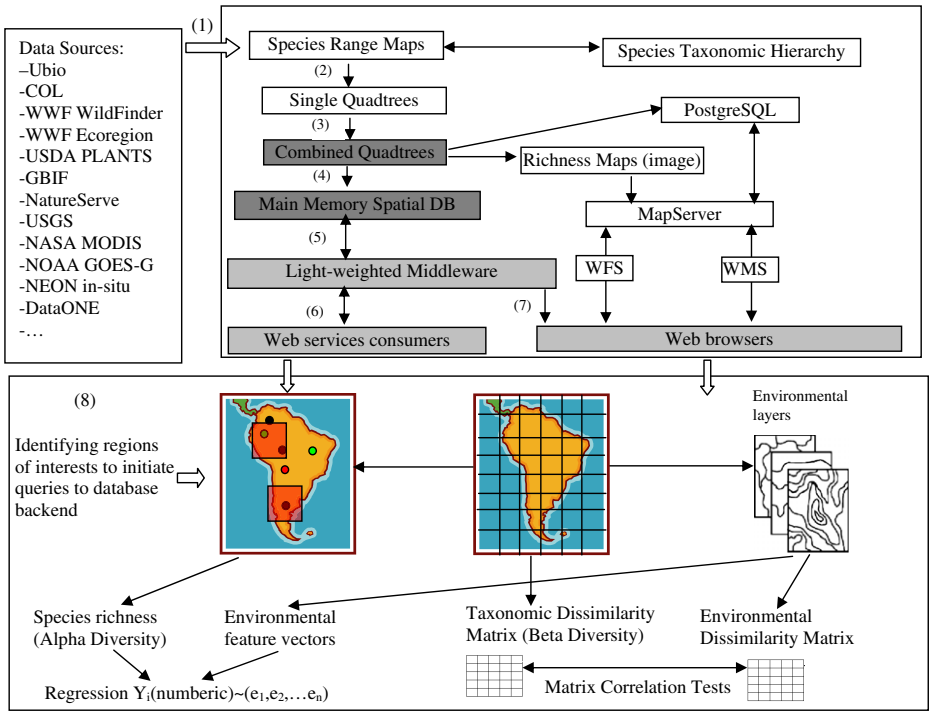


Fig. 2. System architecture and components. (1) Species range maps and taxonomic hierarchy are derived from various data sources (2) Rasterizing species range maps and construct individual quadtrees (3) Combining individual quadtrees to reduce memory footprint (Section 3.1) (4) The combined quadtrees in the main-memory database is used to speed up query processing (Section 3.2) (5) Light-weighted middleware to communicate with the query processing engine and the Web-based applications (Section 3.3) (6) Web services based applications for analytics and modeling (Section 3.3) (7) Web browser based applications for visualizations and visual explorations (Section 3.3) (8) Illustration of two potential applications: regression and matrix correlation between alpha/beta diversities and environmental variables (e.g., Mantel tests).

Unlike normal vector geospatial datasets whose polygons are usually disjoint and spatial indexing is very effective in filtering out a large portion of database records, polygons representing the distributions of a large number of species are highly overlapped which makes spatial indexing on the combined layer much less effective. As demonstrated in the experiments section, these polygons representing the distributions of bird species are very complex in the sense that there might be multiple rings associated with a non-convex polygon (e.g., due to holes) and each ring may have a large number of points. For typical spatial window queries in the order of a few degrees on global datasets, the query response time can be minutes or even longer which renders the solution inadequate for interactive applications.

Our solution is to rasterize these polygons based on quadtree representations (Samet, 2005). By converting polygons into tree-represented quadrants through pre-computing, retrieving quadrants that fall into a spatial query window involves only partial tree traversal which is considerably faster. An example of representing a polygon as a quadtree has been shown at the top of Fig. 3. A parent quadtree tree node has four children. A leaf node can be either black or white. A black node indicates the presence of the species in the quadrant (as part of a polygon) and a white node is the opposite. Correspondingly, the intermediate nodes are gray which can have white, black or gray children. To save storage space, only black nodes are stored and their identifiers are derived by concatenating quadrant indices (numbered 0–3) for all the nodes from the root to the black node. The identifiers are termed as quadtree paths for the respective black nodes. It is not difficult to see that quadtree is an efficient representation of large polygons (Samet, 2004). While many GIS software packages provide modules for vector-raster conversions (Theobald, 2005), usually no quadtree representations are explicitly generated or accessible to application developers. As such, we reuse the Variable-Fanout Space Partition (VF-SP) tree construction module that we have developed previously (Zhang, 2009) and configure it as a quadtree data structure. As polygons representing the distribution of a single species do not overlap, it is straightforward to build a quadtree to represent the range map of a single species. We next present the proposed quadtree data structure for indexing a large number of species to reduce memory footprint while facilitate efficient cross-layer query processing.

The motivation to use a specialized quadtree for a set of species rather than use a set of quadtrees each representing an individual species is obvious: many species are collocated in quadrants of different sizes and it is more efficient to associate a quadrant with a set of species as illustrated in Fig. 3. Instead of using classic quadtree data structures that stores the presence of quadrants only at the leaf nodes for the combined quadtree (mid-right part of Fig. 3, termed as classic combination), we associate a set of species identifiers with both leaf and intermediate nodes of the combined quadtree if the species distributions fully cover the respective quadrants (lower-right part of Fig. 3, termed as proposed combination). Storing species identifiers at the upper level of quadtree nodes eliminates the need to store the respective identifiers at the all of their descendent nodes. For the example shown in Fig. 3, while we need to store 18 identifiers in 9 quadtree nodes for the combined quadtree using the classic combination, we only need 6 identifiers in 6 quadtree nodes for the combined quadtree using the proposed combination. As the number of species gets larger, the memory saving will be more significant if the quadtree data structures are stored in main-memory. As we shall show in the experiment section, for the 4000+ NatureServe bird species data (NatureServe, 2010b), the average number of identifiers associated with a quadtree node is reduced from 110.7 to 4.8 using the proposed combination, a significant saving of memory consumption.

3.2. Spatial window query processing

Given a query window *w*, starting from the root of the combined quadtree *t*, the query processing algorithm is performed in a recursive manner to retrieve all the species distributed in the query window, including species identifiers and their area sizes measured as the numbers of raster cells, as shown in Fig. 4. The algorithm first (Step 1) checks whether the query window intersects with the quadrant

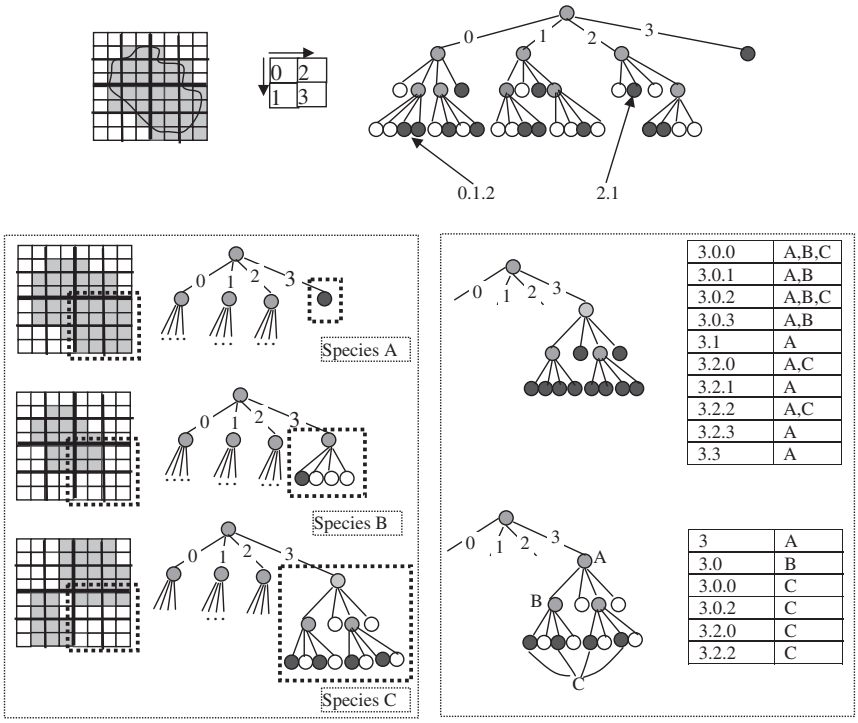


Fig. 3. Illustration of quadtree representation (top) and comparisons of representations of large-scale species range maps using individual quadtree (left), classic combination (mid-right) and proposed combination (lower-right).

```
Procedure WindowQuery(t, w, v)
Input:
•   t: a combined quadtree
•   w: a query window
Output:
•   v:  a vector of (species identifier, area size) pairs

Step 0: set n to t
Step 1: if w intersects the spatial extent of n
Step 2.1:   compute the intersection of w and the spatial extent of n and calculate area size s.
Step 2.2    for ach of the species identifies associated with n (id)
                add (id, s) pair to v and update the total area of the species.
Step 3:     if w completely contains if the spatial extent of n
Step 3.1        call procedure  output_within(n,w,v)
            else
Step 3.2        for each child node of n (n[i])
                    call procedure WindowQuery(n[i],w,v)
Step 4: output the (id, s) pairs in v

Procedure output_within (t, w, v)
Input and output: the same as WindowQuery

Step 0: compute the area size of the quadrant that t represent (s)
Step 1: for ach of the species identifies associated with n (id)
        add (id, s) to v and update the total area of the species
Step 2: for each child node of n (n[i])
                call procedure output_within(n[i],w,v);
```

Fig. 4. Spatial window query processing algorithm.

of the node that is being examined represents (assuming node n). If not, the algorithm skips n. The intersection area between n and w will then be computed and species identifiers associated with n and the intersection area will be added to the output in the form of (identifier, area) pairs (Step 2). The algorithm further checks whether the query window completely contains the quadrant that n represents. If so, the query algorithm recursively adds the (identifier, area) pairs for all the descendants of node n after intersecting the quadrants and the query window by using the helper procedure output_within (Step 3.1). Otherwise, each of the four non-empty child nodes of node n will be recursively processed by applying the query processing algorithm to the child nodes (Step 3.2). The process is also illustrated in Fig. 5 by using an example. The nodes (and the quadrants they represent) in dashed squares in Fig. 5 partially intersect with the query window and applying the query algorithm to all its child nodes recursively is needed. Differently, the nodes (and the quadrants they represent) inside the solid squares are completely contained by the query window and the identifiers associated with all its descendents need to be added to the output.

When a quadrant is completely contained in the query window, the number of raster cells at the finest resolution covered by the quadrant can be immediately computed based on the level of the quadrant and no real intersection computation is needed. However, when a quadrant partially overlaps with the query window, computing the intersection area is needed. Assuming the coordinates for the quadrant and the query window are (nx1, ny1, nx2, ny2) and (wx1, wy1, wx2, wy2), respectively, the intersection can be easily performed in two steps. The first step is to find the larger of nx1 and wx1 (i.e., maxx1), the larger of ny1 and wy1 (i.e., maxy1), the smaller of nx2 and wx2 (i.e., minx2), and, the smaller of ny2 and wy2 (i.e., miny2). The second step is to compute the intersection area by using the simple formula area = (minx2-maxx1)*(miny2-maxx2). The performance of the main memory spatial database we have developed will be reported in the experiment section.

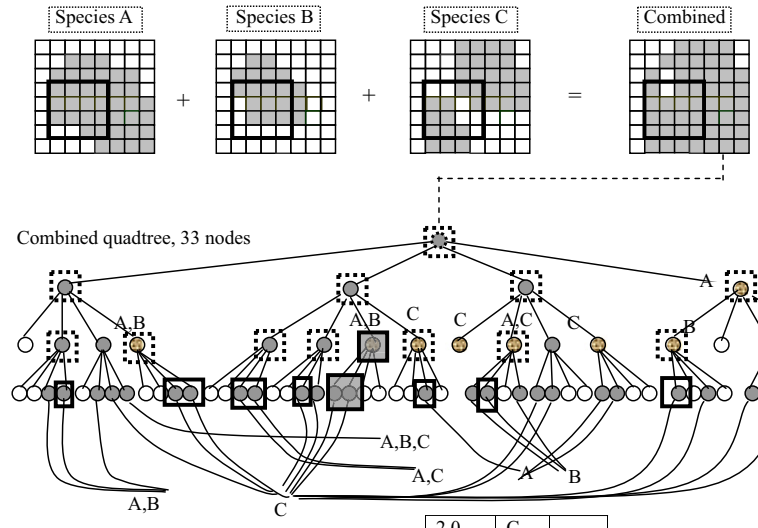
3.3. Web Applications: Mapping and Querying

The main-memory database serves as an efficient query processing engine to support Web applications. As shown in Fig. 2, two types of Web applications are supported, i.e., browser based Web applications for visual explorations and Web services based ones for analytics and modeling. The system has derived a richness map for predefined taxonomic groups across the study area, e.g., Western Hemisphere for the NatureServe bird dataset. The richness maps are then published as OGC Web Map Services (WMS, OGC, 2010a) and can be displayed in a variety of Web-mapping packages. The richness map serves as a base map for further visual exploration. Currently we use open source software MapServer (OSGeo, 2010) for serving base maps but other software, such as ESRI ArcGIS Server (ESRI, 2010a), can be used as well. It is relative easy to improve the response time of visualizing the base map as an open source package called TileCache (MetaCarta, 2008) is readily available to use. TileCache can convert arbitrary WMS requests to MapServer into tiled ones and cache the tiled images derived from the tiled requests. The cached images can be returned to clients directly without further WMS requests to MapServer (also see Fig. 2). This essentially transforms online queries into offline lookups which is very efficient. ESRI ArcGIS Server provides similar functionality (ESRI, 2010a).

To support interactive Web-based visual explorations, e.g., draw a rectangle on the richness base map interactively and returns a list of species fall within the rectangle, two addition components have been developed. The first component, as shown in Fig. 2, is a PHP-based light-weighted middleware to receive queries from Web clients, format them properly before sending the queries to the main memory spatial database that we have developed. The middleware is also responsible for converting database query results into proper formats that can be easily parsed by Web clients. The practical reason to have a middleware sitting between the Web clients and the main-memory spatial database backend is to reconcile two different types of communication protocols (TCP/IP for main-memory spatial database backend and HTTP for Web clients), and, to enable the communications between the Web clients and the database backend that usually sits behind a firewall. The second component is a customer module inside the Web client to capture user interactions, forward the query requests to the middleware in proper format and visualize the query results in a Web browser. In our system, the Ajax framework is used to communicate between Web clients and the middleware. The module is also responsible for visualizing interactively drawn query windows as well as caching query results.

While the database backend and the middleware can work with quite a few Web mapping APIs (Chow, 2008), including Google Map (Google, 2010) and ESRI ArcGIS Flex/Silverlight (ESRI, 2010b; ESRI, 2010c), we choose an open source package called OpenLayers (OpenLayers, 2010) as an example. A snapshot of the Web client user interface is presented in Fig. 6. When users interactively draw an arbitrary rectangle, all the species that are distributed in the query window will be retrieved by querying the main-memory spatial database through the middleware. By mapping the species identifies with the URLs of NatureServe InfoNatura Web information system (NatureServe, 2010c), users can obtain more species specific information at the NatureServe Web sites. The system also allows users to keep previous query results with respect to geographical extent, species and their areas (within the query window) for each of the query results. Users can go back to any of the previous query results by selecting the respective box. This not only improves system performance by caching previous query results at the Web browser side but also provides context of query histories. Internally this is implemented by treating a query result (point location query and spatial window query) as a feature in a temporary OGC Web Feature Service (WFS) (OGC, 2010b) layer that is supported by OpenLayers.

Our system also supports Web services by extending the middleware to support OGC WFS standard or W3C WSDL (Web Service Definition Language) standard (W3C, 2001). The query result of a Web service request can be parsed by software packages that support the respective standards and used in a variety of traditional biodiversity analysis and modeling modules. We are also in the process of integrating some of the analytics and modeling modules into browser-based Web applications to enhance the functionality of visual explorations and visual analytics (Keim and Mansmann, 2006, Thomas and Cook, 2005).



0.1.2	A,B	
0.1.3	A,B	*(1)
0.2.1	A,B	
0.2.2	C	
0.2.3	A,B,C	
0.3	A,B	X(2)
0.3.2	C	
0.3.3	C	X(1)

1.0.2	A,C	*(1)
1.0.3	A,C	*(1)
1.1.2	C	*(1)
1.1.3	C	
1.2	A,B	*(4)
1.2.0	C	*(1)
1.2.1	C	*(1)
1.3	C	X(2)
1.3.2	A	*(1)

2.0	C	
2.1	A,C	X(1)
2.1.0	B	
2.1.1	B	*(1)
2.1.3	B	
2.2.0	C	
2.2.1	C	
2.3	C	
2.3.0	A	
2.3.1	A	

3	A	X(3)
3.0	B	X(2)
3.0.0	C	*(1)
3.0.2	C	
3.2.0	C	
3.2.2	C	

Final outputs: A-14, B-10, C-10

Fig. 5. Example of spatial window query processing on a combined quadtree. Quadrants marked with solid squares denote they are completely within the query window and quadrants marked with dashed squares denote they partially intersect with the query window. Additional information of the quadrants in the combined quadtree that intersect with the query window is provided in the third columns of the quadrant tables. Quadrants that partially intersect with the query window are prefixed with X in the third columns of the quadrant tables. Similarly, quadrants that are completely within the query window (including leaf nodes) are prefixed with *. Numbers inside the parentheses in the third columns of the quadrant tables indicate the numbers of cells of the quadrants that are within the query window.

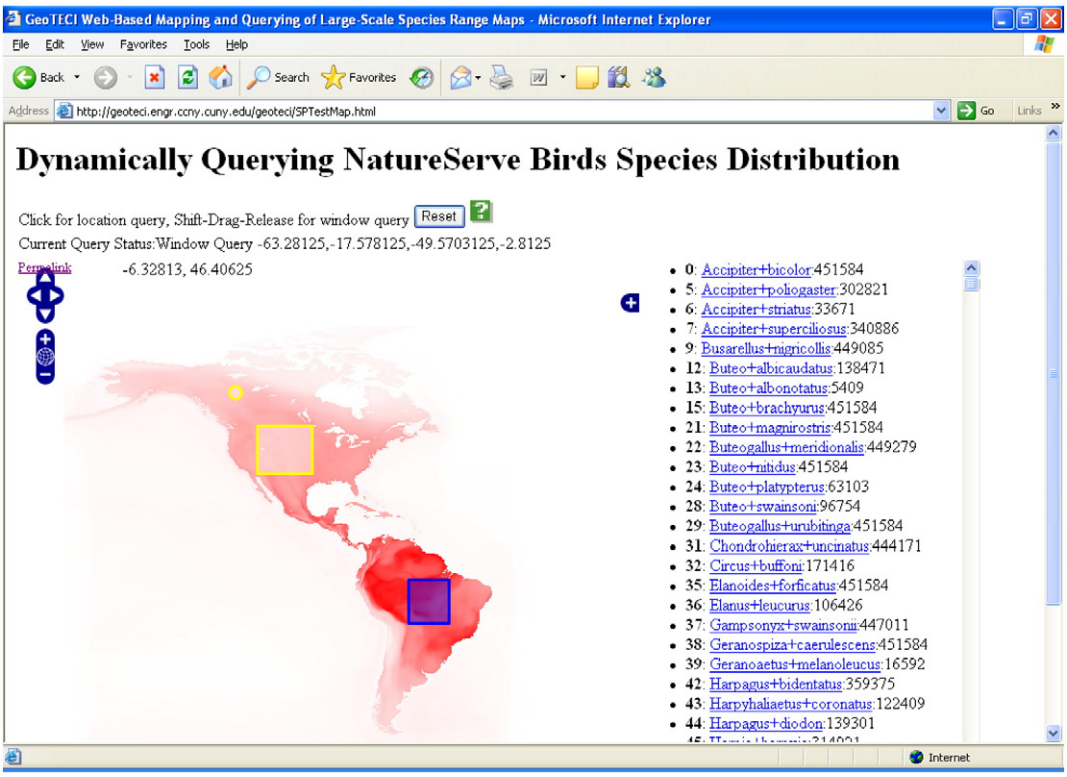


Fig. 6. Graphic user interface of the system: supporting arbitrary spatial window query and linking query results with NatureServe InfoNatura.

4. Experiments

Among the species distribution data published by NatureServe, we choose the bird dataset for our experiments as it is more complex than mammal and reptile datasets. Results of two groups of experiments will be reported. The first group of experiments examines memory requirements for the main memory spatial database and the second group of experiments reports query response times for using different query window sizes.

4.1. Datasets and experiments setup

We have downloaded the bird species distribution shapefiles from NatureServe's website (NatureServe, 2010b). The geographical range of the bird species in the datasets are limited to the Western Hemisphere, i.e., (−180, −90, 0, 90). Thus the number of cells along both latitude and longitude at 1 min resolution is $180*60=10{,}800$. We set the depth for the quadtree for the experiments to $d=14$ as $2^{14}=16384$, which is already greater than 10,800. As such, the effective resolution of cells at the finest scale is $180/2^{14}°$, i.e., nearly 40 arc-seconds. We note that the system allows use arbitrary grid resolutions (r) as long as $r*2^d$ is less than the width and height of the gird of the study area. However, practically the highest resolution of the derived raster tessellation should be limited by the original vector data and should be carefully chosen. We discard species that have only point data and species whose ranges are less than the size of a single cell at the finest resolution. The final number of datasets for the bird species tested is 4062. All experiments are performed on a Dell Precision T5400 workstation with 16 G memory. To better understand the datasets used in the experiments, we have derived a set of statistics of the bird range maps which are listed in Table. 1. The statistics may help understand the sizes of the polygonal datasets and the derived quadtree.

4.2. Experiments on Quadtree Indexing

Combining all the 4062 rasterized bird species ranges maps generates 46,139,247 raster cells and 1,318,136,140 pairs of (cell, identifier) combinations at the finest resolution, i.e., about 28.7 species per cell. The classic combination associates a total of 831,903,250 identifiers with 7,511,823 leaf quadtree nodes, an average of 110.7. The proposed combination associates a total of 23,865,343 identifiers with 4,957,050 quadtree nodes (leaf and non-leaf), an average of 4.8. From these numbers we can see that, quadtree representations, including both the classic combination and the proposed combination, are very effective in representing species distribution data. The 46,139,247 non-empty cells (out of the 10,800*10,800 total number of cells) at the finest resolution are represented by 10,363,457 quadtree nodes (including both non-empty and empty nodes). The representation achieves a compression rate of 4.57 for non-empty cells and a compression rate of 11.25 for all cells. The number of leaf nodes of the combined quadtree using the proposed combination (6,261,549) is about 20% fewer than that of the combined quadtree using the classic combination (7,511,823) due to different tree construction policy applied. The total number of identifiers to store in the classic combination is 34.9 times larger than that of the modified combination (831,903,250 versus 23,865,343). The average number of identifiers to associate with quadtree node for the classic combination is 23.0 times larger than that of the modified combination (110.7 versus 4.8). The proposed combination significantly lowers the memory consumption as detailed next.

In our implementation of the main memory spatial database, a quadtree node has a pointer to its parent (*parent*), a pointer to the array of its four child nodes (*children*), a short integer value indicates the number of species associated with the node (*id_size*), a pointer to the array of the species identifiers (*ids*), a short integer to indicate the level of the node (*level*), and, finally two short integers (*index[0]* and *index[1]*) to indicate the quadrant that the node is located relative to its parent quadrant along both the x and y directions. Since our experiments are performed on a 32-bit machine, a pointer variable takes 4 bytes ($l_p=4$) and a short integer takes 2 bytes ($l_s=2$). For leaf nodes, *children* will be set to NULL and thus the memory footprint for a single quadtree node is 20 bytes ($s_l=4*l_s+3*l_p=20$). On the other hand, for the non-leaf nodes, we will need to allocate additional four pointers to point to a node's four chidren and assign the start postion of the meory block to *children.* As such, the memory footprint for a non-leaf node is $s_{nl}=s_l+4*l_p=20+4*4=36$ bytes. Assuming that there are n_l leaf quadtree nodes and n_{nl} non-leaf nodes, then the memory footprint for the quadtree nodes are $s_{nl}*n_{nl}+s_l*n_l$. Further assuming that there are n_i species that are associated with each of the $n_t=n_{nl}+n_l$ quadtree nodes and each species identifiers takes l_d bytes, then the memory footprint for storing all the identifiers will be $\left(\sum_{i=1}^{n} n_i\right)*l_d$. Thus the total memory footprint for the main memory spatial database is $m=s_{nl}*n_{nl}+s_{nl}*n_{nl}+\left(\sum_{i=1}^{n} n_i\right)*l_d$. Since l_d is proportional to the logarithmic of the number of species identifiers, we use short integer ($l_d=2$) to represent species identifiers as the number of species used in the experiments is 4062 which is far less than $2^{16}=65{,}536$. In general, we expect l_d to be between 2 and 4 bytes as 2^{24} ($l_d=3$) can already represent more than 4 million species identifiers. For comparison purpose, the calculations of memory consumptions for both the classic combination and the proposed combination are listed in Table 2.

From Table 2 we can see that the memory footprint is dropped from 1827.9 MB to 305.8 MB which is about six times saving. Note that the majority of the memory is used for storing species identifiers in the classic combination (86.8%) while the percentage drops to only 14.9% in the proposed combination. Since the data volume of species identifiers dominates, the total memory footprint has dropped to 1/6 for the proposed combination. The results clearly show the memory efficiency of the proposed quadtree indexing approach. While it is difficult to accurately estimate how the average numbers of species identifiers associated with quadtree nodes scale with the numbers of species (in a way similar to, but not the same as, the species–area relationships) based on the proposed combination, we believe that a typical desktop computer equipped with 8–16 G memory may hold global range maps of tens to hundreds of thousands of species

Table 1
Statistics of the bird dataset.

#of species	4062
Volume of geometrical data (.shp files in the original data files)	1.3 G
#of polygons	708,509
#of points	77,699,991
Average # of polygons per species	174.4
Average # of points per polygon	109.7
Average # of points per species	19,128.5
Average # of cells (finest resolution) per species	4,131,931.1

Table 2
Memory footprints for the classic and the proposed combinations.

	Unit size	Classic combination	Proposed combination
Total number of quadtree nodes		10,363,457	10,363,457
Leaf nodes	$s_l=20$ bytes	$n_l=7{,}511{,}823$	$n_l=6{,}261{,}549$
Non-leaf nodes	$s_{nl}=36$ bytes	$n_{nl}=2{,}851{,}634$	$n_{nl}=4{,}101{,}908$
Species identifier	$l_d=2$ bytes	$(\Sigma n_i)=831{,}903{,}250$	$(\Sigma n_i)=23{,}865{,}343$
Memory footprint $m=s_{nl}*n_{nl}+s_l*n_1+\left(\sum_{i=1}^{n} n_i\right)*l_d$		1,916,701,784 (1827.9 MB)	320,630,354 (305.8 MB)

based on our quadtree data structure. Given that the memory capacities for typical commodity computers have increased 100–1000 times over the past ten years (Hennessy and Patterson, 2006), we are quite optimistic that our main-memory spatial database can hold the global range maps of all known species on a typical personal computer when the data becomes available.

4.3. Experiments on query processing

We have performed 100 tests for each of the four window sizes, i.e., 0.2, 1.0, 2 and 10° with randomly generated query window centers. Among the 400 tests, only 36 of them have response times greater than 10 ms. We record the 158 tests that have resulted in at least one species in the queries (note that 70% of the Earth surface is covered by oceans and usually no birds are recorded in oceans unless there are islands). The average query response time is 10.06 ms and the largest one is 290 ms. The excellent query processing performance leaves quite some rooms for achieving end-to-end performance below the level of 1 s for even larger numbers of species range maps on the hardware configurations that similar to the one used in our experiments.

For comparison purposes, we have put the bounding boxes of the quadrants in the combined quadtree into a PostgreSQL database (PostgreSQL, 2010), indexed the boxes and performed the same set of queries (also see Section 3.1 for the exact SQL syntax). The average query response time for the disk-resident PostgreSQL database is 4786 ms and the largest one is 134,960 ms. It is clear that the performance of our main-memory spatial database is about 2–3 orders better than using the disk-resident PostgreSQL database.

5. Summary and conclusion

We have developed a high performance Web-based information system for efficiently publishing and disseminating large scale species range maps based on a novel quadtree data structure. Using the NatureServe's 4000+ bird species range map dataset, experiment results have shown that the memory footprint of the proposed quadtree data structure representing the range maps of all the species is a fraction of the classic combination of individual quadtrees each representing a species range map. The experiment results have also demonstrated that the query response times of our main-memory spatial database are 2–3 orders better than using a disk-resident PostgreSQL database based on the 400 randomly generated spatial query windows. The response times are well beyond a second for query windows as large as $10 \times 10°$.

We envision that more and more large-scale scientific datasets at global scale will become available with increasing spatial resolutions. These datasets need to be published over the Web for easy Web-based visualization and exploration as well as data integration, analysis and modeling. Developing specialized data structures and query processing algorithms for certain domain-specific applications to improve system performances might provide preferable alternatives to utilizing generic software packages when overall cost-effectiveness factors are taken into considerations. This work represents the first step towards the direction.

For future work, we plan to support Region of Interests (ROIs) queries that extend rectangles in window queries to arbitrarily shaped polygons. This can be achieved by constructing a quadtree on an ROI polygon using the same raster tessellation that has been used to construct the combined quadtree for all species range maps. The query procedure discussed in Section 3.2 subsequently needs to be modified to efficiently support synchronized traversal of both the ROI quadtree and the species range map quadtree and derive the list of species that are distributed in the ROI. Second, we plan to further investigate the scalability of our prototype system. There are more than a million known species and their distribution maps are becoming increasingly available. This number is about two to three orders larger than that have been experimented in this study. More experiments are needed to research on memory consumption and query response times as the number of species scales up. Finally, we plan to explore the Rich Internet Application (RIA) frameworks such as Adobe Flex and Microsoft Silverlight as well as HTML 5 to enrich the information visualization and visual exploration functionality of browser based Web applications. For example, automatically compute and visualize top K quadrants that are most different from a focal cell/quadrant based on different types of beta diversity measurements.

References

Berrick, S.W., Leptoukh, G., et al., 2009. Giovanni: a web service workflow-based data visualization and analysis system. IEEE Transactions on Geoscience and Remote Sensing 47 (1), 106–113.

Best, B.D., Halpin, P.N., et al., 2007. Geospatial web services within a scientific workflow: predicting marine mammal habitats in a dynamic environment. Ecological Informatics 2 (3), 210–223.

Bisby, F.A., 2000. The quiet revolution: biodiversity informatics and the internet. Science 289 (5488), 2309–2312.

Boyd, D.S., Foody, G.M., 2011. An overview of recent remote sensing and GIS based research in ecological informatics. Ecological Informatics 6 (1), 25–36.

Chow, T.E., 2008. The potential of maps APIs for Internet GIS applications. Transaction on GIS 12 (2), 179–191.

COL, 2010. Catalogue of Life: 2010 Annual Checklist from http://www.catalogueoflife.org/.

ESRI, 2010a. ArcGIS Server. from http://www.esri.com/software/arcgis/arcgisserver.

ESRI, 2010b. ArcGIS API for Flex. from http://help.arcgis.com/en/webapi/flex/index.html.

ESRI, 2010c. ArcGIS API for Microsoft Silverlight/WPF. from http://help.arcgis.com/en/webapi/silverlight/.

Field, R., Hawkins, B.A., et al., 2009. Spatial species-richness gradients across scales: a meta-analysis. Journal of Biogeography 36 (1), 132–147.

Flemons, P., Guralnick, R., et al., 2007. A web-based GIS tool for exploring the world's biodiversity: The Global Biodiversity Information Facility Mapping and Analysis Portal Application (GBIF-MAPA). Ecological Informatics 2 (1), 49–60.

Foody, G.M., 2008. GIS: biodiversity applications. Progress in Physical Geography 32 (2), 223–235.

Fook, K.D., Monteiro, V., et al., 2009. Geoweb services for sharing modelling results in biodiversity networks. Transactions in GIS 13 (4), 379–399.

Frehner, M., Brandli, M., 2006. Virtual database: spatial analysis in a Web-based data management system for distributed ecological data. Environmental Modelling & Software 21 (11), 1544–1554.

Gaede, V., Gunther, O., 1998. Multidimensional access methods. ACM Computing Surveys 30 (2), 170–231.

GBIF, 2012. GBIF Data Portal. from http://data.gbif.org/.

Gonzales, R., Cardille, J.A., et al., 2009. SFMN GeoSearch: an interactive approach to the visualization and exchange of point-based ecological data. Ecological Informatics 4 (4), 196–205.

Goodall, J.L., Horsburgh, J.S., et al., 2008. A first approach to web services for the National Water Information System. Environmental Modelling & Software 23 (4), 404–411.

Google, 2010. Google Map API. from http://code.google.com/apis/maps/.

Greene, S.L., Minoura, T., et al., 2007. WebGRMS: prototype software for web-based mapping of biological collections. Biodiversity and Conservation 16 (9), 2611–2625.

Grund, M., Kruger, J., et al., 2010. HYRISE: a main memory hybrid storage engine. Proceedings of VLDB Endowment 4 (2), 105–116.

Guisan, A., Zimmermann, N.E., 2000. Predictive habitat distribution models in ecology. Ecological Modelling 135 (2–3), 147–186.

Guralnick, R., Hill, A., 2009. Biodiversity informatics: automated approaches for documenting global biodiversity patterns and processes. Bioinformatics 25 (4), 421–428.

Halpin, P.N., Read, A.J., et al., 2009. OBIS-SEAMAP The world data center for marine mammal, sea bird, and sea turtle distributions. Oceanography 22 (2), 104–115.

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

He, K.S., Zhang, J.T., et al., 2009. Linking variability in species composition and MODIS NDVI based on beta diversity measurements. Acta Oecologica-International Journal of Ecology 35 (1), 14–21.

Hennessy, J.L., Patterson, D.A., 2006. Computer Architecture: A Quantitative Approach, 4th Edition. Morgan Kaufmann.

Jaeger, E., Altintas, I., et al., 2005. A Scientific Workflow Approach to Distributed Geospatial Data Processing using Web Services. Proceedings of SSDBM 2005.

Keim, D.A., Mansmann, F., 2006. Challenges in Visual Data Analysis. Proceedings of the conference on Information Visualization.

Koleff, P., Gaston, K.J., et al., 2003. Measuring beta diversity for presence–absence data. Journal of Animal Ecology 72 (3), 367–382.

Kooistra, L., Bergsma, A., et al., 2009. Development of a dynamic web mapping service for vegetation productivity using earth observation and in situ sensors in a sensor web based approach. Sensors 9 (4), 2371–2388.

MetaCarta, 2008. TileCache. from http://tilecache.org/.

Mittelbach, G.G., Steiner, C.F., et al., 2001. What is the observed relationship between species richness and productivity? Ecology 82 (9), 2381–2396.
Moreno, C., Zuria, I., et al., 2006. Trends in the measurement of alpha diversity in the last two decades. Interciencia 31 (1), 67–71.
NatureServe, 2010a. NatureServe Explorer - An Online Encyclopedia of Life. from http://www.natureserve.org/explorer/.
NatureServe, 2010b. NatureServe Animal Data. from http://www.natureserve.org/getData/animalData.jsp.
NatureServe, 2010c. InfoNatura: Animals and Ecosystems of Latin America. from http://www.natureserve.org/infonatura/.
OGC, 2010a. OpenGIS Web Map Server Implementation Specification. from http://www.opengeospatial.org/standards/wms.
OGC, 2010b. OpenGIS Web Feature Service (WFS) Implementation Specification. from http://www.opengeospatial.org/standards/wfs.
OpenLayers, 2010. OpenLayers: Free Maps for the Web. from http://openlayers.org/.
OSGeo, 2010. MapServer. from http://www.mapserver.org/.
PostgreSQL, 2010. PostgreSQL Database. from http://www.postgresql.org/.
Proches, S., 2005. The world's biogeographical regions: cluster analyses based on bat distributions. Journal of Biogeography 32 (4), 607–614.
Qian, H., 2010. Environment-richness relationships for mammals, birds, reptiles, and amphibians at global and regional scales. Ecological Research 25 (3), 629–637.
Ricotta, C., 2005. Through the jungle of biological diversity. Acta Biotheoretica 53 (1), 29–38.
Samet, H., 2004. Object-based and image-based object representations. ACM Computing Surveys (CSUR) 36 (2), 159–217.
Samet, H., 2005. Foundations of Multidimensional and Metric Data Structures. Morgan Kaufmann Publishers Inc.
SEDAC, 2010. Columbia University Social-Economic Data and Application Center Species Distribution Grid. from http://sedac.ciesin.columbia.edu/species/.
Soberon, J., Peterson, A.T., 2004. Biodiversity informatics: managing and applying primary biodiversity data. Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences 359 (1444), 689–698.
Steiniger, S., Hay, G.J., 2009. Free and open source geographic information tools for landscape ecology. Ecological Informatics 4 (4), 183–195.
Theobald, D.M., 2005. GIS Concepts and ArcGIS Methods, 2nd Ed. Conservation Planning Technologies, Inc.
Thomas, J.J., Cook, K.A., 2005. Illuminating the Path: The R&D Agenda for Visual Analytics. National Visualization and Analytics Center.
uBio, 2010. uBio: Indexing and Organizing Biological Names. from http://www.ubio.org/.
USDA, 2010. PLANTS Database. from http://plants.usda.gov/.
USGS, 2006. Digital Representations of Tree Species Range Maps. from http://esp.cr.usgs.gov/data/atlas/little/.
W3C, 2001. Web Services Description Language (WSDL) 1.1. from http://www.w3.org/TR/wsdl.
Waide, R.B., Willig, M.R., 1999. The relationship between productivity and species richness. Annual Review of Ecology and Systematics, pp. 257–300.
WWF, 2006. World Wildlife Fund WildFinder: Online database of species distributions, ver. Jan-06. from http://www.worldwildlife.org/WildFinder.
WWF, 2010. WWF WildFider Portal. from http://gis.wwfus.org/wildfinder/.
Yang, C.W., Raskin, R., et al., 2010. Geospatial Cyberinfrastructure: past, present and future. Computers Environment and Urban Systems 34 (4), 264–277.
Zhang, J., 2009. Efficient managing large scale species range maps in a spatial database environment. Proceedings of 17th International Conference on Geoinformatics.
Zhang, J., 2011. Speeding Up Large-Scale Geospatial Polygon Rasterization on GPGPUs. Proceedings of the 2nd ACM workshop on High-Performance and Distributed GIS (HPDGIS).
Zhang, J., Gruenwald, L., 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.
Zhang, J., Pennington, D.D., Liu, X., 2007. GBD-Explorer: extending open source java GIS for exploring ecoregion-based biodiversity data. Ecological Informatics 2 (2), 94–102.
Zhang, J.T., Pennington, D.D., Michener, W.K., 2005. Using web services and scientific workflow for species distribution prediction modeling. Proceedings of Advances in Web-Age Information Management, LNCS 3739, Springer, 2005, 610–617.
Zhang, J.T., Hart, Q., et al., 2009a. Sensor data dissemination systems using Web-based standards: a case study of publishing data in support of evapotranspiration models in California. Civil Engineering and Environmental Systems 26 (1), 35–52.
Zhang, J., Gertz, M., Gruenwald, L., 2009b. Efficiently managing large-scale raster species distribution data in PostgreSQL. Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 316–325.