"The Road and the River Should Cross at the Bridge" Problem: Establishing Internal and Relative Topology in an MRDB

Barbara P. Buttenfield and Eric B. Wolf
University of Colorado – Boulder
babs@colorado.edu (corresponding author)

Abstract The search for a robust vector data model that can generate a hierarchy of multiple representations, which are valid for analysis at multiple resolutions, from a single highly detailed version, challenges GIS and cartographic research communities. Hierarchic decomposition of raster data is commonly accomplished by pyramid building algorithms. It is not possible in current GIS environments to create pyramids for vector data. The reason is that raster data contain pixels that can be resampled at regular or randomized intervals. Vector data in contrast contain features that nest, that connect, that have intrinsic contextual meanings with respect to each other. This paper presents a working implementation of a pyramid architecture (MRVIN) for vector data that preserves line length, local coordinate density, and valid topology at multiple levels of resolution. The paper describes decomposition and reconstruction routines for simple vectors, stream networks, and compound vectors (registered road and river networks); presents the database schema for the pyramid, and describes MRVIN architecture.

Keywords MRDB, vector geospatial data, pyramid architecture, internal topology, relative topology.

1.0 Problem Context and Objectives

The search for a robust vector data model that can generate a hierarchy of valid multiple representations from a single highly detailed version has challenged GIS and cartographic research communities for decades. Whether the content is vector coastlines, transportation, hydrography or administrative boundaries, vector data archives remain highly resistant to changing levels of detail on-the-fly. Moreover, a lack of user knowledge about appropriate data reduction leads all too often to generalized versions of data that do not hold up to reliable measurements, nor retain geographic logic in analysis. Solutions using regularized and randomized coordinate reduction to speed computation or display have mixed success, and most existing solutions do not correct for topological corruptions introduced during generalization.

Many online data sources (e.g. *GISDataDepot, TerraServer)* offer downloads of geospatial data compiled at standardized resolutions or compilation scales[1], as for example in USA, 30-meter DEMs, 1:24,000 USGS DLG data, or 1:100,000 US Census TIGER files. The downloaded data contains a single representation at the original (compiled) level of resolution. Ironically, the first GIS user task that customarily follows delivery is generalization, to eliminate some proportion of coordinates, to reduce data volume, and/or modify resolution either for display aesthetics or computational efficiency. One objective of the current research is to generate a vector data architecture that delivers multiple representations across a range of resolutions, and that offers systematic guidelines for choosing an appropriate resolution. The solution we are developing is hierarchical, analogous to raster pyramids.

With raster data, the pyramid architecture relies on hierarchical structures (image pyramids, quad-trees or R-trees) or on spectral or wavelet decomposition. User-determined cell size determines the pyramid layer at which queries, computations, and modeling proceed. Without pyramids, the entire dataset must be

[1] The terms "resolution" and "scale" are not interchangeable. Resolution refers to the smallest item that can be isolated. In raster data, this is the size of a pixel. It applies to data collection and GIS modeling. Scale is a ratio between data measurement and ground measurement, usually length). The ratio refers to a map display, or to the proportions at which vector data are compiled. Tobler (1987) offers useful conversions between resolution and scale, including detectable and average resolution. He suggests an estimate to determine the resolution of a vector data set is to divide the denominator of the scale ratio by 1000. This gives the size of the smallest detectable item, in meters. For a 1:100,000 vector file, the smallest item one should expect to detect is roughly 100 m. in size.

queried or acted upon, slowing performance significantly. With pyramids, a single raster data file provides the source for working with a series of data representations at multiple resolutions.

GIS environments do not incorporate functionality to create pyramids for vector data. The reason is that raster data contain pixels that can be resampled at regular or randomized intervals. Vector data in contrast contain features that nest, that connect, that have intrinsic contextual meanings with respect to each other. A lake is bounded by a shoreline, which nests within a layered set of contours. Road networks must precisely intersect, to be useful for measuring traffic flow. These semantics must be preserved at all levels of resolution, if the data are to remain useful for spatial analysis. A second objective of the work is to demonstrate how to protect feature intersections, nested features, and other intrinsic semantics, such as relationships between point and area features. For example, building locations must not "jump" across roads when transportation networks are simplified.

The third objective follows from the second. The successful vector architecture must preserve all properties that characterize sound principles of generalization: line length, local coordinate density, and valid topology. The first two are important to cartographic display; and all three are prerequisite to robust spatial analysis and modeling. In particular, the solution must incorporate two kinds of topology. *Internal topology* means that line segments must connect at fine and coarse levels of resolution; and isolated segments must not intersect. *Relative topology* preserves feature registration between layers: the road feature in one layer must register to the river feature in another layer at the bridge feature in a third layer, no matter what resolution they are retrieved. Preservation of internal and relative topology at multiple levels of resolution remains the foremost research challenge to be addressed.

This paper moves beyond our previous work on progressive data delivery (Buttenfield, 2002; 2004) showing how to construct a vector data pyramid. This paper presents algorithms for effective processing of more complicated features, such as compound vectors (eg., stream tributaries); nested multi-scale feature descriptions (eg., road networks and settlement features), and linked multi-layer archives (eg., networks of roads and streams). We have created a working architecture called **MRVIN** (pronounced "Marvin", for "**M**ultiple **R**epresentations of **V**ector **In**formation") that builds and maintains vector data pyramids, such that from an archive of georeferenced datasets (such as all vector layers in the City of Boulder database) a user can retrieve complete and georeferenced representations at multiple levels of resolution. Retrieved representations preserve line length and local coordinate density, and sustain valid internal and relative topologies between retrieved data themes, regardless of whether those themes are retrieved from one or from multiple levels of resolution.

The status of the MRVIN project is as follows. The base architecture is complete, with data pyramids maintained in a relational data model that handles not only individual features (a single stream channel) but also nested features (all tributaries in a watershed) and multiple data themes (road networks and point landmarks, in addition to the stream network). We are currently benchmarking county level data sets (on the order of tens of megabytes) with an expectation to scale up another order of magnitude in the coming year. Testing to date confirms that internal topologies can be established and maintained. We anticipate beginning tests on relative topologies this summer, and will report on these at the presentation.

2.0 Requirements for a Multi-Resolution Vector Architecture

To be useful for analysis, pyramid construction must do more than sample or resample the data at regular or randomized increments. The algorithms must adhere to specific requirements underlying GIS and spatial analysis, and to sound principles of cartographic generalization. The first requirement for a multi-resolution vector architecture then is a pyramid-building strategy that can maximize line length globally and locally, and preserve original coordinates.

The second requirement protects logical consistency. It is important to insure that delivered data retains a level of detail that is locally consistent with the original file, and locally proportional to the original compilation. Most geospatial data is heterogeneous, meaning that density of detail is non-uniform. Figure 1 shows the eastern boundary of the United States, which aligns with naturally occurring geomorphic

processes. The boundary at Lake Erie (a) (geologists call it a simple coastline) lacks the high frequency crenulations at Chesapeake Bay (b), or the space-filling path followed by the depositional features near Brownsville Texas (c). All of these feature geometries differ from the repeating crescent pattern formed by shoreline processes in North Carolina (d). Geometrically, the differences can be quantified by comparing density of detail locally throughout a vector representation (Buttenfield, 1989). Any algorithm that modifies resolution must preserve local coordinate density to maintain line length for modeling and to protect geographic and visual logic for display.

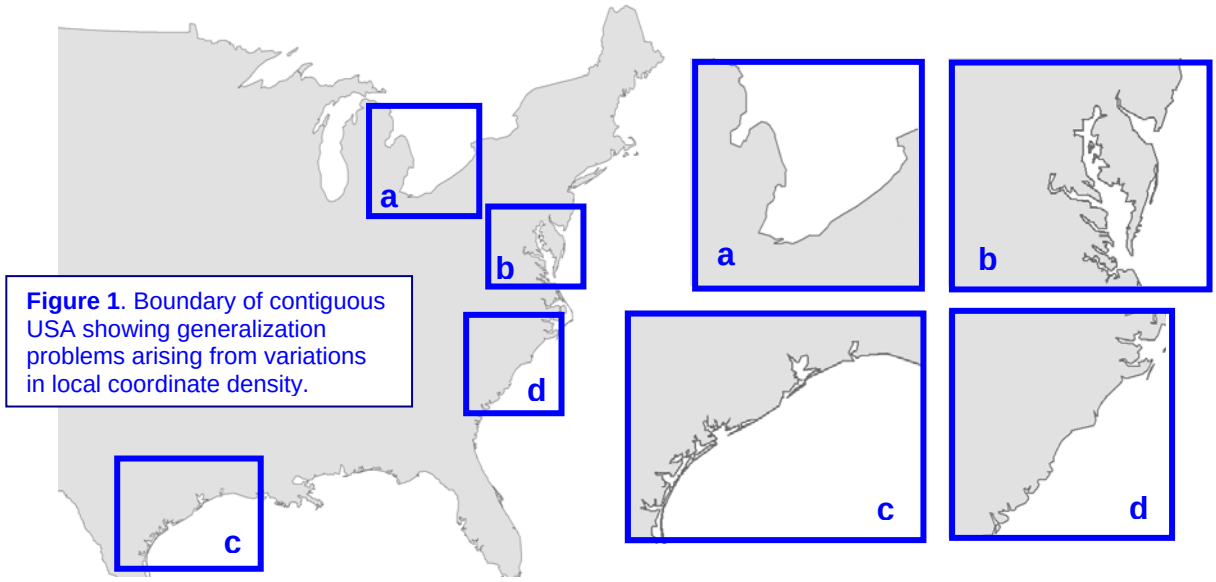

Figure 1. Boundary of contiguous USA showing generalization problems arising from variations in local coordinate density.

A third requirement relates to user-control. Ideally, the level of resolution should be specified prior to access, for example, the user should be able to request a version equivalent to 30 meter resolution, or for representation on a 1:25,000 scale display; or data conforming to a horizontal positional accuracy of 1 part in 1,000. For real time data delivery especially, a handheld client should be able to request from online servers a vector delivery at a specified resolution, within a given transmission time, or below a threshold number of bytes. These specifications should occur at the machine level, automatically.

3.0 Design for a Multi-resolution Architecture

The first algorithms reorganize the vector data into a pyramid, recording metadata on geometry and flagging topological errors. The second set access the data progressively and reconstruct vectors in a form acceptable for GIS or cartographic use and correct internal topology where necessary. These are discussed only briefly: they were initially presented at a GIScience presentation, but we include them here since they form a foundation for the new algorithms that preserve topology. The new work includes a set of algorithms correcting internal topology for compound vectors and extensions to preserve relative topology.

3.1 Reorganizing the Vectors into a Pyramid

A hierarchical coordinate selection routine deconstructs each vector feature (a road segment, or a stream tributary) into a representation of the complete vector, albeit at some intermediate level of resolution. The upper (root) level of the hierarchy contains an initial coarse-resolution version of the feature. Lower levels “fill in” details that would become apparent if the viewer were observing the feature more closely, or measuring it at finer resolution.

Each representation uses a hierarchical subdivision based on the well-known Ramer-Douglas-Peucker (RDP) simplification algorithm (Ramer, 1972; Douglas and Peucker, 1973) (Figure 2). Cromley and Campbell (1990) show that a hierarchical algorithm that optimizes some geometric property will be less effective than a non-hierarchical solution preserving the same property. Cromley counters his own work

five years later (Barber et al, 1995: 276) with an empirical comparison showing that "little geometric quality is lost by using a hierarchical operator … and given the greater flexibility of scale representations that is possible, hierarchical methods seem to be more satisfactory." In spite of longstanding and often heated debate, the RDP algorithm provides default vector simplification routine in many GIS packages.

RDP orders selected coordinates in a stack or a simple array. For raster data, the common logical schema is a pyramid. Many alternatives are available, dating back more than two decades (Ballard, 1981). Becker and Widmayer's (1990) Priority Rectangle (PR) files and van Oosterom's (1990) Reactive-Tree structures move high priority data towards the tree root. Jones and Abrahams's (1986; 1987) Multi-Scale Line tree is partitioned using RDP, and selects the same points as Ballard's strip tree. Their method is intended for cartographic display rather than analysis, as is Cromley's (1991) hierarchical simplification method, which requires *a priori* specification of the desired output mapping scales. Our algorithm design is modified from Ballard (1981), because the requirements for relative topology require fast determination of line intersections between data themes, which Ballard's design can accommodate.

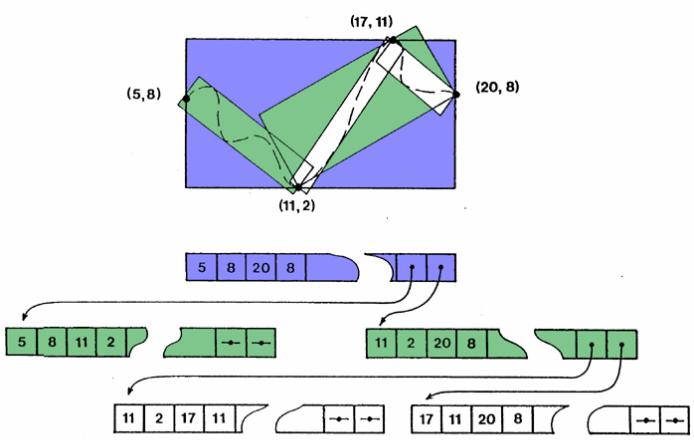

Figure 2. Hierarchical subdivision by RDP and multi-resolution storage in a Ballard (1981) strip tree.

The modifications implement added flexibility for embedding metadata at all levels of the data model. We store a description of the geometry at each level of resolution, including the Minimum Bounding Rectangle encompassing the original coordinates (the 'strip'), a strip identifier, the coordinate endpoints, the maximum deviation from the line connecting these endpoints. Other characteristics may be recorded, for example local coordinate density or the length in ground units between endpoints (the anchor length). Attributes and metadata may be stored in each strip as well. This hierarchical reorganization algorithm is previously implemented and tested (Buttenfield, 2002).

Subdivision by RDP meets the initial stated requirements for multi-resolution vector data delivery. A complete representation at a given resolution can be generated by traversing any single layer of the tree. The resulting representation preserves line length. Cromley and Campbell (1990) show preservation rates of 90% line length in their RDP experiments. RDP is therefore likely to preserve geometric details adequately. Preserving line length protects high frequency detail that is necessary for accurate measurement at all resolutions, and for preserving detail demonstrated to be necessary for visual feature recognition (Attneave, 1954). Local density of detail is preserved by nature of the RDP subdivision, and local tree depth reflects this. Note that the strip tree stores some vertices redundantly: in Figure 2, coordinate (11,2) is stored twice, for example. Redundancy is necessary for topological checking, discussed later in the paper.

3.2 Data Retrieval and Vector Reconstruction

Data are retrieved to insure delivery of a complete representation of the vector feature at any given resolution (Figure 3). Efficiency is improved in each iteration by retrieving only newly partitioned strips, in the style of a UNIX *dif* operation. Later iterations (paradoxically) retrieve more detail but less data, simply because as the algorithm progresses deeper into the tree, representations tend to contain fewer new strips. The retrieval process delivers no more data than requested, with qualifications that will be covered in Section 3.3 on protecting topology.

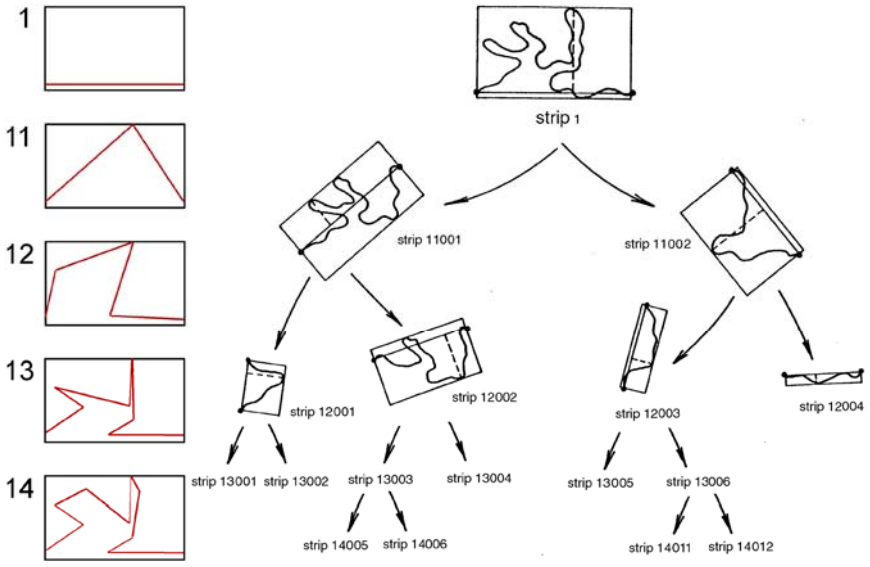

Figure 3. How the vectors are reconstructed. Black line drawing shows the pyramid and how the original vector is decomposed into strips. Red line drawing shows the anchor line representations, and how the line is reconstructed at each pyramid level.

On the left side of Figure 3 are shown anchor lines that connect endpoints of each strip in each representation. Anchor lines are generated from the straight line segments connecting the beginning and ending coordinate of each strip. With completion of each iteration, the set of anchor lines more closely approximate the original coordinate string. A sample vector is shown in Figure 4.

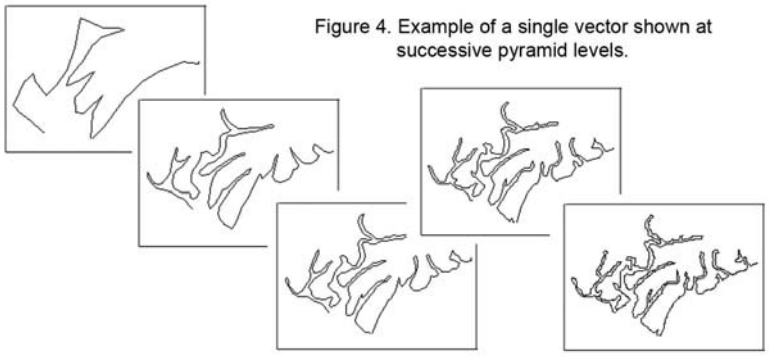

Figure 4. Example of a single vector shown at successive pyramid levels.

3.3 Modifications to Preserve Internal Topology

Three modifications insure partial preservation of internal topology. As defined above, correct internal topology means that the vectors at any resolution do not coalesce, self-cross, or introduce slivers or other topological nonsense. Topological checking can resolve graphical conflicts in line simplification (deBerg et al, 1998; Harrie, 1999; Jones et al, 2000; see the online demonstration at http://www.cs.cf.ac.uk/user/S.Zhou/MSDBDemo/. However, "there has been little progress in the application of such procedures for priority labeling of vertices in a multi-scale database in order to guarantee topological consistency across retrieved levels of detail of the line and area primitives". (van der Poorten et al 2002: 210)

The first modification speeds the RDP execution (Herschberger and Snoeyink, 1992) to worst-case $O(n\log_2 n)$ runtime, and generates convex hulls in the hierarchical tree. Convex hulls insure that a coordinate string will not self-cross at simplified levels of resolution. Saalfeld (1999) formalizes a mathematical proof that if two convex hulls overlap, the coordinate strings contained within them may converge. If the hulls do not overlap, no topological problems exist locally. Saalfeld proves that with convex hulls, RDP always converges to a topologically correct solution, assuming the original vector is free of topological errors.

Convex hulls may be constructed by incremental, gift-wrap, divide-and-conquer, or quick-hull algorithms (http://www.cse.unsw.edu.au/~lambert/java/3d/hull.html). Even though the incremental method has the slowest runtime ($O(n^2)$), it works well for this specific application for the following reasons. Coordinates in

any representation are retrieved in incremental order, obviating the need to store all coordinates in memory at once. Because coordinates are ordered, the algorithm runs in amortized linear time (Saalfeld personal communication, 2001). The incremental routine builds a complete convex hull only once at the tree root and then updates the hull incrementally as it partitions the vector (Buttenfield, 2002).

The second modification adds Saalfeld's (1999) topology check to test for internal errors during RDP partitioning. As each row of the tree is generated, adjacent convex hulls are compared. If they overlap, a "dirty bit" flag is inserted in the convex hull record to alert for possible internal topology conflict.

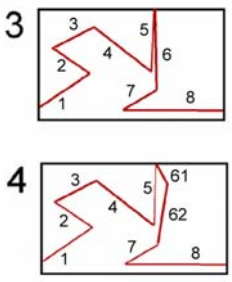

Figure 5. Internal topology is corrected by delivering extra detail locally.

The third modification occurs on data retrieval. The presence of a dirty bit flag in any convex hull in the finest resolution strip initiates retrieval of that hull's children, until the delivered hulls are topologically clean (that is, until no flags are detected). For this region of the vector, more data is delivered than was requested, and the feature resolution will be finer than what the user requests. In pathological cases this could result in delivering and reconstructing most of the original coordinate string for a nominally coarse level of detail. But all internal topological conflicts will be resolved.

Figure 5 illustrates a hypothetical transmission at iterations 3 and 4. Iteration 3 delivers eight convex hulls: anchor lines for hulls 5 and 6 converge. The topology check sets the flag as the pyramid is created. The vector reconstruction routine detects the flag and delivers two additional hulls in iteration 4 (segments 61 and 62), adding enough detail to displace hull 6 and protect internal topology.

Figure 6 shows a small portion of a stream network in Boulder County reconstructed at specified levels of the partitioning process, including convex hulls (in red). Internal topology has been corrected. This network comprises 341 strips in three trees. The trees have a maximum depth of 10 rows. Below row 5, hulls conflate so closely with the anchor lines that even for this small example, they are not really visible.

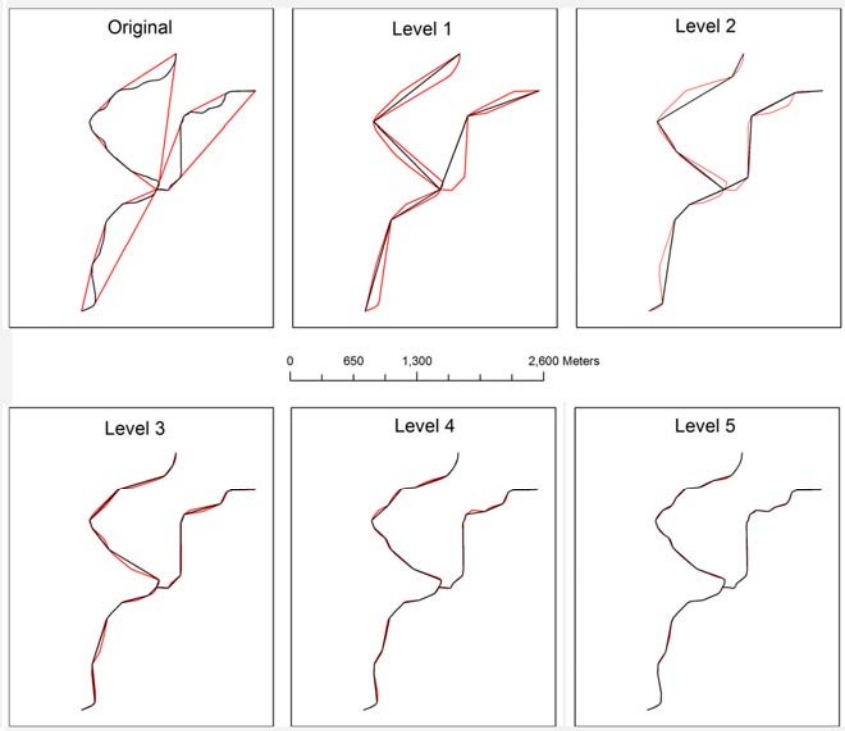

Figure 6. Portion of a stream tributary from Boulder County, with hulls. Pyramid rows are displayed for the first five rows of the pyramid.

4.0 Dealing with more complicated vector features

4.1 Preserving internal topology for a file containing compound vectors (e.g., stream tributaries, or road networks). This aspect of the proposed work handles compound polylines such as in a census tract boundary file, or in an EPA River Reach file. We incorporates multiple trees (each describing a single tributary) into a "grove" of trees, each grove describing (for example) a stream network. Because tributaries may be characterized by differing geometry that does not mirror other trees' resolutions, one cannot assume that the n^{th} level in every tree corresponds to a uniform level of resolution. Instead, we work from measurements of horizontal displacement (in section 5, we refer to this as width and summarize minimum, maximum and average widths for trees and for groves). Internal topology within a tributary (within a tree) will be preserved by Saalfeld's topology check. A nearly equivalent algorithm will be designed to check for and resolve overlap between tributaries (within a grove). The trees must be initially defined to begin and end at topological nodes (stream confluences). Routines to access data at a specified level of resolution by definition retrieve complete representations; and this insures that vectors intersect at the correct coordinate at all levels of resolution.

Second, overlaps between trees must be resolved. Following the initial definition of polylines, a convex hull must be generated to surround the combined footprint of all polylines, forming a hull for the grove (Figure 7). Convex hulls for lower layers in each tree must be constructed from this initial hull. It is likely that the incremental hull method will once again prove optimal, given that coordinates in any single polyline will enter consideration in sequence. This procedure must be reapplied to the entire archive whenever a new tree is added (for each new tributary in the network). As a consequence, archive update carries an extra computational load, but topological checks proceed efficiently.

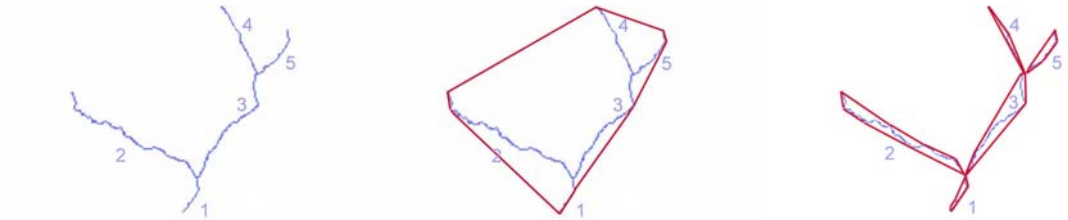

Figure 7. Establishing internal topology for a compound vector. On the left, see five tributaries of a stream network. In the middle, a convex hull is established for the entire network (in the database, this is a grove of trees). On the right, convex hulls for each tree are constructed, and tested for overlaps using Saalfeld's (1999) rule.

Checks for overlapping convex hulls utilize Saalfeld's topology check. The same rule applies: if two convex hulls do not overlap, then internal topological problems do not exist at this or any finer level of resolution. If overlap is detected, a flag is set for both hulls, and the search is repeated for the child hulls in each tree. The check is constrained as follows: Test both children from one tree for spatial overlap with the parent of the other, and continue checking and setting flags only for child hulls where overlap occurs. Where no overlap is detected, the procedure terminates. During data retrieval, when a flag is detected, retrieval continues to lower tree levels. By Saalfeld's proof, retrieved representations will neither self-cross nor intersect, and internal topology for compound vectors is preserved.

4.2 Preserving relative topology between data themes. A multi-layer archive can be defined as a suite of data themes, each containing a different feature class (roads, rivers, landmarks, etc.) This aspect of the work alludes to the scenario of the road crossing the stream channel at the bridge. Three compound vector files must be verified for internal topology, one each for roads, streams, and point landmarks, so that the data retrieval will always resolve to a topologically correct solution, assuming that the vectors are free of topological errors to begin with.

In Figure 8, the first step constructs a convex hull bounding the union-ed footprint for three root convex hulls, generating new partitions wherever the convex hulls are found to overlap. This step is non-trivial, as it involves re-generating the two polyline multi-resolution trees, and resetting all internal topology flags. The Figure shows a vector database containing layers for streams (blue), roads (black) and bridges (red).

The internal topology can be preserved using three trees for the stream channels and three for the roads. Internal topology in this case is equivalent to conventional arc-node topology. To correct relative topology between the layers, the original trees must be re-partitioned into five trees each for the streams and roads, as shown by the labels in the figure.

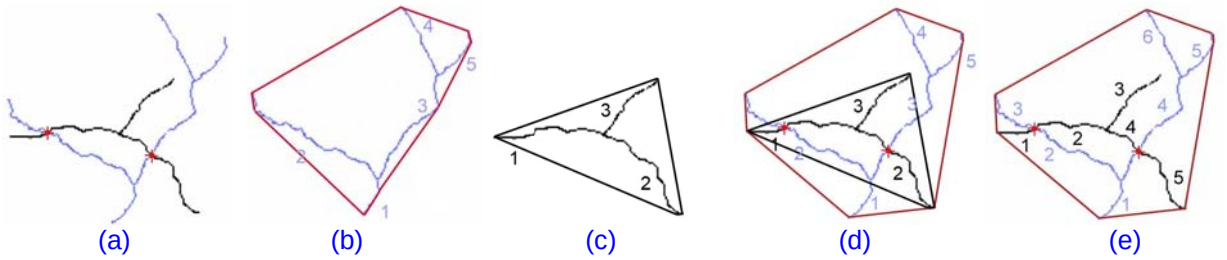

Figure 8. Establishing relative topology between data themes. (a) georeferenced streams, roads and bridges; (b) 5 stream tributaries with common hull; (c) 3 road network with common hull; (d) streams and rivers with bridges and new common hull constructed; (e) spatial intersection of all three themes produces new tributary and road segments (note numbering changes on segments) – relative topology corrected.

An important consideration in generating multi-layer pyramids is determining which data layer should serve as a base framework for the relative topology. Cartographic convention prioritizes naturally occurring data over settlements and transportation nets. For example, a railroad is displaced from a stream (not the other way around) so that the stream will remain synchronized with terrain. This convention would advise one to base relative topology on the stream layer, as opposed to beginning with roads, and that is the solution that will be initially employed. Timpf (1998) presents an alternative and also reasonable strategy for sequencing the transmission of multiple data layers: first transmit the transportation and hydrographic networks to bound regions, then transmit only those objects within regions. A third method would base the relative topologic framework to the layer with lowest dimensionality (the point landmarks).

Another important consideration is that the point landmark file (containing bridges) may also require reconstruction, if any points expand to polylines as described in Section 6.2. In the road-river-bridge example given above, the topology of several segments (e.g., stream segments 4 and 5 and road segment 3) is unchanged from the internal topology solution. It is anticipated that relative topology can be established efficiently by using this knowledge to advantage (that is, by copying convex hulls and portions of existing trees whenever relative topology permits). Shortcuts observed as this solution is designed will be deployed to improve efficiency. Once completed, and again by Saalfeld's proof, the relative topology will be preserved for subsequent multi-layer data retrieval.

5.0 Implementation

Currently the physical schema stores data pyramids in relational tables (Figure 9). The complete set of *vertices* is stored in a single table. Each portion of the line partitioned by RDP deconstruction resides in a *strip*, delineated only by that portion's endpoints. Metadata is stored in the strip table, with fields such as original length, anchor length, bandwidth, monotonicity, etc. *Representations* are simply collections of strips that reconstruct the entire line at a specific resolution. Representations may access strips at more than one tree level, to preserve internal topology. When a representation is created, summaries of the metadata for all included strips are also computed. These summaries are called "structure signatures" (Buttenfield, 1986, 1987, 1989, 1991). Structure signatures summarizing the minimum, maximum and mean anchor length, bifurcation, or the width deviations (called "res_min", "res_max", and "res_mean" are stored in the Representation table; other metadata summaries are requested less often, and computed on the fly from the Strips table. In Section 6, under future work, we discuss how these metadata summaries may help to establish usable limits of resolution for stored data.

Continuing with elements in MRVIN architecture, *trees* are collections of representations, for example, the set of all representations for a single stream channel at all resolutions. *Groves* are collections of trees that

contain compound vectors, for example the entire set of all stream channels in a watershed. The set of all groves for a given study area, including streams, roads, landmarks, administrative boundaries etc. reside in a *Forest.* Internal and relative topology problems are flagged by the dirty bit (next-to-last field in the strip table) and the Dirty Hulls table, respectively. Relational implementation brings several advantages. It obviates the need for pointers to reconstruct representations, which would of course have to be recomputed every time the set of convex hulls is created or revised, as would be the case for data updates or addition of new features. A relational schema also permits fast reassembly of groves, during relative topology correction.

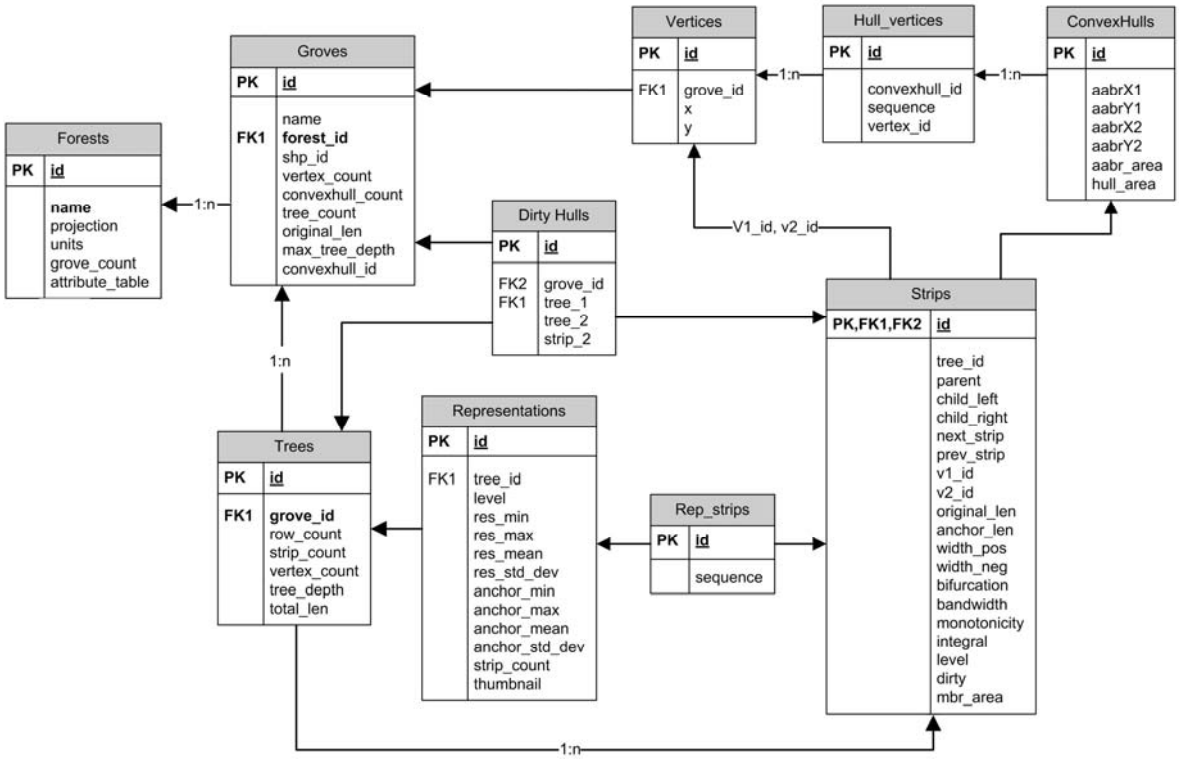

Figure 9 Relational schema for the pyramid architecture. Designations "PK" and "FK" refer to Primary and Foreign keys. Arrows show the directionality of table joins. Other elements of the architecture are described in the text. Vertices and convex hulls are stored explicitly: this creates redundancy in the database, but it speeds some types of search and query. Trees, groves and forest tables may require updating as new data layers are integrated, especially to preserve relative topology.

Figure 10 shows the current status of MRVIN implementation. The RDP decomposition routine can accept either text files or shapefiles. Feature decompositions are stored in a relational database, and routines to test internal and relative topologies are in place and ready to be tested.

The box labeled "calculate geometric structure signatures" is almost complete. This routine records metadata about the geometric properties in strips. The Representation Extraction routines will summarize this metatdata, and create structure signatures described in section 5. On the right, a laptop icon called the MRVIN client indicates an objective to extend our working environment into a distributed client-server architecture, such that mobile applications such as proposed by Neun and Burghardt (2005) could access the vector pyramids. This is an area for future work.

At present the relational database stores only linear features. We acknowledge that polygon topologies will require not only solutions at their boundaries (solved by the work on line features) but also work to preserve adjacency (that is, the continuous fabric covered by the set of all polygons must not be corrupted when internal or relative line topologies are corrected). Our current thinking is that problems

such as buildings jumping across simplified roads, or towns shifting across simplified rivers, can be managed within the convex hull, but to date we have not undertaken systematic testing to establish whether this is the case. This forms an important and challenging area for future work.

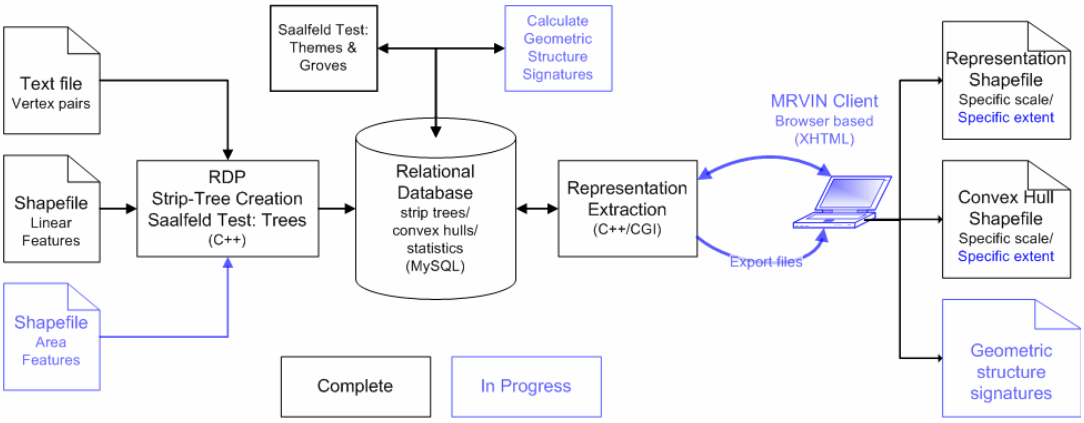

Figure 10. Current MRVIN architecture and status of completion

6.0 Future Work

The progress to date opens many directions for possible work. One establishes a capability common to current GIS environments, namely, spatial clipping, so that users are not restricted to downloading entire trees, groves or forests. A second establishes limits for the range of resolutions beyond which data pyramids lose cartographic or geographic validity. A third direction that we are considering is how to support representations whose dimensionality changes with resolution, for example urban areas.

6.1 Spatial clipping within a multi-resolution vector data model. Any footprint that falls within a compound vector or multi-layer vector database can be spatially clipped, implementing a routine that can operate on either a specified upper right and lower left coordinate pair, or by GIS feature selection. For example, to respond to a query “what is the total length in meters of the tributaries of Clear Creek that lie within Gilpin County?”, the grove of stream features named Clear Creek must be clipped to the county boundary. Clipping must be constrained to a specified resolution. The clip routine must identify the convex hull that contains specified limits and then search child hulls until one of two conditions is reached. Either a set of hulls are identified that are contained completely within the specified clip window; or the leaf nodes of the tree(s) are visited. This solution may require an additional pass through the clipped data subset that adjusts vector topologies to the specified resolution.

6.2 Using metrics to establish ‘usable limits of resolution’ for a multi-resolution, multi-layer vector dataset. This is a more difficult capability to implement, in part because quantitative measures assessing the limits of fitness for use have not been rigorously established. Solutions have been proposed (at previous sessions of this workshop, for example) but none have been widely accepted. Our approach is based on measuring the rate at which details are lost at coarser resolutions. We know that rates vary for different data domains: hydrography is more sensitive to changing resolution than road networks, for example. To compare rates at which details are lost between trees, groves and forests, we plan to utilize the metadata summaries and structure signatures described in Section 5.

Our challenge in doing so can be described as follows: it is likely that in a multi-layer, multi-resolution archive, trees in the grove of roads will be much shallower (contain fewer rows) than trees in the streams grove; and the bridge tree will be very shallow indeed (contain only one or possibly two rows). Metadata comparisons therefore cannot proceed simply by comparing, for example, row 8 for streams with row 8 for roads; instead, we plan to base comparisons on the minimum and maximum resolutions summarized for representations. In coming months, we will conduct experiments with a multi-layer archive of hydrography, transportation networks and administrative boundaries for Boulder County Colorado, to

systematically integrate data representations, and to determine how far one can "push" non-homogeneous levels of detail, addressing the question, how discrepant can representations be without irreconcilable topological problems.

6.3 Linking database descriptions for vector features whose geometric dimensionality changes with resolution. This refers to the issue of the settlement that explodes from a point (a coordinate pair) to an areal region (a compound polyline bounding an area). The problem has also been posed by Devogele et al (1998), and by Haunert and Sester (2004). The solution is to create and link dual feature descriptions, and modify pointers in the pyramid. Instead of pointing to a left or right child, the pointer should be set to another tree. Data groves can be linked just as effectively for a stream segment that metamorphoses from a single channel to a braided channel. The dual descriptions are stored within a single compound vector archive; and internal topological checking proceeds in identical fashion as described above. This is a somewhat different solution than the multi-resolution object identifiers as proposed by Bertolotto and Egenhofer (2001), who did not nest or interleave data models.

Acknowledgments

This work forms a portion of the project "Internal and Relative Topologies for Multi-Resolution Vector Data**",** funded by the US National Science Foundation (NSF BCS 04-51509). Funding by NSF is gratefully acknowledged.

References

Attneave, F. 1954 Some informational aspects of visual perception. *Psychological Review*, 61, 183-193.

Ballard, D. 1981 Strip Trees: A Hierarchical Representation for Curves. *Communication of the Association for Computing Machinery*, vol. 14: 310-321.

Barber, C., Cromley, R.G., and Andrle, R. 1995 Evaluating Alternative Line Simplification Strategies for Multiple Representations of Cartographic Lines, *Cartography and Geographic Information Systems* 22: 276-290.

Bertolotto, M. and Egenhofer, M. 2001 Progressive Transmission of Vector Map Data over the World Wide Web. *GeoInformatica*, 5(4): 345-373, 2001.

Buttenfield, B.P. 2002 Transmitting Vector Geospatial Data Across the Internet Progressively. *Proceedings, Second International GIScience Conference (GIScience 2002)*, Boulder Colorado, September 2002: 51-64.

Buttenfield, B.P. 1991 A Rule for Describing Line Feature Geometry. In: Buttenfield, B.P. and McMaster, R.B. (eds.) *Map Generalization: Making Rules for Knowledge Representation.* London: Longman: 150-171.

Buttenfield, B.P. 1989 Scale-Dependence and Self-Similarity in Cartographic Lines. *Cartographica* 26(1), 79-100.

Buttenfield, B.P.1987 Automatic Identification of Cartographic Lines. *The American Cartographer* 14: 7-20.

Buttenfield, B.P. 1986 Digital Definitions of Scale-Dependent Structure. *Proceedings, AUTO-CARTO LONDON 1: 497-506.*

Cromley, R. G. 1991 Hierarchical Methods of Line Simplification, *Cartography and Geographic Information Systems* 18: 125-131.

Cromley, R.G. and Campbell, G.M. 1990 A Geometrically Efficient Bandwidth Line Simplification Algorithm. *Proceedings 4th International Symposium on Spatial Data Handling*, Zurich: 77-84.

de Berg, M,. van Kreveld, and Schirra S. 1998 Topologically correct subdivision simplification using the bandwidth criterion. *Cartography and Geographic Information Science* 25: 243--257.

Devogele, T., Parent, C. and Spaccapietra, S. 1998 On Spatial Database Integration. *Internatonal Journal of GIScience* 12(4): 335-352.

Douglas, D.H. and Peucker, T.K. 1973 Algorithms for the Reduction of the Number of Points Required to Represent a Line or Its Caricature. *The Canadian Cartographer* 10(2): 112–122.

Fabrikant and Buttenfield, 2001 Formalizing Spaces for Information Access. *Annals, Association of American Geographers,* 91(2): 263–280.

Harrie, L., 1999 The Constraint Method for Solving Spatial Conflicts in Cartographic Generalization. *Cartography and Geographic Information Science* 26(1): 55-69.

Haunert, J. and Sester, M. 2004 Using the Straight Skeleton for Generalization in a Multiple Representation Environment. Paper presented ICA Workshop on Generalization and Multiple Representations, Leicester, UK, August 2004.

Hershberger, J. and Snoeyink, J. 1992 Speeding Up the Douglas-Peucker Line-Simplification Algorithm. *Proceedings 5th International Symposium on Spatial Data Handling*, Charleston, SC: 134 – 143.

Jones, C. B., Abdelmoty, A.I., Lonergan, M.E., Van Der Poorten, P.M. and Zhou, S. 2000 Multi-Scale Spatial Database Design for Online Generalisation. *Proceedings, International Symposium on Spatial Data Handling*, Beijing, PRC, August 2000.

Neun, M. and Burghardt, D. 2005 Web Services for an Open Generalization Research Platform. Paper presented 8th ICA Workshop on Generalization and Multiple Representations, A Coruna, Spain, July 2005.

Peucker, T. K.1975 Theory of the Cartographic Line. *Proceedings AUTO-CARTO 2,* Reston VA: 508-518.

Ramer, U. 1972 An Iterative Procedure for the Polygonal Approximation of Plane Curves. *Computer Vision Graphics and Image Processing* 1: 244-256.

Saalfeld, A. 1999 Topologically Consistent Line Simplification with the Douglas-Peucker Algorithm. *Cartography and Geographic Information Science* 26(1): 7-18.

Tobler, W. R. 1987 Measuring Spatial Resolution. *Proceedings, International Workshop on Geographic Information Systems, Beijing, PRC: 42-48.*

van der Poorten, P. M., Zhou, S. and Jones, C. B. 2002 Topologically Consistent Map Generalization Procedures and Multi-scale Databases. Proceedings GIScience 2002: 209:229.

Van Putten, J. and van Oosterom P. 1998 New Results with Generalized Area Partitionings. *Proceedings 8th International Symposium on Spatial Data Handling, Vancouver Canada: 485-495.*