DEVELOPMENT OF GIS-BASED CONFLATION TOOLS FOR DATA INTEGRATION AND MATCHING

Final Report: Executive Summary

Research Center
Florida Department of Transportation
605 Suwannee Street, MS 30
Tallahassee, FL 32399-0450

Submitted by
Ram M. Pendyala, Ph.D.
Principal Investigator
Department of Civil and Environmental Engineering
University of South Florida, Tampa, FL 33620

In Collaboration with
Oak Ridge National Laboratory
URS Corporation
GIS Software and Consulting, Inc.

Contract No. BC353, RPWO #21
2002

Acknowledgements

The project team is grateful to the members of the Statewide Model Task Force for their technical guidance and advice throughout the course of this research project. The project team thanks Mr. Robert G. McCullough, P.E. and Vidya Mysore of the Systems Planning Office for their help in coordinating this research effort with other ongoing model development projects in the state.

Disclaimer

Executive Summary

Background

There are a variety of data sources and networks that support the transportation planning and modeling processes in Florida. Quite often, there is a need for integrating data and matching networks and their associated attributes across different data sources. The process of integrating, matching, and merging disparate data sets and networks, which is often quite time consuming and labor intensive, is referred to as "conflation". As the number of data sources and networks from which attributes and variables are drawn continues to increase, the need for performing conflation rises too. At this time, there are a few statistical packages and GIS packages that offer some limited conflation capabilities. However, these capabilities are few and far between and require considerable manual work on the part of the user. In recent times, databases and networks have become increasingly complex and therefore manual data conflation is extremely difficult, time consuming, and prone to errors.

There is a need for a comprehensive set of conflation tools that allow the matching, merging, and integration of data and networks across disparate data sources. The conflation tools should be GIS-based and operate on standard database formats so that they are applicable across a wide variety of contexts and applications.

The statewide model uses the basemap created and maintained by the Transportation Statistics office. This basemap is in ESRI coverage format using route features that are linked to the RCI database stored in Oracle. A network model has been created using the graphic representation of the routes from the basemap and using the node and route break points in the RCI database. This resulted in the development of two ESRI shapefiles.

The first is a point shapefile representing every node in the statewide model network. The second is a line shapefile containing the links that connect each node. The major RCI attributes are stored with each link in the link shapefile. These attributes include, but are not limited to, the Average Annual Daily Traffic (AADT), number of lanes, maximum speed, functional class, area type, etc... More importantly, each link has the name of the route that it represents in the Transtat basemap, and the beginning and ending milepost of that route that the link represents.

These attributes (Roadway, Begin_post, End_post) are the key attributes that allow the RCI database and other databases at the Florida Department of Transportation to join with the basemap. The RCI database, Work Program, Traffic Information System, and other databases all use the roadway-begin_post-end_post format for identifying portions of the roadway on the State roadway system.

The first application of the conflation tools developed in this project resulted in the development of a relationship between the statewide basemap model and the FSUTMS models in the state. By having this relationship in place, FDOT is able to transfer attributes from one model to another. This will allow attributes stored in the FDOT databases (RCI, WPA, TRIS, etc...) to be transferred to the FSUTMS model. The opposite will also be true, allowing FSUTMS model data to be transferred to a Roadway-Begin_post-End_post format that can be used with other FDOT databases.

Expected Use of the Conflation Software

The average user should not have to work with the conflation software. It is expected that the conflation will be performed either in the central office on all FSUTMS models, or the individual districts will perform the conflation process on their FSUTMS models using a copy of the basemap model. *The conflation process should only be run whenever the geometry of the basemap model or the geometry of the FSUTMS model changes.* Normally, the results of the conflation process will be used to transfer data between FSUTMS and RCI style databases. It is expected that once the conflation is done, the user will be able to transfer attribute data between the FSUTMS and Basemap data formats.

To transfer data from FSUTMS to Basemap format, the user is prompted to select the FSUTMS model and the attribute to be extracted. The result is a table of data in Roadway, Begin_post, End_post, and Attribute format.

To transfer data from the Basemap format, the user starts out with a table containing Roadway, Begin_post, End_post, and Attribute data. The user is then prompted for the FSUTMS model into which the attributes should be transferred. The FSUTMS model is then updated with the new attribute.

To achieve this simple data transfer mechanism, the results of the conflation process resemble a joining table. This joining table has the key to each link in the FSUTMS model and a corresponding Roadway-Begin_post-End_post for each FSUTMS link. It is possible that multiple basemap roadways and begin/end posts correspond to a single FSUTMS link. In the same vein, a single basemap roadway begin/end post pair may correspond to multiple FSUTMS links. It is also assumed that some links won't have an equivalent in the basemap model and vice-versa.

The joining table looks similar to the following:

Link ID	Start %	End %	Roadway	Begin_post	End_post
12345	0	100	87040000	0.000	4.231

This table has the link id from the FSUTMS data along with a beginning position (percentage) and ending position (percentage). This allows a portion of a FSUTMS link to be associated with a portion of a roadway in the basemap. A table like this exists for every basemap/FSUTMS network pair. Since only a single basemap network exists, a table for each FSUTMS network conflated to the basemap network can be built and maintained.

Automated Conflation with Manual Processing

As conflation is not an exact science, the automated routine can not give the complete conflation between two maps. The conflation tools attempt to give accurate results up to a certain percentage. After that point, the user has to direct the software on the unusual or complex geometries that are being conflated. The conflation software can be tweaked and controlled up to the point where the user has reached an optimal percentage of correctness. This percentage is determined on a case-by-case basis depending on the complexity of the two networks.

At that point, the user will begin to “fill in the gaps” of the conflation process. This will also include overrides of the conflation routines for known relationships. When the user starts to input these overrides, the conflation process will respect these and the user will still be able to run the conflation routines during the iterative process of getting a 100% correct conflation result.

This procedure will also help with the maintenance of the systems. As the basemap network and the FSUTMS networks are changed over time, the user will be able to use the conflation routines to discover the differences of a FSUTMS or basemap network over time. This comparison analysis of similar format networks will help the user during the maintenance and upkeep of the conflation results between FSUTMS and basemap networks.

Successful Conflation

Success in the conflation process would be that every link in both models has been matched. The amount of manual intervention is kept to a minimum. The automated conflation routines can be run easily and users can change the control files to ensure an accurate conflation. A tool to find errors that are created by the conflation process has been incorporated to measure the effects of changes to the control files.

Procedures for Conflation

The product emanating from this effort is different than most conflation efforts. Most conflations focus on improving the spatial accuracy of one coverage and consider the transfer of attributes between layers ancillary. This project has adopted a reverse approach. The goal is to be able to, with the help of database tools, if necessary, transfer attributes between networks.

Using the approach of a lookup table to relate one network to another allows the geometry of both networks to remain intact. In fact, a spatial conflation can be performed at any time by placing attributes from one network onto the other's linework. This, of course, does not prevent the use of scratch coverages for performing the conflation itself.

The lookup table in the conflation tools addresses the issue of many-many relationships. Many to one relationships between local models and the statewide model are common as the statewide model is more general. Statewide model links also have a many to one relationship with local model links where significant attributes in the statewide model are not reflected in the local model.

In order for the lookup table to work, the link identifiers of the conflated models have to be static. It is necessary for the user to be able to refer to each link with an unique identifier. Updates should be possible, but it is envisioned that these will require some additional manual processes and some batch routines.

The lookup table contains the least common denominator between the two networks. In other words, a new record is generated by the tool any time there is a change in either of the models' structure. The drawing below illustrates this point:

```
0  A 50 B 100
*-----*----*
*----------*
     X
```

The two overlapping sets of features would be represented in a table as:

network1id	network2id	nw1from	nw1to	nw2from	nw2to
A	X	0	100	0	50
B	X	0	100	50	100

The table contains records for every coincidence between the two networks. Links not found in the table have no corresponding link. There is an issue of whether to include the union or the intersection of the two networks in the table. So long as one is dealing only with two networks, the intersection works as well as the union. If we start using a 'floating platform' (e.g., GDT databases and networks) to relate various networks, however, it becomes important that the floating platform include the union of all geometry for the various related layers.

The development of the lookup table in the conflation tools makes it possible to conflate many networks. A new table would be created for each new relationship. If networks 1 and 2 are both conflated to one common 'floating platform' then relating 1 to 2 is a matter of performing a database query. This assumes, however, that the 'floating platform' contains the universe of all network geometries.

1. INTRODUCTION

Matching road networks that come from different sources or have been created at different times is an important function for transportation data management and manipulation. Frequently, applications need to integrate data that are referenced by different networks for the same area (Rosen and Saafeld, 1985; Saafeld, 1988; Brown, Rao, and Baran, 1995; Nystuen, Frank, and Frank, 1997; Walter and Fritsch, 1999). A common procedure used to realize this integration is to establish correspondences among different networks; then data that are referenced on different networks are transferred to and integrated into one of the networks. The same procedure is necessary in situations where an existing database grows older and has to be updated with a newer version of the network. In this case, the newer version of the network functions as the control layer, and matches between this newer version and the earlier versions are established. Then data on the earlier versions of networks are conflated to the newer version.

Network matching also has important applications in image processing (Novak, 1992; Stilla, 1995; Wang, 1998). Raw images obtained from remote sensors usually contain various kinds of distortions. To geometrically rectify a distorted image, this image can be overlaid with a network map. By establishing correspondences between linear features on the image and on a network map, the image geometry can be transformed and rectified. Matching with a network map is also a good strategy for image recognition. When unknown linear features on an image can be matched with known features on a network map, the unknown features on the image can be recognized as the same features as the ones on the network map. After matching, the characteristics of the matched features can be extracted, and by referencing these characteristics, other features on the image that may not directly correspond to an existing map can be recognized.

Because of the importance of these applications, considerable effort has been devoted to studying and developing automated procedures for network matching. This research focuses on an investigation of an algorithm that consists of three types of matching: node matching, segment matching, and edge matching. Node matching establishes node correspondences between two networks using Euclidean distance and angle patterns formed by incident edges. Segment matching tracks segment pairs along potential matching edges. The result of segment matching can be used to locate node locations on the edges of the counterpart network when no node correspondences are established. In addition, measures of segment matching can be aggregated to facilitate higher-level matching (e.g., edge matching). Edge matching takes input from segment matching and then derives matching measures at the edge level. Applying these matching measures allows identification of the best matching pairs.

In this research, a computational strategy that combines bottom-up and top-down computations is adopted. The bottom-up computation starts with node matching, then proceeds to segment matching, and finally ends up with edge matching. The top-down procedure is the reverse. At the initial stage of the top-down computation, potential edge-matching pairs are first hypothesized using screening criteria such as distance and angle difference. Then segment matching proceeds. Through segment matching, matching measures for those hypothesized matches are computed, like edge matches are confirmed, and unlike matches are rejected. The purpose of combining the bottom-up and top-down computations is that the bottom-up computation will find matches where node matches can be quickly established, while the top-down computation will find matches where node matches fail or network structures differ.

This three-stage matching procedure promises good improvements to previous matching procedures for two reasons. First, previous research has frequently used node matching and edge matching, but not segment matching. Segment matching is important because it can be used to evaluate correspondences between each pair of segments on potentially matched edges. Through segment matching, matching measures between each pair of segments are first computed; then, overall matching measures at the edge level are obtained. Because of these detailed considerations, edge-matching measures derived from segment matching can be more sensitive in recognizing differences and similarities among different edge pairs. Using these measures, there is a better chance to find the best matches.

The other advantage of the current algorithm is in its overall computational procedure. Previous research has often used a bottom-up procedure as the main computational strategy. There are some exceptions [e.g., the two-stage matching method developed by Gabay and Doytsher (1994)]. The current algorithm not only combines the bottom-up and top-down computations but also considers interaction and feedback between different stages of matching, especially in segment matching and edge matching, thereby providing some refinement to those procedures that already have a computational mechanism in place such as the one developed by Gabay and Doytsher.

In the following discussions, Section 2 provides an overview of previous research on the subject of network matching; Section 3 describes the proposed matching algorithm; Section 4 reports on the matching experiment; and Section 5 provides conclusions and discussions of the current algorithm with respect to its overall performance and necessary improvements needed in the future.

2. PREVIOUS RESEARCH

The problem of network matching has been studied in different disciplines, and related literature can be found in the areas of GIS, cartography, transportation, and image processing. Due to diversified references, different terms have been used in the literature. These terms include *map matching, conflation, and linear alignment*. In this section, alternate terms will be used as well.

One of the earliest matching algorithms was developed by the U.S. Census to integrate data from the U.S. Geological Survey and the Census Bureau (Rosen and Saalfeld, 1985; Saalfeld, 1988). This algorithm was applied to match maps that have a link-node structure. A bottom-up computational procedure is utilized in this algorithm, which uses node matching as the starting point. After node matching, a rubber sheeting operation is carried out. In this operation, the geometry of one of the maps is adjusted to make it correlate better with the other map. Finally, matches of the corresponding links on the two maps are identified.

The method introduced by Gabay and Doytsher (1994) represents a major departure from previous matching algorithms. The major advantage of their method is that it is able both to find common elements on two maps and to reveal unique elements appearing on only one of the maps. This capability allows geometric inconsistency and differences of topological characteristics to be recognized and handled during the map matching process. The two-stage matching procedure proposed by Gabay and Doytsher is also a blueprint of the strategy that combines the bottom-up and top-down computations. In this two-stage matching procedure, lines that have matched end nodes are matched first, then unmatched lines are further evaluated, and final matches are obtained.

Brown, Rao, and Baran (1995) described a conflation system that was developed in a GIS environment. This system makes use of existing GIS functions such as rubber sheeting and dynamic segmentation to adjust network geometry and establish node and link correspondences of two matching networks. With the system, linear mappings along network edges are possible, and network attributes such as direction flags and distances can be assigned or computed automatically when these attributes are transferred from one network to another network.

Nystuen, Frank and Frank (1997) studied a method that compares two networks for the purpose of evaluating the quality of digital network databases. In their research, they developed an automated procedure to associate network elements. Through nearest node analysis and buffering, corresponding nodes and edges on two participating networks are identified and statistics showing how well two networks correspond each other are generated. Their studies identified several problems arising from network comparison. These problems, including scale effects, definitional discrepancies, data model differences, and the need for pattern recognition, represent major research directions in network matching.

More recently, Walter and Fritsch (1999) developed a statistical matching algorithm that provides significant advancement to the state of the art. Walter and Fritsch’s algorithm has two important properties. First, their algorithm takes a relational matching approach that bases matching decisions not only on the characteristics of a matching pair, but also on whether their neighbors are matched or not. Second, information theory is utilized to derive sophisticated matching measures to facilitate matching decisions. By transforming the map-matching problem into a problem of searching for the best communication channel in a communication system, the algorithm obtains optimal matches by selecting matching pairs whose mutual information is maximal. With a combination of these two properties and a bottom-up implementation, matches can be derived first at the local level, then for clusters, and eventually for the entire network.

Matching techniques have also been studied in image processing for purposes of image registration and recognition. Considerable literature can be found in the review article by Fonseca and Manjunath (1996). The method of relational matching or structural matching that was developed in computer vision (Shapiro, 1980; Shapiro and Haralick, 1981) is particularly relevant to network matching. Ventura, Rampini, and Schettini (1990) and Wang (1998) successfully applied this method in image registration. Walter and Fritsch (1999) incorporated the concept of relational matching in their network-matching algorithm. The major strength of relational matching is that it utilizes not only characteristics of individual participating components but also topological and geometric relationships between these components and other corresponding components when matches are evaluated.

Despite existing research, several issues remain to be addressed for automated network matching. In previous research, edge matching has usually been approached by matching end nodes of edges and by searching corresponding edges in a given buffer. In cases where multiple edges are close to each other, identifying best matches can be difficult. Segment matching is likely to be useful in dealing with this problem. By comparing edges at the segment level, one can measure similarity and differences between edges precisely. Nevertheless, segment matching has not been adequately discussed in the literature. Integrating segment matching into the network-matching algorithm therefore represents a major contribution to the effort reported here.

As for the computation of network matching, a great deal of research has been done for the bottom-up procedure, but not much for the top-down procedure. The top-down procedure is generally needed in situations where a priori information about potential matches can be obtained. For instance, when matching candidates are identified or hypothesized on the basis of information about proximity or aspatial attributes, the top-down procedure can be conveniently utilized to verify these matches and find the best matches. Considering that the bottom-up procedure has its advantage in other matching tasks, this research takes an approach that combines both the bottom-up and top-down procedures. To do so, different procedures can be used to effectively handle different tasks.

3. THE ALGORITHM

This section introduces some notations and definitions and describes the overall procedure used to carry on the calculation of network matching and the individual algorithms that constitute the overall computational procedure.

3.1. Notations and Definitions

For simplicity, the symbol $G = (N, E)$ is used to represent a directed network graph, planar or non-planar, with $N = \{n_1, \ldots, n_m\}$ representing the node set, and $E = \{e_1, \ldots, e_p\}$ representing the edge set. For matching purposes, each node will have a set of attributes, including an ID, the number of edges that are incident to the node or called degree $D(n)$, the incident edge set, $n(e) = \{e_1, \ldots, e_{D(n)}\}$, and a coordinate pair (x, y) representing the location of the node. An edge-attribute set will include an ID, from-node and to-node (e.g., n_i and n_j), and a set of coordinate pairs $\{(x_1, y_1) \ldots (x_k, y_k)\}$ representing the shape points of the edge. The shape points on an edge also define a set of segments, and this segment set is denoted as $S = (s_1,\ldots,s_{k-1})$.

The matches of two network graphs $G_a(N^a, E^a)$ and $G_b(N^b, E^b)$ are defined with three sets of mappings: node mapping, $M_n = \{(n^a, n^b) \mid n^a \in N^a \text{ and } n^b \in N^b\}$; segment mapping, $M_{es} = \{(s_e^a, s_e^b) | s_e^a$ representing a segment on an edge, e^a, of the G_a, and s_e^b representing a segment on the edge e^b that corresponds to $e^a\}$; and edge mapping, $M_e = \{(e^a, e^b) \mid e^a \in E^a \text{ and } e^b \in E^b\}$. Because precise correspondences between elements (nodes, segments, and edges) of two networks are rare, it is assumed that mappings between network matching pairs are based on edited network graphs rather than on the original ones. That is, network edges may be split or merged (or network nodes inserted or removed) when mappings between two networks are generated.

In this discussion, one of the networks, G_a, that participates in the matching is called the *reference network* [following the definition by Nystun, Frank, and Frank (1997)], and the other participating network, G_b, is referred to as the *matching network*. In general, it is assumed that the reference network functions as a control or reference layer. When data from two networks need to be integrated, usually data from the matching network will be transferred to the reference network. In cases where networks need to be geometrically rectified, the reference network will be used as a control layer, and the geometry of the matching network will be transformed.

3.2. The Computational Strategy

The major objective of network matching is to generate three types of mappings: node mapping, segment mapping, and edge mapping. Naturally, calculations of these mappings involve three types of computations: node matching, segment matching, and edge matching. As network elements (nodes, segments, and edges) are intrinsically related to each other, it is not efficient to compute individual mappings separately. To integrate these three types of computations, we consider an overall computational strategy as shown in Fig. 1.

For this overall computational strategy, the matching computations are divided into two sub-processes, the bottom-up and the top-down. The bottom-up computation starts with matching network nodes using criteria of Euclidean distances between nodes and angle patterns of network edges that are incident to these nodes. After node correspondences are established, segment matching proceeds. Segment matching involves both tracking corresponding segments of a potential match and computing the likelihood of whether one segment indeed forms the counterpart of another segment. Finally, edge matching is carried out. In edge matching, matching measures obtained from segment matching are first aggregated at the edge level. Then these measures are applied to eliminate unlikely matches.

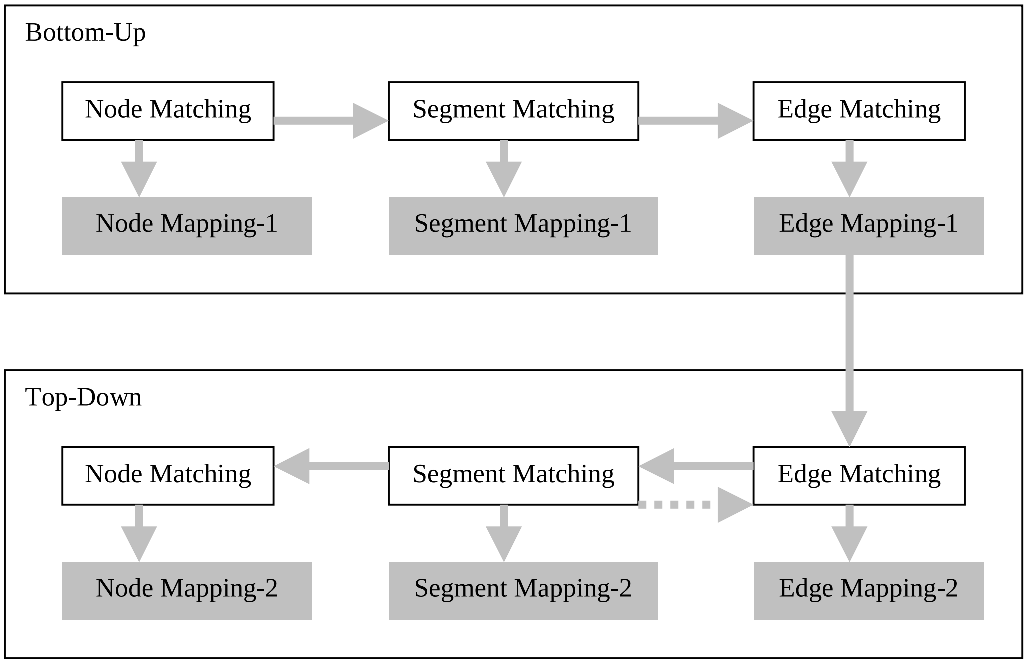

Fig. 1. The overall computational strategy.

By searching nearest neighbors and exploiting network topological relationships, the bottom-up computation can quickly find node and edge matches at locations where node correspondences can be reliably established, but this method may leave good matches unidentified at other locations. To deal with this problem, the top-down computation follows. The top-down computation assumes no node correspondences; instead, it proceeds with generating matches between network edges as the first step. With the bottom-up results, the top-down computation will need only to evaluate those edges that are not matched after the bottom-up matching. Note, however, that edge matching actually proceeds with two stages in the top-down computation. In the first stage, potential candidates on the matching network that may form a counterpart to a given edge on the reference network are first identified. After that, segment matching takes over, and for each pair of edge-matching candidates, segment mappings are generated. With these segment mappings, edge-matching procedure is reactivated to compute matching measures at the edge level (or sub-edge level if edges are split). And finally, decisions are made to find the best edge matches.

The top-down procedure can be independently implemented. In this case, the edge-mapping set from the bottom-up matching, as shown in Fig. 1, is assumed to be empty.

With the overall procedure described above, we can now treat each of the three matching procedures (node matching, segment matching, and edge matching) separately.

3.3. Node Matching

Because of errors and distortions in network databases, the same nodes on different networks usually may not have an exact geometric match, e.g., $x_i^a \neq x_j^b$ and $y_i^a \neq y_j^b$. Instead, a gap exists between these two corresponding nodes:

$$d_{ij}^n = \sqrt{(x_i^a - x_j^b)^2 + (y_i^a - y_j^b)^2}\,. \tag{1}$$

Assuming that geometric errors and distortions of the networks tend to be local and are bounded ($d_{ij}^n < \Delta$), the gap between a corresponding node pair d_{ij} can be used as a criterion to search for candidate node matches.

Because each node has one or more edges incident to the node, if two nodes represent the counterparts of each other on two networks, the similarity of angle patterns formed between the nodes and edges can be used as another matching criterion. Assume the angle between two edges referenced to the same node is calculated by

$$\varphi_{kl}^n = ar\cos\left(\frac{v_k^a \bullet v_l^b}{|v_k^a||v_l^b|}\right), \tag{2}$$

where v_k^a and v_l^b are vectors formed with e_k^a and e_l^b, and ($e_k^a \in E^a$; $e_l^b \in E^b$) and it is assumed that v_k^a and v_l^b always have a direction pointing away from the referenced node. Then an angle-matching measure between two nodes can be evaluated with

$$d(\varphi)_{ij}^n = \min_{kl}(\varphi_{kl}^n)\,. \tag{3}$$

The minimization sign used in equation 3 means that the minimum angle difference is used to determine whether at least one pair of edges is matched for these two corresponding nodes. With the distance and the angle difference criteria, node matching is computed with a decision function:

$$F_{ij}^n = \begin{cases} 0 \ if \ d_{ij}^n > \Delta, \\ 0 \ if \ d(\varphi)_{ij}^n > \Phi, \\ 1 \ if \ d_{ij}^n = \min \ or \ 0 \ otherwise, \end{cases} \tag{4}$$

where Δ and Φ are thresholds set for distance and angle difference, beyond which no matches will be allowed. In equation 4, 0 indicates no matches, and 1 indicates the two nodes are matched. The procedure to derive matches between nodes of two networks is straightforward. The inputs for this procedure include node and edge coordinates of the two networks, the maximum matching distance, and the maximum angle difference. The computation proceeds as follows:

Step 1. Initialize the node-mapping set [e.g., set $M_n = \{(n_1^a, \varnothing), \ldots, (n_m^a, \varnothing)\}$].

Step 2. For each node n^a on the reference network, find all the nodes on the matching network that are within the maximum distance Δ. Then apply the matching criteria using equation 4, and find the best match.

Step 3. Output the mapping M_n, which now contains nodes on the matching network that correspond to the nodes on the reference network.

3.4. Segment Matching

Segment matching establishes detailed correspondences at the segment level between edges. It can find corresponding locations of nodes that are represented on one network but not on the other network, and generate measures that can be aggregated at the edge level for edge-matching decisions. We utilize three criteria to evaluate segment matching: angle difference between segments, distance between segments, and matched segment length. The use of these three criteria is based upon the consideration that angle difference and distance are direct measures of how well two segments can correspond each other, while the length can be used as a weighting factor for both the angle difference and the distance.

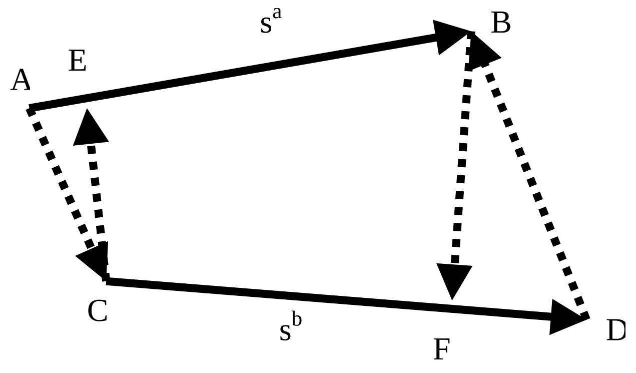

Fig. 2. Segment configuration.

To effectively match segments, a configuration, as shown in Fig. 2, is used to determine relationships between segments of the two participating networks. In this configuration, s^a represents a segment on an edge of the reference network, and s^b represents a segment on an edge of the matching network. The angle difference between s^a and s^b can be computed using an equation similar to equation 2, the only difference is that the vectors used in the equation are now the two segments instead of the two edges.

Before the distance between the two segments is calculated, four distance measures are first derived: d^{AC}, d^{CE}, d^{BF}, and d^{DB}. As shown in Fig. 2, d^{AC} represents the distance from the starting point A of s^a to s^b, d^{CE} represents the distance from the starting point C of s^b to s^a, d^{BF} represents the distance from the ending point B of s^a to s^b, and d^{DB} represents the distance from the ending point D of s^b to s^a. The distance between the two segments then is computed with

$$d_{ab}^{s} = \frac{1}{2}\left(\min(d^{AC}, d^{CE}) + \min(d^{BF}, d^{DB})\right). \qquad (5)$$

Notice that distance between the two segments can be defined in different ways—for instance, min (d^{AC}, d^{CE}, d^{BF}, d^{DB}). Nevertheless, equation 5 provides an average distance measure that takes account of the distances at both ends of the segment pair. By contrast, measures such as min (d^{AC}, d^{CE}, d^{BF}, d^{DB}) give distance measures only at certain points. When corresponding segments stretch into different directions, these measures may fail to provide sensitive information in a particular match. During the computation of the segment distance, the lengths of the effectively matched segments are also determined. In the case of Fig. 2, the matching lengths are d^{EB} on s^a and d^{CF} on s^b.

Segment matching in general involves intensive computations. To carry out these computations effectively, a tracking procedure that can coordinate with the node-matching and edge-matching computations is devised. The major steps for this tracking procedure are as follows:

Step 1. For a given edge pair (that may be derived from node matching or assumed as a potential matching candidate from edge matching), first compare their directions. If they are in opposite directions, the order of the shape points for one of the two edges will be reversed.

Step 2. Search for the first pair of matching segments along the two edges. This matching pair must be within the allowed maximum distance and in the same direction.

Step 3. Compute the angle difference, distance, and matching lengths as described earlier for each segment pair. Continue the process until one or two of the edges end or break away. During the process, the decision on which segment is selected for the next computation is based on whether a segment is an overshot or an undershot (in Fig. 3, s^b is an overshot and s^a is an undershot). The segment that is next to a previously undershot segment will be selected for next computation (in Fig. 3, the segment next to s^a will be selected).

3.5. Edge Matching

For edge matching, we use three matching measures: average angle difference, average distance, and total matched lengths. These three measures are derived from segment matching but aggregated at the edge level. The average angle difference is computed with the length-weighted average of segment angle differences. The average distance is the length-weighted average of segment distances. The total matched lengths are the total lengths of the edges on the reference network and on the matching network respectively. With these measures, the overall matching measure between two edges are calculated with

$$D_{ij}^{e} = \alpha d_{ij}^{e} + \beta d\phi_{ij}^{e} + \delta(L_i^e + L_j^e), \qquad (6)$$

where i and j represent ith edge on the reference network and jth edge on the matching network; d_{ij}^{e}, $d\varphi_{ij}^{e}$, L_i^{e}, and L_j^{e} represent the average distance, average angle difference, total matched length on the edge of the reference network, and total matched length on the edge of the matching network separately; α, β, and δ are weighting factors used to balance the effect of different measures. In program implementation, a decision function is utilized to facilitate matching decisions:

$$F_{ij}^{e} = \begin{cases} 0 \;\; if \; d_{ij}^{e} > \Delta^{e}, \\ 0 \;\; if \; (L_i^{e} < L^{e} \; and \; L_i^{e'} > L^{e}) \, or \, (L_j^{e} < L^{e} \; and \; L_j^{e'} > L^{e}), \\ 1 \;\; if \, D_{ij}^{e} = \min \, or \, 0 \, otherwise, \end{cases} \tag{7}$$

where 1 indicates a match, and 0 indicates no match; Δ^{e} represents the maximum distance allowed between two edges, and L^{e} represents the minimum length required for an edge match; for D_{ij}^{e} = min, it is assumed that for all potential matches, the matching pair with the minimum overall measure will be identified to make the match. $L_i^{e'}$ and $L_j^{e'}$ are the total lengths of the edge pair, which may be different from the matched lengths, L_i^{e} and L_j^{e}. The reason to use $L_i^{e'}$ and $L_j^{e'}$ as constraints in equation 7 is to make sure that short edges are not eliminated from the matching process.

In designing procedures to carry out edge matching, two scenarios must be considered. The first scenario is when the segments between two edges have been matched (e.g., edge matching with the bottom-up computation). In this case, the task of edge matching is to apply the aggregated matching criteria derived from segment matching to validate a potential match. The second scenario is when edge matching starts without segment matching (e.g., edge matching with the top-down procedure). In this case, edge matching proceeds with selecting a set of potential matching candidates as the starting point, then segment matching follows. Through segment matching, edge matching measures are obtained for each of the assumed matching pairs. With these measures, the best match will be finally identified. Since edge matching in the second scenario contains all the steps necessary for edge matching in the first scenario, the procedure of edge matching for the second scenario is outlined below:

Step 1: Initialize the edge-matching mapping, M_e, which starts as an empty set.

Step 2: For each edge on the reference network, find all the edges on the matching network that are located within the maximum matching distance. Before proceeding to segment matching, use a preprocessor to group segments to form sub-edges in cases where part of an edge is included within the maximum matching distance. Then, proceed with segment matching, and compute edge-matching measures for each potential edge pair. Finally, carry out the matching test (*substep 2.1*) to find the best match if there is one.

Substep 2.1: The matching test first checks whether matched edges meet the requirements of the minimum length, the maximum angle difference, and the maximum distance. Then the matching measures of all the potential matching pairs that have met these minimum or maximum requirements are compared, and the pair with the maximum overall matching measure is identified as the best match. If no matching pairs meet the minimum and maximum requirements, there will be no matches.

Step 3: During *Step 2*, the edge-mapping set, M_e, grows into a list that contains corresponding edges on the two networks. This list, however, may also contain entries where edges on the matching network match multiple edges on the reference network redundantly. Before the mapping set is finalized, use matching measures again to eliminate these overlapped mappings.

4. THE EXPERIMENT

To examine how well the proposed algorithm works, a computer program uses the above-described programming logic was developed. This program was built in C and works in the Windows environment. Currently, the program is able to read and write networks in ESRI's shape file format and has the ability to perform fast searches for nearest points and segments. This section reports on the experiment of matching two versions of the U.S. waterway networks using this program.

4.1. The Data

The data used for the matching experiment are waterway networks for the eastern United States that are derived from the 1995 and 1996 versions of the National Transportation Atlas Databases (NTAD) provided by the U.S. Bureau of Transportation Statistics. These two networks were chosen because a major revision occurred between the 1995 and 1996 networks; it was therefore a good test case. Fig. 3 shows the 1995 network, and Fig. 4 shows the 1996 network. Both of the networks are shown on an Albers equal-area projection.

Given the map scales in Figs. 3 and 4, specific differences between these two networks are not obvious. Fig. 5 provides a detailed comparison of these two networks. As shown in Fig. 5, the 1996 network is represented with thick gray lines as the edges and big gray dots as the nodes. The 1995 network is represented with thin dark lines as the edges and small dark dots as the nodes.

Three major differences between the 1996 network and the 1995 network can be identified. First, some of the edges on the 1995 network have a coarse geometry, but these edges have been replaced with edges that have more naturally curved shapes on the 1996 network. Location shifts also occur on some of the edges of the 1996 network, which apparently is in an attempt to improve the accuracy of the network geometry. Second, in some cases, the 1996 network has shorter edges, which means that one edge on the 1995 network may correspond to several edges on the 1996 network. Third, the

1996 network introduces some new edges that do not have correspondences on the 1995 network.

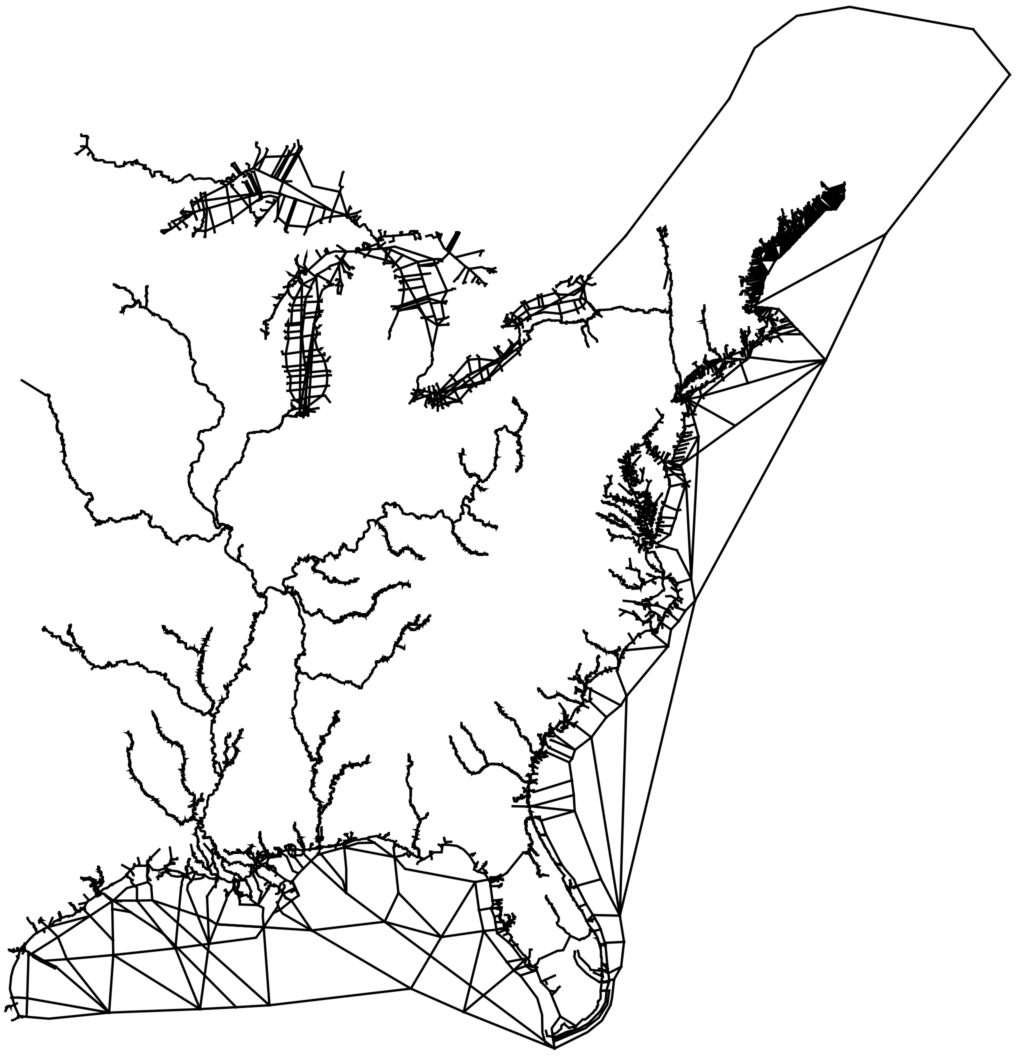

Fig. 3. The 1995 waterway network.

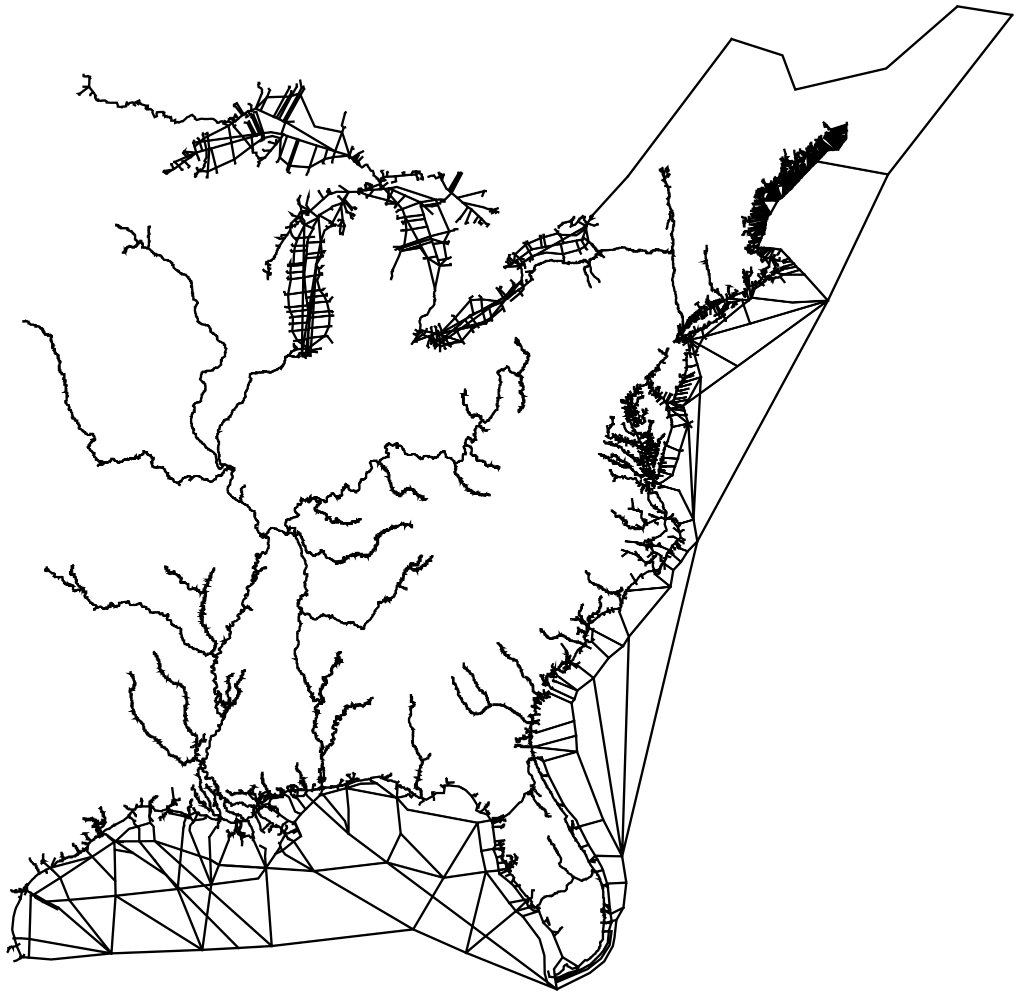

Fig. 4. The 1996 waterway network.

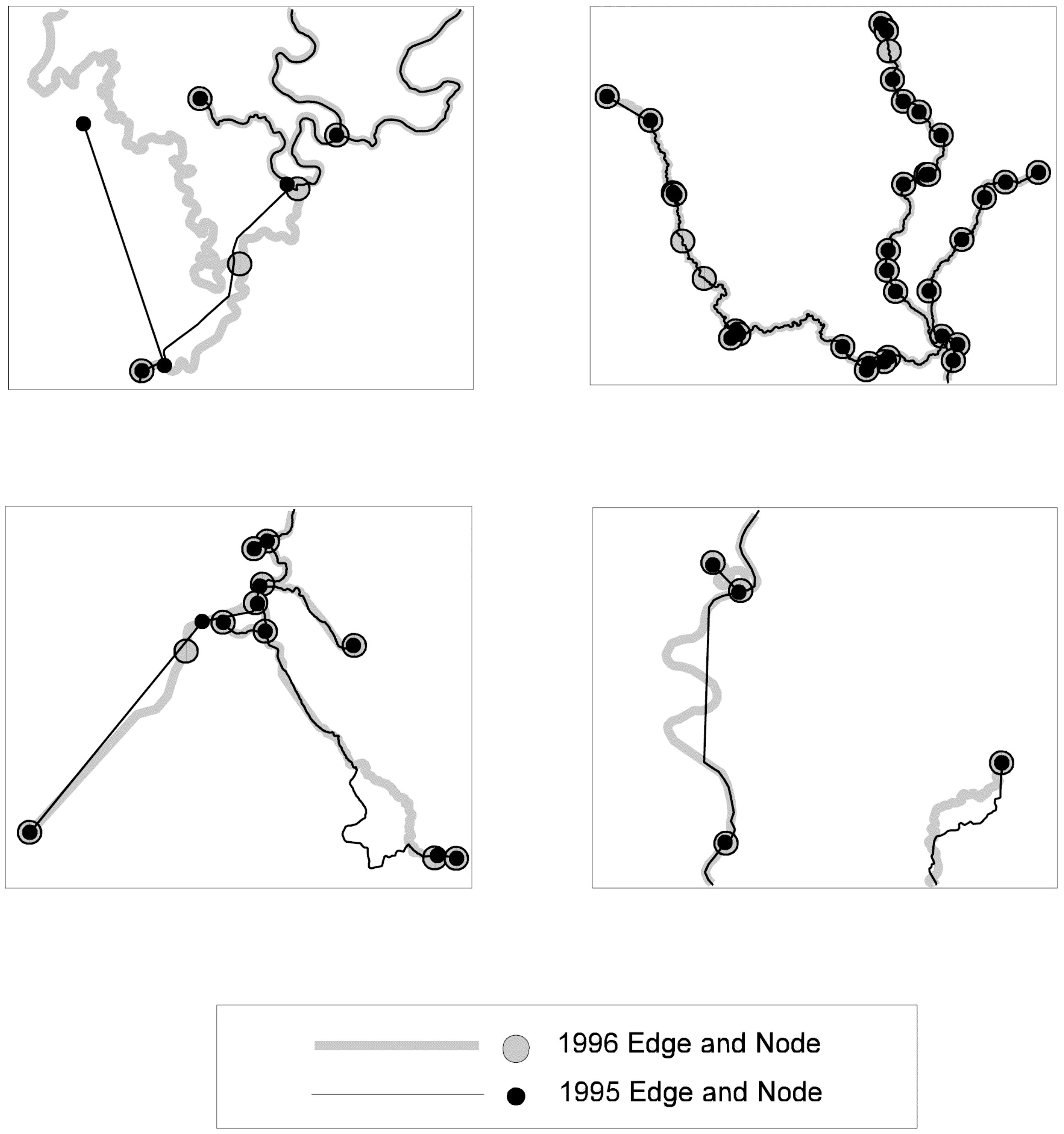

Fig. 5. Comparison of the 1995 and 1996 Networks

4.2. Bottom-up Matching

In spite of differences, these two waterway networks have a very similar node-link structure and have limited coordinate discrepancies, which make them suitable for the use of the bottom-up matching. To apply the bottom-up procedure, 10 km is used as the maximum distance criterion. Angle difference is not used in this particular matching because there are significant differences between the two versions of the networks in the geometry for waterway links, which results in large angle differences. In general, angle difference is a valuable piece of information for distinguishing nodes that are close to each other. To allow effective use of this information, yet to avoid mismatching at nodes with large angle differences, an ideal solution would be an adaptive procedure that is able to determine when the information should be turned on and when the information should be turned off. Such a procedure was not developed in this specific experiment.

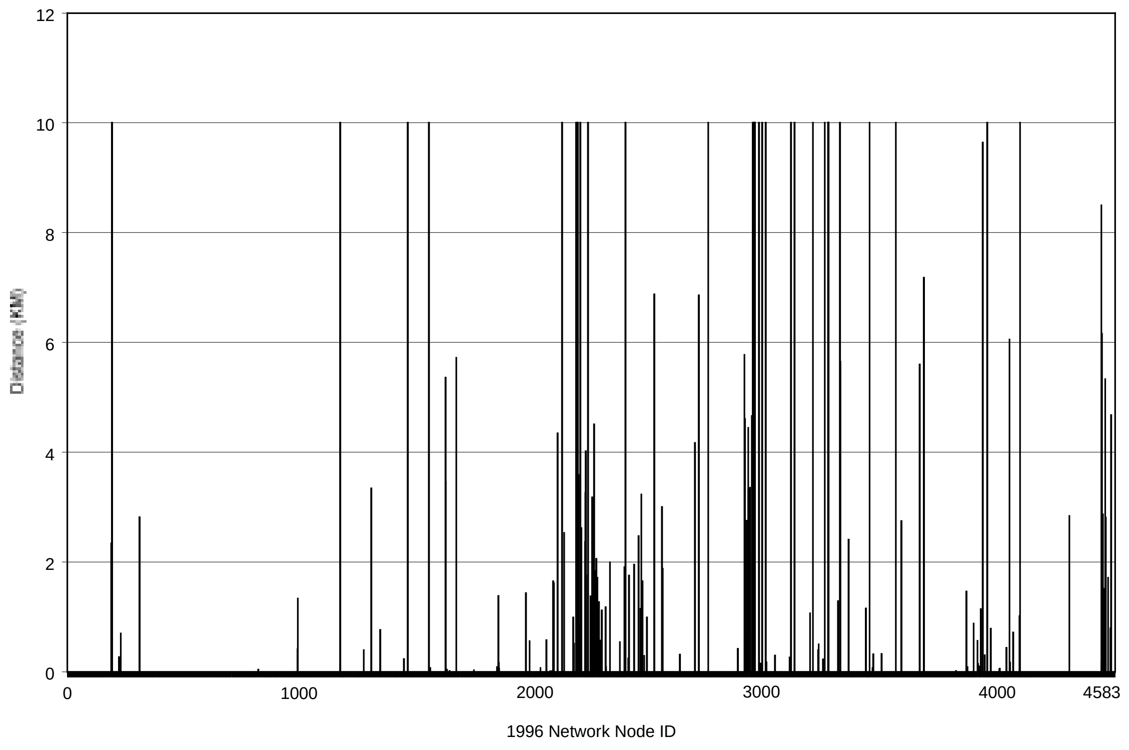

Fig. 6. Distances between nearest node pairs

Fig. 6 shows the distances between each of nearest node pairs. The vertical axis of Fig. 6 represents the distance. The horizontal axis represents the node IDs on the 1996 network. As Fig. 6 shows, a large number of nodes are perfectly matched (e.g., have 0 distance). Among the nodes that are not perfectly matched, most have the distance gap of less than 10 km. (As 10 km is used as the cutoff distance, distances greater than 10 km are shown as 10 km on the diagram.) Using the distance criterion, for all 4584 nodes on the 1996 network, 4551 nodes have a match on the 1995 network.

After node matching, edge correspondences are evaluated. This evaluation starts with segment matching, which generates matching measures for edges that have their end nodes matched. The computations of matching measures follow the general procedures outlined in sects. 3.4 and 3.5, but with some customization. In this specific application, each of the matching measures is transformed into a 0-6 point score, and the weighting factors α, β, and δ, as shown in equation 6, are set to 1.0. Finally the three scores are added together to get the total score for each of the matching pairs.

Table 1 illustrates how the matching measures are transformed into the 0-6 point scores. The purpose of this transformation is to have a discretized score so that each of the measures used for the edge matching can be easily assessed. As is shown in Table 1, different intervals were chosen for the values of thresholds used for the discretization. The purpose was to provide a higher resolution for measures where potential matches are more likely. As no evidence suggests that a particular matching measure needs to be weighted more or less than other measures, 1.0 was chosen for each of the weighting factors to give equal weight to the measures.

Table 1. Transformation of matching measures to 0-6 point scores.

Point	Angle Difference	Distance	Length
6	$d\varphi_{ij}^{e}/\Phi^{e} < 0.125$	$d_{ij}^{e} /\Delta^{e} < 0.125$	$L_i^{e}/L^{e} > 2.0$ and $L_j^{e}/L^{e} > 2.0$
5	$d\varphi_{ij}^{e}/\Phi^{e}$ >= 0.125 and $d\varphi_{ij}^{e}/\Phi^{e} < 0.25$	d_{ij}^{e} /Δ^{e} >= 0.125 and $d_{ij}^{e} /\Delta^{e} < 0.25$	$L_i^{e}/L^{e} > 2.0$ and $L_j^{e}/L^{e} > 1.0$ or $L_i^{e}/L^{e} > 1.0$ and $L_j^{e}/L^{e} > 2.0$
4	$d\varphi_{ij}^{e}/\Phi^{e}$ >= 0.25 and $d\varphi_{ij}^{e}/\Phi^{e} < 0.325$	d_{ij}^{e} /Δ^{e} >= 0.25 and $d_{ij}^{e} /\Delta^{e} < 0.325$	$L_i^{e}/L^{e} > 1.0$ and $L_j^{e}/L^{e} > 1.0$
3	$d\varphi_{ij}^{e}/\Phi^{e}$ >= 0.325 and $d\varphi_{ij}^{e}/\Phi^{e} < 0.5$	d_{ij}^{e} /Δ^{e} >= 0.325 and $d_{ij}^{e} /\Delta^{e} < 0.5$	($L_i^{e}/L^{e} < 1.0$ and $L_i^{e'} < L^{e}$) and ($L_i^{e}/L^{e} > 1.0$ or $L_j^{e}/L^{e} > 1.0$)
2	$d\varphi_{ij}^{e}/\Phi^{e}$ >= 0.5 and $d\varphi_{ij}^{e}/\Phi^{e} < 0.75$	d_{ij}^{e} /Δ^{e} >= 0.5 and $d_{ij}^{e} /\Delta^{e} < 0.75$	($L_i^{e}/L^{e} < 1.0$ and $L_i^{e'} < L^{e}$) and ($L_j^{e}/L^{e} < 1.0$ *and* $L_j^{e'} < L^{e}$)
1	$d\varphi_{ij}^{e}/\Phi^{e}$ >= 0.75 and $d\varphi_{ij}^{e}/\Phi^{e} < 1.0$	d_{ij}^{e} /Δ^{e} >= 0.75 and $d_{ij}^{e} /\Delta^{e} < 1.0$	*N/A*
0	$D\varphi_{ij}^{e}/\Phi^{e}$ <= 1.0	d_{ij}^{e} /Δ^{e} >= 1.0	(L_i^{e}/L^{e} <= 1.0 and $L_i^{e'} > L^{e}$) or (L_j^{e}/L^{e} <= 1.0 and $L_j^{e'} > L^{e}$)

[1]*Refer to sect. 3.5 for symbol representations in the table.*

When the total scores are computed for each of the edge pairs, those edge pairs that have a score less then 1 will be eliminated for a match. The matching result with these scores is that for all 5088 edges on the 1996 network, 4894 edges are matched to the 1995 network. A visual examination of these matched nodes and edges reveals no obvious mismatches.

4.3. Top-down Matching

After the bottom-up matching, 194 edges on the 1996 network are left without a match. Some of these edges may not have a match, but some may have a counterpart on the 1995 network, but their end nodes are not exactly matched. To allow matching of additional edges, the top-down matching procedure is now utilized.

For the top-down matching, only those edges that are unmatched during the bottom-up matching are considered for further matching. The edge mapping set generated from the bottom-up matching is passed to the top-down matching procedure so that those unmatched edges can be identified. In the top-down matching, each edge on the 1996 network is evaluated separately. When a 1996 edge is selected, unmatched edges on the

1995 network that are close and parallel to the selected edge on the 1996 network are identified. Each of the candidate pairs are then matched at the segment level. After segment matching, matching scores between each edge pair are computed, and used to determine the best match. For the 194 unmatched edges on the 1996 network, 58 edges then find their corresponding matches on the 1995 network.

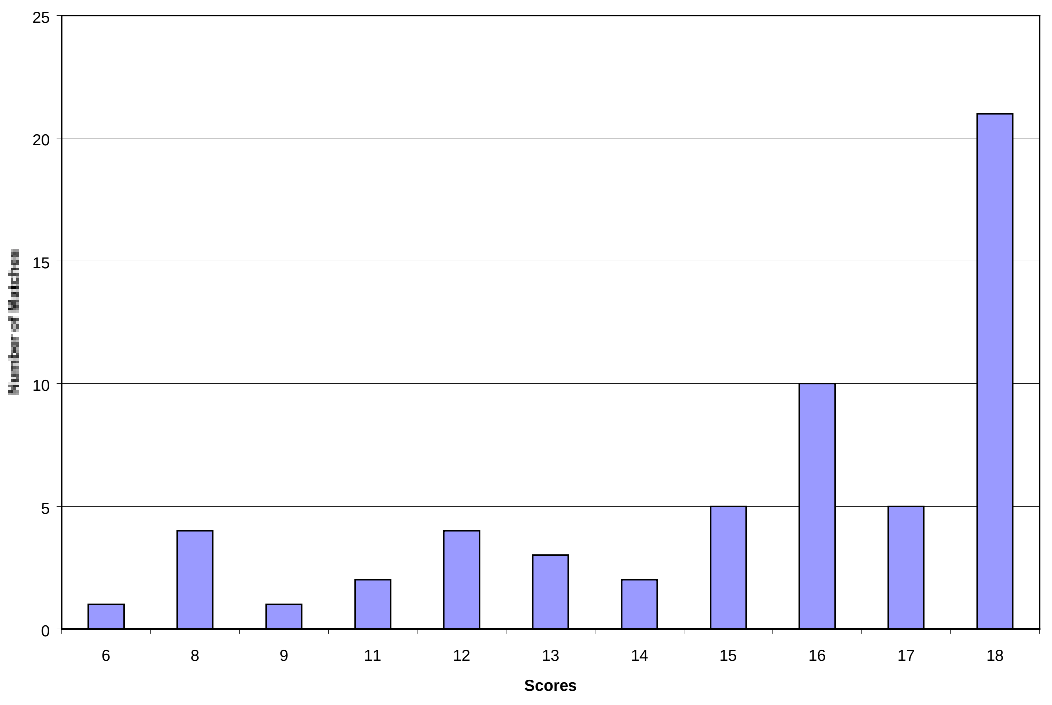

Fig. 7. Score distribution for matched edges

Figure 7 shows the frequency distribution of final matching scores for the 58 matched edges. As shown in Figure 7, 21 out of 58 edges are matched at the highest score (18), and the lowest score where a match has been found is 6, which indicates that all matches are fairly certain and unambiguous. Typical cases for the 136 edges that remain unmatched include (1) edges newly introduced on the 1996 network with no counterpart on the 1995 network; (2) edges whose geometry has changed significantly from the 1995 network to the 1996, making it difficult to determine whether they actually form an match; and (3) edges that are far apart and cannot be effectively matched with the use of the distance criterion.

5. CONCLUSIONS AND DISCUSSIONS

The experiment results show that a reasonable match rate can be achieved with the proposed matching algorithm. The combined use of top-down and bottom-up computations is effective. Because the waterway networks used in our experiment have a similar network structure, the bottom-up computation can quickly identify matches using node correspondences. In cases where node correspondences can not be effectively identified, the top-down computation is able to find additional matches. The experiment also demonstrates that segment matching provides leverage when edges are compared in detail. With segment matching, finer resolution can be achieved when edges shows subtle differences.

Currently, the program used to implement the proposed algorithm is able to read and write a ESRI's shape file format directly, which has proven to be very helpful. The data can be viewed and analyzed conveniently before the algorithm is applied. After matching computations, the results can be quickly evaluated. In addition, it is possible to take advantage of existing GIS functions for preprocessing and postprocessing the network data, which is necessary in many real-world applications.

The computational performance of the proposed algorithm appears satisfactory based upon the current experiment. The entire matching process for the waterway networks took about 2 seconds on an Intel Pentium II processor. These 2 seconds included the time for reading the input files and writing the output files and the time spent on the actual matching computations. This computational effectiveness has a great deal to do with the use of a grid system for effectively indexing network nodes and edges. This grid system is very similar to the one introduced by Franklin et al. (1994). The major usefulness of this grid system is in searching for matching candidates during node and edge matching. Take node matching as an example, if indexes are not used, $m*n$ calculations are necessary to find the nearest matches for two sets of nodes that have the node numbers m and n, respectively. If m and n are assumed to be the same size, the computational complexity is on the order of n^2 or $O(n^2)$. When grid indexes are utilized, however, the computation number can be cut to $9*m*n/g$, where g is the total number of cells. The factor of 9 is based on the assumption that the maximum search range is equal to or less than the edge length of a cell, so a maximum of nine cells will be searched in each of the searches. As the size of g is adjustable and can be made proportional to the size of n, the computational complexity is now actually on the order of n or $O(n)$.

With the use of the spatial indexes, as described above, an increase in network size will not significantly slow down the matching computation. Nevertheless, for extremely large networks such as those for cities like Los Angeles or New York, they may overwhelm computer memories if they are loaded at once. In these cases, special care must be taken. For instance, these large networks can first be divided into several smaller networks; these smaller networks are then matched separately. Another factor that has to be considered in the analysis of computational performance is the similarity and difference of the participating networks. In general, networks with similar geometrical and topological characteristics can be more effectively matched. However, further study of this problem will be necessary in order to arrive at more substantive conclusions.

In spite of the preliminary success of this algorithm, additional improvement or extension of the algorithm is necessary to make it more effective and reliable. In the short term, it appears that improvements should be made in three areas. First, the algorithm uses a segment-matching procedure that is based on distance and angle differences to track corresponding segments. The drawback of this procedure is that when network edges are distorted significantly, segment mappings derived from this procedure cannot faithfully reflect actual segment correspondences. If this occurs, matching measures derived from these segment mappings are not accurate, and node positions represented by the segment mappings cannot be determined precisely. Filin and Doytsher (1998) studied the problem and suggested that edges or curves should be divided into separate smooth sub-edges or sub-curves, and then mappings should be established between these sub-curves. Incorporation of this matching procedure into the current algorithm is likely to improve the accuracy of segment matching.

Second, the current research mainly deals with spatial charateristics of the participating networks. Yet, aspatial properties of these networks such as road names or mileposts also provide critical information for network matching. To make effective use of the aspatial properties, integration of aspatial matching with spatial matching represents an important strategy. In general, attributes of networks can be first evaluated to determine matches or potential matches (e.g., they have the same road name or the same street address range). When uncertainties arise, matching candidates can be handed over for spatial matching. In this case, the top-down procedure can be utilized to verify whether the matching candidates can be spatially matched. Although spatial matching and aspatial matching can be implemented independently, an integrated procedure will likely generate better results.

The third area for improvement is in higher-level matching (e.g., matching at the cluster or sub-network level). The matching strategy proposed in this paper is focused on local matches, using Euclidean distance in searching for and matching individual matching pairs. In situations where counterparts are not immediate neighbors, correct matching will be difficult. If matching decisions can be evaluated at a higher level (e.g., clusters or sub-networks evaluated as a whole before individual matches are determined), there will be a better chance for higher matching accuracy. In this case, the relational matching approach adopted by Walter and Fritsch (1998) represents a good solution.

Acknowledgments: Sincere thanks are due to the three anonymous reviewers for their valuable comments.

REFERENCES

Brown, J., Rao, A., and Baran, J. (1995) Automated GIS conflation: coverage update problems and solutions. *Proceedings of Geographic Information Systems for Transportation Symposium (GIS-T),* pp. 220–229, Sparks, Nevada, USA.

Fonseca, L. M. G. and Manjunath B. S. (1996) Registration techniques for multisensor remotely sensed imagery. *Photogrammetric Engineering and Remote Sensing*, **62**, 1049–1056.

Filin, S. and Doytsher, Y. (1998) Conflation—the linear matching issue. *Proceedings of 1998 Annual Convention and Exhibition*, American Congress on Surveying and Mapping, Baltimore, Maryland, USA.

Franklin, W. R., Sivaswami, V., Sun, D., Kankanhalli, M., and Narayanaswami, C. (1994) Calculating the area of overlaid polygons without constructing the overlay. *Cartography and Geographic Information Systems*, **21**, 81–89.

Gabay, Y. and Doytsher, Y. (1994) Automatic adjustment of line maps. *Proceedings of the GIS/LIS'94 Annual Convention*, pp. 333–341, Arizona, Phoenix, USA.

Novak, K. (1992) Rectification of digital imagery. *Photogrammetric Engineering and Remote Sensing*, **58**, 339–344.

Nystuen, J. D., Frank, A. I. and Frank, L., Jr. (1997) Assessing topological similarity of spatial networks. *Proceedings of International Conference on Interoperating Geographic Information Systems*, Santa Barbara, California, USA.

Rosen, B. and Saalfeld, A. (1985) Match criteria for automatic alignment. *Proceedings of Auto-Carto VII*, pp. 1–20, American Congress on Surveying and Mapping and American Society for Photogrammetry and Remote Sensing.

Saalfeld, A. (1988) Automated map compilation. *International Journal of Geographical Information Systems*, **2**, 217–218.

Shapiro, L. G. (1980) A structural model of shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **2**, 111–126.

Shipiro, L. G. and Haralick, R. M. (1981) Structural description and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5**, 504–519.

Stilla, U. (1995) Map-aided structural analysis of aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, **50**, 3–10.

Ventura, A. D., Rampini, A., and Schettini, R. (1990) Image registration by recognition of corresponding structures. *IEEE Transactions on Geoscience and Remote Sensing*, **28**, 305–314.

Walter, V. and Fritsch, D. (1999) Matching spatial data sets: a statistical approach. *International Journal of Geographic Information Science*, **13**, 445–473.

Wang, Y. (1998) Principles and applications of structural image matching. *ISPRS Journal of Photogrammetry and Remote Sensing*, **53**, 154–165.