Data Mining for Selective Visualization of Large Spatial Datasets

Shashi Shekhar [*], Chang-Tien Lu [†], Pusheng Zhang, Rulin Liu
Computer Science & Engineering Department
University of Minnesota
Email: [shekhar, ctlu, pusheng, rliu]@cs.umn.edu

Abstract

Data mining is the process of extracting implicit, valuable, and interesting information from large sets of data. Visualization is the process of visually exploring data for pattern and trend analysis, and it is a common method of browsing spatial datasets to look for patterns. However, the growing volume of spatial datasets make it difficult for humans to browse such datasets in their entirety, and data mining algorithms are needed to filter out large uninteresting parts of spatial datasets. We construct a web-based visualization software package for observing the summarization of spatial patterns and temporal trends. We also present data mining algorithms for filtering out vast parts of datasets for spatial outlier patterns. The algorithms were implemented and tested with a real-world set of Minneapolis-St. Paul(Twin Cities) traffic data.

1. Introduction

Data mining is a process to extract implicit, nontrivial, previously unknown and potentially useful information(such as knowledge rules, constraints, regularities) from data in databases [22, 10]. The explosive growth in data and databases used in business management, government administration, and scientific data analysis has created a need for tools that can automatically transform the processed data into useful information and knowledge. Data mining allows organizations and companies to extract useful information from the vast amount of data they have gathered, thus helping them make more effective decisions.

Spatial data mining [18, 19, 26, 25, 5], a subfield of data mining, is concerned with the discovery of interesting and useful but implicit knowledge in spatial databases. With the huge amount of spatial data obtained from satellite images, medical images, and geographical information systems (GIS), it is a non-trivial task for humans to explore spatial data in detail. Spatial datasets and patterns are abundant in many application domains related to NASA, the Environmental Protection Agency, the National Institute of Standards and Technology, and the Department of Transportation. A key goal of spatial data mining is to partially automate knowledge discovery, i.e., search for "nuggets" of information embedded in very large quantities of spatial data. Challenges in spatial data mining arise from the following issues. First, classical data mining is designed to process numbers and categories. In contrast, spatial data is more complex and includes extended objects such as points, lines, and polygons. Second, classical data mining works with explicit inputs, whereas spatial predicates and attributes are often implicit. Third, classical data mining treats each input independently of other inputs, while spatial patterns often exhibit continuity and high autocorrelation among nearby features.

In this paper, we construct a visualization software package for observing the summarization of spatial patterns and temporal trends. In the underlying data structure, we model the spatial data as a spatial data warehouse to facilitate the query engine for the on-line analytical processing used in the visualization software. We present algorithms for filtering out vast parts of datasets for spatial outlier patterns. The algorithms were implemented and tested with a real-world traffic dataset from Minneapolis-St. Paul(Twin Cities).

Application Domain: The Traffic Management Center(TMC) [28] of the Minnesota Department of Transportation(MNDOT) has a database to archive sensor network measurements from the freeway system in the Twin Cities metropolitan area. The sensor network includes about nine hundred stations, each of which contains one to four loop detectors, depending on the number of lanes. Sensors embedded in the freeways and interstate monitor the occupancy and volume of traffic on the road. The volume is measured as the number of vehicle passing through this sta-

[*] This work is partially supported by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014,the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

[†] Chang-Tien Lu is currently with the Department of Computer Science, Northern Virginia Center, Virginia Tech. Email:ctlu@cs.vt.edu

tion during a 5-minute time interval. Occupancy is the percentage of time during which a station is occupied by a vehicle. At regular intervals, this information is sent to the Traffic Management Center for operational purposes, e.g., ramp meter control, as well as research on traffic modeling and experiments. Figure 1 shows a map of the stations on highways within the Twin Cities metropolitan area, where each polygon represents one station. Downtown Minneapolis is located at the intersection of I-94, I-394, and I-35W, and downtown St. Paul is located at the intersection of I-35E and I-94. With the huge amount of traffic data stored in its traffic database, the Minnesota Department of Transportation needs a high-performance traffic data mining and visualization system that extracts patterns and rules from historical data to support decision making.

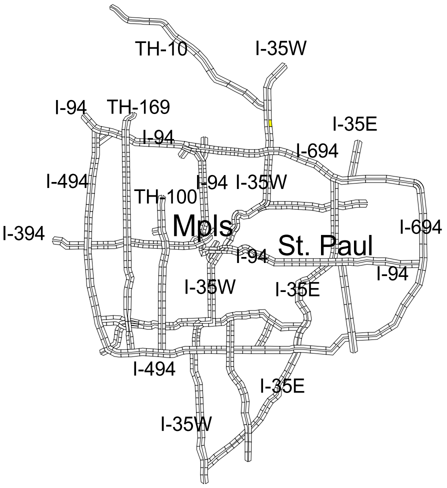


Figure 1. Freeway Map of Twin Cities

Related Work: Fayyad et al. [9, 8] recognized the primary importance of user interaction and graphical representation for data as part of the whole knowledge discovery process to make patterns more understandable by users. Many powerful visualization tools have been developed in databases and data mining. Visualization tools, such as VisDB [17], XGobi [2] and XmdvTool [30], provides a set of predefined visualizations, e.g., histogram, scatterplots, and parallel coordinates, to explore multi-dimensional datasets. These view are augmented with brushing and zooming techniques, however, users can not interactively construct and refine a wide range of displays to suit an analysis task for large spatial datasets.

There have been many studies [8] which focus on the visualization of spatial data, though none of them address the constraints of large data sets on visualization. The growing volume of spatial datasets make it difficult for humans to browse the entire datasets. The effective visualization is needed to explore large spatial datasets. This paper uses data mining, data warehousing and spatial visualization techniques to discover implicit patterns and relationships embedded in the data.

Our Contribution: The major contributions of this paper are as follows: we propose and implement the *CubeView* visualization system, which allow users to quickly gain insight into the data by data exploration using general data cube operations. This system is built on the concept of a spatial data warehouse to support data mining and data visualization. In addition, we provide efficient and scalable spatial outlier detection algorithms to help identify abnormal patterns in data visualization so that the volume of data needed to be analyzed by data analysts can be reduce dramatically.

Outline: The rest of the paper is organized as follows. Section 2 introduces the concepts of spatial data warehouse and spatial data mining. The architecture of the *CubeView* system is illustrated in Section 3. In Section 4, We present algorithms for filtering out vast parts of datasets for spatial outlier patterns to aid data visualization. Finally, we summarize our work in Section 5.

Scope: In this paper, we focus on the development of a spatial visualization system and the design of efficient data mining algorithms to aid data visualization. For data mining techniques, we use outlier detection as an illustrative example. Other data mining techniques, e.g., classification, clustering, association rule discovery, and trend analysis, will be addressed in future work.

2. Basic Concepts

In this section, we introduce the basic concepts of a data warehouse, especially a spatial data warehouse, and spatial data mining techniques.

2.1. Spatial Data Warehouse

A data warehouse(DW) [3, 4, 16] is a repository of subject-oriented, integrated, and non-volatile information whose aim is to support knowledge workers(executives, managers, analysts) to make better and faster decisions. Data warehouses contain large amounts of information collected from a variety of independent sources and are often maintained separately from the operational databases. Traditionally, operational databases are optimized for on-line transaction processing (OLTP), where consistency and recoverability are critical. Transactions are typically small and access a small number of individual records based on the primary key. Operational databases maintain current state information. In contrast, data warehouses maintain historical, summarized, and consolidated information, and are designed for on-line analytical processing (OLAP) [6]. The data in the warehouse are often modeled as a multidimensional space to facilitate OLAP in the query engines, where queries typically aggregate data across many dimensions in order to detect trends and anomalies [21]. The subject of analysis in a multidimensional data model is a set of numeric measures. Each numeric measures is determined by a set of dimensions. In a census data warehouse, for example, the measure is population, and the dimensions of interest include age group, ethnicity, income type, time (year), and location(census tract). Given N dimensions, the measures can be aggregated in 2^N different ways. The SQL aggregate functions and the group-by operators only produce one out of 2^N aggregates at a time. A data cube [12] operator computes all 2^N aggregates in one shot.

Spatial data warehouses contain geographic data, e.g., satellite images, aerial photographs [23, 13, 20, 24], in addition to non-spatial data. Examples of spatial data-warehouses include the US Census dataset [11], Earth Observation System archives of satellite imagery [29], Sequoia 2000 [27], and highway traffic measurement archives. The research on spatial data warehouses has focused on case-studies [7, 20] and on the per-dimension concept hierarchy [13]. A major difference between conventional and spatial data warehouses lies in the visualization of the results. Conventional data warehouse OLAP results are often shown as summary tables or spread sheets of text and numbers, whereas in the case of spatial data warehouses the results may be albums of maps. It is not trivial to convert the alpha-numeric output of a data cube on spatial data warehouses into an organized collection of maps.

In a traffic data warehouse, for example, the measures are volume and occupancy, and the dimensions are *time* and *space*. Dimensions are hierarchical by nature. For example, the *time* dimension can be grouped into “Hour”,“Date”,“Month”,“Week”,“Season”, or “Year”, which form a lattice structure indicating a partial order for the dimension. Similarly, the *Space* dimension can be grouped into “Station”,“County”,“Freeway”, or “Region”. Given the dimensions and hierarchy, the measures can be aggregated in different ways. The SQL aggregate functions and the group-by operator only produce one out of all possible aggregates at a time. A data cube is an aggregate operator which computes all possible aggregates in one shot. The CUBE operator generalizes the histogram, cross-tabulation, roll-up, drill-down, and sub-total constructs. It is the N-dimensional generalization of simple aggregate functions.

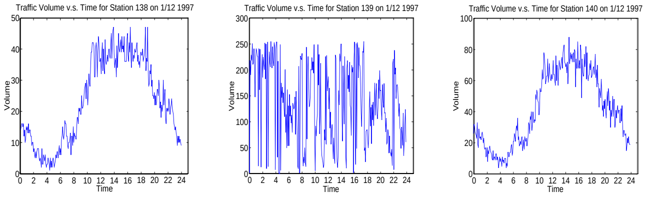
2.2. Spatial Data Mining

Spatial data mining is a process of discovering interesting and useful but implicit spatial patterns. With the huge amount of spatial data obtained from satellite images, medical images, GIS, etc., it is a non-trivial task for humans to explore spatial data in detail. Spatial datasets and patterns are abundant in many application domains related to NASA, EPA, NIST, and USDOT. A key goal of spatial data mining is to partially ‘automate’ knowledge discovery, i.e. search for “nuggets” of information embedded in very large quantities of spatial data. Efficient tools for extracting information from spatial data sets can be of importance to organizations which own, generate, and manage large geo-spatial data sets.

Spatial Outliers Detection

Outliers have been informally defined as observations which appear to be inconsistent with the remainder of that set of data [1], or which deviate so much from other observations so as to arouse suspicions that they were generated by a different mechanism [15]. A spatial outlier is a spatially referenced object whose non-spatial attribute values are significantly different from those of other spatially referenced objects in its spatial neighborhood. Informally, a spatial outlier is a local instability (in values of non-spatial attributes) or a spatially referenced object whose non-spatial attributes are extreme relative to its neighbors, even though they may not be significantly different from the entire population.

In the traffic data set, each station is a spatially referenced object with spatial attributes(e.g., location) and non-spatial attributes(e.g., measurements). Figure 2 shows an example of traffic flow outliers. The X-axis is the time in a day, and the Y-axis is the volume of a station. As shown in Figure 2, the abnormal station(Station 139) had the volume values significantly inconsistent with the volume values of its neighboring stations 138 and 140. With the help of the visualization, we can easily identify the outliers.



(a) Station 138 (b) Station 139 (c) Station 140

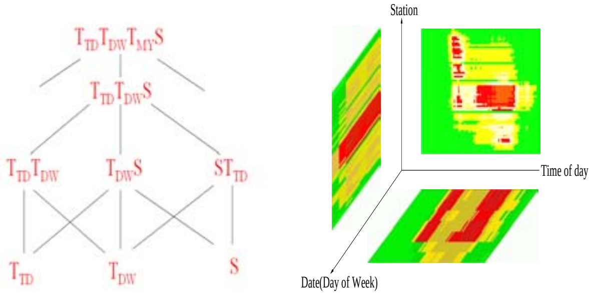
Figure 2. Outlier Station 139 and Its Neighbor Stations on 1/12/1997

3. System Architecture for Data Visualization

Most data mining tools can process data stored in conventional databases. The construction of data warehouses is not required. However, with its support for convenient OLAP multidimensional grouping, aggregation, and queries, data warehousing is helpful to data mining tools. Thus a visualization system supporting data warehousing will help users gain insight and enhance the understanding of the large data.

3.1. *CubeView* Visualization System

We construct the *CubeView* visualization system and use Twin Cities traffic data as an application example to illustrate the design concepts. As shown in Figure 3(a), the basic structure is a data cube, where T_{TD} represents the time of day, T_{DW} represents the day of the week, T_{MY} represents the month of the year and S represents the station. Each node is a visualization style. For example, the S node represents the traffic volume of each station at all times. The ST_{TD} node represents the daily traffic volume of each station. The $T_{TD}T_{DW}S$ node represents the traffic volume at each station at different time of different days. The information is generated as a video in the *CubeView* system. Because of the nature of data cubes, *CubeView* can analyze any traffic data. The software only requires the space and time for each measure, such as volume, occupancy, and speed. Speed can be derived from volume and occupancy. These requirements are very simple and most highway monitoring systems should be able to satisfy them. Figure 3(b) gives examples of nodes in the *CubeView* system. The three dimensions are Station (S), Time of Day (T_{TD}), and Day of Week (T_{DW}), and the three pictures correspond to the three nodes $T_{TD}T_{DW}$, $T_{DW}S$, and ST_{TD}.



(a) Cube View For Traffic Data

(b) Example of Cube View

Figure 3. CubeView Visualization System

We use Figure 4 to illustrate data flows and the required modules of the *CubeView* system. The basic map and raw data are cleaned, transformed, and loaded into the data warehouse module, which provides the multidimensional views and the OLAP operations for data visualization, as well as a variety of data mining analysis tools, e.g, classification, clustering, outlier detection. The discovered patterns or rules are then visually displayed as maps or charts for further interpretation.

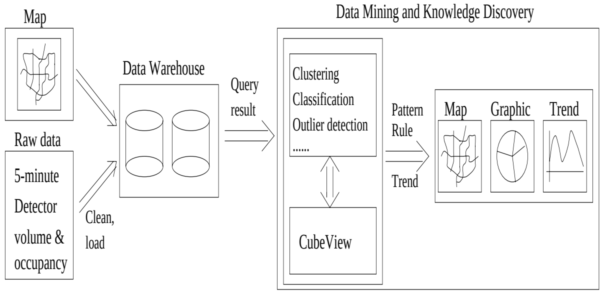


Figure 4. Data-flow and Main Modules in the *CubeView* System

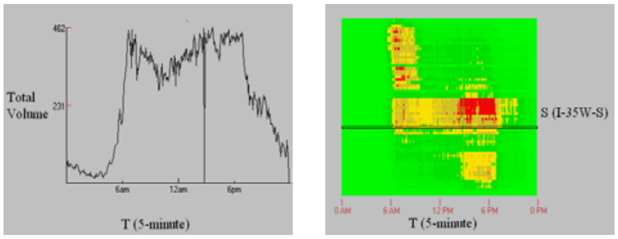
The *CubeView* system provides several essential visualization utilities to support traffic pattern analysis. These utilities include traffic video visualization, traffic volume map, attribute visualization, video map comparison, and traffic flow visualization.

One-Dimensional Visualization T_{TD}

Visualization of traffic attributes (e.g., volume, occupancy) as a function of time allows identification of outliers or the identification of groups of stations with similar behavior. Figure 5 (a) shows the total traffic volume of station 15, located at the intersection of I-35W and Highway 61, on Monday, January 6, 1997. The X-axis is the time interval; the Y-axis is the measure of volume. As can be seen, the average 5-minute traffic volume was higher than 230 from 6:00AM to 6:00PM. (Notice the abrupt drop of the traffic volume from 462 to 0 at 2:09PM, which is caused by the reboot of traffic data recorder).

Attribute Map ST_{TD} (Two-Dimensional Visualization)

Summary traffic maps of the metropolitan area for a variety of aggregate information includes highway-level traffic volume and occupancy. Figure 5(b) shows the summary volume map for highway I-35W South Bound on Monday, January 6, 1997. The X-axis is the 5-minute time slot for the whole day and the Y-axis is the label of the stations installed on the highway, starting from the top in the north end to the bottom in the south end. In this figure, we can easily



(a) One-Dimensional Attribute Visualization T_{TD}

(b) Attribute Map ST_{TD}

Figure 5. Traffic Data Visualization

observe three traffic patterns: 1) the morning rush hour (6-9AM) in the northern part of the highway; 2) the evening rush hour (3-6AM) in the southern part of the highway; and 3) morning to evening (6:00AM to 6PM) busy traffic volume for stations located within downtown to the junction with Highway 61, that is, the route from downtown Minneapolis to the Minneapolis/Saint Paul airport.

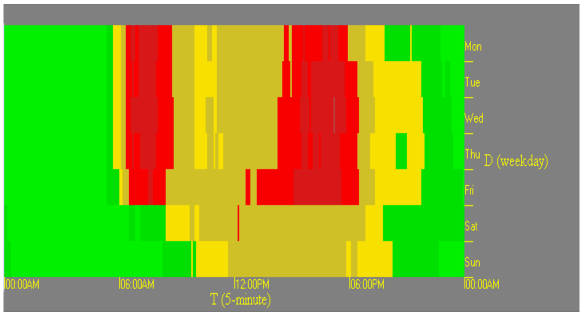


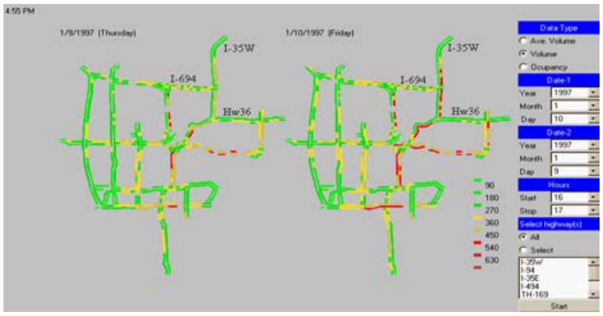
Figure 6. Attribute Matrix $T_{TD}T_{DW}$

Attribute Matrix $T_{TD}T_{DW}$ (Two-Dimensional Visualization)

The visualization of two different time dimensions, such as the time of day and the day of week, can help to identify temporal patterns. Figure 6 shows an example of visualization on two dimensions, T_{TD} and T_{DW}. The X-axis is Time of Day; the Y-axis is Day of Week. The displayed value is the average traffic volume for all stations in January 1997. Some interesting general trends can be observed in this figure, e.g., rush hour starts earlier on Friday than other weekdays.

Multi-Dimensional Visualization $T_{TD}T_{DW}S$ (Animated Map)

CubeView system provides a comparison utility that allows users to simultaneously observe the traffic flow on two different dates. Figure 7 shows the traffic flow on Thursday, January 9, 1997 and Friday, January 10, 1997 from 4:55PM to 5:00PM. The figure indicates that the busy traffic flow started earlier on Friday for Highway I-35W North Bound between the downtown area and its intersection with Highway I-694. An important application of this comparison utility is that users can compare the effect of applying ramp meter control on the same weekday.



Data Type
○ Ave. Volume
● Volume
○ Occupancy

Date-1
Year 1997
Month 1
Day 10

Date-2
Year 1997
Month 1
Day 9

Hours
Start 16
Stop 17

Select highway(s)
● All
○ Select

I-35W
I-94
I-35E
I-494
TH-169

Figure 7. Comparison of traffic video of two different days

Visualization can be used as an convenient way to discover outliers, however, inspecting thousands of pictures is still a challenging task. For example, there are 18 highways in our system. Each highway runs in 2 directions. If one picture is generated per day/highway/direction. This means 2*18=36 pictures are generated per day. To inspect one month's data, the traffic manager need to look at about 1000 pictures, which is tedious and error-prone. The growing volume of datasets make it difficult for humans to browse the entire datasets. Thus we need data mining algorithms to discover suspected outliers so that they can be highlighted by the *CubeView* for further investigations.

4 Data Mining Algorithms for Visualization

Compared with data visualization, which involves much human effort, data mining techniques are automated data analysis tools. By applying these automatic data mining techniques, the volume of data needed to be observed can be reduced dramatically. For example, if users are interested in observing all the spatial outlier stations for the past 10 years, outlier detection algorithms can first be applied to detect days with outliers, and visualization of data for selected days are displayed for further analysis. Due to the large volume(giga bytes/day) traffic data, efficient algorithms to detect spatial outliers are needed to support such filtering.

In this section we use spatial outlier detection as an illustrative example for data mining techniques. We formally define the spatial outlier detection problem and present algorithms to automate the detection process. The algorithms were tested with the months traffic data from Twin Cities.

4.1 Problem Definition

Given a spatial graph $G = \{S, E\}$, where S is a spatial framework consisting of locations $s_1, s_2, \ldots, s_n$ and E is a collection of edges between locations in S; an attribute function f over S; a neighborhood relationship R; a confidence level threshold θ; an aggregate function f_{aggr}: $R^N \rightarrow a\,set\,of\,real\,numbers$ to summarize values of attribute f over a neighborhood relationship $R^N \subseteq R$. we build a model and construct statistical tests to find spatial outliers based on a spatial graph G according to the given confidence level threshold θ. The majors objectives are correctness and efficiency, i.e., the attribute values of outliers identified should be significantly different from those of their neighborhood and the algorithm should minimize the computation time. There are some constrains applied. First, attribute values have a normal distribution. Second, the size of the data set is much larger than main memory size. Third, the range of attribute function f is the set of real numbers.

The formulation shows two subtasks in this spatial outlier detection problem: (a) the design of a statistical model M and a test for spatial outliers (b) the design of an efficient computation method to estimate parameters of the test, test whether a specific spatial location is an outlier, and test whether spatial locations on a given path are outliers.

4.2 Choice of Spatial Statistic

For spatial statistics, several parameters should be predetermined before running the spatial outlier test. First, the neighborhood can be selected based on a fixed cardinality or a fixed graph distance or a fixed Euclidean distance. Second, the choice of neighborhood aggregate function can be mean, variance, or auto-correlation. Third, the choice for comparing a location with its neighbors can be either just a number or a vector of attribute values. Finally, the statistic for the base distribution can be selected from various choices such as normal distribution and chi-square distribution.

The statistic we used is $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$, where $f(x)$ is the attribute value for a data record x, $N(x)$ is the fixed cardinality set of neighbors of x, and $E_{y \in N(x)}(f(y))$ is the average attribute value for neighbors of x. Statistic $S(x)$ denotes the difference of the attribute value of each data object x and the average attribute value of $x's$ neighbors.

The test for detecting an outlier can be described as $|\frac{S(x)-\mu_s}{\sigma_s}| > \theta$. For each data object x with an attribute value $f(x)$, the $S(x)$ is the difference of the attribute value of data object x and the average attribute value of its neighbors; μ_s is the mean value of all $S(x)$, and σ_s is the standard deviation of all $S(x)$. The choice of θ depends on the specified confidence interval. For example, a confidence interval of 95 percent will lead to $\theta \approx 2$.

4.3 Computation of Test Parameters

We now present an I/O efficient algorithm to calculate the test parameters, e.g., mean and standard deviation for the statistics, as shown in Algorithm 1. The computed mean and standard deviation can then be used to detect the outlier in the incoming data set.

Given an attribute data set V and the connectivity graph G, the Test Parameters Computation(TPC) algorithm first retrieves the neighbor nodes from G for each data object x. It then computes the difference of the attribute value of x and the average of the attribute values of $x's$ neighbor nodes. These different values are then stored as a set in the AvgDist_Set. Finally, the AvgDist_Set is used to get the distribution value μ_s and σ_s. Note that the data objects are processed on a page basis to reduce redundant I/O.

Algorithm 1. Test Parameters Computation(TPC) Algorithm

```
Input: $S$ is the attribute space;
    $D$ is the attribute data set in $S$;
    $F$ is the distance function in $S$;
    $ND$ is the depth of neighbor;
    $G = (D, E)$ is the spatial graph;

Output: ($\mu_s$,$\sigma_s$).
for(i=1;i $\leq$ $|D|$ ;i++){
    $O_i$=Get_One_Object(i,D); /* Select each object from $D$ */
    NNS=Find_Neighbor_Nodes_Set($O_i$,$ND$,$G$);
   /* Find neighbor nodes of $O_i$ from $G$ */
    Accum_Dist=0;
    for(j=1;j$\leq$ $|$NSS$|$;j++){
        $O_k$=Get_One_Object(j,NNS); /* Select each object */
        Accum_Dist += $F(O_i, O_k, S)$
    }
    Avg_Dist = Accum_Dist / $|$NNS$|$;
    Add_Element(AvgDist_Set,i); /* Add the element */
}
$\mu_s$ = Get_Mean(AvgDist_Set); /* Compute Mean */
$\sigma_s$ = Get_Standard_Dev(AvgDist_Set);
return ($\mu_s$,$\sigma_s$).
```

4.4 Computation of Test Results

The neighborhood aggregate statistics value, e.g., mean and standard deviation, computed in the TPC algorithm can be used to verify the outliers in an incoming data set. The two verification procedures are Route Outlier Detection(ROD) and Random Node Verification(RNV). The ROD procedure detects the spatial outliers from a user specified route, as shown in Algorithm 2. The RNV procedure checks the outliers from a set of randomly generated nodes. Given route RN in the data set D with graph structure G, the *ROD* algorithm first retrieves the neighboring nodes from G for each data object x in the route RN, then it computes the difference $S(x)$ between the attribute value of x and the average of attribute values of $x's$ neighboring nodes. Each $S(x)$ can then be tested using the spatial outlier detection test $|\frac{S(x)-\mu_s}{\sigma_s}| > \theta$. The θ is predetermined by the given confidence interval. The steps to detect outliers in both ROD and RNV are similar, except that the RNV has no shared data access needs across tests for different nodes. The I/O operations for Find_Neighbor_Nodes_Set() in different iterations are independent of each other in RNV. We note that the operation Find_Neighbor_Nodes_Set() is executed once in each iteration and dominates the I/O cost of the entire algorithm. The storage of the data set should support the I/O efficient computation of this operation. The detailed discussions about I/O efficient computations, the choices for storage structure, and experimental comparisons are available at [26].

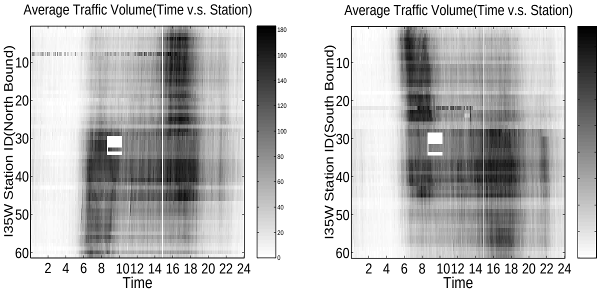
Algorithm 2. Route Outlier Detection(ROD) Algorithm

```
Input: $S$ is the attribute space;
    $D$ is the attribute data set in $S$;
    $F$ is the distance function in $S$;
    $ND$ is the depth of neighbor;
    $G = (D, E)$ is the spatial graph;
    $CI$ is the confidence interval;
    ($\mu_s$,$\sigma_s$) are mean and standard deviation calculated in TPC;
    $RN$ is the set of node in a route;

Output: Outlier_Set.
for(i=1;i $\leq$ $|RN|$ ;i++){
    $O_i$=Get_One_Object(i,D); /* Select each object from $D$ */
    NNS=Find_Neighbor_Nodes_Set($O_i$,$ND$,$G$);
   /* Find neighbor nodes of $O_i$ from $G$ */
    $Accum\_Dist$=0;
    for(j=1;j$\leq$ $|$NSS$|$;j++){
        $O_k$=Get_One_Object(j,NNS); /* Select each object */
        Accum_Dist += $F(O_i, O_k, S)$
    }
    AvgDist = Accum_Dist/$|$NNS$|$;
    $T_{value}$=$\frac{AvgDist-\mu_s}{\sigma_s}$
    /*Check the normal distribution table */
    if( Check_Normal_Table($T_{value}$,$CI$)== True){
        Add_Element(Outlier_Set,i); /* Add the element */
    }
}
return Outlier_Set.
```

4.5 Results

We tested the effectiveness of our algorithms on the Twin Cities traffic data set and detected numerous outliers. Our algorithms automated the spatial outlier detection process from large traffic data, and the outliers detected are highlighted using the *CubeView* for further investigation by traffic managers. Figure 8 shows an example of traffic flow outliers detected. Figures 8(a) and (b) are the ST_{TD} attribute maps for I-35W North Bound and South Bound, respectively, on January 21, 1997. The X-axis is a 5-minute time slot for the whole day and the Y-axis is the label of the stations installed on the highway, starting from 1 on the north end to 61 on the south end. The abnormal white line at 2:45pm and the white rectangle from 8:20am to 10:00am on the X-axis and between stations 29 to 34 on the Y-axis can be easily observed from both (a) and (b). The white line at 2:45pm is an instance of temporal outliers, where the white rectangle is a spatial-temporal outlier. Moreover, station 9 in Figure 8(a) exhibits inconsistent traffic flow compared with its neighboring stations, and was detected as a spatial outlier. The identification of spatial outliers, e.g., station 9 in this example, can help traffic managers to quickly respond to faulty detectors.



(a) I-35W North Bound

(b) I-35W South Bound

Figure 8. An example of spatial outliers detected

5. Conclusions

Data mining and visualization are rapidly expanding fields with many new research results and prototypes for various application have recently been developed [14, 8]. In this paper, we construct a visualization software package for observing the summarization of spatial patterns and temporal trends. In the underlying data structure, we model the spatial data as a spatial data warehouse to facilitate the query engine for the on-line analytical processing used in the visualization software. We also present spatial outlier detection algorithms to filter out large uninteresting parts of spatial datasets for visualization. The algorithms were tested with the traffic data from Twin Cities. To support the visualization of spatial data using currently available techniques, data mining faces many challenges and unsolved problems which pose new research opportunities for further study.

6. Acknowledgment

We are particularly grateful to Professor Vipin Kumar, and our Spatial Database Group members, Dr. Weili Wu, Changqing Zhou, Hui Xiong, Xiaobin Ma, Yan Huang, and Ranga Raju Vatsavai for their helpful comments and valuable discussions.

This work is supported in part by USDOT grant on high performance spatial visualization of traffic data, and is sponsored in part by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

References

[1] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley, New York, 3rd edition, 1994.

[2] A. Buja, D. Cook, and D. Swayne. Interactive High-Dimensional Data Visualization. *J. Computational and Graphical Statistics*, 5(1):78–99, 1996.

[3] S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. In *Proc. VLDB Conference*, page 205, 1996.

[4] S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. (1):65–74, March 1997.

[5] S. Chawla, S. Shekhar, W.-L. Wu, and U. Ozesmi. Modeling spatial dependencies for mining geospatial data: An introduction. In *Harvey Miller and Jiawei Han, editors, Geographic data mining and Knowledge Discovery (GKD)*, 1999.

[6] E. Codd, S. Codd, and C. Salley. Providing OLAP(On-line Analytical Processing) to User-Analysts: An IT Mandate. In *Arbor Software Corporation. Avaliable at http://www.arborsoft.com/essbase/wht_ppr/coddToc.html*, 1993.

[7] ESRI. http://www.esri.com.

[8] U. Fayyad, G. G. Grinstein, and A. Wierse, editors. *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann, 2001.

[9] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The KDD Process for Extracting Useful Knowledge from Volumes of data. *Communications of the ACM*, 39, 1996.

[10] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. MIT Press, Cambridge, MA, 1996.

[11] P. Ferguson. Census 2000 behinds the scenes. In *Intelligent Enterprise*, October 1999.

[12] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total. In *Proceedings of the Twelfth IEEE International Conference on Data Engineering*, pages 152–159, 1995.

[13] J. Han, N. Stefanovic, and K. Koperski. Selective materialization: An efficient method for spatial data cube construction. In *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'98)*, pages 144–158, 1998.

[14] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. The MIT Press, 2001.

[15] D. Hawkins. *Identification of Outliers*. Chapman and Hall, 1980.

[16] W. Inmon, J. Welch, and K. Glassey. *Managing the Data Warehouse*. New York, NY: John Wiley & Sons, 1997.

[17] D. Keim and H.-P. Kriegel. VisDB: Database Exploration Using Multidimensional Visualization. *IEEE Computer Graphics and Applications*, 14(5):40–49, 1994.

[18] K. Koperski, J. Adhikary, and J. Han. Spatial data mining: Progress and challenges. In *Workshop on Research Issues on Data Mining and Knowledge Discovery(DMKD'96)*, pages 1–10, Montreal, Canada, 1996.

[19] K. Koperski and J. Han. Discovery of spatial association rules in geographic information databases. In *Advances in Spatial Databases, Proc. of 4th International Symposium, SSD'95*, pages 47–66, Portland, Maine, USA, 1995.

[20] MICROSOFT. Terraserver: A spatial data warehouse. http://www.microsoft.com.

[21] I. Mumick, D. Quass, and B. Mumick. Maintenance of data cubes and summary tables in a warehouse. In *SIGMOD*, pages 100–111, 1997.

[22] G. Piatetsky-Shapiro and W. J. Frawley. *Knowledge Discovery in Databases*. AAAI/MIT Press, 1991.

[23] S. Shekhar and S. Chawla. *Spatial Databases: A Tour*. Prentice Hall, 2002.

[24] S. Shekhar, S. Chawla, S. Ravada, A. Fetterer, X. Liu, and C. Lu. Spatial databases: Accomplishments and research needs. *IEEE Transactions on Knowledge and Data Engineering(TKDE)*, 11(1):45–55, 1999.

[25] S. Shekhar and Y. Huang. Co-location Rules Mining: A Summary of Results. In *Proc. Spatio-temporal Symposium on Databases*, 2001.

[26] S. Shekhar, C. Lu, and P. Zhang. Detecting Graph-Based Spatial Outlier: Algorithms and Applications(A Summary of Results). In *Computer Science & Engineering Department, UMN, Technical Report 01-014*, 2001.

[27] M. Stonebraker, J. Frew, and J. Dozier. The sequoia 2000 project. In *Proceedings of the Third International Symposium on Large Spatial Databases*, 1993.

[28] the Minnesota Department of Transportantation's Traffic Mangagement Center. http://www.dot.state.mn.us/tmc/tmchome.html.

[29] USGS. National Satellite Land Remote Sensing Data Archive. In *http://edc.usgs.gov/programs/nslrsda/overview.html*.

[30] M. Ward. XmdvTool: Integrating Multiple Methods for Visualized Views. In *Proc. Visualization*, pages 326–331, 1996.