Interactive Visualization of Network Anomalous Events

Yang Cai and Rafael de M. Franco

Carnegie Mellon University
ycai@cmu.edu
www.cmu.edu/vis

Abstract. We present an interactive visualization and clustering algorithm that reveals real-time network anomalous events. In the model, glyphs are defined with multiple network attributes and clustered with a recursive optimization algorithm for dimensional reduction. The user's visual latency time is incorporated into the recursive process so that it updates the display and the optimization model according to a human-based delay factor and maximizes the capacity of real-time computation. The interactive search interface is developed to enable the display of similar data points according to the degree of their similarity of attributes. Finally, typical network anomalous events are analyzed and visualized such as password guessing, etc. This technology is expected to have an impact on visual real-time data mining for network security, sensor networks and many other multivariable real-time monitoring systems.

Keywords: interaction, visualization, network anomaly, anomalous event, clustering.

1 Introduction

The paradigm of data visualizations has shifted from merely visual data rendering to the model-based visual analysis [1-9][29-31]. Examples include: 1) *graph models*, such as the social interaction theory of "the six degrees of separation" [26], the "power law of the linked interactions [22], *minimal graph cuts* for analyzing the outliers in a very large social network [23], 2) *color models* such as the spectrograph [24] of the interaction patterns emerged from gas stations and cellular phone towers, 3) the *geographical profiling model* for investigating serial killer's spatio-temporal patterns [27], 4) the cellular automata model for simulating the dynamics of mass panic in public places[25]. These methods computationally incorporate modeling, rules and visualization in one algorithm, which enables emergent pattern discovery and empirical experimentation.

However, there are limitations to these existing approaches: 1) they are 'off-line' models, which are based on isolated databases without connections to other dynamic systems or continuously updated data streams; 2) they do not normally take physical interactions into account, e.g. location, duration and field strength; and 3) many visual analytics for network conditions are static, one-shot, rather than interactive or iterative.

In this study, we focus on the following scientific problems: First, we develop a real-time network event simulator with realistic data streams for visual analytics. We construct a synthetic real-time network database from a real network server and

G. Allen et al. (Eds.): ICCS 2009, Part I, LNCS 5544, pp. 450–459, 2009.

historical databases. To build such a dynamic database itself warrants scientific research because it involves privacy, fidelity and bandwidth issues. We explore several ways to downsize the data space in orders of magnitude. We then investigate which method is the best for this kind of problem. We start with a shape-based dynamic clustering model for massive multi-attribute data points (e.g. up to 64 attributes in 10,000 data points) in a 2D space. We then experiment with other representations, such as colored micro-arrays, pixel maps and 3D surfaces, etc.

Secondly, we develop interactive visualization algorithms. Most of existing visualization models that we found are non-interactive, and follow the sequence of 'data-process-display'. In this study we develop a set of algorithms that incorporate human latency, manual navigation and affection. For example, the human latency-aware algorithm converts a batch optimization problem to an incremental optimization problem that enables the computer to visualize more network data by an order of magnitude. The manual navigation of data would allow the multi-resolution system to 'hide' unnecessary data in a peripheral area and only display the user interested data at the highest resolution. As a result, the visualization system can handle increased network data by orders of magnitude.

Finally, we study the visual detection of signatures and anomalies. The biggest issue of network security is intrusion detection and recognizing whether a system has been compromised. There are two groups of methods: signature detection and anomaly detection [13-19]. Signature-based threat detection scans network traffic for a set of predefined attack or vulnerability patterns – similar to today's anti-virus checking. It seeks the "known bad" signatures and assumes that everything else is good. On the other side, the behavior-based anomaly detection methods try to define the "known good" or "normal" behaviors and assume any deviations from the normal behavior are possible attacks. There are many pros and cons in these two approaches: signature detection won't detect any undefined problems and it is computationally expensive when the signature definitions increase; the anomaly detection methods can potentially detect unknown attacks, however, they often lead to false alarms because it is not so easy to define normal patterns. In our study, visualization is actually a detection and learning tool that combines signature detection and anomaly detection seamlessly. Compared to those typical Network Intrusion Detection Systems (NIDS), which normally require a lengthy supervised machine learning process and a large volume of historical data, our approach can avoid this preparation and directly perform the learn-by-doing detection. Here we integrate the signature detection for known attack patterns (e.g. password guessing) and anomaly detection for spatial patterns (e.g. shape anomaly) or temporal patterns (e.g. periodicity). This provides a visual way to interpret the network flow and gives a human expert the possibility to make the final decision on the detection of an anomaly.

2 The Real-Time Network Event Simulator

We assume at any given time, the visualization system assimilates a real-time sequence of a network data set, up to 10,000 points, in which each of them has up to 64 attributes. The update interval is within 10 seconds. The key factor in this project is

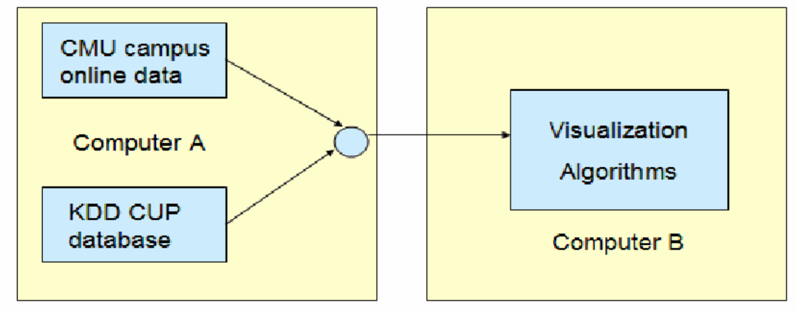


Fig. 1. The architecture of the network data simulator and visualization system

how to obtain a stream of continuous network data in a realistic and non-invasive way. We use two data sources: 1) CMU campus real-time database, and 2) the KDD CUP 1999 database [11].

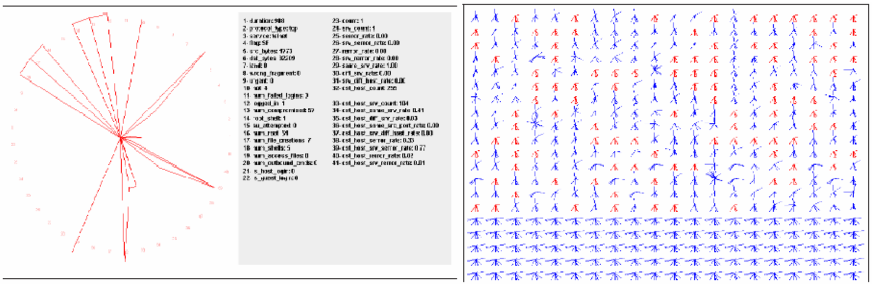
To simulate the dynamics of the network data flow, we use two computers (A as the data source and B as the visualization terminal) to generate the simulated continuous network data. Since 2005, we have been collecting the real-time network data from the Andrew network server in CMU at 1 point per 10 second interval. The database is in XML format. We have developed a Java code to automatically capture the data so that personal identifications are removed. The CMU campus live database gives us the real-world fresh data. Every 10 seconds, computer A passes the data to computer B. The visualization software updates accordingly. To effectively use the information, we add necessary software tools on Computer A, such as the '*tcpdump*' and network traffic analysis utilities.

Since network attacks are not frequent, we also use the captured attack data from the KDD CUP 1999 database [11] as a key reference for designing the algorithm. This data represents thousands of connections with 41 different attributes. There are 2 kinds of attributes: continuous and discrete.

3 Glyph-Based Dynamic Clustering and Visualization

The pre-processing task is to normalize the continuous data so that each attribute can have the same level of influence when comparing one dataset to another (calculating the Euclidean distance). The normalization is performed dividing each attribute by the maximum attribute's value of the whole data scanned during a period of time. The normalized attributes can also be multiplied by a coefficient weight that can be adjusted according to the importance of each attribute for the detection of an anomaly.

The second step of the algorithm is to visualize each network connection and their 41 attributes on a 2 Dimensional graphic. To resolve that, it is necessary to find a visualization technique that represents a multidimensional (n-D) array on a 2-D graphic. The star glyphs technique [12] is an elegant solution for the problem and it fits perfectly for this application. In the glyph's plot, the dimensions are represented as equal-spaced angles from the center of a circle and each axis represents the value of the dimension. Figure 2 illustrates a glyph using network attributes.



1- duration: 980
2- protocol_type: tcp
3- service: telnet
4- flag: SF
5- src_bytes: 1773
6- dst_bytes: 32509
7- land: 0
8- wrong_fragment: 0
9- urgent: 0
10- hot: 4
11- num_failed_logins: 3
12- logged_in: 1
13- num_compromised: 57
14- root_shell: 1
15- su_attempted: 0
16- num_root: 54
17- num_file_creations: 7
18- num_shells: 5
19- num_access_files: 0
20- num_outbound_cmds: 0
21- is_host_login: 0
22- is_guest_login: 0
23- count: 1
24- srv_count: 1
25- serror_rate: 0.00
26- srv_serror_rate: 0.00
27- rerror_rate: 0.00
28- srv_rerror_rate: 0.00
29- same_srv_rate: 1.00
30- diff_srv_rate: 0.00
31- srv_diff_host_rate: 0.00
32- dst_host_count: 255
33- dst_host_srv_count: 104
34- dst_host_same_srv_rate: 0.41
35- dst_host_diff_srv_rate: 0.03
36- dst_host_same_src_port_rate: 0.00
37- dst_host_srv_diff_host_rate: 0.00
38- dst_host_serror_rate: 0.35
39- dst_host_srv_serror_rate: 0.77
40- dst_host_rerror_rate: 0.02
41- dst_host_srv_rerror_rate: 0.01

Fig. 2. Glyph definition (left) and the fixed position glyph display (right)

The right image in Fig. 2 shows 400 network connections displayed in a Glyph form. The glyphs highlighted in red are connections that have similar attributes and consequently a similar glyph's form. The glyphs on the bottom are connections from a DoS (Denial of Service) attack and, comparing to all the other connections, the abnormal form emerges.

The clustering process has two main purposes: first to reduce the large amount of data connections (that can be more than 10 thousand for big networks) and second to detect a possible anomaly. In order to understand how clustering data can perform these two tasks, it is intuitively better to imagine the clusters as galaxies and the network connections as planets. Each planet has its own characteristics that are given by its attributes. Thus, planets with similar characteristics cluster together and form a galaxy. On normal network traffic, it is expected to have a large number of data glyphs with the same normal pattern. This data will create a cluster (like planets of a same galaxy). However, for the case of a network intrusion, different clusters will be created (like a far-away galaxy). These new clusters do not have the same patterns as normal data, thus they are distant from the normal cluster. If the distance between this cluster and the "normal" data is bigger than a certain threshold, the algorithm recognizes it as an anomaly. The creation of a cluster was implemented with a simple algorithm by comparing the similarity among the connections' attributes. This comparison can be implemented by calculating the distance between two connections. The Euclidean distance is the simplest form to measure the distance between two vectors and it is applied.

The first step is to create a cluster using a first data called "reference". The Euclidean distance between this "reference" and the rest of the data is calculated. Data that have a distance smaller than a threshold are added to the cluster and set with a flag telling that they were already selected. The loop continues choosing new unselected "reference" data and so on. The cluster's mean is also calculated during the process and used in the detection part. The anomaly detection for this algorithm is quite simple, and it should be improved in future work. The detection is made by adding an anomaly threshold: if the Euclidean distance between a cluster's mean and the global mean from all the clusters is bigger than the anomaly threshold, all the data from this cluster are anomalies.

We start with a case study of network anomaly visualization. We display clusters of 10,000 data points, in which each point has 64 attributes in real-time on a regular PC. We test Principal Component Analysis, Kohonen Self Organizing Maps (SOMS) and Multidimensional Scaling (MDS). The MDS algorithm is to organize a multidimensional data on a two dimensional graphic by coordinate pairs(x,y). The cartesian plane makes axes explicit and it is easier to organize data according to their characteristics. The idea of the MDS is to measure all the data distance in an N-dimensional space and place it on 2D display so they obey a same mutual distance relationship. However, a 2D perfect configuration is not always possible. Let d_{ij} be the multidimensional distance between a point i and j, calculated by the Euclidean distance. Let also d_{ij} be the 2-dimensional distance between the same point i and j calculated with the Pythogorean Theorem $\delta_{ij} = \sqrt{x^2 + y^2}$. If $d_{ij} \neq \delta_{ij}$ than there is a stress between the data and 2D representation. The stress looks to the multidimensional and 2-dimensional distances between the point P_1 and $P_{2.}$ The stress is calculated and the 2D positions have to be placed on a way that it minimizes this stress. To minimize it, we applied the simplex optimization algorithm.

$$stress = \sum_{i<j} \left(\frac{d_{ij} - \delta_{ij}}{d_{ij}} \right)$$

Figure 3 is an example of the latency-aware algorithm. The glyphs in red are the data from a highlighted cluster. The glyphs in blue and green are the other clusters organized with the MDS algorithm.

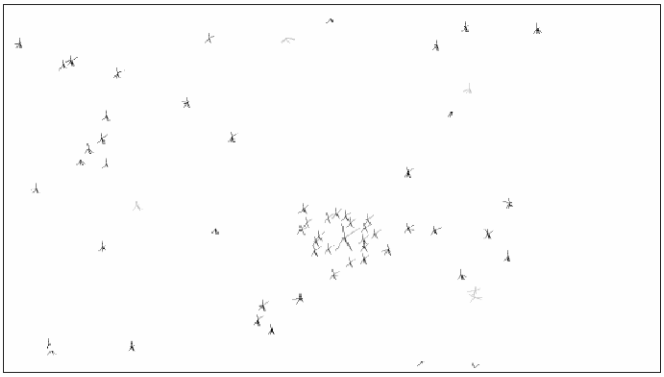


Fig. 3. Example of the clustering algorithm

4 Interactive Visualization Algorithms

Here we develop a set of algorithms that incorporate human latency, manual navigation and affection. For example, the human latency-aware algorithm converts a batch optimization problem into an incremental optimization problem that enables the computer to visualize more network data by an order of magnitude. The manual navigation of data would allow the multi-resolution system to 'hide' unnecessary data in a peripheral area and only display the user interested data at the highest resolution. As a result, the visualization system can handle more network data by multiple orders of magnitude. Furthermore, in a 24/7 pervasive real-time visualization system, psychological boredom is inevitable phenomena.

4.1 Latency-Aware Computing

Human vision has about 0.1 second latency which has been an important factor in modern video and movie technologies [10]. In principle, a system need not update data faster than a human's response time. In light of this, we can use the human latency to design many novel human-centric computing algorithms that incorporate the latency factor. Many visualization methods involve the time-consuming algorithms for clustering and optimization. Instead of waiting for minutes to see the updated results, the latency-aware algorithms are designed to synchronize the speed of human vision by incremental updating. This should create a new philosophy for algorithm design.

In this study, we develop the latency-aware multidimensional scaling model. The algorithm is to organize a multidimensional data on a two dimensional graphic by coordinate *pairs(x,y)*. Described in Section 3, according to the Pythogorean Theorem. The stress looks to the multidimensional and 2-dimensional distances between the points. The stress is calculated and the 2D positions have to be placed on a way that it minimize this stress. To minimize it, we applied the simplex optimization algorithm. The only difference is that here we use an incremental update with a visual pulse interval (e.g. say, between 1/25 sec to 10 sec). Faster than 1/25 second is not necessary because human eyes can't catch up. Slower than a 10 sec interval might cause anxiety.

4.2 Search for Similar Patterns

Human eyes are good for recognition but poor in searching a massive numerical matrix. On the other hand, a computer is poor in recognition but efficient in performing a large searching task. Here we develop an interactive signature or anomaly search algorithm: Given a sample glyph with a certain patterns, the computer will search all the data on the screen and highlight the ones with similar patterns. Figure 4 shows an illustration of the results.

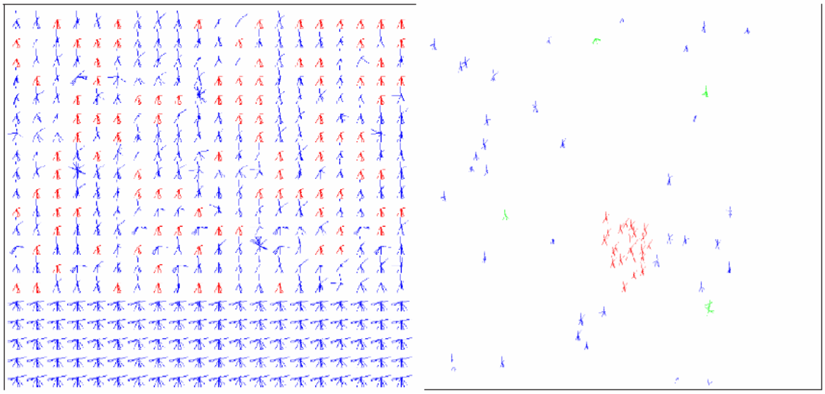
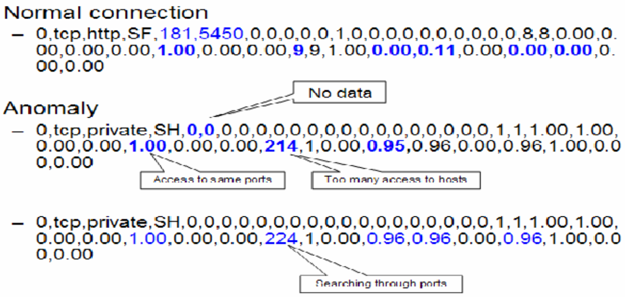


Fig. 4. Highlighted (in red) anomalies with the fixed glyphs (at the bottom of the left image) and dynamically clustered ones (red glyphs in the right image)

5 Visual Transformation for Signature Detection

Signature detection is actually a rule-based expert system, which scans network traffic for a set of predefined attack patterns – similar to today's anti-virus checking. It seeks the "known bad" signatures. With the help of technologies like NIDS (Network Intrusion Detection Systems), Sniffers and SNMP (simple network management protocol), different kinds of network attributes and thousands of data samples can be retrieved in a very short period of time. However, raw numeric data is difficult to interpret. The first scenario focuses on general normal data generation. The connection and data are made from host B to host A services (HTTP, Telnet, FTP, SMTP, HTTPS). We built several connections. Several connections are recorded with '*tcpdump*'. Figure 5 shows a sequence of a normal dataset and two anomalous datasets.



Normal connection

– 0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00

Anomaly

– 0,tcp,private,SH,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.00,214,1,0.00,0.95,0.96,0.00,0.96,1.00,0.00,0.00

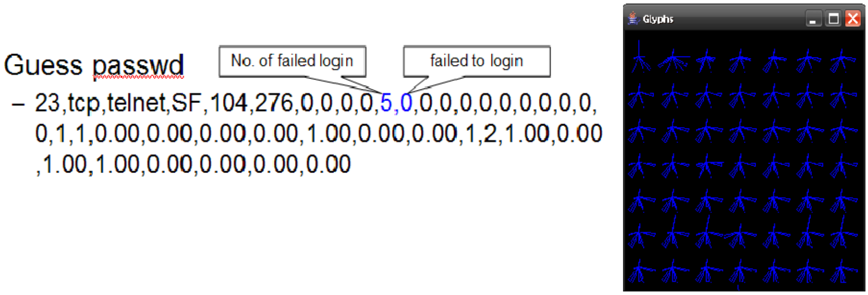
– 0,tcp,private,SH,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.00,224,1,0.00,0.96,0.96,0.00,0.96,1.00,0.00,0.00

No data
Access to same ports
Too many access to hosts
Searching through ports

Fig. 5. Samples of normal string vs. anomalous strings

The second scenario is to make forged data and send out lots of attack data and abnormal data from host A, and then record them with '*tcpdump*'. The typical case for creating abnormal data is to employ *'ipmagic*', with which most IP packets can be manipulated in any way that we want. Following is a command to send a single identical packet for Land attack, which consists of same departing address and port to the destination address and port. It covers all attack groups, where DARPA set the rules to define attack categories. They are categorized with 4 groups below. 1) DoS: denial-of-service, 2) R2L: remote machine to local Machine, 3) U2R: user to root, and 4) Probing: port scanning. For each group, a representative attack is conducted: *'juno'* attack for DoS, 'login' password guessing for R2L, and the attempt for U2R and *'nmap'* scanning for probing attack. All the attack runs on the network are stored.

The next step is to replay stored data in a continuous way where Linux scripts are used continuously. After source data is all set, by running *'tcpreplay*' with options and Linux's scripting, data can be sent in very diverse forms with source data. Following is an example case to send the *'dumped_data'* with a form of randomized IP address and looped 100 times through the interface. After filtering, we can distract data from *'tcpdumped'* data into the computer network. This data is put into visualization software, and then the software makes moving glyphs. Figure 6 shows an example of the anomalous event '*password guessing.*'



Guess passwd

- 23,tcp,telnet,SF,104,276,0,0,0,0,5,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,1,2,1.00,0.00,1.00,1.00,0.00,0.00,0.00,0.00

Fig. 6. Example of 'password guessing' attack

This algorithm uses the Euclidean distance among the network connections in order to create different clusters according to their patterns. Based on the distance among these clusters, anomalies can be detected. It is also possible to visualize the multidimensional data using the star glyphs technique. These glyphs give a visual representation of the numerical attributes, which allows a human to visually detect abnormalities by just looking at the different forms of glyphs.

6 Conclusions

Here we present an interactive visualization and clustering algorithm that reveals real-time network anomalous events. We first build a real-time network event simulator based on the replay of the collected network log data from multiple sources. Then we develop the interactive visualization model that can be connected to the simulator. In the model, glyphs are defined with multiple network attributes and clustered with the recursive optimization algorithm for dimensional reduction. The user visual latency time is incorporated into the recursive process so that it updates the display and the optimization model according to the human vision delay factor and maximizes the capacity of the real-time computation. The interactive search interface is developed to enable the display of similar data points according to their similarity in attributes. Finally, typical network anomalous events are analyzed and visualized such as password guessing, etc.

This technology is expected to have an impact on visual real-time data mining for network security, sensor networks and many other multivariable real-time monitoring systems.

Acknowledgement

The authors would like to thanks the sponsors for this project: US Army Research Office (ARO) and Korea Information Security Agency (KISA). In addition, the authors are grateful to the help from Brian Zeleznik for his comments and editing.

References

1. Cai, Y. (ed.): Ambient Intelligence for Scientific Discovery. LNCS (LNAI), vol. 3345. Springer, Heidelberg (2005)
2. Cai, Y., Abscal, J. (eds.): Ambient Intelligence for Everyday Life. LNCS (LNAI), vol. 3864. Springer, Heidelberg (2006)
3. Cai, Y., Terrill, J.D.: Visual Analysis of Human Dynamics. Journal of Information Visualization 5(4) (2006)
4. Cai, Y.: Ambient Intelligence for Knowledge Discovery: Editorial. To appear on International Journal of Human-Computer Studies (2007)
5. Cai, Y., et al.: Visual Transform for Spatiotemporal Data Mining. Accepted and to appear on Journal of Knowledge and Information Systems (KAIS) (2009)
6. Laws, J., Cai, Y.: Feature Hiding for Human 3D Scan Data. Journal of Information Visualization 5(4) (2006)
7. Cai, Y., Snel, I., Bharathi, B.S., Klein, C., Seetharaman, J.K.: BioSim: A Character-Based Biomedical Problem Solving Environment. Journal of Future Generation Computer Systems 21(7), 1145–1156 (2005)
8. Cai, Y.: Minimalism Context-Aware Displays. Journal of CyberPsychology and Behavior 7(3) (2004)
9. Tanz, O., Shaffer, J.: Wireless local area network positioning. In: Cai, Y. (ed.) Ambient Intelligence for Scientific Discovery. LNCS (LNAI), vol. 3345, pp. 248–262. Springer, Heidelberg (2005)
10. Boff, K.R., Kaufman, L., Thomas, J.P. (eds.): Human Performance Measures Handbook. Wiley and Sons, Chichester (1986)
11. KDD CUP 1999 data (1999), `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`
12. Matthew Ward. Mavis home page (1996), `http://web.cs.wpi.edu/~matt/research/MAVIS/`
13. Neill, D.B., Moore, A.W.: Anomalous spatial cluster detection. In: Proc. KDD 2005 Workshop on Data Mining Methods for Anomaly Detection, pp. 41–44 (2005)
14. Neill, D.B., Moore, A.W.: Rapid detection of significant spatial clusters. In: Proc. 10th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining, pp. 256–265 (2004)
15. Salvador, S., Chan, P.: Fastdtw: Toward accurate dynamic time warping in linear time and space. In: KDD Workshop on Mining Temporal and Sequential Data (2004)
16. Shyu, M.L., Chen, S.C., Sarinnapakorn, K., Chang, L.W.: A novel anomaly detection scheme based on principal component classifier. In: Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop (2003)
17. Zhang, J., Zulkernine, M.: Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection. In: Symposium on Network Security and Information Assurance – Proc. of the IEEE International Conference on Communications (ICC), Istanbul, Turkey (June 2006)
18. Burbeck, K., Tehrani, S.N.: ADWICE: Anomaly Detection with Real-time Incremental Clustering. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 407–424. Springer, Heidelberg (2005)
19. Gomez, J., Gonzalez, F., Dasgupta, D.: An Immuno-Fuzzy Approach to Anomaly Detection. In: The proceedings of the 12th IEEE International Conference on Fuzzy Systems (FUZZIEEE), May 25-28, 2003, vol. 2, pp. 1219–1224 (2003)
20. Levin, I.: KDD 1999 Classifier Learning Contest: LLSoft's Results Overview. In: ACM SIGKDD Explorations 2000, pp. 67–75 (January 2000)

21. Pfahringer, B.: Winning the KDD 1999 Classification Cup: Bagged Boosting. In: ACM SIGKDD Explorations 2000, pp. 65–66 (January 2000)
22. Barabasi, A.L.: Linked: The New Science of Networks, Perseus (2002)
23. Cowell, A., et al.: Understanding the Dynamics of Collaborative Multi-Party Discourse, IVJ, 5(4) (2006)
24. Chakrabarti, D., Zhan, Y., Blandford, D., Faloutsos, C., Blelloch, G.: NetMine: New Mining Tools for Large Graphs. In: The SDM 2004 Workshop on Link Analysis, Counterterrorism and Privacy (2004)
25. Eagle, N., Pentland, A.: Reality Mining: Sensing Complex Social Systems. Personal and Ubiquitous Computing (September 2005)
26. Helbing, D., Farkas, I.J., Vicsek, T.: Simulating dynamical features of escape panic. Nature 407, 487–490 (2000)
27. Guare, J.: Six Degrees of Separation, Vintage (1990)
28. Rossmo, K.: Geograpical Profiling. CRC Press, Boca Raton (1990)
29. NML, Visible Human Project (2008), `http://www.nlm.nih.gov/research/visible/visible_human.html`
30. Wong, P.C., Rose, S.J., Chin Jr., G., Frincke, D.A., May, R., Posse, C., Sanfilippo, A., Thomas, J.: Walking the Path - A New Journey to Explore and Discover through Visual Analytics, IVJ, 5(4) (2006)
31. Beale, R., Hendley, B., Pryke, A., Wilkins, B.: Nature-inspired Visualisation of Similarity and Relationships in Human Systems and Behaviours. IVJ 5(4) (2006)