Visual analysis of large-scale network anomalies

Q. Liao
L. Shi
C. Wang

The amount of information flowing across communication networks has rapidly increased. The highly dynamic and complex networks, represented as large graphs, make the analysis of such networks increasingly challenging. In this paper, we provide a brief overview of several useful visualization techniques for the analysis of spatiotemporal anomalies in large-scale networks. We make use of community-based similarity graphs (CSGs), temporal expansion model graphs (TEMGs), correlation graphs (CGs), high-dimension projection graphs (HDPGs), and topology-preserving compressed graphs (TPCGs). CSG is used to detect anomalies based on community membership changes rather than individual nodes and edges and therefore may be more tolerant to the highly dynamic nature of large networks. TEMG transforms network topologies into directed trees so that efficient search is more likely to be performed for anomalous changes in network behavior and routing topology in large dynamic networks. CG and HDPG are used to examine the complex relationship of data dimensions among graph nodes through transformation in a high-dimensional space. TPCG groups nodes with similar neighbor sets into mega-nodes, thus making graph visualization and analysis more scalable to large networks. All the methods target efficient large-graph anomaly visualization from different perspectives and together provide valuable insights.

Introduction

The traditional computer-to-computer communication networks have been evolving to include the ubiquitously connected device-centric networks, the so-called Internet of Things (IoT), which consists of an unprecedented quantity of emerging end hosts, such as smart phones, sensors, environmental meters, wearable devices, appliances, and vehicles. Another indication of large networks is the growing popularity of social networking [1], with one single social network approaching one billion users. Graphs can be used to naturally represent the relationship of the above kinds of data and provide a useful approach for analyzing big data on networks. We refer to such graphs as "big graphs." Understanding these big graphs is crucial in many cases. For example, an administrator of a cloud-computing system needs to keep track of the traffic distribution among servers and hosts for better network and virtual-machine optimization during a capacity planning stage. Administrators also need to monitor the latest traffic graphs to increase situation awareness for the purposes of responsive troubleshooting and security-related investigations. In a broader Internet scenario, the operators of a social networking service (SNS) website depend on knowledge of the social network to design more effective promotion strategies and online advertising.

Graph anomaly detection [2–4] can both help network operators and managers improve situation awareness of their networks and help detect what the *abnormal* changes are. In order to understand patterns represented by large graphs, data-mining techniques are usually applied. For example, graph-mining tasks [5–7] involve the detection of patterns from graphs, nodes, and edges. Some characteristics of complex networks include power laws, degree distribution, small-world features, communities (or clusters) [8], random graph models, and network growth models [6, 9]. Those models can be used to either generate new graphs following

Corresponding author: L. Shi, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Haidian, Beijing, China (shil@ios.ac.cn).

Digital Object Identifier: 10.1147/JRD.2013.2249356

©**Copyright** 2013 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the Journal reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied by any means or distributed royalty free without further permission by computer-based and other information-service systems. Permission to republish any other portion of this paper must be obtained from the Editor.

0018-8646/13/$5.00 © 2013 IBM

certain distributions or to detect abnormalities in a given graph. Visualization is a well-known technology to help understand and analyze the data. Visual data analysis differs in essence from automated methods by incorporating human perceptions into the data mining process [10] so that researchers can detect patterns that could be missed by traditional automatic data-mining methods.

Nevertheless, visualization for anomaly analysis of large graphs is challenging because of the nonlinear increase of *complexity* and the highly *dynamic* nature of such large networks. For example, nodes in most sensor networks are deployed in hostile environments, which have a great impact on the performance and reliability of network routing and topology. In a large enterprise network, users and applications are initiating network connections and joining and leaving networks at any time. Because networks are constantly changing, visualizing only anomalous changes on a busy network is a nontrivial task. Anomaly visualization in large-scale networks requires additional thinking about how to present meaningful results to users without overwhelming them with less relevant information. Finally, even from a graph-drawing point of view, visualizing a graph with more than approximately 100 nodes may face two fundamental challenges. First, the classical force-directed algorithm, which often positions the graph nodes so that edges are of similar length and so that crossing of edges is reduced, may fail to calculate an aesthetic graph layout in real time. Second, even if a big graph layout is computed, the visual clutters (mainly the edge crossings) created by the straight-line node-link representation prohibit the user from understanding the graph in detail, which may be important for analytical tasks.

In this paper, we present a brief overview of five potentially useful graph visualization techniques for anomaly analysis on large-scale network data. Specifically, our visual analytic tool (which provides users with overview and details) includes community-based similarity graphs (CSGs), temporal expansion model graphs (TEMGs), correlation graphs (CGs), high-dimension projection graphs (HDPGs), and topology-preserving compressed graphs (TPCGs). We examine the difficult problem of detecting abnormal changes based mostly on snapshots of time-series network graphs without any prior knowledge of anomalies. The key challenge is how to effectively visualize the *dynamics* and important changes among the networks. In CSGs, nodes are connected on the basis of their similarity scores so that difficult-to-find anomalies can be detected with respect to the membership changes within communities. More importantly, by considering an appropriate balance of granularity and complexity, the CSG algorithm is more tolerant to the large changes and high dynamics usually associated with larger networks by treating communities rather than individual nodes or edges.

Second, we make use of the TEMG to track the highly dynamic nature of the network data resulting from the unpredictable network behavior and routing topology. The basic idea of TEMG is to render network topology graphs as directed trees according to their time-dynamic routing paths. One major benefit is that temporal changes to the network are encoded to the graph, making it easier to analyze network dynamics. With a hierarchical tree structure, TEMG is more efficient and easier for data exploration than a regular graph and therefore may be used to identify the anomalies in a time-efficient manner even with the routing dynamics of large-scale networks.

Third, we make use of the CG, in which nodes are the data dimensions of original graph nodes, and edges represent their correlation scores. In addition, HDPGs are designed to map the high dimensionality of node attributes into visible patterns and demonstrate the temporal dynamics. Original graph nodes are transformed into a different dimensional space using start coordinates [11]. The locations of nodes are determined by the distance to all dimensional anchor nodes on a circle, and the edges show the temporal evolution of changes. Together, CG and HDPG provide a useful alternative for understanding anomalies among the higher data dimensions associated with larger graphs.

Last, we make use of TPCGs to address the fundamental visualization challenge for big graphs. TPCG groups nodes with similar neighbor set in the graph together into a larger mega-node and then regenerates a new compressed graph for subsequent visualization and analysis. With lower computational complexity, TPCG scales to support graphs with a million nodes.

We hope that this brief overview of useful methods serves as background to other researchers who want to employ similar methods for visual analysis of data involving large networks. Additional information can be found in the various references.

Visualization challenges for big data and related work

Visualization plays an important role in the top three stacks of the big data taxonomy defined by International Data Corporation (IDC) [12]. Because of the increase in data with respect to volume, velocity, and variety, visualization is facing unprecedented challenges in almost all aspects to deliver huge potential insight from big data. In addition to large volume, velocity is another important dimension of big data, often involving the streaming of "fast data." In other words, velocity involves *dynamically* updated information. This poses a challenge to the traditional batch-mode visualization pipeline over the entire dataset. Because straightforward animation methods have limitations, effective visualizations designed to demonstrate dynamic changes, rather than case-by-case examples, are of great importance in facilitating the analytics.

Visual analytics is analytical reasoning facilitated by interactive visual interfaces [13]. Big data visual analytics

combine both automatic machine analysis (e.g., data-mining and machine-learning algorithms) and manual human analysis, which employs expert domain knowledge and interaction. In the big data era, there is a desperate need with respect to both aspects for paradigm-shifting thinking to cope with the changing data properties. We need linked views combining all visualizations and take advantage of parallel algorithms and efficient data transformations.

The recent surge of numerous heterogeneous information sources generates *uncertainty* or even data errors. Innovation with respect to semi-automated visualization tools is therefore needed to help information managers quantify the data quality and integrate multiple data sources into one coherent corpus. Different from the traditional methods, often correcting every data-quality problem, tools for big data focus on the patterns and correlations of the data anomalies for strengthening information visualization.

In this paper, we focus on the visual analysis of big graphs (networks) and their anomalies [14, 15]. Previous literature on visualizing large graphs falls into three categories. The first class studies the efficient drawing algorithms of large graphs [16–19]. Most practical methods [16, 17] follow a multi-scale approach. Instead of computing the large-graph layouts directly, they iteratively coarsen the graphs into smaller abstractions until feasible for use by classical layout algorithms. The original large-graph layout is restored by recursively adding the coarsened graph elements back and refining their layouts. The second class alleviates the visual clutter of a large-graph representation through view transformations. They include topological edge bundling [20], hyperbolic views [21, 22], and fisheye distortion [23]. The third class reduces the visual complexity through data transformation, notably multi-level graph clustering [24–26] and partitioning [27], geometric clustering [28], and filtering [29–31]. The entire graph can be navigated through graph hierarchy traversals [24].

To understand the graph anomalies, which can represent abnormal or malicious network behaviors, the classical diagnosis methods have been based on the network itself and its graph properties, such as degree distributions, sub-graph isomorphism [32], graph edit distance [33], difference graphs [34], or discrepancies [35]. While there are techniques dealing with multidimensional data (e.g., parallel coordinates [36], star coordinates [11], TimeWheel, and MultiComb [37]) and time series data visualizations [38] (e.g., GrowthRingMaps [39] and SpiralGraph [40]), few of them are designed for effectively analyzing spatiotemporal anomalies in large graphs. There also exist visualization tools for packet-level or flow-level information [41–45], sensor networks [46], communication networks [47], and biological and general networks [48, 49]. Notably, we focus on the *dynamics* and *anomaly* of large-scale networks by combining both automatic analytic algorithms and the interactive visualization process. While the above existing visualization tools can be useful to visualize networks if people know exactly what to locate in the first place, we believe presentation of "need-to-know" information will guide users to detect spatiotemporal anomalies that are not easy to find otherwise, and most importantly to find the root causes of those anomalies.

Community-based similarity graphs

While graph anomaly visualization that is based on each node and edge gives a maximum level of detail, often it provides too much detail and can be obfuscating, thus less effective in the face of larger networks with higher dynamics and "churn" rates. Therefore, as we mentioned, in order to analyze spatiotemporal anomalies in larger networks, we develop CSGs by considering an appropriate balance of granularity and complexity. One can view the CSG approach as an *intermediate* similarity metric between whole graph properties (coarse) and nodes/edges (fine). One immediate advantage of comparing networks at the community level is the attenuation of noise from individual node and edge changes.

As suggested, one key challenge of anomaly analysis is the understanding of which changes are normal and which changes are abnormal, and what are the reasons behind these anomalies. In the scheme of CSGs, no matter how dynamic the nodes are (nodes may join and leave the network at any time), if nodes consistently belong to the same community (or consistently belong to different communities), this is considered normal change; otherwise, this is considered abnormal change.

Algorithm

Intuitively, if a user suddenly uses a different set of applications, appears on a different set of hosts, or contacts many different target machines causing his membership change with respect to other users, then the behavior of that user is possibly suspicious and needs the administrator's attention for further investigation. In CSG visualization, users may switch their memberships in a temporal manner such that the distance between two snapshot graphs increases as some user nodes change their memberships from one community to the other.

We use a community-based algorithm [50] to measure graph similarity, that is, given any two sets of communities or clusters C_1 and C_2 that do not need to contain exactly the same number of communities, the distance between communities is based on an idea derived from the Rand Index [51]. We consider the ratio of 1) the number of nodes *consistently* belonging or not belonging to the communities to 2) those belonging to the *same* community in C_1 but ending up in different communities in C_2 (SD), or vice versa (DS). Expressed mathematically, $D(C_1, C_2) = 1 - (SS + DD)/(SS + SD + DD + DS)$. The larger the ratio, the smaller the distance is. Here, *SS* refers to two nodes that

belong to the *same* community before and still belong to the *same* community after. *DD* refers to two nodes that belong to *different* communities before and still belong to *different* communities after.

Once we compute a distance matrix over all pairs of communities, a multidimensional scaling (MDS) [52] view can be mapped in an efficient way that allows investigators to observe any changes (or anomalies) over the entire data range. With the help of the intelligence provided by the visual analytic tool, administrators' domain knowledge can then play an important role when determining the root cause of the suggested anomaly (i.e., the actual cause of the changes), by interacting with the data through user-friendly mouse clicks and queries.

Visualization design and examples

Suppose an administrator of a large enterprise network wants to know if there are any suspicious user behaviors that possibly violate acceptable usage policies (AUPs). The administrator opens our graph similarity visualization tool, sets the granularity of time window as one day, and 30 daily network activity snapshot graphs are generated automatically by the visualization tool. Because the administrator only wants to see user behaviors, he chooses to generate *user similarity graphs*. For example, users are connected if they share at least one common application, one common destination host, etc. With a click of a menu option, various graph community detection algorithms can be applied. In this case, the *Walktrap* [53] algorithm is selected to compute communities with the best modularity. The *Walktrap* is especially appealing because the administrator does not need to specify the exact number of communities in advance.

First, a line chart of graph distances over the past month is plotted to help the administrator get started with the investigation. Graph distances are in proportion to community membership changes between consecutive snapshot graphs. Therefore, the spikes in the distance indicate potential anomalies. From charts, the administrator easily determines that his network from day 9 to day 10 has the highest variation in terms of community membership changes.

A natural question following the above observation is who is responsible for those changes. The interactive exploration capability of the tool allows the investigator to visually explore the two graph communities (**Figure 1**) that correspond to the time of change (day 9 to day 10). In this example, the edges in the user similarity graph represent the common destination IP (Internet Protocol) addresses users have contacted. In Figure 1(a), user *nyadav* (highlighted in solid red cycle at the top) belongs to the graduate student community. A quick querying for *nyadav* and one of its neighbors *hlu* reveals the overlapping target domains are *google.com*, *mozilla.com*, *deploy.akamaitechnologies.com*, *ist.psu.edu*, and *wizard.cse.nd.edu*. On the other hand, user *psempoli* (highlighted in the solid blue cycle at the bottom) belongs to the *condor* [54] community. A query on one of his neighbor users, *condor*, reveals users share target machines in *cse* and *crc* subdomains.

Interestingly, in the following snapshot graph [Figure 1(b)], user *nyadav* (highlighted in solid red cycle) and *psempoli* switch their community memberships. User *nyadav* becomes a member of condor community with new destination server *directory.nd.edu*. User *psempoli*, on the other hand, becomes a member of graduate ("grad") student community with overlapping destination domains in *google.com*, *rackspace.com*, and *deploy.akamaitechnologies.com* with one of his neighbors (*hwang6*). The visual exploration process answers what causes the changes, i.e., the users switch their community membership due to the changes of the destination domains they contacted. While the data in the above example are from users consisting of mostly students and faculty and may not contain malicious attacks, the methodology of the proposed algorithms and visualization framework allows the detection of potential malicious user behaviors. Graph similarity visualization based on community membership changes can serve as a promising alternative in analyzing hard-to-detect anomalous network activities that warrant further investigation. More importantly, the CSGs are more tolerant to the high dynamics of a large network by treating communities rather than individual nodes.

Temporal expansion model graphs

The *dynamics* of routing topology is the key indicator to the performance of many large-scale wireless communication networks including sensor networks. Traditionally, it is difficult to compose an analytics-friendly visualization for such a graph because by nature this topology is a time-varying graph because of the instability of ad hoc communications and routing schemes. Packets delivered from one node to an immediate neighbor may have to cross multiple hops next time, even for the same destination. The resulting graph drawn by normal methods is quite cluttered with either traditional *geographical* or *logical* layouts.

We make use of TEMG [55] for constructing a more intuitive graph to track a time-dynamic, evolving network topology. The basic idea of TEMG is to split a *physical* node into multiple *logical* nodes according to the dynamic routing paths to the common destination (e.g., the sink node in a sensor network). The advantages of TEMG are twofold. First, the resulting topology graph is essentially a directed tree, enabling more user-friendly visualization and human navigation. Second, temporal changes to the network are encoded in the graph itself, providing input for further analytics with respect to network dynamics.

Algorithm

In contrast to traditional graphs, TEMG nodes are defined by using the path N from the original node (S) to the

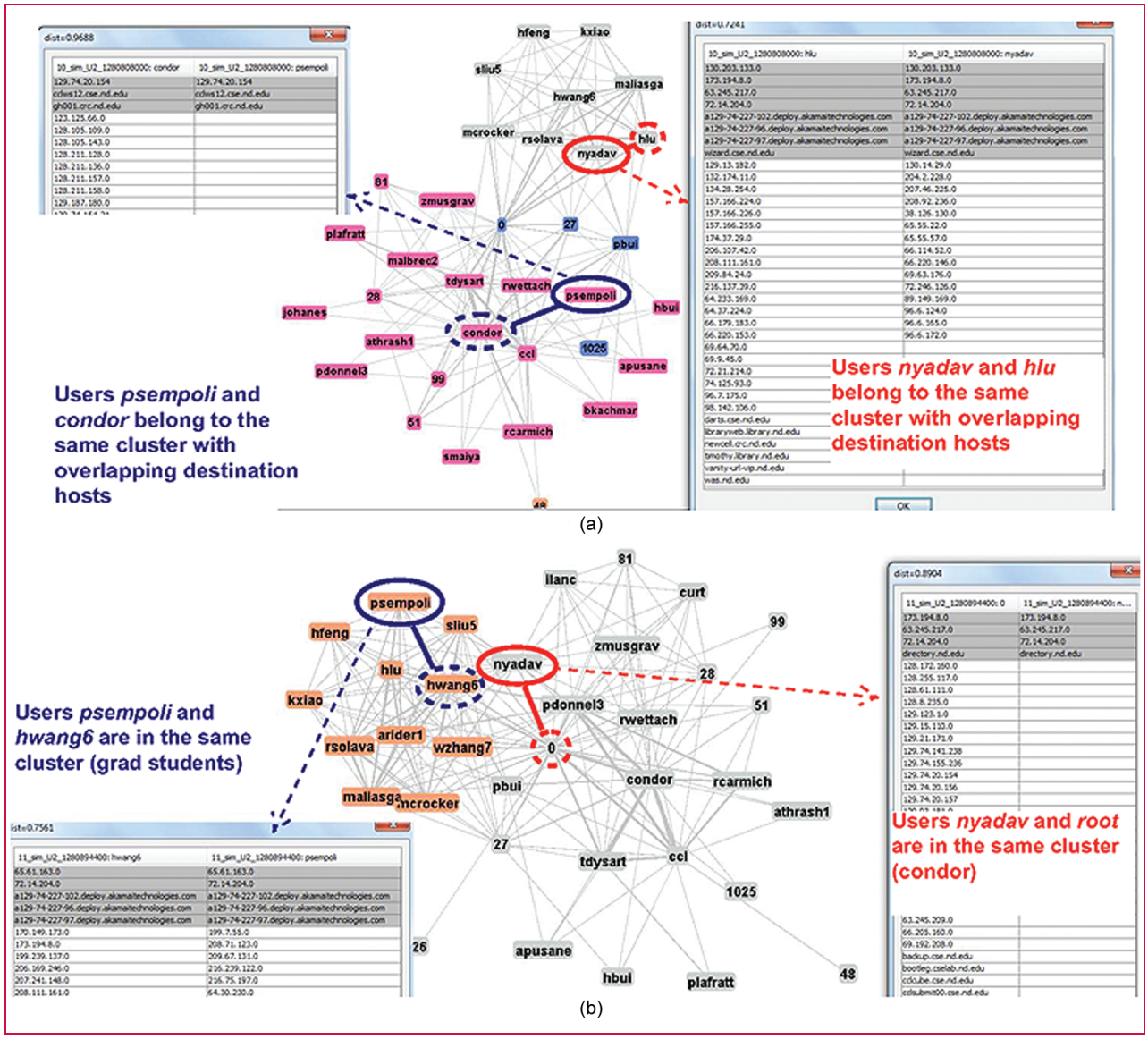

Figure 1

User similarity graphs with edges representing common destinations. In community-based similarity graph (CSG) visualization, anomalies are suggested by users' community membership changes. (a) User communities at time t. (b) User communities at time $t+1$.

destination node (R). Two nodes are identical only if their paths are the same. Another difference involves keeping track of the event time series (T) of each node (packet sent from S). An m-hop path from source node S at time t is represented as $\Gamma(S,t) = (S, p_1, p_2, \ldots, p_{m-1}, R)$, where p_i is a node on the path. TEMG graphs are generated in two steps. First, we split each input path into multiple nodes and add them to the node set. For the m-hop path $\Gamma(S,t), m+1$ path nodes are generated: $N_1 = (S, p_1, p_2, \ldots, R)$, $N_2 = (p_1, p_2, \ldots, R), \ldots, N_m = (p_{m-1}, R)$, and R (the sink node). Second, we truncate the raw path data into edges to construct the final graph. For the path data $\Gamma(S,t)$, m edges are generated: $(N_1, N_2), (N_2, N_3), \ldots, (N_{m-1}, N_m), (N_m, R)$.

We detect spatial-temporal anomalies as follows. Topologically, a major node is identified from the path nodes belonging to the same physical node as the one with the longest time series, i.e., the one with the most packet deliveries through its associated path. Temporally, we detect changing points over the packet delivery time series $T = (t_1, t_2, \ldots, t_k)$ of each sensor node S. The life cycle of node S is partitioned into multiple fix-length bins, and the number of values accumulated in each bin is recorded as

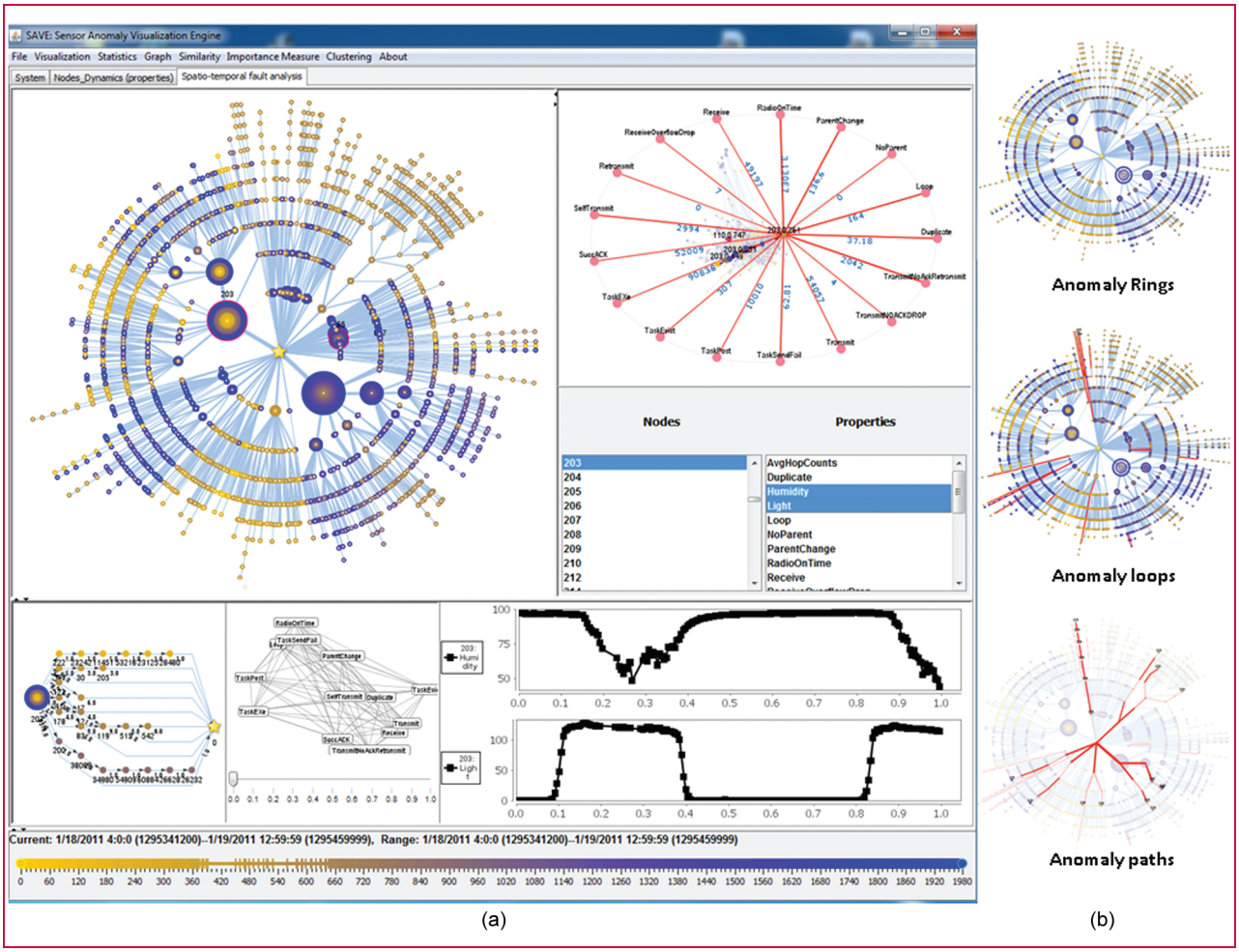

Figure 2

Temporal expansion model graph (TEMG) visualization with "overview + detail" design. (a) Tool overview. (b) Examples of TEMG views [from upper left corner of (a)]. The *color* and *size* of a node represent the *time* and *magnitude* of network traffic. The routing dynamics is encoded into directed trees instead of regular graphs, simplifying search and visual exploration process. Highlighted in red in (b), anomaly rings, loops, and paths simplify anomaly analysis.

$(v_1, v_2, \ldots, v_m)$. We then compute the changes of these bin sizes as $(\Delta_1, \Delta_2, \ldots, \Delta_m)$, where $\Delta_i = v_i - v_{i-1}(v_{-1} = 0)$, representing the changes of node S during the time period of bin i. This indicator reflects the topology changes centric to node S in TEMG.

Visualization design and examples

Figure 2 provides an overview of the visualization interface [Figure 2(a)] and a few examples of TEMG [Figure 2(b)]. The combined view shares the similar concept of linking and brushing [56], which is useful for user visualization and interaction. To visualize packet sending patterns, we use the idea of GrowthRingMap [39] and compose a temporal ring for each node. Starting from the earliest one to the latest, each normalized time-series value is drawn as a ring on the node. In our example, we choose *orange* to indicate the *earliest* time and *blue* to indicate the *latest* time. The resulting drawings could effectively guide the users in detecting temporal topology changes. In Figure 2, for example, there are clusters of orange nodes and blue nodes with similar temporal patterns.

Spatiotemporal anomalies can also be readily visualized using TEMGs. For example, Figure 2(b) highlights loops in paths detected in the topology in red. Once a node under investigation is selected, major paths with higher traffic volume are automatically rendered in the TEMG with thicker strokes, and non-major paths are rendered with dashes. The packet-sending time-series anomalies within each node are drawn by *anomaly rings*. The time points with ascending packet deliveries are rendered in the solid-line color rings,

whereas the time points with descending deliveries are drawn in dashed rings. In both rings, the color hue still follows the temporal ring color at its time point. Our graph visualization design (which provides users with overviews and details) allows the scenario such that once the user selects a path node in TEMG, an egocentric graph (i.e., a graph originating from one node with its immediate neighbors and associated interconnections) is highlighted, which essentially represents the path changes from the selected node to the sink.

It is obvious that because each node now has only one parent, TEMGs reduce most visual clutter compared with traditional, straightforward graph-drawing based on physical location or network connections. While TEMGs have the advantage over traditional graph drawing, by default each timestamp of the reporting time of a node is treated as one data point mapped in the view. Although the total number of nodes may increase, the complexity of search for anomalous nodes is reduced to logarithmic complexity due to the hierarchical structure of the directed, temporal tree. In other words, instead of performing linear search in unstructured graphs, the investigator can perform binary search-like operations to quickly identify the problem node in network troubleshooting. We have tested six entire-day datasets from a real-world large-scale wireless sensor network (WSN)-GreenOrbs [57]. The experimental data suggests that TEMG is effective for large-scale sensor networks.

Correlation graphs and high-dimension projection graphs

According to trends, network data not only becomes large in size but increasingly complex in terms of higher dimensions on each node. For example, in enterprise networks, nodes can contain such attributes as IP addresses, port numbers, applications, users, and files. In sensor networks, each node may have as high as 30 dimensions [55] ranging from normal sensor readings (e.g., temperature, light, humidity, and voltage) to network routing counters [e.g., radio-on-time, transmit, receive, retransmit, and succACK (successful acknowledgment)]. Naturally, the values of each dimension of data in graph nodes change over time, suggesting a high temporal dynamics. Understanding the causality relationship between dimensions, i.e., how one dimension changes in relation to another, is often essential for detecting and analyzing network anomalies. For example, in sensor networks, one might ask whether the number of errors and packet transmission increases in proportion to voltage decrease of sensor nodes. This relationship or correlation can be used as one of the metrics for anomaly detection and analysis. We introduce the concept of CGs to understand the complex relationship among data dimensions and to detect anomalies. Furthermore, we want to know both the spatial and temporal anomalies of high-dimensional graph nodes. For s*patial anomalies*, we may ask how does a graph node differ from others in terms of dimensions of data? For *temporal anomalies*, we may ask how does a single graph node change through time? In addition, note that an HDPG is also designed to map the multi-dimensions of graph node attributes into visible patterns and render the spatial-temporal dynamics.

Algorithm

CGs are constructed in the following way. First, time-series datasets of property-value vectors are extracted from the raw data in real-time on the basis of the selection of the nodes, as well as start and end timestamps in the visual analytic tool. Second, our tool computes the correlation scores according to the Pearson product-moment coefficient [58], i.e.,

$$Correlation(p_1, p_2) = \frac{|p_1| \cdot \sum_{i=1}^{|p_1|} p_{1_i} \cdot p_{2_i} - \sum_{i=1}^{|p_1|} p_{1_i} \cdot \sum_{i=1}^{|p_2|} p_{2_i}}{\sqrt{|p_1| \cdot \sum_{i=1}^{|p_1|} p_{1_i}^2 - \left(\sum_{i=1}^{|p_1|} p_{1_i}\right)^2} \cdot \sqrt{|p_2| \cdot \sum_{i=1}^{|p_2|} p_{2_i}^2 - \left(\sum_{i=1}^{|p_2|} p_{2_i}\right)^2}},$$

where p_1 and p_2 are the property vectors, whose lengths equal the number of timestamps. A correlation matrix can then be computed for each pair of properties. A CG associated with the correlation matrix can then be constructed and visualized [59].

HDPGs are constructed in the following way. We first categorize the dimensions of node data into property vectors defined by intervals of timestamps. Each high-dimensional sensor node is then transformed into a data point onto a two-dimensional star-coordinates [11] space (or RadViz [60–62]). The dimensions equally distribute on the circumference of a cycle $(x^2 + y^2 = r^2)$, with the coordinates of those dimensional anchors on the circumference being defined by $(x_i, y_i) = (r \cdot \cos(\theta_i), r \cdot \sin(\theta_i))$, and $\theta_i = i \cdot (2\pi/dim),\ i = 0, 1, 2, \ldots, dim - 1$, where r is the radius, θ is the angle measured in radians, and dim is the number of dimensions. Temporal dynamics of nodes can be presented by different xy locations at two different timeslots (from t to t'). In this view, not only are the relative locations among the sensor nodes quickly visualized, but the temporal evolution of the same node at different times is also analyzed.

The property value P for each graph node i at time t is a three-tuple, i.e., $P_{(i,t)} = \{V, V_{[0,1]}, \Delta_{[0,1]}\}$, where V is the original property value (e.g., the actual sensor readings or counter values v). To consider different scaling of properties, each dimension is normalized between 0 and 1, computed as $V_{[0,1]}$. The normalization is based on the minimum (min) and maximum (max) observed values reported by each or all the sensors during the entire time range, i.e., $(v - min)/(max - min)$, depending on the emphasis on either temporal or spatial anomaly detection. Further, $\Delta_{[0,1]}$ represents the normalized delta or changes of property value from the previous timestamp, e.g., a value of 0 means no change. In this way, it is easier to visualize the turning points when the state changes.

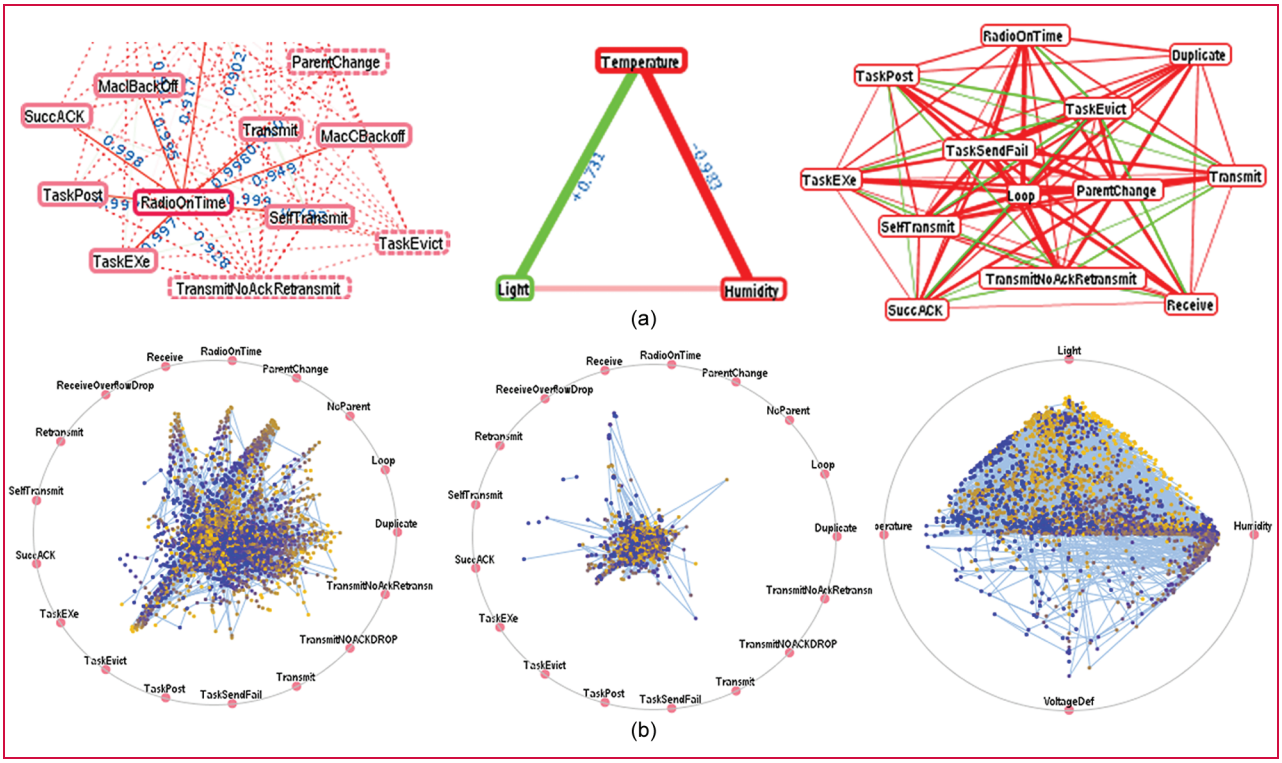

Figure 3

Multi-dimensional anomaly visualization using (a) correlation graphs (CGs) and (b) high-dimension projection graphs (HDPGs). In (a), each node represents one dimension of sensor readings and edges represent positive or negative correlations. The right picture of (a) is a *delta* correlation graph, e.g., green edges for increases in correlation and red edges for a decreasing correlation trend. In (b), data dimensions are represented by anchor nodes on a ring. If a node has the same value for all dimensions, it will be at the center. Scatter plots show temporal movements (orange for earlier and blue for later) of graph nodes as the value for each dimension changes.

Once the spatiotemporal mapping of properties of the nodes is performed, further analysis can be performed by computing the clustering of nodes. For example, the visual analytic tool includes state-of-art clustering modules, which can detect any data points deviating from the centroid by testing if the distance $(dist)$ of i-th node (n_i) from k-th cluster (C_k) is beyond a threshold value [e.g., a constant (m) multiplied by the standard deviation (σ) of distances from all other nodes within the same cluster to the centroid], and identify these points as outliers. Expressed mathematically, this deviation is represented by $dist(n_i, C_k) > m \cdot \sigma$.

Visualization design and examples

Figure 3(a) shows the concept of CGs, in which each node represents one dimension of data associated with a node, and the weights of edges represent the correlation scores between dimensions. The layout takes a force-directed approach by setting the optimal length of the edge inversely proportionally to the correlation value. For example, a CG with the mixture of sensor readings and sensor counters may be positively correlated because high humidity could cause problems in transmission, resulting in packet loss and thus more retransmissions. To better diagnose network dynamics, we also introduce the *delta* CG visualization, in which changes in correlations (rather than the absolute correlation values) are added to the graph. We encode correlation increases into *green* edges, decreases into *red* edges, and magnitude of changes into edge thickness.

Figure 3(b) shows screenshots of HDPGs of sensors. The anchor nodes on the bounding circle are the data dimensions of graph nodes. Inside the circle, the location of each node depends on the value of each dimension. HDPGs use a modified version of a spring-force model of a graph layout. Intuitively, the higher the values of data dimensions, the stronger is the force to pull the node into the dimension anchors along the cycle. Theoretically, if the values of all dimensions are equal, the node will be located at the center of the circle. The color of the plot indicates the time of the measurement, i.e., orange indicates early measurements, whereas blue indicates late measurements. Edges are added to visualize the temporal changes of the dimensional values.

The left graph of Figure 3(b) shows the temporal movement of sensor nodes based on all reported values. As shown, there are trends associated with some dimensions, e.g., more nodes reported *ParentChange* at early times than late times. Because values of many data dimensions are designed to be cumulative (e.g., counters in sensor networks), the distribution of the values may not accurately track the anomalies at their exact time. The higher value of one counter could be due to a burst from a previous event. To solve this problem, we used a *temporal-HDPG* [the middle graph in Figure 3(b)], in which *normalized delta* values are computed from the last known measurement. From the graph, more anomalies can be potentially identified as the burst of value changes are located, such as *Receive, Transmit*, *SelfTransmit*, and *SuccAck*. Finally, the right graph of Figure 3(b) provides an example of four dimensions of sensor readings, i.e., *light*, *temperature*, *humidity,* and *voltage*.

Topology-preserving compressed graphs

As we suggested, modern networks can be quite large in size, containing hundreds of thousands or even millions of nodes. Although visualization is an effective technology to analyze network data, visualizing a graph with merely a few hundred nodes is extremely challenging due to the following reasons. First, the classical force-directed methods in most cases fail to calculate an optimally aesthetic graph layout in real time. Second, even if a graph layout is computed, the visual clutter of big graphs (mainly the edge crossings) created by the straight-line node-link representation prohibit the user from understanding the graph in detail. Traditional methods either create the layout with an abundance of visual clutter, or oversimplify the topology through graph clustering and filtering.

Unlike graph clustering, our idea is to condense the graph by removing the redundancy (without any loss) in topology. We make use of TPCGs, which group the graph nodes with the same neighbor set together as a larger *mega-node* and regenerate a compressed graph for the subsequent visualization and analysis. While reducing visual complexity, TPCG preserves the topology and other critical features from the original graph, making human understanding and analysis easier and more accurate.

Algorithm

The basic idea of our approach is to aggregate nodes with similar connection patterns (e.g., same neighbor sets) in the graph into groups and then construct a new graph for visualization. For a formal definition, let $G = (V, E)$ be a directed, weighted, and connected original graph, where $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$ denote the node and link set. Let W be the graph adjacency matrix where $w_{ij} > 0$ indicates a link from v_i to v_j, with w_{ij} denoting the link weight. In each row of W, $R_i = \{w_{i1}, \ldots, w_{in}\}$ denotes the row vector for node vi, representing its connection pattern. (w_{in} is the link weight from node v_i to v_n.) The compressed graph is denoted as $G^* = (V^*, E^*)$. The compression ratio is defined by $\Gamma = 1 - |V^*|/|V|$.

The basic algorithm takes the graph as a simple, undirected, and unweighted one by setting $w_{ii} = 0$ and $w_{ij} = w_{ji} = 1$ for any $w_{ij} > 0$. On graph G, order its node list by the corresponding row vectors R_i $(i = 1, \ldots, n)$. For any collection of nodes with the same row vector (including the single outstanding node), aggregate them into a new mega-node $G_{vi} = \{v_{i1}, \ldots, v_{ik}\}$. All G_{vi} form the node set V^* for the compressed graph G^*. In addition, let $fv_i = v_{i1}$ denote the first sub-node in G_{vi}. The link set E^* in G^* is generated by simply replacing all fv_i with G_{vi} in the original link set and removing all the links not incident to any fv_i. We have also extended our compression algorithm to support directed, weighted, and dynamic graphs by generalizing the definition of adjacency matrix and the corresponding row vectors.

Visualization design and examples

TPCG uses a force-directed layout algorithm in the Kamada-Kawai model [63] with the stress majorization solver [64] and the multi-level force-directed layout algorithm from the GraphViz tool [17], with potential adaptation to other possible applicable layout algorithms such as multipole graph drawing [65].

A comparison of TPCGs with the original graphs is shown in **Figure 4** for three different datasets. The first is based on a VAST (Visual Analytics Science and Technology) 2011 Mini Challenge-II dataset, which contains traffic logs from a multinational corporate network [i.e., a firewall log (similar to NetFlow data)], an intrusion-detection system (IDS) log, a system log, and a Nessus network vulnerability scan report. The left graph in Figure 4(a) is a network connectivity graph with anomaly icons rendered on nodes for anomaly analysis. However, the view is cluttered, thus obfuscating the analysis. The left graph in Figure 4(b) shows a compressed graph derived from the original graph above. It contains much fewer nodes and edges due to the properties of many security attacks, such as port scans and denial-of-service (DoS) attacks, that exhibit a broadcasting pattern. With significantly fewer nodes and edges in the transformed compressed graph, it may be easier for the investigator to analyze the anomalies, e.g., compromised machines 192.168.2.174/175 conducting port scan activities and DoS attacks on web server 172.20.1.5.

The middle graphs in Figure 4 are based on honeypot (intrusion-detection trap) monitoring [66], which contains 14 million labeled flows and 7 million alerts from a single honeypot lasting about six days. Take a seven-hour trace as an example, the initial view [the middle graph of Figure 4(a)] is visually overwhelmed by the connected hosts. To access the dynamic patterns of the attackers, the user can check the "temporal" option and generate a dynamic version of the

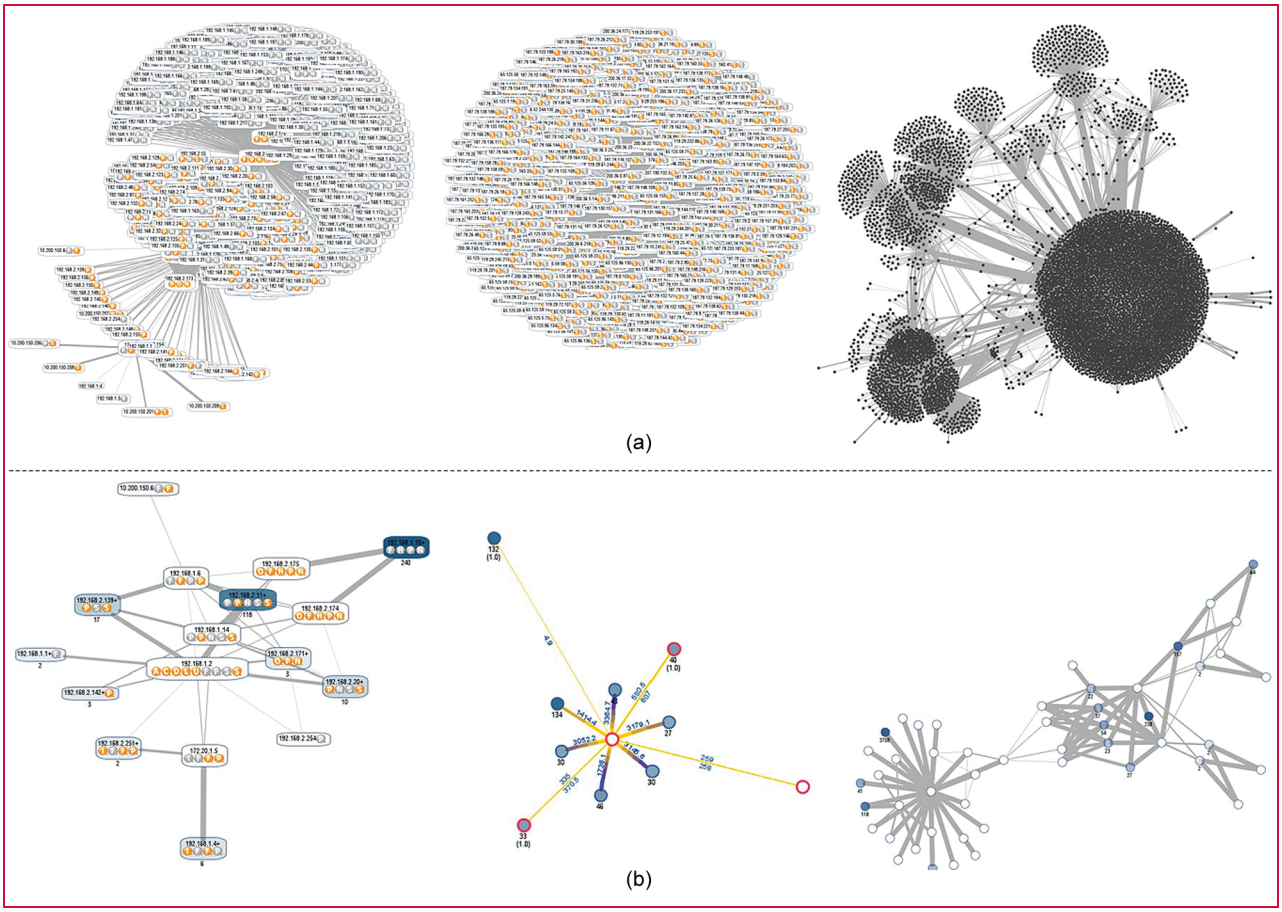

Figure 4

Network visualizations. Comparison of (a) original graph visualization and (b) compressed graph visualization from an enterprise network (left), a honeypot network (middle), and a large data center (right). Nodes of the left graph have anomaly icons, each representing one unique type of security violation, attack, firewall, or intrusion-detection system warning. The reduced size of compressed graphs reduces visualization complexity for better anomalies analysis of big graphs.

compressed graph, which exploits different groups connecting to and from the honeypot at different times. A temporally compressed graph [the middle graph of Figure 4(b)] is clearer, with the middle node being the honeypot within the in and out connections. A group of nodes in this graph, e.g., 187.79.2.4 + (206 hosts), show an abnormal traffic burst at the beginning of the investigation time period. Combined with the outstanding *F*-icons (File Transfer Protocol, or FTP) on the node, it suggests that the honeypot has been compromised early with weak Secure Shell (SSH) passwords (S-icons) and then distributes malware through numerous FTP channels. Further splitting and re-compressing this node group with both the "weighted" and "temporal" options reveals the suspected attackers [the red nodes in the middle graph of Figure 4(b)] with a significant traffic volume to the honeypot. Finally, the rightmost graph of Figure 4(a) shows a network flow graph from traces that are collected from a large corporate data center. For a "cleaner" visualization of the overall topology, a compressed graph [the rightmost graph of Figure 4(b)] is generated with nodes filtered down to 50 while still preserving the backbone topology of the network.

Conclusion

With the fast growth of modern networks in terms of size, dynamics, and complexity, how to understand, analyze, troubleshoot, and manage large networks has become increasingly important and challenging. Although there is no solution that fits all spectrums of networks, we discuss a few promising visualization methods for time-efficient detection and analysis of spatiotemporal anomalies in big graphs using several useful techniques in a unified manner. CSGs can be used to detect anomalies on the basis of membership changes rather than individual nodes or edges and are therefore more tolerant to the high dynamics of large

networks. TEMGs transform the network topology into directed trees, upon which a more efficient search can be performed to detect anomalous changes in network behaviors and routing topologies of a large dynamic network. Although the above approaches study only topological information, CGs and HDPGs can be used to examine the complex relationship of data attributes among graph nodes through a transformation in the dimensional spaces. Finally, TPCGs group the nodes with similar neighbor set into mega-nodes while preserving the topological structures of the original graphs, thus making graph visualization and analysis more scalable to larger networks.

Acknowledgments

We thank Dr. Aaron Striegel for his support at the University of Notre Dame, as well as colleagues at IBM Research - China.

References

1. J. Heer and D. Boyd, "Vizster: Visualizing online social networks," in *Proc. IEEE Symp. INFOVIS*, Minneapolis, MN, USA, Oct. 23–25, 2005, pp. 32–39.
2. C. C. Noble and D. J. Cook, "Graph-based anomaly detection," in *Proc. 9th ACM SIGKDD Int. Conf. KDD*, Washington, DC, USA, Aug. 24–27, 2003, pp. 631–636.
3. L. Akoglu, M. McGlohon, and C. Faloutsos, "OddBall: Spotting anomalies in weighted graphs," in *Proc. 14th Pac.-Asia Conf. Knowl. Discov. Data Mining*, Hyderabad, India, Jun. 21–24, 2010, pp. 410–421.
4. W. Eberle and L. Holder, "Anomaly detection in data represented as graphs," *Intell. Data Anal.*, vol. 11, no. 6, pp. 663–689, Dec. 2007.
5. D. Cook and L. Holder, "Graph-based data mining," *IEEE Intell. Syst.*, vol. 15, no. 2, pp. 32–41, Mar./Apr. 2000.
6. D. Chakrabarti and C. Faloutsos, "Graph mining: Laws, generators, and algorithms," *ACM Comput. Surv.*, vol. 38, no. 1, pp. 1–69, Mar. 2006.
7. L. Getoor and C. P. Diehl, "Link mining: A survey," *ACM SIGKDD Explor. Newslett.*, vol. 7, no. 2, pp. 3–12, Dec. 2005.
8. T. Falkowski, *Community Analysis in Dynamic Social Networks*. Goettingen, Germany: Sierke Verlag, 2009.
9. M. E. J. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.
10. S. T. Teoh, K.-L. Ma, S. F. Wu, and T. Jankun-Kelly, "Detecting flaws and intruders with visual data analysis," *IEEE Comput. Graph. Appl.*, vol. 24, no. 5, pp. 27–35, Sep./Oct. 2004.
11. E. Kandogan, "Visualizing multi-dimensional clusters, trends, and outliers using star coordinates," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, San Francisco, CA, USA, Aug. 26–29, 2001, pp. 107–116.
12. B. Woo, D. Vesset, C. W. Olofson, S. Conway, S. Feldman, and J. S. Bozman, "IDC's worldwide big data taxonomy," Int. Data Corp., Framingham, MA, USA, Oct. 27, 2011. [Online]. Available: http://www.idc.com/getdoc.jsp?containerId=231099
13. J. J. Thomas and K. A. Cook, *Illuminating the Path: The Research and Development Agenda for Visual Analytics*, Nat. Visual. Anal. Ctr., Pacific Northwest Nat. Lab., Richland, WA., 2005.
14. T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. van Wijk, J. Fekete, and D. Fellner, "Visual analysis of large graphs: State-of-the-art and future research challenges," *Comput. Graph. Forum*, vol. 30, no. 6, pp. 1719–1749, Sep. 2011.
15. P. C. Wong, P. Mackey, K. A. Cook, R. M. Rohrer, H. Foote, and M. Whiting, "A multi-level middle-out cross-zooming approach for large graph analytics," in *Proc. IEEE Symp. VAST*, Atlantic City, NJ, USA, Oct. 12/13, 2009, pp. 147–154.
16. P. Gajer and S. G. Kobourov, "GRIP: Graph drawing with intelligent placement," *J. Graph Algorithms Appl.*, vol. 6, no. 3, pp. 203–224, 2002.
17. Y. Hu, "Efficient and high quality force-directed graph drawing," *Math. J.*, vol. 10, no. 1, pp. 37–71, 2005.
18. Y. Koren, L. Carmel, and D. Harel, "ACE: A fast multiscale eigenvector computation for drawing huge graphs," in *Proc. IEEE Symp. INFOVIS*, Boston, MA, USA, Oct. 28/29, 2002, pp. 137–144.
19. D. Harel and Y. Koren, "Graph drawing by high-dimensional embedding," *J. Graph Algorithms Appl.*, vol. 8, no. 2, pp. 195–214, 2004.
20. D. Holten, "Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 741–748, Sep./Oct. 2006.
21. J. Lamping, R. Rao, and P. Pirolli, "A Focus+context technique based on hyperbolic geometry for visualizing large hierarchies," in *Proc. SIGCHI CHI*, Denver, CO, USA, May 7–11, 1995, pp. 401–408.
22. T. Munzner, "H3: Laying out large directed graphs in 3D hyperbolic space," in *Proc. IEEE Symp. INFOVIS*, Phoenix, AZ, USA, Oct. 20/21, 1997, pp. 2–10.
23. E. Gansner, Y. Koren, and S. North, "Topological fisheye views for visualizing large graphs," *IEEE Trans. Vis. Comput. Graphics*, vol. 11, no. 4, pp. 457–468, Jul./Aug. 2005.
24. L. Shi, N. Cao, S. Liu, W. Qian, L. Tan, G. Wang, J. Sun, and C.-Y. Lin, "HiMap: Adaptive visualization of large-scale online social networks," in *Proc. IEEE PACIFICVIS*, Beijing, China, Apr. 20–23, 2009, pp. 41–48.
25. J. Abello, F. van Ham, and N. Krishnan, "ASK-GraphView: A large scale graph visualization system," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 669–676, Sep./Oct. 2006.
26. G. Kumar and M. Garland, "Visual exploration of complex time-varying graphs," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 805–812, Sep./Oct. 2006.
27. D. Auber, Y. Chiricota, F. Jourdan, and G. Melancon, "Multiscale visualization of small world networks," in *Proc. 9th Annu. IEEE Conf. INFOVIS*, Seattle, WA, USA, Oct. 19–24, 2003, pp. 75–81.
28. A. Quigley and P. Eades, "FADE: Graph drawing, clustering and visual abstraction," in *Proc. 8th Int. Symp. GD*, Williamsburg, VA, USA, Sep. 20–23, 2000, pp. 197–210.
29. Y. Jia, J. Hoberock, M. Garland, and J. C. Hart, "On the visualization of social and other scale-free networks," *IEEE Trans. Vis. Comput. Graphics*, vol. 14, no. 6, pp. 1285–1292, Nov./Dec. 2008.
30. F. van Ham and M. Wattenberg, "Centrality based visualization of small world graphs," *Comput. Graph. Forum*, vol. 27, no. 3, pp. 975–982, May 2008.
31. M. Wattenberg, "Visual exploration of multivariate graphs," in *Proc. SIGCHI CHI*, Montreal, QC, Canada, Apr. 22–27, 2006, pp. 811–819.
32. R. C. Read and D. G. Corneil, "The graph isomorphism disease," *J. Graph Theory*, vol. 1, no. 4, pp. 339–363, Winter 1977.
33. H. Bunke, P. J. Dickinson, M. Kraetzl, and W. D. Wallis, *A Graph-Theoretic Approach to Enterprise Network Dynamics—Series: Progress in Computer Science and Applied Logic (PCS)*. Boston, MA, USA: Birkhauser, 2007.
34. D. Archambault, "Structural differences between two graphs through hierarchies," in *Proc. Graph. Interface*, Kelowna, BC, Canada, May 25–27, 2009, pp. 87–94.
35. J. Abello, T. Eliassi-Rad, and N. Devanur, "Detecting novel discrepancies in communication networks," in *Proc. IEEE ICDM*, Sydney, Australia, Dec. 14–17, 2010, pp. 8–17.
36. A. Inselberg, "The plane with parallel coordinates," *Vis. Comput.*, vol. 1, no. 2, pp. 69–91, Aug. 1985.
37. C. Tominski, J. Abello, and H. Schumann, "Axes-based visualizations with radial layouts," in *Proc. ACM Symp. Appl. Comput.*, Nicosia, Cyprus, Mar. 14–17, 2004, pp. 1242–1247.
38. W. Aigner, S. Miksch, H. Schumann, and C. Tominski, *Visualization of Time-Oriented Data*. Berlin, Germany: Springer-Verlag, 2011.
39. P. Bak, F. Mansmann, H. Janetzko, and D. A. Keim, "Spatiotemporal analysis of sensor logs using growth ring

maps," *IEEE Trans. Vis. Comput. Graphics*, vol. 15, no. 6, pp. 913–920, Nov./Dec. 2009.
40. J. V. Carlis and J. A. Konstan, "Interactive visualization of serial periodic data," in *Proc. 11th Annu. ACM Symp. User Interface Softw. Technol.*, San Francisco, CA, USA, Nov. 1–4, 1998, pp. 29–38.
41. J. R. Goodall, W. G. Lutters, P. Rheingans, and A. Komlodi, "Focusing on context in network traffic analysis," *IEEE Comput. Graph. Appl.*, vol. 26, no. 2, pp. 72–80, Mar./Apr. 2006.
42. G. Conti, *Rumint–Open Source Network and Security Visualization Tool*. [Online]. Available: http://www.rumint.org/
43. P. Minarik and T. Dymacek, "NetFlow data visualization based on graphs," in *Proc. 5th Int. Workshop VizSec*, Cambridge, MA, USA, Sep. 15, 2008, pp. 144–151.
44. F. Fischer, F. Mansmann, D. A. Keim, S. Pietzko, and M. Waldvogel, "Large-scale network monitoring for visual analysis of attacks," in *Proc. 5th Int. Workshop VizSec*, Cambridge, MA, USA, Sep. 15, 2008, pp. 111–118.
45. D. Phan, J. Gerth, M. Lee, A. Paepcke, and T. Winograd, "Visual analysis of network flow data with timelines and event plots," in *Proc. Workshop VizSeC*, Sacramento, CA, USA, Oct. 29, 2007, pp. 85–99.
46. N. Senechal, S. Hong, and P. Eades, "Display of Sensor Networks: A Feasibility Study," Daintree Netw., Los Altos, CA, USA, Jul. 2006.
47. L. Shi, C. Wang, and Z. Wen, "Dynamic network visualization in 1.5D," in *Proc. IEEE PacificVis*, Hong Kong, Mar. 1–4, 2011, pp. 179–186.
48. P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: Analyzing and visualizing network data," *Genome Res.*, vol. 13, no. 11, pp. 2498–2504, Nov. 2003.
49. W. de Nooy, A. Mrvar, and V. Batagelj, *Exploratory Social Network Analysis with Pajek*, Cambridge, U.K.: Cambridge Univ. Press, Mar. 2005.
50. Q. Liao and A. Striegel, "Intelligent network management using graph differential anomaly visualization," in *Proc. IEEE/IFIP NOMS*, Maui, HI, USA, Apr. 16–20, 2012, pp. 1008–1014.
51. W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Amer. Stat. Assoc.* , vol. 66, no. 336, pp. 846–850, Dec. 1971.
52. T. F. Cox and M. A. A. Cox, *Multidimensional Scaling*, 2nd ed. London, U.K.: Chapman & Hall, 2000.
53. P. Pons and M. Latapy, "Computing communities in large networks using random walks," *J. Graph Algorithms Appl.*, vol. 10, no. 2, pp. 191–218, Apr. 2006.
54. D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: The condor experience," *Concurrency Comput., Pract. Exp.*, vol. 17, no. 2–4, pp. 323–356, Feb.–Apr. 2005.
55. L. Shi, Q. Liao, Y. He, R. Li, A. Striegel, and Z. Su, "SAVE: Sensor anomaly visualization engine," in *Proc. IEEE Conf. VAST*, Providence, RI, USA, Oct. 23–28, 2011, pp. 201–210.
56. D. A. Keim, "Information visualization and visual data mining," *IEEE Trans. Vis. Comput. Graphics*, vol. 8, no. 1, pp. 1–8, Jan. 2002.
57. Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, J. Zhao, and X.-Y. Li, "Does wireless sensor network scale? A measurement study on GreenOrbs," in *Proc. IEEE INFOCOM*, Shanghai, China, Apr. 10–15, 2011, pp. 873–881.
58. S. M. Stigler, "Francis Galton's account of the invention of correlation," *Stat. Sci.*, vol. 4, no. 2, pp. 73–79, May 1989.
59. X. Miao, K. Liu, Y. He, Y. Liu, and D. Papadias, "Agnostic diagnosis: Discovering silent failures in wireless sensor networks," in *Proc. IEEE INFOCOM*, Shanghai, China, Apr. 10–15, 2011, pp. 1548–1556.
60. P. Hoffman, G. Grinstein, K. Marx, I. Grosse, and E. Stanley, "DNA visual and analytic data mining," in *Proc. 8th Conf. Vis.*, Phoenix, AZ, USA, Oct. 19–24, 1997, pp. 437–441.
61. P. Hoffman, G. Grinstein, and D. Pinkney, "Dimensional anchors: A graphic primitive for multidimensional multivariate information visualizations," in *Proc. Workshop New Paradigms Inf. Vis. Manipul. Conjunct. 8th ACM Int. Conf. Inf. Knowl. Manage.*, Kansas City, MO, USA, Nov. 2–6, 1999, pp. 9–16.
62. D. Lalanne, E. Bertini, P. Hertzog, and P. Bados, "Visual analysis of corporate network intelligence: Abstracting and reasoning on yesterdays for acting today," in *Proc. 4th Int. Workshop VizSec*, Sacramento, CA, USA, Oct. 29, 2007, pp. 115–130.
63. T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Inf. Process. Lett.*, vol. 31, no. 1, pp. 7–15, Apr. 1989.
64. E. R. Gansner, Y. Koren, and S. North, "Graph drawing by stress majorization," in *Proc. 12th Int. Conf. Graph Drawing*, New York, NY, USA, Sep. 29/Oct. 2, 2004, pp. 239–250.
65. A. Godiyal, J. Hoberock, M. Garland, and J. C. Hart, "Rapid multipole graph drawing on the GPU," in *Proc. 16th Int. Symp. Graph Drawing*, Heraklion, Greece, Sep. 21–24, 2008, pp. 90–101.
66. A. Sperotto, R. Sadre, F. van Vliet, and A. Pras, "A labeled data set for flow-based intrusion detection," in *Proc. 9th IEEE Int. Workshop IPOM*, Venice, Italy, Oct. 29/30, 2009, pp. 39–50.

Received July 13, 2012; accepted for publication August 9, 2012

Qi Liao *Department of Computer Science, Central Michigan University, Mount Pleasant, MI 48859 USA (liao1q@cmich.edu).* Dr. Liao is an assistant professor of Computer Science at Central Michigan University (CMU). He graduated with a B.S. degree and Departmental Distinction in computer science from Hartwick College, New York, with a minor concentration in mathematics in 2005, and a M.S. and Ph.D. degrees in computer science and engineering (CSE) from the University of Notre Dame, Indiana, in 2008 and 2011, respectively. Prior to joining CMU, he worked as a research intern in the Visual Analytics Group at IBM Research in Beijing in 2010. Dr. Liao's research interests include computer security, anomaly detection, visual analytics, graph data mining, and economics of cybersecurity. His research was recognized by awards such as the best paper award from USENIX at the 22nd Large Installation System Administration Conference (LISA'08), the second prize winner at the 3rd National Security Innovation Competition (NSIC'09), the Center for Research Computing Award for Computational Sciences and Visualization in 2011, and the IEEE Visual Analytics Science and Technology (VAST) Mini-Challenge 2 Award in 2012. Dr. Liao is a member of the Institute of Electrical and Electronics Engineers, Kappa Mu Epsilon (national mathematics honor society), Upsilon Pi Epsilon (international honor society for the computing and information disciplines), and Tau Beta Pi (national engineering honor society).

Lei Shi *State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Haidian District, Beijing 100190 China (shil@ios.ac.cn).* Dr. Shi is an Associate Research Professor at the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences. Previously, he was a Research Staff Member and Research Manager at IBM Research - China, working on information visualization and visual analytics. He holds B.S. (2003), M.S. (2006), and Ph.D. (2008) degrees from the Department of Computer Science and Technology at Tsinghua University. His research interests span information visualization, visual analytics, network science, and networked systems. He has published more than 30 papers in refereed conferences and journals. He is the recipient of an IBM Research Accomplishment Award on "Visual Analytics," the IEEE Visual Analytics Science and Technology (VAST) Challenge Award in 2010, and the IEEE VAST Mini-Challenge 2 Award in 2012.

Chen Wang *IBM Research Division, China Research Laboratory, Haidian District, Beijing 100193 China (wangcwc@cn.ibm.com).* Mr. Wang received his B.S. and M.S. degrees in computer science from Fudan University, Shanghai, China, in 2003 and 2006, respectively. He is currently a Research Staff Member and Research Manager at IBM Research - China, and his areas of research are in machine learning, data mining, data management, and analytic applications across industries. He is a member of the IEEE Computer Society.