SAVE: Sensor Anomaly Visualization Engine

Lei Shi Qi Liao Yuan He Rui Li Aaron Striegel Zhong Su*

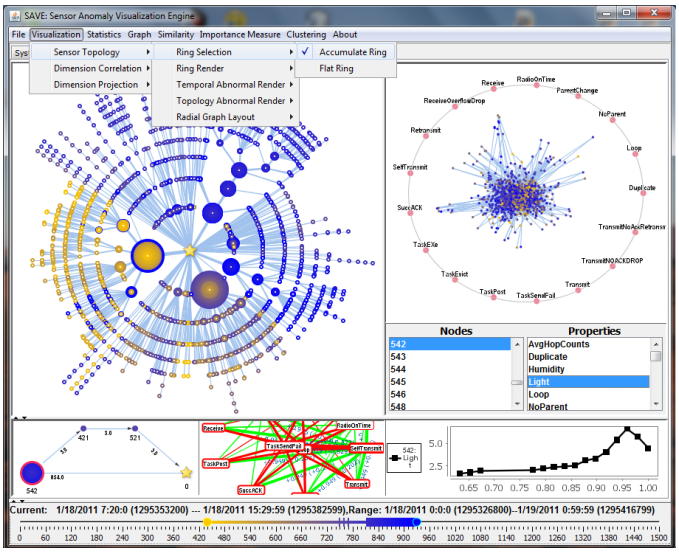


Figure 1: SAVE user interface visualizing sensor network topology (top-left, bottom-left), high-dimensional sensor measurements and statuses (top-right), dimension correlations (middle of bottom) and the dimension temporal trends (bottom-right). Various visual settings and analytic functionalities can be accessed from the menu.

Abstract

Diagnosing a large-scale sensor network is a crucial but challenging task. Particular challenges include the resource and bandwidth constraints on sensor nodes, the spatiotemporally dynamic network behaviors, and the lack of accurate models to understand such behaviors in a hostile environment. In this paper, we present the Sensor Anomaly Visualization Engine (SAVE), a system that fully leverages the power of both visualization and anomaly detection analytics to guide the user to quickly and accurately diagnose sensor network failures and faults. SAVE combines customized visualizations over separate sensor data facets as multiple coordinated views. Temporal expansion model, correlation graph and dynamic projection views are proposed to effectively interpret the topological, correlational and dimensional sensor data dynamics and their anomalies. Through a case study with real-world sensor network system and administrators, we demonstrate that SAVE is able to help better locate the system problem and further identify the root cause of major sensor network failure scenarios.

*Lei Shi and Zhong Su are with IBM Research - China, E-mail:{shllsh,suzhong}@cn.ibm.com

Qi Liao and Aaron Striegel are with Computer Science and Engineering Department, University of Notre Dame, USA, E-mail: {qliao,striegel}@nd.edu

Yuan He is with Computer Science and Engineering Department, Hong Kong University of Science and Technology, HK, E-mail:heyuan@cse.ust.hk

Rui Li is with Computer Science and Technology Department, Xi'an Jiao Tong University, E-mail:rli@mail.xjtu.edu.cn

IEEE Symposium on Visual Analytics Science and Technology
October 23 - 28, Providence, RI, USA
978-1-4673-0014-8/11/$26.00 ©2011 IEEE

1 Introduction

With the recent surge of sensor technology, networks have expanded from the traditional computer-centric networks, e.g. the Internet, to real world networks centered around physical objects, namely the Internet of Things (IoT) [1]. A sensor network consists of a number of sensor nodes, each of which has the capability of sensing, computing, and communication. Sensor networks are already playing a key role in numerous emerging applications, such as smart grid, logistics, traffic control and health care.

Driven by the need for sustainable operations of the network, the diagnosis of sensor network problems has become a crucial task. For example, consider the case of a sensor network deployed in the forest for the purpose of measuring carbon emissions and conducting ecological surveillance, there is a need to know the percentage of abnormal sensors to guarantee an accurate overall measurement. By detecting significant changes in routing paths, one can use that as a trigger to look into the network status, identify broken wireless links or suddenly failed sensor nodes, and take immediate measures to repair problems and maintain system integrity.

However, the diagnosis of sensor networks can be tremendously challenging when faced with the dynamics in full scale deployments. First, sensor networks consist of numerous low-end embedded computing devices and are often resource-constrained. The resource constraints of sensor networks tend to include limited capacity of computing, storage, and bandwidth. Furthermore, most remotely deployed sensor networks are powered by non-rechargeable batteries introducing additional energy constraints. Second, many sensor networks are deployed in outdoor or even hostile environments. Unlike the Internet and traditional networks, various environmental factors could have significant impacts on the performance and reliability of sensor networks. For example, changes in temperature, humidity, wind, rain, and physical damages by human or wild animals, can cause performance degradation or even system failures of the sensor networks. Third, the wireless nature of most sensor networks is extremely likely to be lossy, thus making perfect

data collection impossible or at least highly unlikely.

Most existing diagnostic approaches for the Internet or other traditional networks (e.g. SNMP) cannot be directly applied to sensor networks. These approaches are often resource-consuming and demand intensive data collection. On the other hand, the current algorithms to diagnose sensor networks from the collected data mainly rely on certain inference models which link symptoms to the underlying root causes [2]. These inference models depend on the evidence-based sensor data fault taxonomy to characterize the symptoms. However, due to the resource constraints and the existence of salient failures, such inference models can only focus on a portion of symptoms, restricting their adaptability and scalability in real-world usages.

In this paper, we present the Sensor Anomaly Visualization Engine (SAVE) (Figure 1), an integrated system that tackles the sensor network diagnosis problem by using visual analytics technologies. Compared to the existing suite of algorithmic approaches, SAVE has the following advantages that include:

- The visualization of sensor network data from multiple perspectives, at various scales of size and time, helps the administrator collect evidence in a more comprehensive and consolidated manner;
- The advanced visual interface provides a more intuitive display of data, which by its nature is more suitable for human diagnosis of large-scale data intensive systems;
- The visual analytics solution combines human knowledge with algorithmic results to work better in diagnosing salient sensor network failures, whose symptoms and root causes are previously unknown;

As noted earlier, visual analytics for sensor networks is a distinctly non-trivial task. The high-dimensional and dynamic nature of the sensor data, the unpredictable network behavior, and the error-prone transmissions and operations all bring great challenges. This paper offers the following contributions to help address these challenges.

First, we propose the Temporal Expansion Model (TEM) graph to track the evolving sensor network topology. TEM leverages the key feature of a typical sensor network, i.e. all the sensor nodes (or those in one autonomous system) only send packets to a central sink node for information fusion. The basic idea is to split each physical sensor node into multiple logical nodes according to their separate routing paths to the sink node. Therefore, the resulting topology graph is essentially a directed tree, enabling more friendly visualization and human navigation. Simultaneously, temporal changes to the network are surfaced to the graph, providing input for further analytics over network dynamics. We also introduce an overview+detail visualization design to the TEM graph to alleviate its limitation in identifying physical connections. A clear delivery path from each node to the sink is offered to the user as a detailed view.

Second, we introduce a correlation graph to monitor temporal correlation patterns and detect anomalies between numerical sensor data dimensions. Third, a novel dimension projection view is designed to map the high-dimensional sensor counters/readings into visible patterns and render the temporal dynamics.

We have deployed SAVE into a real-world large-scale Wireless Sensor Network (WSN) system – GreenOrbs [3]. Based on long-term iterative studies with domain experts there, we have elaborated a suite of interactions into SAVE to help the network operators better diagnose network failures from a comprehensive perspective. We further demonstrate through the link-failure diagnosis case study that SAVE is effective in both identifying salient failures and providing the critical guidance for the user to drill-down to root causes of the anomalies. In most cases, SAVE indeed saves a great amount of time for the operators in completing their tasks.

2 Related Work

Traditionally, analysis of sensor networks have been based on either general (ns-2) or in-domain (TOSSIM [4]) network simulators. For example, MOTE_VIEW [5] tracks the health and status of nodes in small simulated networks as well as visualizes the sensor readings of humans and vehicles movements through a simulated hazardous area. NetTopo [6] provides both simulation and visualization functions to test and validate algorithms in WSNs. In particular, TOSSIM, a discrete event simulator for TinyOS sensor networks, includes TinyViz [4], a Java visualization and actuation environment that interacts with TOSSIM through its network protocol. While visualization tools like TinyViz allow users to select and manipulate simulated nodes through a GUI, the tools control and visualize the results of network simulations but not the experimental routing data from an actual deployment of a large scale sensor network. The result is that such tools tend to miss network anomalies that occur unpredictably in real deployments.

Products such as Sensor Network Analyzer (SNA) [7] are available as commercial solution for developing, decoding, debugging and deploying wireless embedded networks. They supports various protocols including IEEE 802.15.4 and ZigBee. SNA includes filtering, labeling and color-coding to make it easy to locate packets of interest and measure performance. Surge Network Viewer [8] is an application that comes standard in the TinyOS Tools distribution and is useful for monitoring a sensor network and analyzing mesh network performance. In addition, SpyGlass [9] visualizes the network topology and the state of sensors via basic graph drawing and node labeling. In the Sensor Network Analysis and Management Platform (SNAMP) [10], data emitted by individual sensor nodes is collected by a multi-sniffer data collation network and passed to a flexible multi-view visualization mechanism. In a sense, the debugging functions included in SNAMP "visualize" the development of WSNs applications. While products like SNA allow users to visualize packets and fields at the byte level, it lacks intelligent analytic capability to detect and analyze the abnormalities of wireless sensor nodes. Key questions focused in this study, e.g., what are the abnormal changes in terms of both spatial and temporal nature, have not been addressed in previous studies.

Sensor network data have been known for many types of faults [2], e.g., the most common faults observed in a sensor network include outliers, spikes, stuck-at and noise. Other than these predefined faults, many times the sensor networks themselves exhibit silent failures [11] that are unknown beforehand. To understand general network behaviors, diagnosis methods have been based on the network itself and its graph properties, such as degree distributions, subgraph isomorphism [12] and graph edit distance [13]. While there are techniques dealing with time series data visualizations, e.g. the GrowthRingMaps [14] and SpiralGraph [15] for temporal patterns in serial periodical data, very few of them are designed specially for the diagnosis of performance issues in sensor networks and for the easy exploration of root causes for which there may be no prior established knowledge.

Techniques exist for visual analysis of network graphs, spatiotemporal, and multidimensional data sets, such as Parallel coordinates [16], star coordinates [17] or RadViz [18], TimeWheel and MultiComb [19]. Rather than duplicate snapshot graphs at different timestamps, the spatiotemporal sensor graph (STSG) model [20] has been proposed to discover spatiotemporal patterns by constructing one super graph with the properties of edges and nodes containing the time series of measurement data. Graph-based wavelets coefficients [21, 22], which aggregate or difference adjacent nodes/edges in the neighborhoods of network graphs, are proposed to analyze and understand the spatial and temporal behav-

Figure 2: The bird's-eye view of the GreenOrbs deployment in Zhejiang Forestry University. The latest deployment includes 500 physical sensor nodes in total. A portion of the sensor nodes have been in continuous operation for over one year.

ior of internet traffic anomalies. Interactive visualization has been shown to be useful to fault diagnosis in various networks including enterprise networks [23]. Our study involves some problems of graph analytics which is among the most important research areas within the visual analytics community [24, 25]. In the setting of sensor networks, we apply as well as develop novel views for spatiotemporal anomalies visual analysis, e.g., TEM graph visualization for sensor network routing topology, correlation-based projection views for high-dimensional sensor properties (readings vs. counters), and a differential visualization framework for sensor routing graphs and node-correlation graphs.

3 System Overview

SAVE is a visual analytics system for realtime and playback data diagnosis of real-world sensor networks. In this section, we first introduce the underlying GreenOrbs deployment and its data collection mechanism. Second, we explain the building blocks and working process of the SAVE system.

3.1 Sensor Deployment

GreenOrbs [3] is a long-term large-scale wireless sensor network system in the forest. The system realizes all-year-round forest ecological surveillance and supports various forestry applications, e.g. canopy closure estimation, fire risk prediction and carbon sequestration and emission. Figure 2 shows the bird's-eye view of the GreenOrbs deployment.

In hardware, GreenOrbs employs the TelosB mote with a MSP430 processor and CC2420 transceiver. On each sensor node, a 48KB program flash memory and a 1024KB measurement serial flash is installed. An elaborate design of the enclosure for each sensor node is introduced (also shown in Figure 2) to protect the nodes from being corrupted by possible inclement weather or destruction. A typical GreenOrbs node in the current deployment is equipped with four sensors, providing five types of sensor readings, namely temperature, humidity, illumination, MCU-internal voltage, and the concentration of carbon dioxide in the air. In software, GreenOrbs employs the Collection Tree Protocol (CTP) [26] to collect the data in a distributed multi-hop manner. The transmission of the configuration packets from the sink to the other sensor nodes are delivered using the DRIP protocol [27].

3.2 Data Measurement and Collection

GreenOrbs sensor nodes work in a periodic manner. The operational period is set to 10 minutes and can be flexibly configured by external control commands issued from the sink. In each operational period, a sensor node collects four categories of data, namely sensor readings, routing path, wireless link status, and the networking/system statistics. This data is encoded into three data packets and sent to a central sink node via multi-hop routing through CTP.

The first category of data is sensor readings. Each sensor node invokes the read interfaces of corresponding sensor components periodically to collect the readings. The second category is the routing path, which records the IDs of all the nodes that appear in the multi-hop routing path from the source node to the sink. This is realized by piggybacking the forwarding node's ID at each hop during the packet relaying process with CTP. The third category is link status, which includes the Received Signal Strength Index (RSSI), link quality index (LQI) and ETX value of all the neighboring nodes. The fourth category is a large collection of statistics information on each sensor node, including the cumulative time of radio power on, the cumulative number of received/transmitted/dropped(due to receive buffer overflow, transmit queue overflow or timeouts)/not-ACKed/retransmitted/duplicate packets and the cumulative number of parent changes and no parent events with CTP.

GreenOrbs began continuous operation in late 2010. In this work, most of the visualizations for SAVE are built on a four-day trace data set collected in early 2011 when the entire system had converged to a stable mode. Statistically, this trace contained 349 physical nodes with 140k reports for each kind on sensor reading, sensor mote status update and routing dynamics. Provided that the sensor data format stays unchanged, the SAVE system could extend its usage to data sets over longer periods of time or even to acting on real-time data. The stability issue of the proposed visualizations are further discussed in Section 5.

3.3 SAVE System Framework

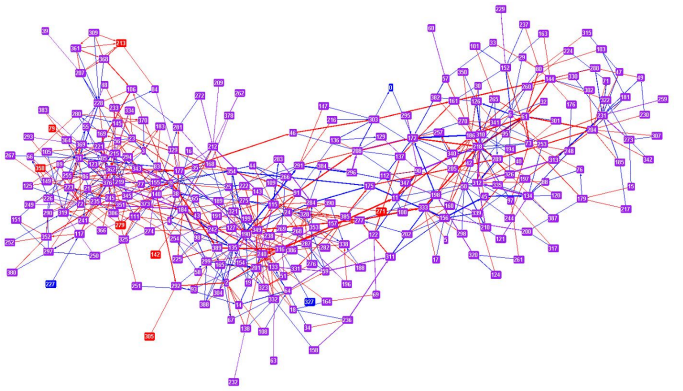
From a system perspective, SAVE is designed as a three-stage pipeline, encompassing data preprocessing, anomaly detection and multi-view visualization stages. In the first stage, the multi-facet raw data is preprocessed and the temporal network topology and dimension correlations are built from distributed data pieces. Next, anomalies such as measurement outliers and correlation changes are computed and cached online. Finally, at the core stage, the data facets and the prepared analytics are brought together and consumed by the visualization components integrated with SAVE.

From the user's perspective, they access SAVE from selecting a collection of data traces in the offline analysis mode. SAVE then presents the user an overview to the whole data set with multiple coordinated views interpreting different facets of the trace data, including topology, correlation and dimension projection views. The user can navigate in each view with standard data brushing and filtering interactions. The correlations among data facets are illustrated through coordinated updating in all the views. A time range control is also provided to allow the user specify the time period of interest in the data. In this way, the user is able to drill-down to the interesting data portion jointly from temporal, spatial and dimensional perspectives so that analysis over the details can be further conducted with standard tools such as trend lines.

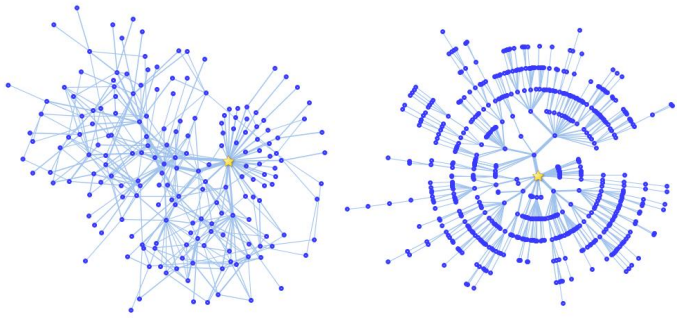
4 Data Engineering and Anomaly Detection

4.1 Data Preprocess

The raw data reported by the sensing nodes, i.e. link quality, routing, sensing, and status data, are first parsed and fed into the SAVE visualization tool for analysis. For link quality data, each sensor



(a) Geographical view



(b) Force-directed view (c) TEM graph view

Figure 3: A comparison of sensor network topology visualizations.

node would normally have 16 neighbors with different link quality metrics (LQI, RSSI, ETX). A *link graph* is generated and updated when the link quality changes. For routing data, each path up to 10 hops is parsed to record the entire routing path from the source to the sink (node ID 0). Each node along the path is added into the *routing graph* data structure, which is stored in memory and later transformed into GraphML format that can be readily read by most graph visualization libraries.

For sensing data, the actual sensor readings reported are time-stamped and saved as the properties of each sensor. Each property is a four-tuple, i.e., [nodeID, time, propertyName, propertyValue]. The available sensing properties include *temperature*, *humidity*, *light*, *voltage*, and CO_2. The available sensor status properties are a group of predefined metrics for diagnosis purposes, e.g., *RadioOn*, *ReceiveOverflowDrop*, *Retransmit*, *ParentChange*, etc. Indices on the four-tuple properties using hashtable data structure allow quick searching, sorting and displaying updated statistical charts/graphs in response to the user interactions, e.g., clicking nodes in the graph, selecting properties and dragging the time sliders for the investigation.

4.2 Topology Analytics

The dynamics of routing topology is the key indicator to the performance of a sensor network. By nature, this topology is a large-scale time-varying graph due to the instability of ad-hoc communications and routing scheme employed. Traditionally, it is hard to compose an analytics-friendly visualization for such a graph. The most promising approach aggregates the temporal graphs together and draws an accumulated connection picture [28], e.g. for community visualization through a social graph. However, this approach fails in our case as the connection changes here are extremely random. One sensor node delivering a packet to another node in one hop could need multiple hops just in the next time for the same destination. The resulting graph drawn in normal methods is quite messy either with geographical or logical layouts, as shown in Figure 3(a)(b).

We propose the Temporal Expansion Model (TEM) graph in this paper to prepare a more intuitive graph for the following visualization stage. TEM leverages the key feature of the sensor network studied here – all the sensor nodes only send packets to the central sink node for information fusion. The basic idea is to split one physical sensor node into multiple logical nodes according to the separate routing paths to the sink. The advantages of TEM are two-fold: first, the graphs generated are directed trees, much better for visualization and navigation; second, temporal changes to the network are surfaced to the graph, providing input for further analytics.

The TEM graph still follows the node-edge paradigm of general graphs. However, the node here is defined differently by using the path N from the original node (S) to the sink node (R):

$$N = [(S, p_1, p_2, ..., R), T] \tag{1}$$

where p_i is a node on the path. The unique node ID of N is encoded by concatenating its paths together. Two nodes are identical only if their paths are the same. The physical ID S is also kept for each node to link to its original physical node. Another difference in TEM graph is that it keeps track of the event time series (T) of each node, rather than only maintaining numerical accumulated weight. In our case, the events are packet sendings from S.

We further introduce the TEM processing in the sensor network case for its better understanding. The raw topology data input are the packet delivery paths to the sink node from all distributed sensors. A m-hop path from source node S at time t is represented as

$$\Gamma(S,t) = (S, p_1, p_2, ..., p_{m-1}, R) \tag{2}$$

Next we construct the TEM graph from the path data in two steps.

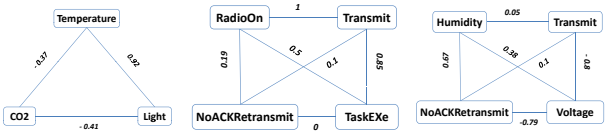
- **Path-node set generation:** We first split each input path into multiple nodes and add them to the TEM node set. For the m-hop path $\Gamma(S,t)$, $m+1$ path nodes are generated: N_1 = $(S, p_1, p_2, ..., R)$, N_2 = $(p_1, p_2, ..., R)$, ..., N_m = (n_{m-1}, R), R. Time t is added to the time series associated with these $m+1$ nodes. In a flat variation of TEM (F-TEM), we only add time t to the time series of N_1, as in some cases, we are more concerned with the time series when the nodes behave as source.

- **Temporal graph building:** In the second step, 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), ..., (N_{m-1}, N_m), (N_m, R). We do not maintain time series for each edge, since the resulting graph is essentially a tree, the time series of the source node of an edge will indicate the edge dynamics.

A static example of the generated TEM graph is shown in Figure 3c, reducing most visual clutter compared with its original drawing in Figure 3b. The temporal dynamics of TEM is visualized through ring-based node rendering, which is detailed in Section 5.1.

We incorporate some basic analytics over the dynamics of the TEM graph. 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 change points over the time series of each node. Given the packet delivery time series $T = (t_1, t_2, ..., t_k)$ on a sensor node S, we find abnormal time points as below:

Temporal detection: We partition the life cycle of node S into multiple fix-length bins according to the preset sensing intervals, e.g. 10 minute in which the sensors wake up regularly and report status information. Each value within the time series of a node S is fit into its particular bin. Finally, the number of values accumulated



(a) Sensor readings (b) Sensor counters (c) Mixed

Figure 4: Hypothetical examples of correlation graphs. The nodes are sensor node properties and the edges represent the correlation between each pair of properties.

in each bin is recorded as $(v_1, v_2, ..., v_m)$. We then compute the changes of these bin sizes as $(\Delta_1, \Delta_2, ..., \Delta_m)$, where

$$\Delta_i = v_i - v_{i-1} \quad (v_{-1} = 0) \tag{3}$$

Then Δ_i represents the changes of node S during the time period of bin i. This indicator reflects the topology changes centric to node S in the general TEM graph, while reporting more on the packet delivery changes of node S when applied to the F-TEM graph.

With the splitting mechanism, the TEM graph will expand multiple times from the original physical sensor graph, which may lead to scalability issues. There are three key factors affecting the size of a TEM graph: the duration of the temporal graph, the size of the original physical network and the fluctuation of the routing protocol. However, we show that none of these factors will grow unboundedly, hence the overall TEM graph size is controlled as a result. First, users are more interested to the short-term dynamics of a sensor topology which give evidences to potential sensor faults. Due to the ad-hoc nature of the sensor network, the long-term topology changes do not possess significant temporal trends. A one-day window would be enough for user to locate the fine-grained topology anomalies. Second, as the physical sensor network grows, hierarchical routing schemes are generally employed where the entire network is partitioned into several autonomous systems operating independently. Each system for which a TEM graph is drawn, is composed of up to several hundreds of physical nodes, similar to our current data size. Third, the routing fluctuation, as shown in our experiments are not as random as we may expect. We have tested 6 whole-day data sets from GreenOrbs, with an average physical node size of 337. The generated TEM graph size reaches 4534 in average. The mean graph layout time is 0.11s and the mean graph rendering time is 2.79s, both linear to the TEM graph size. With these statistics, we argue here the TEM approach is feasible for most sensor topology scenarios we considered in this paper as the resulting graph will not grow indefinitely and the overhead is shown to be limited.

4.3 Correlation Graphs

Another temporal dynamic occurs from the various dimensions of sensor data. Naturally, the values of each dimension change over time. One question is to ask how one dimension (or property) changes in relation to another dimension (or property). For example, should the number of packets in transmission increase in the same proportion of voltage decrease of sensor nodes? What are the relationships between those sensor readings? This relationship or correlation can be used as a metric for anomaly detection and analysis.

To this end, two time-series data sets of property value vectors (i.e., one for sensor readings and one for sensor counters) are extracted from the raw reported sensor data in real time based on the selection of the node, start and end timestamps in the visual analytic tool. SAVE computes the correlation scores according to the Pearson's product-moment coefficient [29], i.e., $Correlation(p_1, p_2) =$

$$\frac{|p_1| \cdot \sum_{i=1}^{|p_1|} p_{1i} \cdot p_{2i} - \sum_{i=1}^{|p_1|} p_{1i} \cdot \sum_{i=1}^{|p_2|} p_{2i}}{\sqrt{|p_1| \cdot \sum_{i=1}^{|p_1|} {p_1}^2 - \left(\sum_{i=1}^{|p_1|} p_1\right)^2} \cdot \sqrt{|p_2| \cdot \sum_{i=1}^{|p_2|} {p_2}^2 - \left(\sum_{i=1}^{|p_2|} p_2\right)^2}} \tag{4}$$

where p_1 and p_2 are the property vectors, whose lengths equal the number of timestamps. For example, if the investigation window is set as two hours in the visualization tool, the property vector size will be twelve if each sensor reports every ten minutes. A correlation matrix can then be computed for each pair of properties. Naturally, a correlation graph [11] associated with the correlation matrix can be constructed and visualized.

Figure 4 shows the concept of correlation graphs (CGs), in which each node represents one dimension of data associated with a sensor node, and the weights of edges represent how much pairwise properties are correlated. Specifically, Figure 4a shows a CG with only sensor readings, e.g., *temperature* and *light* are very much positively correlated since intuitively the *temperature* rises in the date and drops at night. CO_2 is negatively related with daylight as the forest emits oxygen and consumes CO_2 under sunshine for photosynthesis. Figure 4b shows a CG with only senor network routing counters, e.g., *RadioOn*, *Transmit* and *NoACKRetransmit* are all positively correlated. In addition, Figure 4c shows a mixed scenario with both sensor readings and counters, e.g., high humidity may cause some problem in packet loss and may be correlated to some of the retransmission. Transmitting / relaying packets or retransmitting packets are negatively correlated with sensor voltage due to increased load and power consumption.

4.4 Dimensional Outliers and Dynamics

The data reported from sensor nodes is also multi-dimensional. The dimensions of sensor data in the deployment can be as high as 30 dimensions ranging from normal sensor readings (e.g., *Temperature*, *Light*, *Humidity*, *Voltage*, etc) to network routing counters (e.g., *RadioOnTime*, *Transmit*, *Receive*, *Retransmit*, *SuccACK*, etc). Since the wireless sensors deployed in a wild environment are quite dynamic, how to detect and analyze the anomalies in high-dimensional space becomes a challenging task. Specifically, we want to know both the spatial and temporal anomalies of high-dimensional sensor nodes:

- *Spatial anomalies*: How is each individual sensor node different from others in terms of the dimensions of data?
- *Temporal anomalies*: How does each individual sensor node evolve from its status of previous time?

To solve the problem, we first divide the dimensions of sensor data into two categories of properties, i.e., sensor readings and sensor counters. Each sensor node reports data at an interval of approximately ten minutes and is uniquely identified by *nodeID_time*. Therefore, each sensor node at each reporting time will then have two property vectors, whose elements represent the values for each dimension.

Each high-dimensional sensor node *nodeID_time* is mapped as a data point onto a 2D space using a concept similar to Star Coordinates [17] or RadViz [18]. The projection uses an improved spring-force model detailed in Section 5.3. As an illustration, Figure 5 shows a unit cycle with function $x^2 + y^2 = 1$. The circumference is equally divided by the number of dimensions, where the dimensions represent either the total types of sensor readings or total types of sensor counters. The figure shows an example of temporal dynamic of nodes 2 and 35 that change at two different timeslots (from t to t'). In this view, not only the relative locations among the sensor nodes are quickly visualized, but the temporal evolution of the same node at different time is also analyzed.

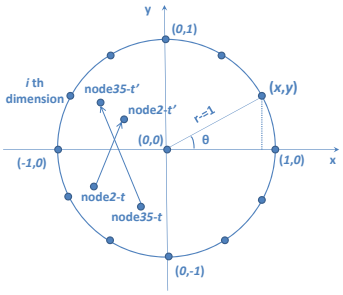


Figure 5: Dimension nodes (readings and counters) are along the circumference, with the high-dimensional sensor node values projected within it, showing the temporal dynamics.

The function for the coordinates of the dimensional anchors on the circumference is defined as:

$$\begin{aligned}(x_i, y_i) &= (r \cdot \cos(\theta_i), r \cdot \sin(\theta_i)) \\ \theta_i &= i \cdot \tfrac{2\pi}{dim},\ i = 0, 1, 2, ..., dim-1\end{aligned} \tag{5}$$

where r is the radius, θ is the radian, and dim is the number of dimensions.

The property value P for each sensor 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). 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 and maximum observed values reported by each or all the sensors during the entire time range, i.e., $\frac{v-min}{max-min}$, 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., 0 means no change. In this way, it is easier to visualize the state-changing turning points. Totally, there are "readings/counter" $(2)\times$ "$\{V_{[0,1]}, \Delta_{[0,1]}\}$" (2) different views for the high-dimensional sensor data display.

By default, each timestamp of a sensor's reporting time is treated as one data point mapped in the view, which means there will be $N * T$ points where N is the total number of sensor nodes and T is the number of timestamps. To support scalable visual exploration, we also introduce the level of granularity to automatically adjust based on the zoom-in/out requirement. An additional parameter *slot* is added in addition to *start* and *end*. Arithmetic mean is applied to the sensor reading/counter values spanning each recurring time *slot* for compression of multiple data points into one point, i.e., $P_{(i,t)} \rightarrow \overline{P}_{(i,slot)}$.

Once the spatiotemporal mapping of sensor nodes' properties is performed, further analysis can be performed by computing the clusters on the sensor nodes. For example, the tool can easily be extended to include state-of-art clustering modules. These modules can detect any data points deviated from the centroid (i.e., $d(n_i, C_k) > m \cdot \sigma$) and identify these points as outliers.

5 Visualization Design

The visualization for SAVE aims to interpret all the three sensor data facets: topology data, high-dimensional sensor data and the correlations among dimensions. For each facet, it also encodes analytics results described in Section 4. Due to the complex nature of these facets and their analytics, the visualization for SAVE is designed in multiple coordinated views as in Figure 1. Each view illustrates one dynamic data facet evolving over time. All the three views are snapped together through a shared time slider in the bottom of the interface. Once a new time range is selected, all the views will synchronously update to reflect the new data.

5.1 Sensor Topology View

The topology view shows the TEM graph generated from the raw packet delivery paths. As the input TEM graph is strictly a directed tree, we employ the traditional radial layout to place the sensor node, as shown in Figure 6a. The nodes in each graph hierarchy are ordered according to their path ID, such that the nodes with the same path ID will be kept stable in the graph layouts upon different time range selections. We also set the radius of graph hierarchies to be decreasing from the sink to the brim, so as to allocate more space for the nodes closer to the sink.

Recall that in the TEM graph, there is time series data attached to each node. In the sensor network scenario, it records the sending times of packets initiated or relayed through the node. To visualize this packet sending pattern, we borrow the idea of GrowthRingMap [14] and compose a temporal ring for each node. Each time value in the raw time series is first normalized into [0,1] according to the current selected time range. Then, starting from the earliest one to the latest, each normalized value is drawn as a ring on the node, with each ring radius increased by one from the previous ring. The color of each ring is selected by interpolating two boundary colors using the normalized value. In our case, 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 6a, we may notice that there are a cluster of orange nodes on the bottom right and a cluster of blue nodes on the top right, with quite similar topologies. It could be inferred that these two clusters contain the same set of physical nodes, and have experienced a major route change at their shared ancestor.

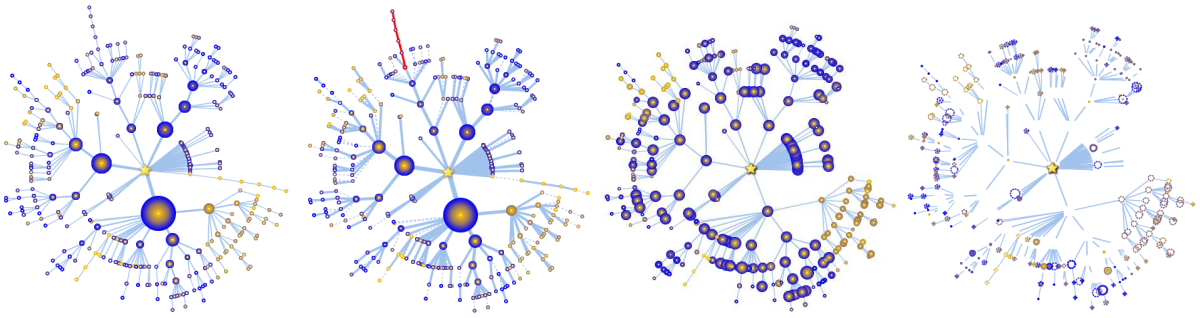
The anomaly analytics results in Section 4 can also be drawn on the TEM graph. Figure 6b highlights the loop paths detected in the topology in red color. It also renders the major paths in TEM with thicker stroke and non-major paths with dashes. The packet sending time series anomalies within each node are drawn by anomaly rings. The time points with ascending packet deliveries are rendered to the solid-line color rings while the time points with descending deliveries are drawn in dash rings. In both rings, the color hue still follows the temporal ring color at its time point. As shown in Figure 6d, it is more clear that the top right node cluster are counterpart of the cluster in bottom right after the major route change, because their anomaly rings have the same color hue.

One deficiency of the TEM graph is that each physical node may split to multiple path nodes which may confuse the user. To alleviate that, we introduce an overview+detail graph visualization approach. Once the user selects a path node in TEM graph, an egocentric graph of its corresponding physical node is also composed and presented in another coordinated view, as shown in Figure 9a. The egocentric graph is made up of physical nodes and essentially represents the path changes from the selected node to the sink.

5.2 Dimension Correlation View

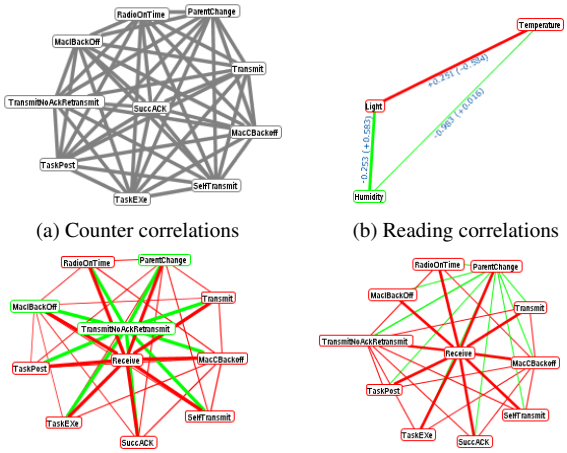
We design the correlation graph to visualize the data dimension correlations in one or a group of sensor nodes within the selected time range, as introduced in Section 4.3. For the case of selecting multiple sensor nodes, we compute the average correlation graph for these nodes. Based on the sensor data available, two types of correlation graph are presented, one among sensor network counter values (Figure 7a) and the other among sensor readings (Figure 7b).

Figure 7a gives an example of the correlation graph. Each node indicates one data dimension and each edge indicates a correlation calculated between the connected dimensions during the selected time range. In this graph, the layout is computed in force-directed



(a) TEM with temporal rings (b) TEM with loop and major paths (c) Flat TEM (d) Flat TEM with anomaly rings

Figure 6: TEM graph visualization of sensor network routing topologies and their anomalies.



(a) Counter correlations (b) Reading correlations

(c) Delta correlations (d) Correlation anomalies

Figure 7: Correlation graph views.

approach by setting the optimal length of the edge inversely proportional to the correlation value. We also encode the correlation value to the edge thickness for better data perception.

When the users diagnose network dynamics with correlation graphs, it is sometimes difficult for them to tell the differences as most of the correlation graphs are quite similar. To overcome this limitation, we introduce the delta correlation graph visualization, as shown in Figure 7c. In this mode, only correlations that increase or decrease significantly compared with the previous time window are added to the graph. The graph layout in this case computes the optimal node distance from correlation changes, rather than from the absolute correlation value. To help the user identify positive and negative changes, we map correlation increases into green edges and decreases into red edges, with the edge thickness encoding the absolute change value. Similarly, green nodes correspond to those with increased overall correlations with the other dimensions, and red nodes correspond to the opposites. In additional to the delta graph from the previous time window, we also support another delta graph by comparing the current correlation graph with the average correlation graph throughout the data set. It is believed that anomalies are rare in the entire data set, so this delta graph mode will effectively represent the anomalies in the sense of dimension correlations. Figure 7d shows an example anomaly correlation graph. In all correlation graph modes, a slider is included in the view to filter out the edges with low correlations (changes).

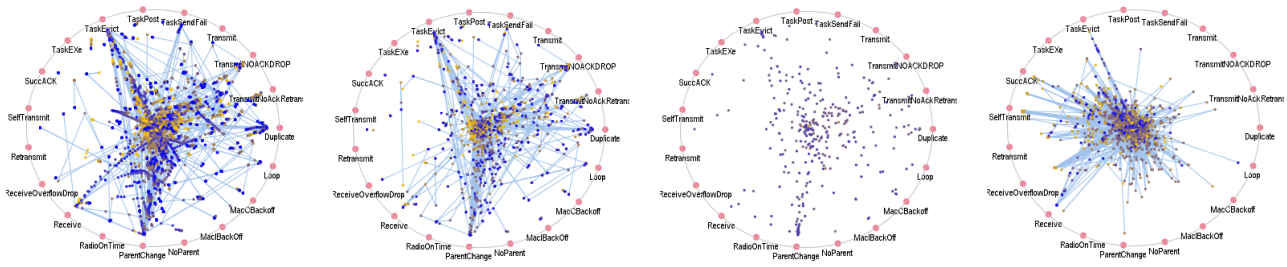
5.3 Dimension Projection View

Apart from the correlation graphs, we also incorporate the dimension projection view to visualize the high-dimensional data values associated with the sensor nodes and their changes over time. The basic idea is to draw a scatter plot to show the distribution of high-dimensional data value for each sensor node at each sensing time, so that both spatial and temporal outliers are identified in the view and operable further by the user.

Figure 8a gives an example of the dimension projection graph. There are several super nodes in red connected together as a bounding circle in the outside part of the graph, showing the data dimensions measured in the application. Inside the circle, numerous plots are drawn, each corresponds to one multi-dimensional data measurement for a particular sensor in a given time point. The color of the plot indicates the time of the measurement, with the same color mapping scheme as the topology graph view: orange indicates early measurements in the selected time range, blue indicates late measurements. Edges are added between each consecutive measurement time point of the same sensor node to visualize the temporal changes of the dimensional value. As introduced in Section 4.4, each value in the multi-dimensional measurement is normalized to [0,1] among all the plots. The placement of each plot is further calculated from these normalized multi-dimensional values. We apply a graph drawing approach here. Initially, optimal distances between each plot and the corresponding dimension super nodes are computed with the normalized value. Further, spring embedders are added between the plot and all the dimension super nodes. The final layouts are computed by finding an equivalence for all the embedders installed on the plot.

From Figure 8a, we can find that the most increasing dimensions, identified by the accumulated blue plots, are *ParentChange*, *Duplicate* and *TaskEvict*, which infer the most significant issues of the sensor network. However, due to the large number of plots drawn (in this case more than 200 $\times$ 48 plots), the graph appears quite cluttered and the user finds it hard to further diagnose with it. Also, from rendering performance perspective, it is costly to plot every time point. Herein, we introduce an interaction to let users choose the projection granularity as they like. Figure 8b shows the example for the same data but with only 5 averaged measurement plots, the visual clutter is alleviated in some degree. Further aggregating to one measurement per node, we create Figure 8c which highlights the spatial dynamics among sensor nodes. In this graph, we can tell that apart from the previous three major dimensions, *TransmitNoAckRetransmit* also shows up as another distinguishing factor.

Another concern from the user is that as the counter values are designed to be cumulative, the distribution of the value itself may not accurately track the anomalies at its exact time. The higher value of one counter could be due to a burst from a previous issue. To solve this problem, we introduce the dimension temporal change projection graph. The only difference from the original projection is that we use normalized delta value compared with the last known measurement in each sensor for the projection. Details can also be found in Section 4.4. The temporal change projection graph on the same data is shown in Figure 8d. From the graph, more anomalies



(a) Original counter projections (b) 5 aggregated measurements (c) 1 aggregated measurement (d) Temporal change projections

Figure 8: Correlation-based radial dimension projection views.

as busty value changes are located, such as *Receive*, *SelfTransmit* and *SuccAck*. Also, some previous identified dominant counters, such as *ParentChange* and *Duplicate*, actually has very rare extreme dynamics in short time.

The dimension projection view provides multiple customized user interactions for better navigation and diagnosis of the data:

- **Dimension sorting:** Upon clicking one dimension super node, all the other dimensions are reordered according to the correlations with the selected dimension. The dimension with smaller correlation is placed to the further location in the dimension circle. The resulting plots will better present the distribution and temporal trend of the selected dimension value.
- **Dimension manipulation:** The user can manually drag the dimension super node to reorder it with another dimension. In this way, projection graph with better customization can be created for user's specific scenario. The user could also delete the existing dimension node or add them back to simplify the projection graph.
- **Drill-in to values:** The user can hover or select one data plot to access the numerical multi-dimensional data value. Each dimension value is shown as one edge connecting the data plot to the corresponding dimension node. Both the label and the edge thickness indicate the value.

5.4 Discussion

SAVE introduces three novel visualizations to illustrate the multi-facet sensor data. We further show here, compared to their alternative designs, the visualizations of SAVE perform much better to reveal anomalies within the data.

One standard visualization of the dimension correlations would be the correlation matrix [11]. In the matrix, both the columns and the rows represent the list of dimensions. The correlation value between any two dimensions is color-coded to the cell at the intersection of the corresponding row and column. This correlation matrix is equivalent to the correlation graph in SAVE, however, is less informative and intuitive for our case. In the GreenOrbs scenario, the number of varying sensor data dimensions in the correlation graph reaches up to 13 in average, the number of dimension correlations larger than 0.1 sums up to 40 in average and will be much smaller if we set the correlation threshold to 0.5 to only reveal the significant correlations. Based on researches in [30], for such small graphs with sparse connections, the node-link representation performs better than the matrix visualization in most graph visualization tasks.

The tabular representation would be an alternative to the dimension projection view on the multi-dimensional sensor data. A standard design assigns the sensor dimensions to the columns of the table and the sensor nodes to the rows. In our scenario, the sensor data in one measurement cycle will lead to a table with 25 columns and more than 300 rows. Such a large table with numeric value in each cell is hard to interpret and detect anomalies. To track the trends in a full day, 144 such tables need to be created and compared. Comparatively, in our dimension projection view design, each multi-dimensional value is reduced to a single point and the points of the same node are connected together to show the temporal trends. Although still several thousands of points are rendered in a view and lead to visual clutter, the few abnormal measurements, which show significant bias in certain dimensions, will be placed far from the center of the view and are identifiable by the user.

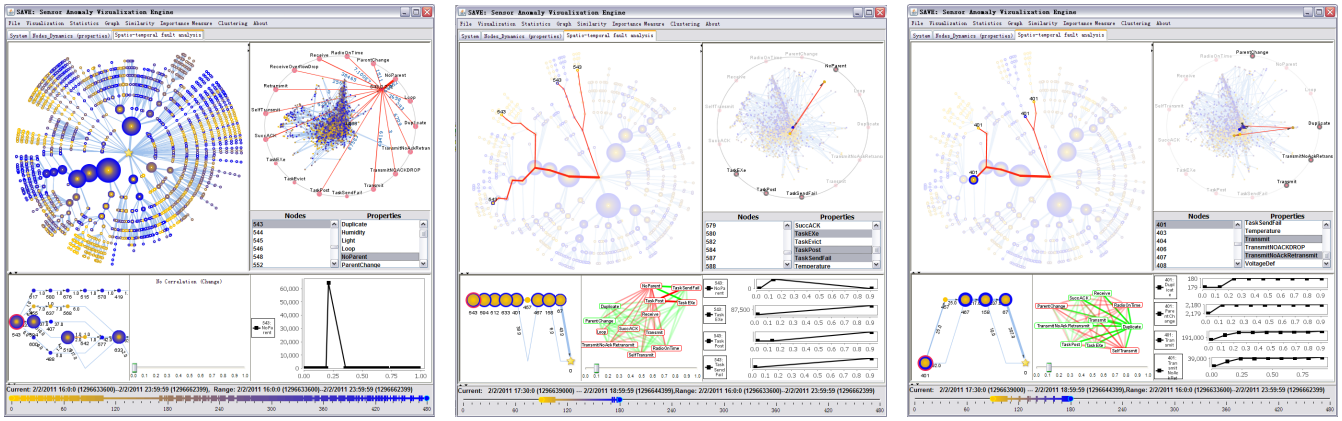
6 Evaluation

We have deployed the SAVE system to the GreenOrbs site to help the sensor network developers and administrators better operate the network. Through several period trials, the operators found the system to be quite good in identifying potential network anomalies. Below we give an detailed case demonstrating the usage of SAVE in the diagnose of salient link failures. We also report the initial feedbacks from domain users.

6.1 A Case on Sensor Failure Diagnosis

Consider John, the GreenOrbs system scientist, who was analyzing a one-day data collection from GreenOrbs sensor network deployment. His task was to investigate the data to find out the potential failed sensor nodes, and if possible discover the root cause of such failures. After preprocessing the data with SAVE, he started by selecting the data set from the afternoon, because he could infer some issues there from the changing size of the temporal data. Upon John's operation, an overview of the sensor data from afternoon was composed by SAVE, as depicted in Figure 9a, showing both the network topology and the sensor counter value dynamics. John quickly identified that there were several spatiotemporal outliers in the sensor counter projection view, compared with the other sensors and the other time points. With his extensive knowledge on GreenOrbs deployment, he first selected the data plot most close to *NoParent* dimension node (shown in Figure 9a) since this symptom told that the sensor could not find a parent node as next hop in its transmissions and generally led to serious routing problems. From the bottom-right details panel, he learned that it was sensor node 543 and indeed it experienced a burst in *NoParent* value.

John found from the scented time slider that there was a blank time period just corresponding to the *NoParent* burst, when no packet was successfully transmitted to the sink node for recording. He then refined the time slider to the exact time range containing such temporal anomalies. SAVE returned the data view as in Figure 9b. John went to the details by checking the correlation graph view of sensor 543. He filtered out some low-strength correlations with the slider. It turned out that the most significant correlation changes from the normal case included: *NoParent* which has increased correlations with *TaskSendFail* and *TaskEXe*; *TaskPost* which has decreased correlations with the latter two counters. Expanding these four counter values in the details panel, his hypothesis was further validated that due to the trouble in finding the routing parent, a portion of tasks issued by sensor 543 finally failed.



(a) Locate the *NoParent* anomaly (b) Drill-down to a node and its dimensions (c) Trace up to the suspected parent node

Figure 9: Link failure diagnosis: the user starts by locating the *NoParent* anomaly in node 543 (Figure (a)), drills down to the node and abnormal time range to show the topology, dimension and correlations (Figure (b)), traces up to the suspected ancestral node 401 for in-situ analysis (Figure (c)), finally validates node 457 to be normal and concludes with the link failure between node 401 and 457 (Figure 11).

However, with SAVE, John continued his diagnosis in the faith that the root cause of this symptom may not be sensor 543, because he found in the egocentric topology graph that sensor 543 did not have multiple parents in this time period. Actually if it were the failure in its best routing path to the sink, the symptom would have been similar since it took some time for the ETX-based routing scheme to converge to the new path, before which all the packets may be dropped. With this idea, he turned to click on the most suspected node 401 in the egocentric topology graph which had two routing paths during this time period. Figure 9c showed the returned view centric to the new node. Through the delta correlation graphs, John further detailed the *Duplicate*, *ParentChange* and *TransmitNoAckRetransmit* in the right charts, which are anomaly identifiers compared with the other status counters such as *Transmit* and *TaskPost*. The juxtaposing curve graphs proved that node 401 had both parent change and duplicate packet, also most of its transmissions are not ACKed. From the transmitted packet trends, it could be inferred that due to its sending failures, its child sensor nodes did not send packet up anymore during the blank period before it found another good path directly through sensor 467.

However, there exists another possibility that its previous parent – sensor 457 failed, which led to the similar symptom. To validate that, John clicked on sensor 457 in the egocentric graph view and got result in Figure 10. To validate its normality, he also switched the topology graph to flat mode where only self-initiated packets were visualized. Through the graph, he found that there was no problem with sensor 457: the periodical reporting packets, the routing path, the data correlations and the *TaskPost* vs. *TaskEXe* effect did not show major fluctuations. Finally, John concluded that the simple *NoParent* symptom of node 543 actually traced upward to the link failure between node 401 and 457.

6.2 User Feedback

During the period of deployment, the domain users provided us a lot of valuable feedbacks for SAVE on the useability, functionality and scalability of the tool. The most attractive features of SAVE in their opinion are the user-friendly interfaces that present the data and the practical function to drill down to a fault's root cause in a visual manner. Previously, most of their diagnostic tools simply focus on detecting certain types of faults, with little or no illustration of the source data for the potential diagnosis of another type of fault. What they liked most is the dimension projection view that could present the distribution of all the dimensions in a single view, especially the interaction to show the detailed values upon hovering the plot. They are positive to the TEM graph view for the intuitive radial

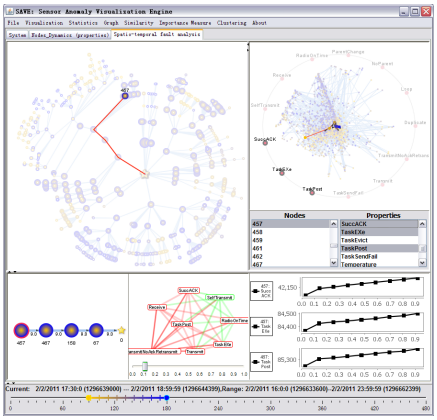


Figure 10: Validate the health of the parent node.

way to describe the topology, although they do need some time and tutorials to understand the solution.

The users suggested several improvements to the tool for our future work. In their daily jobs, a basic requirement is to check the status of the system regularly and report on demand. It will be quite helpful if SAVE can provide a simpler view to automatically report the faults that can be detected routinely. They also mentioned the data quality issue. The current visualizations in SAVE are mostly over processed data, assuming the correct data transformation from the raw sensor data. How to visualize the anomalies raised due to the low sensor data quality, which is quite common in reality, is still a open question. One user with good visualization experience suggested to consider the radar chart as an alternative to the dimension projection view. Another wanted a query box to search the sensor node by ID to locate it in the TEM graph.

7 Conclusion

In this paper, we have presented SAVE, a visual analytics system that provides a new way for operators and administrators to identify sensor network data patterns, diagnose potential failures, and locate root causes of these failures in a user-friendly and interactive manner. In designing SAVE, we have developed several novel visualizations based on generic anomaly detection methods, for effective representation of the dynamic high-dimensional sensor data. Through deployments and case study with a real-world sensor network system, we demonstrate that SAVE is able to save great human efforts and optimize their exploratory analysis process in conducting the sensor network diagnosis task.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, pp. 2787–2805, 2010.

[2] K. Ni, N. Ramanathan, M. N. H. Chehade, S. Nair, S. Z. E. Kohler, G. Pottie, M. Hansen, and M. Srivastava, "Sensor network data fault types," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, pp. 25:1–29, May 2009.

[3] "Greenorbs: A long-term kilo-scale wireless sensor network system in the forest." `http://greenorbs.org`.

[4] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: Accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems (ACM SenSys '03)*, (Los Angeles, CA), November 5-7 2003.

[5] M. Turon, "Mote-view: A sensor network monitoring and management tool," in *the 2nd IEEE workshop on Embedded Networked Sensors*, pp. 11–17, 2005.

[6] L. Shu, C. Wu, Y. Zhang, J. Chen, L. Wang, and M. Hauswirth, "Nettopo: beyond simulator and visualizer for wireless sensor networks," *ACM SIGBED Review*, vol. 5, no. 3, 2008.

[7] N. Sénéchal, S. Hong, and P. Eades, "Display of sensor networks: a feasibility study." Daintree Networks, July 2006.

[8] "Surge network viewer." Crosshow Technology Inc.

[9] C. Buschmann, D. Pfisterer, S. Fischer, S. P. Fekete, and A. Kröller, "Spyglass: a wireless sensor network visualizer," in *ACM SIGBED Review Special issue: Best of sensys 2004 work-in-progress*, vol. 2, pp. 1–6, 2005.

[10] Y. Yang, L. Huang, Q. Zhou, Y. Xu, and X. Li, "Snamp: A multi-sniffer and multi-view visualization platform for wireless sensor networks," in *1st IEEE Conference on Industrial Electronics and Applications (ICIEA '06)*, (Singapore), pp. 1523–1526, May 24-26 2006.

[11] X. Miao, K. Liu, Y. He, Y. Liu, and D. Papadias, "Agnostic diagnosis: Discovering silent failures in wireless sensor networks," in *IEEE INFOCOM*, (Shanghai, China), April 10-15 2011.

[12] R. C. Read and D. G. Corneil, "The graph isomorphism disease," *Journal of Graph Theory*, vol. 1, no. 4, pp. 339–363, 1977.

[13] H. Bunke, P. J. Dickinson, M. Kraetzl, and W. D. Wallis, *A Graph-Theoretic Approach to Enterprise Network Dynamics (Progress in Computer Science and Applied Logic (PCS))*, vol. 24. Birkhäuser Boston, 2007.

[14] P. Bak, F. Mansmann, H. Janetzko, and D. A. Keim, "Spatiotemporal analysis of sensor logs using growth ring maps," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, pp. 913–920, November/December 2009.

[15] J. V. Carlis and J. A. Konstan, "Interactive visualization of serial periodic data," in *11th annual ACM Symposium on User Interface Software and Technology*, pp. 29–38, 1998.

[16] A. Inselberg, "The plane with parallel coordinates," *The Visual Computer*, vol. 1, no. 2, pp. 69–91, 1985.

[17] E. Kandogan, "Visualizing multi-dimensional clusters, trends, and outliers using star coordinates," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, (San Francisco, CA), pp. 107–116, 2001.

[18] P. Hoffman, G. Grinstein, and D. Pinkney, "Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations," in *Proceedings of workshop on new paradigms in information visualization and manipulation*, (Kansas City, Missouri), pp. 9–16, 1999.

[19] C. Tominski, J. Abello, and H. Schumann, "Axes-based visualizations with radial layouts," in *ACM Symposium on Applied Computing*, pp. 1242–1247, 2004.

[20] B. George, J. M. Kang, and S. Shekhar, "Spatio-temporal sensor graphs (stsg): A data model for the discovery of spatio-temporal patterns," *International Journal of Intelligent Data Analysis (JIDA)*, vol. 13, no. 3, pp. 457–475, 2009.

[21] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *IEEE INFOCOM*, vol. 3, (San Francisco, CA), pp. 1848–1857, March 30-April 3 2003.

[22] V. Bandara, A. Pezeshki, and A. P. Jayasumana, "Modeling spatial and temporal behavior of internet traffic anomalies," in *the 35th Annual IEEE Conference on Local Computer Networks (LCN)*, (Denver, Colorado), pp. 392–399, October 11-14 2010.

[23] Z. Liu, B. Lee, S. Kandula, and R. Mahajan, "Netclinic: Interactive visualization to enhance automated fault diagnosis in enterprise networks," in *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, (Salt Lake City, UT), pp. 131–138, Oct 25-26 2010.

[24] P. C. Wong, H. Foote, G. C. Jr., P. Mackey, and K. Perrine, "Graph signatures for visual analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, pp. 1399–1413, November-December 2006.

[25] 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 *Proceedings IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pp. 147–154, October 2009.

[26] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2009.

[27] G. Tolle and D. Culler, "Design of an application-cooperative management system for wireless sensor networks," in *Proceeedings of the Second European Workshop on Wireless Sensor Networks*, pp. 121–132, Jan. 31-Feb.2 2005.

[28] J. Heer and D. Boyd, "Vizster: Visualizing online social networks," in *IEEE Symposium on Information Visualization (InfoVis)*, 2005.

[29] S. M. Stigler, "Francis galtons account of the invention of correlation," *Statistical Science*, vol. 4, pp. 73–79, 1989.

[30] M. Ghoniem, J.-D. Fekete, and P. Castagliola, "A comparison of the readability of graphs using node-link and matrix-based representations," in *IEEE Symposium on Information Visualization (InfoVis)*, 2004.