Social Plane for Recommenders in Network Performance Expectation Management

Yuanxun Zhang, Prasad Calyam, *Senior Member, IEEE*, Saptarshi Debroy, *Member, IEEE*, and Sai Shreya Nuguri

***Abstract*—Multi-domain end-to-end network performance monitoring federations such as perfSONAR are increasingly being used in Big Data application management. They rely on trustworthy collaborative measurement intelligence to identify and diagnose network anomaly events that impact application performance. Large volumes of end-to-end measurement traces are generated on a daily basis, and new Big Data analysis techniques are needed to isolate network-wide anomaly event(s) and to diagnose the root-cause(s). In addition, not all network operators and application users have enough knowledge and experience to understand the anomaly events. The lack of a platform for sharing knowledge and working collaboratively makes it difficult to isolate and diagnose network-wide anomaly events quickly and accurately. In this paper, we define a "social plane" that relies on recommended measurements based on "content-based filtering" and "collaborative filtering" approaches to enable network performance expectation management. Based on similarity analysis, the content-based filtering facilitates users to subscribe to useful measurements, and the collaborative filtering promotes users to share knowledge on anomaly symptoms. Using real perfSONAR measurements and synthetic events, we show the effectiveness of our social plane approach within a SoyKB Big Data application case study using social network creation and mingling of experts. Our experimental results show that our measurements recommendation scheme has high precision, recall, and accuracy, as well as efficiency in terms of the time taken for large volume measurement trace analysis.**

***Index Terms*—Network monitoring, socio-technical methods, measurement recommenders, performance management.**

I. Introduction

SCIENTIFIC communities (e.g., LHC in high-energy physics [1], SoyKB in Bioinformatics [2]) have extensively deployed multi-domain Network Performance Monitoring (NPM) federations that use passive and active measurements for troubleshooting network bottlenecks. Among these NPM federations, perfSONAR [3] is the most widely instrumented framework with over 1400 instances deployed worldwide. It uses network diagnostic tools, such as Ping, Traceroute, OWAMP (for one-way delay measurements), and BWCTL (for TCP/UDP throughput measurements) to collect measurements such as end-to-end delay, jitter, loss, and bandwidth. With growing perfSONAR deployments within multi-domain federations, the initial focus of intra-campus network monitoring has shifted towards end-to-end performance monitoring and troubleshooting of Big Data applications. In multi-site Big Data collaborations, the application traffic generated within a network (domain) traverses several different domains or autonomous systems before reaching a destination. As a result, end-to-end performance troubleshooting becomes significantly harder for standalone perfSONAR measurement instances (Measurement Point Appliances or MPAs) installed at the source and destination domains.

Current end-to-end performance monitoring systems have many limitations: Firstly, an end-to-end measurement trace normally traverses different domains and different service providers. A single measurement trace is typically insufficient to isolate and diagnose the root-cause of anomaly events. Users usually need enough data to make an accurate and trustworthy judgment. However, current infrastructure does not provide the necessary tools or services for users to filter out the relevant data from vast archives of measurements. Secondly, multiple measurement traces from different domains may not be calibrated and trustworthy in cases such as invalid data (e.g., negative one-way delay values due to faulty clock synchronization), missing data or too dense/sparse or irregular (i.e., long data collection gaps) measurement data sampling frequency. These measurement data "veracity" issues can result in erroneous features [4], missing events or even exponential anomaly event detection time [7]. Lastly, there are no existing frameworks for users to create social networks and mingle for trustworthy knowledge sharing and collaborative work to efficiently and effectively diagnose anomaly event root-causes.

Figure 1 illustrates the effect of these limitations from an end-user perspective in an example scenario involving users (i.e., $A \sim F$) from different domains. We assume users A, B and D have to depend on their own knowledge to diagnose their problem of inferior performance due to an anomaly event. Without knowledge sharing, we can see that most of these users fail to diagnose or even recognize their problem,

Manuscript received May 17, 2017; revised September 23, 2017; accepted November 4, 2017. Date of publication November 13, 2017; date of current version March 9, 2018. This material is based upon work supported by the U.S. Department of Energy under Award Numbers: DE-SC0001331 and DE-SC0007531, and National Science Foundation under Award Number: ACI-1440582. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or any agency thereof. The associate editor coordinating the review of this paper and approving it for publication was Y. Diao *(Corresponding author: Prasad Calyam.)*

Y. Zhang, P. Calyam, and S. S. Nuguri are with the Department of Computer Science, University of Missouri–Columbia, Columbia, MO 65211 USA (e-mail: yzd3b@mail.missouri.edu; calyamp@missouri.edu; snd45@mail.missouri.edu).

S. Debroy is with the Department of Computer Science, City University of New York, New York, NY, USA (e-mail: saptarshi.debroy@hunter.cuny.edu).

Digital Object Identifier 10.1109/TNSM.2017.2772905

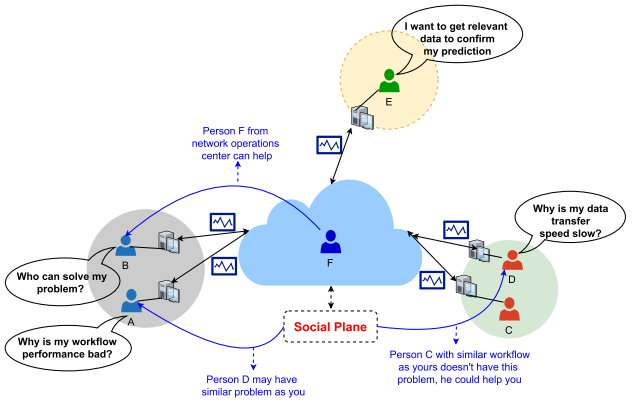


Fig. 1. Limitations of current end-to-end performance monitoring system due to: (a) challenges in diagnosing the root cause of anomaly events; (b) lack of services to find relevant measurements; (c) lack of frameworks to share knowledge or work collaboratively.

and those users (e.g., user C and F in Figure 1) who may know the cause of anomaly events could not be connected. In addition, many users (e.g., user E in Figure 1) need more relevant data to confirm their anomaly event detection results. Without pertinent measurements recommendation, users usually collect data randomly, which could result in erroneous detection and ineffective diagnosis.

To overcome such situations that lead to users' dilemma during bottleneck anomaly root-cause diagnosis, we propose a "social plane" approach. The goal of our social plane approach is to provide a framework to allow expert knowledge and other measurement intelligence information to be integrated/customized into scalable anomaly detection tools. We propose the use of "measurement recommenders" for processing large volumes of measurement traces collected on a daily basis to improve the efficiency of application users and network operators to troubleshoot and work collaboratively on root-cause diagnosis. The novelty of our approach is to combine: (i) *content-based filtering* which recommends pertinent traces based on user's measurements analysis objective to achieve more accurate detection results, along with (ii) *collaborative filtering* to recommend the most pertinent traces which may have similar network issues or possibly the same root-cause issues with user's targeted trace.

The *content-based filtering* is used to rank and recommend the most pertinent traces based on measurements temporal and spatial similarity matching with a target trace/path. The target trace is the one for which the operator/user needs help to perform some specific measurement correlation analysis. In order to strengthen the measurement data "veracity" information, we propose a data sanity checking scheme. The measurement data sanity scores are extended to use a Bayesian Inference-driven scheme for historical domain/community reputation. This scheme acts as a "confidence indicator" to help the operators and users in relevant long-term measurement subscriptions. The data sanity together with domain reputation ultimately provides the essential meta-information on top of content-based filtering recommendation for the operators/users to take informed decisions.

The *collaborative filtering* scheme is used to seek the measurements which have similar network issues, or possibly the same root-cause issues with a user's targeted trace. Through relevant measurement analysis, we could find potential users who face similar problems, so that those users could share diagnosis experiences or work collaboratively on the issue. Our algorithm does so by measuring the similarity in anomaly symptoms between users' targeted measurements and candidate measurements. Lastly, we implement a "social network" for measurements, on top of filtering scheme to demonstrate how our collaborative filtering scheme helps users to enhance knowledge sharing and trustworthy collaboration for network performance expectation management.

In our evaluation studies, we use real measurements with synthetic events to validate the effectiveness and efficiency of our scheme in recommending pertinent measurement traces by combining content-based filtering and collaborative filtering. In addition, we also use real measurements to show how users can use content-based filtering to filter pertinent measurements in terms of their measurements analysis objective. Further, we simulate the SoyKB [8] Big Data application's network environment in a GENI Cloud Testbed [9] to show how our scheme recommends the most pertinent traces. The evaluation results demonstrate how our proposed measurement recommendation scheme will enable network operators and application users to intelligently use subscribed community traces for diagnosing network events that impact Big Data applications.

The rest of the paper is organized as follows: Section II discusses the related works. In Section III, we briefly describe our "social plane" approach, and basic components in our approach. In the Sections IV and V, we describe the major components of our proposed schemes in detail. The "social plane" implementation is described in Section VI. Section VII discusses the evaluation and results using real and synthetic data. Our approach effectiveness in several Big Data application scenarios is presented in Section VIII. Section IX concludes the paper.

II. Related Works

A. Measurement Intelligence Sharing

New challenges are emerging for analyzing measurements in large-scale multi-domain infrastructures in order to resolve bottleneck anomaly events and facilitate efficient root-cause diagnosis. Govindan *et al.* [10] from Google show the benefits of expertise sharing and knowledge management of network failures as vignettes within a complex and high-scale environment. Kanuparthy and Dovrolis [11] show that it is possible to have experts identify symptoms of anomaly events based on measurement trace characteristics, which can be shared as measurement intelligence to understand performance bottleneck events in perfSONAR deployments. Our work uses these symptoms knowledge and the idea of expertise sharing within the social plane implementations of Big Data applications such as SoyKB [8].

In addition, our work builds upon [12], where the authors present an automatic network fault detection and diagnosis

system for end-users called DYSWIS ("Do You See What I See"). A distributed hash tree network is used to search the collaborative nodes with appropriate properties required to diagnose a failure in large-scale home networks. Contributions of expert knowledge (diagnosis rules and probes) by application developers, vendors, and network administrators are used to enable crowdsourcing of diagnosis strategy. Our work is also comparable to the work in [13], where the authors implement a "social angle" to monitor and detect problems in a large collaborative network environment involving multiple domains. Their solution was limited to a social media platform setup to allow communication of users to share issues, ideas, concerns and problems with other users or network experts and does not feature any recommender algorithms. Our social plane implementation for measurement intelligence sharing is inspired by the work in [14]. Therein, the authors propose a social-media approach to monitor virtualized environments by creating a "community" that includes various entities along with an administrator.

B. Data Veracity and Domain Reputation

The risks of using potentially misleading data and the related guidelines to trustworthy measurement best practices were first highlighted in [4], wherein the author explains the methods implemented in handling errors and inaccuracies; the importance of associating meta-data with measurements; the technique of calibrating measurements by examining outliers and testing for consistencies; difficulties that arise with large-scale measurements; among other issues. Our previous work on using sanitized measurement data for anomaly detection [5] is closest to the work by [6] where an anomaly detection system is developed based on prediction of upper and lower dynamic thresholds of various time-varying data trends. Tang *et al.* [15] proposed an overlay fault diagnosis framework with a diagnosis uncertainty reasoning analysis based on evidences.

Similarly, reputation-based trust schemes have long been used by the scientific community for decision making in shared environments. Xiong and Liu [16] present a reputation-based trust model for peer-to-peer eCommerce communities. Whereas, Mui *et al.* [17] described a similar scheme to use Bayesian Inference to build reputations for agents in the e-business community. Further, works such as [18] extend such reputation models by introducing an age factor in Bayesian Inference as it is desirable to give greater weight to more recent ratings. Ivanov *et al.* [19] propose a tagging and trust mechanism in social networks based on users and their contents that is similar to our research of building a reputation scheme for multi-domain measurement domains within scientific Big Data communities.

C. Recommender System

Filter-based (mostly content-based and collaborative) social/community environments have been proposed in different areas of computing. However, in relation to measurement frameworks, such works are limited. Our work is the first to propose a recommender framework to systematically analyze large number of measurement traces to identify bottlenecks and resolve them in multi-domain network monitoring. Collaborative filtering is a technique to filter large sets of data for information and patterns. Lim and Finkelstein [20] propose a novel method that uses social networks and collaborative filtering to identify and prioritize requirements in large-scale software projects. More specifically, they address information overload problems in requirements elicitation of software development activities by using collaborative filtering to recommend relevant requirements to stakeholders and prioritizing the requirements and stakeholders. Wang and Blei [21] use collaborative filtering and a probabilistic topic model to recommend relevant scientific articles to users of online communities. With collaborative filtering, their model can recommend articles for a particular user based on other users who liked similar articles.

Our approach builds upon the above prior works and aims to combine both content-based and collaborative filtering techniques for finding: (a) measurement traces with similar anomaly symptoms, and (b) relevant people who can mingle and resolve any identified bottlenecks. In our previous work [22], we have proposed a content-based recommender to filter relevant measurements based on user's analysis objective, such as temporal analysis and spatial analysis. This work extends the prior content-based filtering in combination with collaborative filtering to find measurements with similar anomaly symptoms such as those identified in [11]. Furthermore, we show the effectiveness of our social plane approach for a Big Data application case study, i.e., SoyKB that involves distributed computing across HPC sites, and Big Data processing within large knowledge bases in bioinformatics.

III. Social Plane Approach Overview

The anomaly symptom space can be broad in real network situations, and could pose challenges for effective root-cause analysis. However, it is possible to obtain expert knowledge of common network symptoms that affect, e.g., large file transfer applications over wide-area network paths as in [11]. The goal of our social plane approach is to provide a framework to allow such expert knowledge and other measurement intelligence information to be integrated/customized into scalable anomaly detection tools. We propose a subscription of large-scale monitoring of measurement data sets collected on a daily basis and timely identification/notification of critical anomaly events accurately. We suppose that the root-cause diagnosis of the critical anomaly events require mingling of relevant stakeholders for diagnosis. Our social plane approach thus provides a "systematized" method is to: (i) help users to obtain relevant measurements for analysis which may enhance measurements subscription and sharing, and (ii) find potential people who may solve a given multi-domain problem which could enhance knowledge sharing and trustworthy collaborations.

In the following, we describe our "social plane" architecture that is depicted in Figure 2, which combines the content-based and collaborative filtering techniques to accomplish the purpose.

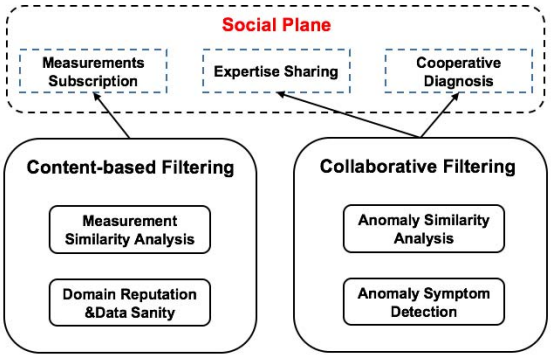


Fig. 2. Framework of *social plane* approach.

A. Content-Based Filtering

Content-based filtering is applied when users do some correlation analysis and want to find pertinent measurements according to their temporal or spatial correlation analysis objectives. We propose the *measurements similarity analysis* algorithm to calculate measurement temporal and spatial attributes similarity score between users' target measurement trace and candidate measurement traces from the pool of measurements archives. Using a decision tree, we can rank relevant measurements based on user's temporal or spatial analysis objectives. Because "data veracity" may affect accuracy of analysis results, we provide *data sanity checking* for users to indicate the trustworthiness of measurements and analysis results. With *domain reputation estimation*, users could decide if they should subscribe their measurements from long-term perspective. Hence, content-based filtering scheme could encourage measurements sharing and subscription bases on user's interests.

B. Collaborative Filtering

Collaborative filtering is applied when users need helps from others who may solve their problem or have similar problem which could involve collaborative work. In order to find similar problem, we classify it with different types of anomaly symptoms (e.g., "delay level shift", "high delay utilization") by using an *anomaly symptom detection* algorithm, and then rank relevant measurement traces with most similar anomaly symptom with target trace using *anomaly similarity analysis*. Through filtering out the measurement traces with similar anomaly events, we connect most relevant persons who face a similar problem. This in turn allows those persons to share diagnosis knowledge and work collaboratively. Consequently, a collaborative filtering scheme encourages diagnosis knowledge sharing and trustworthy collaboration.

With the help of "content-base filtering" and "collaborative filtering" schemes, the "social plane" could be formed to encourage measurements sharing/subscription, expertise sharing and cooperative diagnosis. In the following sections, we explain the details of each of these schemes.

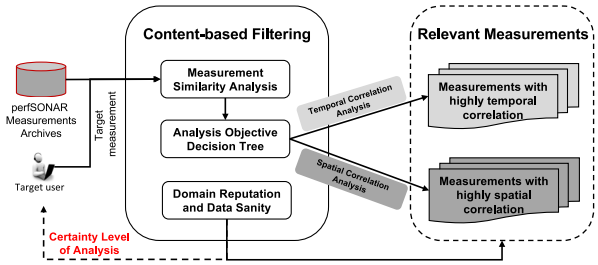


Fig. 3. Major components of *content-based filtering* scheme, which filters out the relevant measurements according to users' analysis objective.

IV. Content-Based Filtering Measurement Recommendation

Content-based filtering is applied when users perform correlation analysis and want to find pertinent measurements according to their temporal or spatial correlation analysis objectives. In this section, we propose our content-based filtering measurements recommendation scheme. Our filtering approach is derived from the concepts of content-based filtering techniques used in many recommendation systems, especially by online retail enterprises, such as Amazon, eBay. Content-based filtering, also referred to as cognitive filtering, recommends items based on a comparison between the contents/features of the items and users' profiles or interests.

As shown in Figure 3, our proposed content-based filtering recommendation scheme filters and ranks most relevant measurement traces from a pool of traces based on *Measurement similarity analysis* and *Analysis objective decision tree*. In addition, due to current qualities of measurements, "data veracity" is an important factor to affect the accuracy of measurements analysis, especially in correlation analysis. Thus, we provide *Data sanity checking and Domain reputation estimation* in content-based filtering scheme for users to have a level of certainty regarding their measurements and analysis results.

A. Measurements Similarity Analysis

As proposed in our previous work [22], we argue that for any broad type of correlation analysis objective, i.e., spatial or temporal, there are four important factors that define time-series measurement traces' attributes. They are: 1) *Topology*: the path taken by the measurement probe packets; 2) *Metric*: the measurement tool (such as one-way delay, throughput) used to collect the samples; 3) *Time Range*: the time range of the time-series measurements; 4) *Alignment*: the relative positions of the measurement sampling time stamp instances.

For each of these factors, we will quantify the relative similarity between target trace and candidate traces and then create an overall similarity score from individual factor similarities. The overall similarity score will help the network operators to rank the candidate traces in the order of their relevance.

1) Measurements Topology Similarity: The topology attribute of any measurement trace is the traceroute information between two sites, which is made of intermediate nodes or hops.

This topology information is very important for correlating measurements traces because - higher the similarity between traces' topologies, higher the probability of common network events of interest. Therefore, we express the topology similarity $topo\_simi_{i,j}$ between target trace i and candidate trace j as:

$$topo\_simi_{i,j} = \frac{topo_i \cap topo_j}{topo_i \cup topo_j}. \quad (1)$$

2) Measurements Metric Similarity: The measurement *metric* indicates the network performance measurement tool used for monitoring, such as Ping, OWAMP, BWCTL, etc. The measurement metrics similarity between traces is of importance based on the type of measurement analysis sought. We consider measurement metric similarity for recommendation of measurement traces of two different tools that use different configurations/methods to provide similar metrics, e.g., latency or loss. We express measurement metric similarity $metric\_simi_{i,j}$ between traces i and j as a boolean representation:

$$metric\_simi_{i,j} = \begin{cases} 1, & (m_i = m_j) \\ 0, & (m_i != m_j). \end{cases} \quad (2)$$

3) Measurements Time Range Similarity: The measurement *time range* of a trace is one of the most important measurement attributes for temporal analysis, when the duration of the traces becomes critical to detect a time-specific network event. Thus, if two traces' duration are not aligned temporally, their time range similarity should be equal to zero. Therefore, we express the time range similarity $range\_simi_{i,j}$ between traces i and j as:

$$range\_simi_{i,j} = \frac{r_i \cap r_j}{r_i}. \quad (3)$$

4) Measurements Alignment Similarity: Measurement alignment, i.e., the relative positions of measurement sample instances is also significant for correlation analysis with multiple dimension time-series measurements. Samples that are closely aligned are easier to correlate and have better chances of accurate detection of network events upon analysis. Relative alignment of sample instances between two traces is a better metric to quantify their relative similarity. In an illustrative example shown in Figure 4, we show one target and three candidate traces with different periods and sampling patterns. As far as the similarity is concerned, candidate trace 1 is best aligned to the target trace as the mean relative displacement between the trace 1 and the target trace is minimum.

Thus, for generic quantification of alignment between measurement traces, we define an alignment displacement metric d that denotes the mean relative distance between sample instances of target trace i and candidate trace j. The metric d is expressed as:

$$d_{i,j} = \sum_T |ts_i - \hat{ts_j}| \quad (4)$$

where ts_i denotes target trace time stamp, $\hat{ts_j}$ denotes the candidate time stamp closest to the target trace time stamp ts_i, and

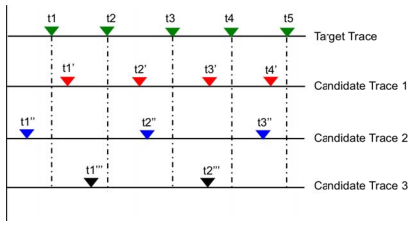


Fig. 4. Measurements alignment illustration between traces with varied periodicity and non-aligned sample time stamp instances.

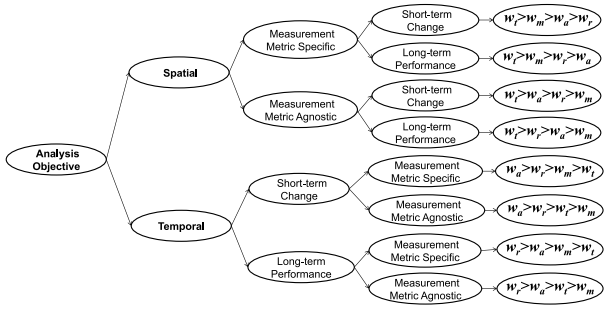


Fig. 5. Decision tree for different measurement analysis objectives and the corresponding relative measurement attributes' weights.

T denotes the number of such time stamps in the target trace. The range of metric d varies between $[0, +\infty)$ with smaller value indicating better alignment between traces. In order to normalize d and to be consistent with other similarity factors, we transform the range of metric d between [0, 1] using min-max normalization method, which can be expressed as:

$$align\_simi_{i,j} = \frac{max_{d_{i,j}} - d_{i,j}}{max_{d_{i,j}} - min_{d_{i,j}}} \quad (5)$$

where $max_{d_{i,j}}$ and $min_{d_{i,j}}$ are the maximum and minimum values of $d_{i,j}$.

B. Overall Similarity Scores Based on Analysis Objective Decision Tree

The overall measurements similarity is expressed as a weighted product of the aforementioned four similarity factors:

$$overall\_simi_{i,j} = \sum_k w_k * factor\_simi^k_{i,j} \quad (6)$$

where $factor\_simi^k_{i,j}$ denotes each of the aforementioned similarity factors and w_k denotes their respective weights for the overall measurements similarity score. The values of the weights depend on the relative importance of these attributes to achieve different *measurement analysis objectives*. In order to simplify the weights but at the same time to have a more comprehensive list of analysis objective scenarios, we design a decision tree of different generic measurement analysis objectives and corresponding relative importance of attributes' weights (w_t, w_m, w_r, and w_a respectively for each of the similarity factors), as shown in Figure 5.

Broadly, we divide the entire analysis objective space into temporal and spatial analysis. In temporal analysis, the weights for temporal factors, such as *time range* and *alignment* are more important than spatial factors

such as *topology*. For example, one of the most common measurement analysis for end-to-end data-intensive application management is the correlated anomaly event detection [25], which involves a spatial analysis and falls under 'Analysis Objective'→'Spatial'→'Measurement Metric Specific'→'Short-term Change'. So, the relative weights should be $w_t > w_m > w_a > w_r$. Hence, using a content-based filtering scheme, users can easily obtain the relevant measurements according to their analysis interests.

C. Data Sanity Checking and Domain Reputation Estimation

"Data veracity" factor is important to determine the accuracy of any measurement analysis results. Hence, we design a "data sanity checking" scheme to check the quality of data, and use it with a "domain reputation" algorithm to verify whether a measurements from a certain domain should be subscribed from a long-term perspective or ignored for critical analysis.

1) Data Sanity Checking: We implement the two-pronged approach from [5] to sanitize measurement data: a reputation analysis scheme for collected samples, and a filter framework to intelligently prune the potentially misleading samples.

In the context of detecting and diagnosing potential correlated anomaly events within time-series measurements, it is important that the sample data has a desired nature expected by the monitoring objective in terms of two aspects: (i) Sampling Pattern, and (ii) Data Validity. To identify potentially misleading features of measurements, we use the reputation-based data sanity checking scheme which analyzes the measurement samples for sampling pattern, and collected sample validity.

2) Domain Reputation Estimation: The concept of reputation of data is closely linked to its trustworthiness; an entity's reputation is generally a subjective proof of its historical actions and in most cases, a measure of expectations of future behavior. The main objective of proposing a domain reputation scheme is to generate domain-centric expectations for network operators when they subscribe to measurement data from different domains. Such a domain reputation scheme can encourage trustworthy measurement practices (e.g., sharing of calibrated measurement tool data) among multiple domains supporting Big Data applications.

Reputation of any domain is a function of the quality of measurement data generated from that domain. We observed that in any random collection that was publicly accessible, some measurements exhibit non-periodic sampling patterns, i.e., they are either too dense or too sparse, and some were invalid due to faulty clock synchronization between measurement servers or data corruption (e.g., negative one-way delay values). For spatial analysis with OWAMP, the characteristics are: *sampling pattern* and *data validity*. Thus, in our earlier work [5], we have defined the sanity score of any trace with path (source, destination) i as:

$$s_i = \frac{N_i - \left(N_i - n_i^{majority}\right) - \left(N_i - n_i^{valid}\right)}{N_i} \tag{7}$$

where N_i denotes the number of measurement samples in path i, n_i^{valid} denotes the number of valid data samples in path i, and $n_i^{majority}$ denotes the number of samples showing consistent sampling pattern.

Through Bayesian Inference, new or an updated reputation score (i.e., posteriori) of an entity can be computed by combining the old/previous reputation score (i.e., priori) with a new belief. In order to translate sanity scores into domain reputation, we first discretize the measurement data sanity scores into data sanity ratings of a particular domain using boolean variables such as 'Good' (variable x) and 'Bad' (variable y), and some sanity threshold ϵ. The value of ϵ is a measure of the degree of conservativeness of the reputation scheme that is kept constant for the entire system. The value of ϵ can be set based on the distribution of measurements' sanity scores in the system. If the average sanity score of measurement data in the system is very high, ϵ value is kept high to differentiate between good and bad measurements, and vice versa. Usually for all practical purposes, ϵ value is around $[\mu+\sigma, \mu+2*\sigma]$.

$$x = |i| \ \forall \ s_i >= \epsilon; \ \ y = |i| \ \forall \ s_i < \epsilon \tag{8}$$

Therefore, at any given time t, the measurement data sanity rating of any domain is represented as $\rho^t = [x, y]^t$. Now if there are T such discrete data sanity ratings collected over a period of time, then the overall data sanity rating after T collection is given as $\rho^T = [x, y]^T$, where x^T and y^T are expressed as:

$$x^T = \sum_{t=1}^{T} \lambda^{T-t} x^t \text{ and } y^T = \sum_{t=1}^{T} \lambda^{T-t} y^t \tag{9}$$

where $0 \leq \lambda \leq 1$ is called the 'forgetting factor' and keeps the recent history of data sanity rating more relevant in reputation calculation than ancient history. The value of λ represents how forgetful a system is, $\lambda = 1$ means the system forgets nothing. Thus, this value depends on the user's opinion such that, if the user thinks the historical reputation is also very important, this value should be close to 1; however, if the user thinks current reputations are more important, the value should be close to 0.

Now after collecting T such discrete data sanity ratings, the reputation of the domain responsible is expressed as a posterior expectation of beta distribution of ρ^T and is represented as:

$$\begin{aligned} R^T &= \mathrm{E}[beta(\rho^T)] \\ &= \frac{x^T + 1}{x^T + y^T + 2}. \end{aligned} \tag{10}$$

V. Collaborative Filtering Measurement Recommendation

Collaborative filtering is applied when users need help from other users to solve a multi-domain performance issue. When network anomaly events are detected in their monitoring system, the application users or network administrators want to diagnose the reasons of anomaly events. However, anomaly event diagnosis is a challenging problem, especially in and end-to-end multi-domain monitoring system. Due to lack of skills or knowledge in network diagnosis and complicated reasons to cause network issues involving large measurement archives, most users fail to identify many anomaly events, which may affect their performance. If there was a "social"

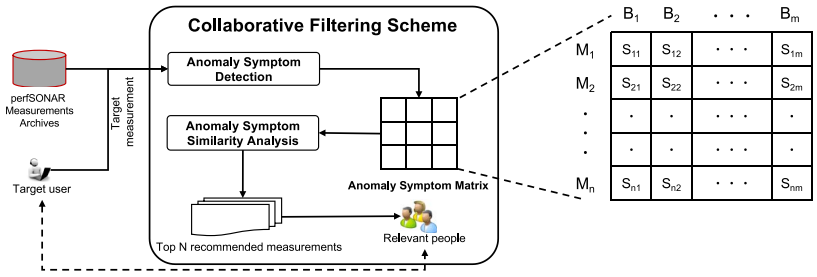


Fig. 6. Major components of *collaborative filtering* scheme, which filters out relevant measurement traces to connect relevant users to the target user.

platform for users to share experience and diagnosis skills, it could improve the efficiency of trouble-shooting anomaly events greatly. This is the reason why we leverage the concept of collaborative filtering to add a "social plane" to anomaly event detection in the overall monitoring system.

Collaborative filtering is widely used in recommendation systems, such as eBay, Amazon, to find the possible interesting products for particular users based on finding other users with similar interests, by using all the users' ratings in their database (also called as user-based collaborative filtering). Similarly, when anomaly events are found in a monitoring system, the users want to find those people who faced similar network anomaly events and find the root-cause of the anomaly events. Thus, we leverage the concept of collaborative filtering in our anomaly detection for finding those network measurement traces with similar anomaly symptoms. Our approach is also based on the reasonable assumption that the people who have encountered similar networking issues previously have a higher probability to know the root-cause of these anomaly events.

The original collaborative filtering (user-based model) usually involves two procedures: Finding the Top-N users with similar tastes; Using Top-N users' ratings for a particular item to predict a targeted user's rating. The collaborative filtering in our scenario similarly, has two steps:

- Finding those measurement traces having similar anomaly symptoms;
- Using those measurement traces to connect the potential people who may understand the network issues and provide solutions and suggestions, assuming each measurement trace is managed by the network administrators or Big Data application community users.

As shown in Figure 6, target users put their suspicious measurements traces into our collaborative filtering, and then our system pulls all measurements traces as candidate measurements from a relevant perfSONAR measurements archive. The anomaly symptom detection module scans target and candidate measurements to detect anomaly symptoms for each measurement trace. After detecting anomaly symptoms, we quantify each anomaly symptom and construct an anomaly symptom matrix. Finally, the anomaly symptom similarity analysis module filters out those measurements traces with most similar anomaly symptoms. Hence, through similar anomaly symptoms, we can connect those users who also have similar network problems such that they can diagnose collaboratively and in a trustworthy manner.

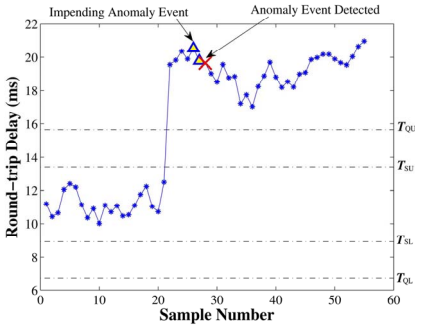


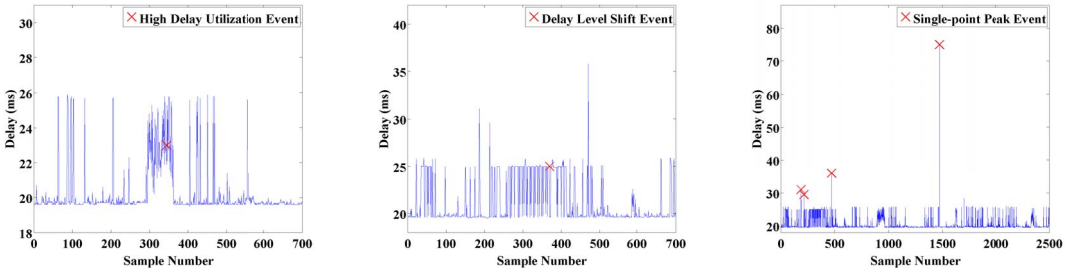
Fig. 7. Plateau-detector thresholds illustration: normal state threshold is $[T_{SL}, T_{SU}]$, and abnormal state threshold is $> T_{SU}$ or $< T_{SL}$.

In the following subsections, we describe how our collaborative filtering based recommender detects and classifies anomaly events and similarity between measurement traces.

A. Anomaly Symptom Detection

The first challenge of our collaborative filtering approach is how to detect anomaly symptoms in measurements, i.e., how to classify anomaly events based on their features. Anomaly events could be visualized and analyzed in different approaches such as time-series, clusters or heat maps. In our research, we apply anomaly detection algorithms to network time-series measurements. As described in [11], network malfunctions are caused by different reasons. Moreover, different network malfunctions may have different anomaly symptoms (or signatures) in the time-series measurements. In this section, we describe how to detect anomaly symptoms and create an anomaly profile for each anomaly symptom, which can help us quantify anomaly symptoms for collaborative filtering and content-based filtering.

In addition, our anomaly symptoms detection algorithm is based on our previous work on the Adaptive Plateau Detector algorithm (APD) [7]. Before describing the anomaly symptoms detector, we describe the working of the APD algorithm briefly. Plateau events are used to detect performance change events in network environment, which look like a "plateau" in the measurements plots, as shown in Figure 7. A plateau trigger event is detected if the most recent measurement sample value crosses the upper or lower thresholds of the summary (i.e., T_{SU}, T_{SL}) and quarantine (i.e., T_{QU}, T_{QL}) buffers as determined by the settings of sensitivity and trigger elevation parameters. The summary buffer is used to maintain sample



(a) High delay utilization (b) Delay level shift (c) Single point peak

Fig. 8. The process of adjusting different parameters in the APD configurations to detect different anomaly symptoms.

TABLE I
PARAMETERS CONFIGURATION OF APD TO DETECT ANOMALY SYMPTOMS

Parameters	High delay utilization	Delay level shift	Single point peak
summary window count (swc)	100	100	100
trigger duration (td)	50	15	1
trigger threshold (tt)	Same as original APD	1.2	1.5
trigger method (tm)	Persistent and Intermittent	Persistent	Same with original APD

history that indicates the normal state values and a quarantine buffer is used to store outlier data samples that are more than twice the normal state sample values. A plateau anomaly is triggered when the count of trigger events reaches a value of pre-configured trigger duration, which is indicated by the cross mark in Figure 7.

High delay utilization: "High delay utilization" may be caused by network congestion overload, which is a case of a significant and persistent queue backlog in one or more links along the path. As shown in Figure 8(a), the feature of "high delay utilization" has a high delay utilization duration, and we configure the parameter *trigger duration* of APD algorithm as shown in Table I to detect this symptom.

Delay level shift: The "delay level shift" may be caused by route changes or clock synchronization issue. It will be a significant change above normal baseline shown in Figure 8(b) and related configuration shown in Table I.

Single point peak: The "Single point peak" may be affected by end-points, such as NIC (Network Interface Card) buffering issue, I/O issue and OS manual operations issues. So, this symptom is usually made of a single (or few) point(s) higher than normal delay as shown in Figure 8(c) and a related configuration is shown in Table I.

For each anomaly symptom s, we quantify it as $s = (p, g)$, where the p denotes the type of anomaly symptom (such as "high delay utilization", "delay level shift"), and the value of p will be denoted as enumerative value $(0, 1, 2)$; And the g denotes the magnitude of anomaly symptom, which can be expressed as $(1 - \frac{\overline{normal}}{\overline{abnormal}})$ means the difference ratio between average value of abnormal events and average value of normal events during a certain defined window size, so the value of $g \in [0, 1]$.

Hence, after anomaly symptom detection, we obtain a set of anomaly symptoms s for each measurement trace M_i, which could be expressed as $M_i = \{s_1, s_2, \ldots, s_m\}$.

B. Anomaly Symptom Matrix

In Section V-A, we obtain a set of anomaly symptoms for each measurement trace by using anomaly symptom detection. The next step is to construct an anomaly symptom matrix D to compare similarities of measurement anomaly events.

However, constructing an anomaly symptom matrix is a challenging problem. A naive approach may scan all the anomaly symptoms and look for the most similar one to be aligned with target anomaly symptoms. However this approach is hard to operate and requires lots of computation resources.

To solve this problem, we propose using bins to align similar anomaly symptoms. This approach is easier to operate and requires lesser computation resources. The overall computation time of alignment processing could be achieved in $O(n)$. As the value of symptom magnitude g is between 0 and 1, our bin size will be defined within a 0.2 scale. However, a challenge with this approach is the lack of a way to analyze similarity with different anomaly symptom types. The value should be zero, if the types are different. Hence, we adjust them with different ranges. The magnitude of anomaly symptom S_i for particular anomaly symptom could be expressed as,

$$S_i = p_i + g_i \tag{11}$$

where p_i equals to 0, 1, or 2, so the adjustment magnitude of anomaly symptom with different anomaly type will be shifted to different range.

Hence, our anomaly symptom matrix is defined as: the X-axis has 15 number of bins with 0.2 scale for each bin; and the Y-axis is the measurements traces as shown in Figure 6.

C. Anomaly Symptom Similarity Analysis

After generating anomaly symptom matrix, we apply Pearson correlation coefficient to compute the anomaly

symptom similarity score between two measurement trace M_i and M_j, which can be expressed as:

$$M\_sim(i,j) = \frac{\sum(M_i - \overline{M_i})(M_j - \overline{M_j})}{\sqrt{\sum(M_i - \overline{M_i})^2}\sqrt{\sum(M_j - \overline{M_j})^2}} \quad (12)$$

Through anomaly symptom similarity analysis, we could filter out the measurements with most similar anomaly symptoms when compared to the target measurement, so as to connect the relevant users to the target user for expertise sharing and cooperative mingling as shown in Figure 6.

VI. Social Plane Implementation

We implemented our "social plane" approach social networking user interface in HumHub [27], which is social network open source development kit. When HumHub is deployed, it works as a common social network website as Facebook, Twitter, and it supports basic social network interfaces: users can create their own domains or communities; users can post their messages, which could be shared among their domains and their followers. We address challenges that could be faced by our proposed architecture implementation in a real deployment. The main challenge is to adapt our implementation of our proposed architecture such that it can be used routinely on a regular basis by many stakeholders. One strategy is to allow users to subscribe and receive anomaly event notifications via email pertaining to various communities' measurement resources and data archives that are of direct interest in terms of performance monitoring objectives. Another strategy is to create relevant user interfaces and visualizations that allow stakeholders to use the social plane tools to collaborate around critical anomaly events in a 'systematic' manner through knowledge sharing and expert mingling for root-cause diagnosis.

In the following, we will describe the basic features of our "social plane" implementation.

A. Content-Based Filtering Features

When users want to perform correlation analysis, the users have the option to find other measurements which are interest to them. The measurements can be searched based mainly on two objectives: Temporal and Spatial correlation analysis objectives. Based on the objectives selected, our recommendation scheme will recommend measurements which match the user's measurements. In our application, users in a particular domain only need to choose their analysis objectives (such as temporal, spatial), thereafter our system will automatically analyze a domain's measurements attributes (refer to Section IV) and recommends relevant measurements based on a user's analysis objective.

B. Collaborative Filtering Features

Users can subscribe to measurements by inputting, e.g., the IP address of measurements archive, which are related to their services. After subscribing the measurements, users can observe these measurements, their end-to-end performance.

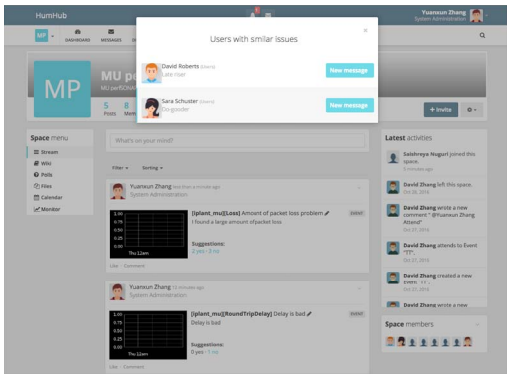


Fig. 9. Screenshot of social plane implementation in Humhub.

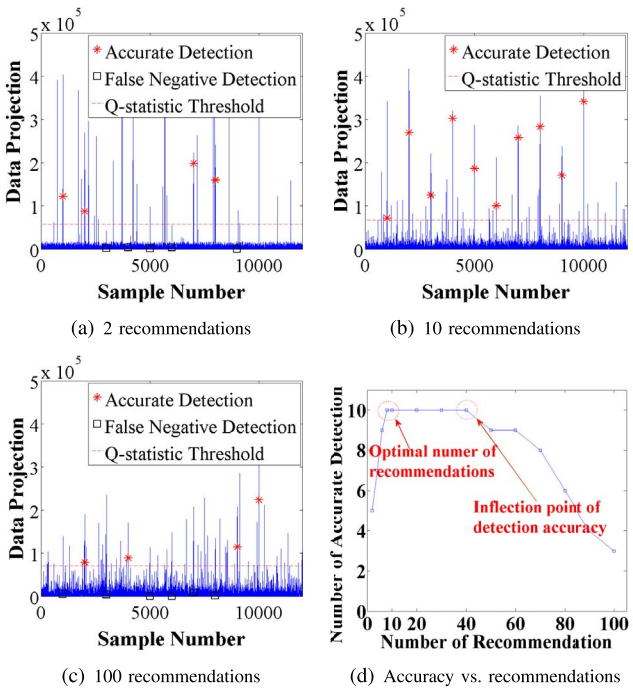
When users find any suspect events, they can post the suspect events on the social Web site by providing basic events information. The Collaborative filtering scheme is executed when users post a suspect event as described in the procedure in Section V. After collaborative filtering, the users with similar anomaly will be filtered out as shown in Figure 9. Further, users can directly send messages to those for asking for help. Consequently, those users can build up connections and share diagnosis knowledge or work collaboratively.

VII. Performance Evaluation and Results

In the evaluation section, we separately evaluate the effectiveness of *content-based filtering* and *collaborative filtering* to show the effectiveness of our measurements recommendation scheme in accurately identifying anomaly events for a user's measurements analysis objective, and accurately identifying those anomaly events with similar symptoms. Because of the challenges in finding the ground-truth of measurements in real-world deployments, we create a simulation environment to synthesize measurements and anomaly events to prove our recommendation schemes. The data sets used in our evaluation experiments are publicly accessible at [28].

A. Content-Based Filtering Recommendation Scheme Functionality With Synthetic Traces

In order to examine the effectiveness of the proposed recommendation scheme in accurately identifying anomaly events upon analysis, we perform experiments with synthetic perfSONAR data. The synthetic data is carefully generated to closely mimic the actual perfSONAR OWAMP measurement traces. And then, we inject two types of anomaly events: correlated and uncorrelated anomaly events. Correlated anomaly events are temporal correlated anomaly events, which means anomaly events from different traces happened at same time. In order to inject correlated anomaly events, we generate 100 traces and then inject anomaly events in those traces at the same time. We also inject events at random times as uncorrelated anomaly events. The percentage of anomaly events in each trace varies from 0.1%-1% of the trace sample population. The magnitudes of anomaly events vary from 10% - 60% over normal measurements with higher magnitudes causing sharper spikes.



(a) 2 recommendations (b) 10 recommendations

(c) 100 recommendations (d) Accuracy vs. recommendations

Fig. 10. Accuracy of correlated anomaly detection in terms of false alarms with varying number of recommended traces.

In the first experiment, we use the traces recommended by our scheme to detect network anomaly events using an exemplary temporal correlation analysis. For this experiment, we use different number of recommendations and see whether analysis with such recommendations can successfully identify the correlated/uncorrelated anomaly events that we injected. In Figure 10(a), 10(b), and 10(c), we show the accuracy of such anomaly event detection with our scheme recommending 2, 10 and all 100 traces, respectively. We observe that in this particular scenario, 10 recommendations accurately detect all the anomaly events with no false alarms. Whereas analysis with only 2 recommendations lack the necessary data to establish correlation, thus causing false alarms. Further, all 100 recommendations suffer from too much noise in anomaly detection caused by undesired traces resulting in false alarms. In Figure 10(d), we showed the nature of detection accuracy with the number of recommended traces exhibiting an inflection within the 10 - 40 recommendations range. Thus, we argue that there exists: an optimal number of recommendations (in this case 10) for accurate detection, and an inflection point (in this case 40) beyond which too many traces contribute to high levels of noise resulting in false alarms.

Figure 11 shows the benefits of our content filter based recommendation scheme over the greedy recommendation approach (random selection), and recommendation strategy with filtering based on partial measurement features, such as temporal aspects (time range) of the traces. For this experiment, we vary the density of anomaly events and recommend equal number of traces for each approach being compared. We see that on an average, the content filter performs consistently better than the greedy and temporal filter based approaches. It is interesting to observe that for a very small

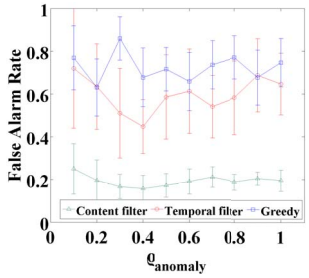


Fig. 11. False Alarm Rate comparison among the three different schemes evaluated.

density of anomaly events, the false alarm rate is higher for all approaches. This is because, too few anomaly events with minimal anomalous features are difficult to detect and are neither related to the number of recommendations nor the filtering approaches.

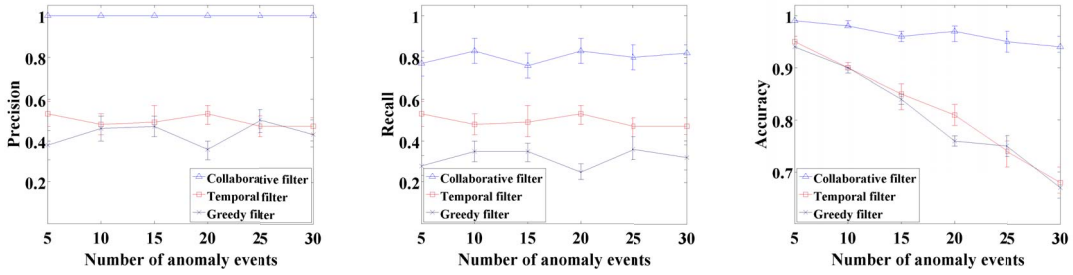
B. Collaborative Filtering Recommendation Scheme Functionality With Synthetic Traces

To evaluate our collaborative filtering scheme for finding the most similar symptoms, we perform experiments with perfSONAR data as detailed in Section VII-A. We also evaluate the efficiency of our scheme to show the overload for accuracy.

- *Effectiveness:* We use three different metrics (Precision, Recall, Accuracy) in the effectiveness evaluation. These metrics are widely used in pattern recognition and information retrieval fields.
  - *Precision* is to indicate the fraction of recommended measurement traces that have similar anomaly symptom as the target measurement trace.
  - *Recall* is to indicate the fraction of measurement traces that have similar anomaly symptom as the target measurement trace that are recommended.
  - *Accuracy* is to indicate the fraction of measurement traces that have and don't have similar measurement traces that are accurately estimated.
- *Efficiency:* We use the average running time to evaluate efficiency of our scheme.

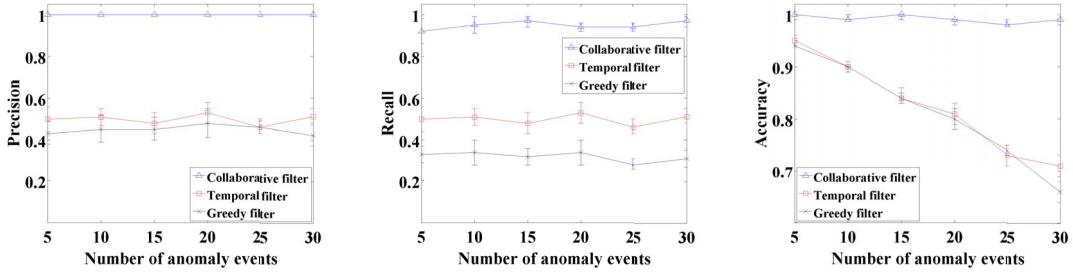
1) Effectiveness Evaluation: We compare our collaborative filtering scheme with other two schemes: temporal filtering and greedy filtering schemes. temporal filtering scheme recommends measurements with similar anomaly symptoms by checking temporal correlation level; and greedy filtering scheme recommends measurements by random selection.

In order to show the effectiveness of our scheme in dealing with each anomaly symptom, we separately evaluate our scheme in dealing with each anomaly symptom in terms of different number of anomaly events injected in candidate measurement traces. First, we select one measurement trace as a target measurement trace and 100 measurement traces as candidate measurement traces. Second, for each evaluation scenario, we inject one anomaly symptom into the target measurement trace, and then we randomly select number of measurement traces from candidate measurement traces to inject different types of anomaly symptoms.



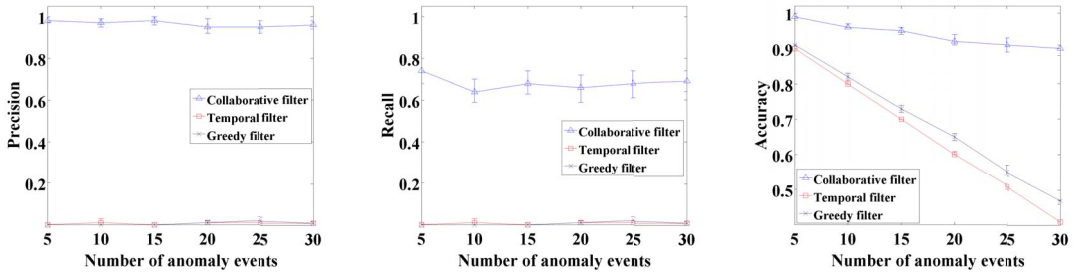
(a) Precision (b) Recall (c) Accuracy

Fig. 12. Results of recommendation schemes to find anomaly symptom: "high delay utilization" with varying number of anomaly events.



(a) Precision (b) Recall (c) Accuracy

Fig. 13. Results of recommendation schemes to find anomaly symptom: "delay level shift" with varying number of anomaly events.



(a) Precision (b) Recall (c) Accuracy

Fig. 14. Results of recommendation schemes to find anomaly symptom: "single point peak" with varying number of anomaly events.

For example, as shown in Figure 12, we inject the "high delay utilization" anomaly event into target measurement trace and inject different types anomaly symptoms to candidate measurement traces. The goal of these schemes is to find those measurements in candidate measurement traces that have similar anomaly symptoms as the target measurements. In Figure 12, we show that our collaborative filtering scheme has much better performance than the other two schemes with respect to the *Precision* and *Recall* metrics. We see that the results won't change as the number anomaly events are increased. However, in Figure. *Accuracy* the collaborative filtering scheme also has higher performance, but it is not obvious as the previous two metrics. The reason for why temporal filtering and greedy filtering also have good accuracy is because of the number of true negatives, which means without any positive predication it can achieve 0.95 accuracy if there are 5 measurement traces with similar anomaly symptom. Also, the accuracy result decreases as the number of anomaly events increase. The other two evaluation scenarios for "delay level shift" in Figure. 13 and "single peak point" in Figure. 14 show the similar results. We also find that the "Temporal filter" and "Greedy filter" schemes are unable to make correct positive predictions, so the "precision" and "recall" results are almost close to zero.

2) Efficiency Evaluation: In the efficiency evaluation studies, we compare the computation time for finding similar anomaly symptoms among these three approaches in terms of different number of measurements traces. Table II shows scalability results of our filtering algorithms when used on a large number of traces in a production scale environment. Specifically, we show computation time performance comparison of our filtering algorithm with greedy filtering and temporal filtering from 100 to 100,000 measurement traces. For this, we use different Amazon Web Services instances (see descriptions provided in Table III) for analyzing scalability. We can observe that the computation time increases linearly as the number of measurements traces increases. Also, the overhead of our algorithm is minimal in production scale

TABLE II
COMPUTATION TIME (SECONDS) COMPARISON OF COLLABORATIVE FILTERING (CF), GREEDY FILTERING (GF) AND TEMPORAL FILTERING (TF)

Instance Type	100 Traces			1000 Traces			10000 Traces			100000 Traces		
	CF	GF	TF	CF	GF	TF	CF	GF	TF	CF	GF	TF
t2.small	4	3	2	33	20	21	340	209	211	22492	16347	16936
c4.large	3	2	2	28	18	17	277	175	174	3081	1942	1890

TABLE III
DESCRIPTIONS OF AMAZON EC2 INSTANCES; *Legend:* ECU (EC2 COMPUTE UNIT), GIB (GIBIBITE), EBS (ELASTIC BLOCK STORE)

Instance Type	Descriptions
t2.small	Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 2 GiB memory, EBS only
m3.medium	3 ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon E5-2670v2, 3.75 GiB memory, 1 x 4 GiB Storage Capacity
c4.large	8 ECUs, 2 vCPUs, 2.9 GHz, Intel Xeon E5-2666v3, 3.75 GiB memory, EBS only

environments and is on the order of a few seconds even for several thousands of measurement traces. The computation time results of greedy filtering and temporal filtering are similar, and the computation time of collaborative filtering is ≈ 1.5 times larger than the other two schemes. This is an expected result, because the greedy and temporal filtering schemes need to run the APD algorithm only once. Whereas, our APD algorithm runs $1 \sim 3$ times in the case of the collaborative filtering scheme depending on how we randomly inject three different anomaly symptoms in the target measurement trace for efficiency evaluation.

VIII. APPLICATION TESTBED CASE STUDIES

In this application testbed case studies section, we will show how our collaborative filtering scheme creates a "social plane" for helping application users or network administrators to share experience and diagnose collaboratively. We also show how our content-based filtering helps users find best matched measurements based on their measurement analysis objectives.

A. Case Studies of Collaborative Filtering in a Social Plane

SoyKB [8] is a comprehensive Web resource developed at University of Missouri (MU) for soybean translational genomics and breeding, which handles the integration of soybean genomics and multi-omics data along with gene function annotations, biological pathway and trait information. The SoyKB has been featured as a model distributed computing use case in the systems biology area within the Open Science Grid community. It represents a Big Data application in bioinformatics due to the large volume of distributed data sets that need to be integrated and analyzed.

In order to simulate various network issues in a Big data application, we use the GENI Cloud Testbed [9] to develop our case studies. First, we simulate the SoyKB Big Data application [2] related network environment in GENI. As shown in Figure 15, SoyKB Application imports raw data from other counties (such as China, Brazil). Researchers at MU use iPlant and TACC resources to store and process. They may also choose public cloud (AWS, Azure) to achieve better performance. Same storage (iPlant) and computing (TACC) resources could also be used by researchers from Iowa and Arkansas. Second, we deploy an end-to-end monitoring infrastructure to collect measurements using our OnTimeMeasure software for GENI [24]. The measurements are collected by a 'Measurements engine' shown in Figure 15. Third, we use the "netem" command [26] to control delay values and duration time to simulate different anomaly symptoms, as shown in Table IV. Note that the arrows in Table IV denote the increase in the value of "delay" to simulate different types of anomaly symptoms. Lastly, we use our collaborative filtering scheme to create a social plane for sharing knowledge or collaborative diagnosis in a trustworthy manner.

TABLE IV
SIMULATION ANOMALY SYMPTOMS WITH "NETEM" COMMANDS

Anomaly symptom type	variation	duration(s)
High delay utilization	$\uparrow (20 \sim 40)\%$	3600s
Delay level shift	$\uparrow (55 \sim 60)\%$	Persistence
Single point peak	$\uparrow 60\%$	10s

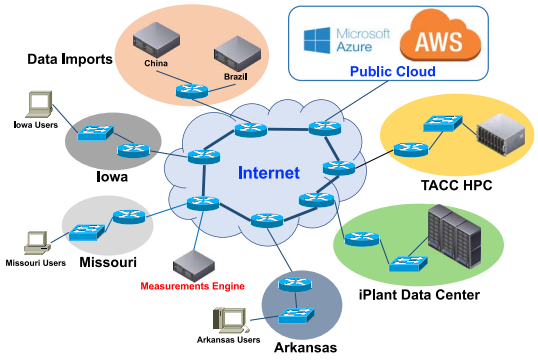


Fig. 15. Physical simulation of SoyKB application testbed in GENI Cloud Infrastructure.

We will now present a few scenarios which are most common cases in scientific Big Data applications, such as different user experiences with same applications; issues located at sources (e.g., users' side) which may impact on those users' application performance; and issues located at sinks (e.g., data centers or computation centers) which may impact on most users' application performance.

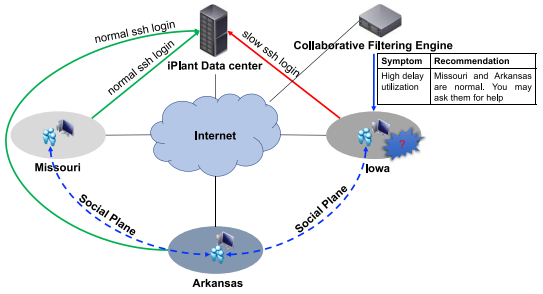


Fig. 16. Logical illustration of User Experience bottleneck case: users in "Iowa" feel the high latency SSH operation on iPlant server.

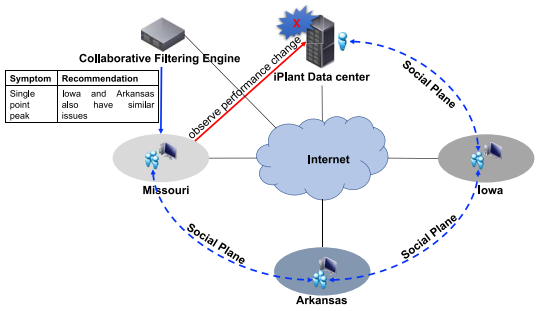


Fig. 17. Logical illustration of data center issue case: data center iPlant has some issues, which would affect all the iPlant users.

TABLE V
REAL PERFSONAR MEASUREMENTS TRACES' ATTRIBUTES DESCRIPTION

Trace Name	Metric	Periodicity	Time Range
bnl↔fnal	**One-way delay**	**[58, 62]**	**09-01 00:57 ↔ 09-09 23:58**
lbl↔ornl	One-way delay	[51, 68]	09-03 05:39 ↔ 09-03 22:19
aofa↔bost	One-way delay	[51, 161]	09-01 00:00 ↔ 09-01 05:35
bnl↔bois	One-way delay	[57, 63]	09-01 00:00 ↔ 09-09 23:59
sacr↔bois	One-way delay	[53, 150]	09-01 00:00 ↔ 09-01 16:42
bnl ↔bost	One-way delay	[53, 67]	09-01 00:00: ↔ 09-01 05:35
hous ↔srs	One-way delay	[53, 69]	09-05 05:31 ↔ 09-05 22:08
bnl ↔lbl	One-way delay	[61, 66]	09-01 00:00 ↔ 09-09 23:59
newy ↔sacr	Throughput	[54, 67]	09-01 00:00 ↔ 09-09 23:59
anl ↔newy	Throughput	[52, 64]	09-01 00:01 ↔ 09-09 23:59
bnl ↔nash	Throughput	[57, 71]	09-03 05:58 ↔ 09-03 22:38
denv ↔fnal	Throughput	[58, 67]	09-01 00:00 ↔ 09-09 23:59

TABLE VI
MEASUREMENT ATTRIBUTES SIMILARITY SCORE DESCRIPTION

Trace Name	Topology Similarity	Metric Similarity	Alignment Similarity	Time Range Similarity
lbl↔ornl	0	1	0.456	0.077
aofa↔bost	0.077	1	0	0.026
bnl↔bois	0.385	1	0.999	1
sacr↔bois	0	1	0.103	0.077
bnl↔bost	0.4	1	0	0.026
hous↔srs	0	1	0.767	0.53
bnl↔lbl	0.333	1	0.999	1
newy↔sacr	0.125	0	0.999	1
anl↔newy	0	0	1	0.999
bnl↔nash	0.143	0	0.979	0.862
denv↔fnal	0.154	0	0.999	1

1) User Experience Issue: User experience issue suggests that a user has bad or unsatisfactory experiences of application performance, such as long data transfer times, excessive execution times, or slow SSH login speed. The cause of those issues are usually complicated and irregular, which may be caused by mis-configurations, network bottlenecks, or firewall security issues.

As shown in Figure 16, the users in Iowa find the speed of using SSH to login into the iPlant data center for SoyKB related computations is very slow. After analysis using our collaborative filtering engine (labeled in Figure 16), the engine detects a "high delay utilization" symptom in the measurement trace from "Iowa" to "iPlant", and the engine also finds measurements traces from "Missouri" or "Arkansas" to iPlant without this kind of issues. So, the engine may recommend users in "Iowa" ask users in "Missouri" or "Arkansas" for help, so that a "social plane" is created among those users for sharing knowledge or collaborative troubleshooting. This example also could be the scenario of issues located at sources, because only certain users face them.

2) Data Center Issue: Data center issue (or called issues located at data sink) means that the data center faces some issues which are caused by data centers' device failures or link failures, as a result of which, many users or entities will be affected. Those issues may not directly impact on user's experiences, however, those issues may be manifest as notable changes in measurements, which catch users' attention.

As shown in Figure 17, the users in Missouri observe the performance change in their measurements. After analysis with our collaborative filtering engine, we detect a "Single point peak" symptom in the measurement trace from "Missouri" to "iPlant", and also detect similar anomaly symptoms in measurements traces from "Arkansas" or "iPlant" and from "Iowa" to "iPlant". Consequently, the engine may suggest diagnosing collaboratively work with users in "Arkansas" and "Iowa". And because they have same sink (iPlant), they may also check this issues with administrators of iPlant. Consequently, a "social plane" is created among those users and service provider for sharing knowledge or collaborative troubleshooting of the root-cause issue.

B. Case Studies of Studies of Content-Based Filtering Scheme With Real Traces

We collect hundreds of perfSONAR traces from National labs and ESnet sites with perfSONAR end-points with different measurement attributes as inputs to our proposed measurement recommendation scheme. In Table V, we show the attributes for only a small subset of collected samples. In this subset, we have kept the trace BNL↔FNAL as the target trace (in bold font) and the rest as candidate traces. Applying our proposed recommendation scheme, we seek to find the most relevant traces for two separate measurement analysis objectives: a) Temporal analysis for correlated anomaly event detection [25] where complete topology information is not available; and b) Spatial analysis for topology-aware correlated anomaly event detection [23]. We remark that recommendations in a spatial analysis context are as useful as the amount of topology details available between the various measurement end-points. In reality, it is not always possible to obtain topology information for the measurement traces in the publicly

TABLE VII
DATA SANITY SCORE AND DOMAIN REPUTATION RESULTS FOR SELECTED TRACES USED IN EXEMPLAR ANALYSIS CASE STUDY

Trace	Sanity	Temporal Correlation Analysis		Spatial Correlation Analysis		Domain Reputation	
Name	Score	Overall Similarity Score	Ranking	Overall Similarity Score	Ranking	Source	Destination
bnl↔bois	0.993	0.938	**1**	0.938	**1**	0.983	0.984
bnl↔lbl	**0.642**	0.933	**2**	0.933	**2**	0.985	**0.611**
denv↔fnal	**0.972**	**0.765**	3	**0.327**	5	0.991	0.984
newy↔sacr	0.982	0.762	4	0.312	7	0.979	0.986
anl↔newy	0.984	0.221	5	0.273	10	0.987	0.986
bnl↔bost	0.953	0.197	10	0.453	3	0.983	0.964
hous↔srs	0.976	0.666	7	0.418	4	0.934	0.986

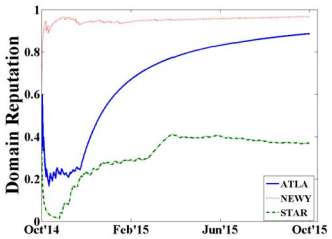


Fig. 18. Historical reputation characteristics comparison of 3 exemplar DOE lab with real perfSONAR traces over one year period.

available perfSONAR data archives. Consequently, recommendations might not be useful for root-cause diagnosis in cases where there is an absence of topology information.

Table VI shows the measurement attributes similarity scores for the candidate measurement traces with the target trace described in Table V. The score evaluations follow the scheme described in Section IV and due to the varied attributes of the collected traces, we observe that the similarity scores of candidate traces based on different attributes can vary by a considerable margin.

In Figure 18, we show the historical reputation characteristic comparison of 3 exemplar National Lab sites based on one year (Oct 2014 - Oct 2015) traces' data sanity scores using the scheme discussed earlier. We observe that although these are well known and seemingly reputed perfSONAR end-points within the Big Data application communities deploying perfSONAR, not all the sites produce trustworthy data at all times. Thus, we establish the need for a domain's reputation as a key factor in subscribing to that domain's measurement data for accurate detection and effective troubleshooting. For example, according to Figure 18, subscribing to data from domain 'NEWY and 'ATLA' is likely to yield data with high veracity; whereas, subscribing to STAR data may not always lead to accurate correlated anomaly event detection and diagnosis.

Subsequently, we apply the relative weights of the measurement attributes on the similarity scores based on the two monitoring objectives: temporal and spatial. For temporal analysis, we focus on detecting correlated anomaly events in time series measurements where measurement metric needs to be of similar type. Hence, we follow the path: 'Analysis Objective' → 'Temporal' → 'Short-term Change' → 'Measurements Metric Specific', and assign the weights according to the rule $w_a > w_r > w_m > w_t$. Whereas for spatial analysis, focus is on using measurements topological information to drill down the location of events. Hence, it follows the path: 'Analysis Objective' → 'Spatial' → 'Measurements Metric Specific' → 'Short-term Change', with final relative weights following the rule $w_t > w_m > w_a > w_r$.

The final recommendation outcomes and corresponding ranking of a subset of candidate traces are shown in Table VII along with the traces' data sanity scores, and corresponding source and destination domains' reputation scores. Table VII is a snapshot of the actual manifestation of our proposed recommendation scheme to assist the operators and application users to better gauge the relevance and veracity of collected samples. For example, the Trace BNL↔BOIS (shown in Table VII) is the best choice among the candidates (1^{st} ranked) in terms of similarity and high instantaneous data sanity score (0.993). Whereas, Trace BNL↔LBL, although being 2^{nd} ranked for both temporal and spatial analysis, may not be a good choice for candidacy as the low sanity score (0.642) suggests sub-par data quality which can be attributed to low reputation (0.611) of the destination domain (LBL). Thus, operators should be advised to use Trace DENV↔FNAL over Trace BNL↔LBL for temporal analysis as the data quality of the former is much better (0.972) in spite of having a slightly lower similarity (0.765). However, DENV↔FNAL will not be a significantly better choice over BNL↔LBL for spatial analysis due to the former's very low similarity score (0.327).

IX. CONCLUSION

In this paper, we showed the need of a "social plane" for network operators and users to effectively troubleshoot bottlenecks affecting Big Data applications in a trustworthy manner. We leveraged the concept of recommendation system to enhance measurements sharing/subscription, diagnosis expertise sharing and collaborations. Using content-based filtering, we filter and rank best relevant measurement traces from a pool of candidate traces. The recommendation scheme was complemented by a Bayesian Inference based domain reputation calculation scheme that indicates the trustworthiness of the collected samples among the involved domains. With collaborative filtering, we find those measurement traces having similar anomaly symptoms with target measurements, and through measurements traces, we can connect those people for sharing knowledge, or working collaboratively in a trustworthy manner. Using real measurements traces and synthetic events, we showed how our content-based filtering scheme enables operators to intelligently use less but relevant measurement samples to accurately detect and diagnose performance bottleneck causing network events. In addition, we showed how

our collaborative filtering scheme finds similar anomaly issues efficiently and effectively.

Our future work is to adapt our social plane approach to be more deeply integrated into diverse Big Data application communities, to help them better socialize around measurements and achieve expected performance.

REFERENCES

[1] L. Evans and P. Bryant, "LHC machine," *J. Instrum.*, vol. 3, Aug. 2008, Art. no. S08001. [Online]. Available: http://lhc.web.cern.ch/lhc

[2] Y. Liu *et al.*, "PGen: Large-scale genomic variations analysis workflow and browser in SoyKB," *BMC Bioinformat.*, vol. 17, no. 13, p. 337, 2016, doi: 10.1186/s12859-016-1227-y.

[3] A. Hanemann *et al.*, "PerfSONAR: A service oriented architecture for multi-domain network monitoring," in *Proc. 3rd Int. Conf. Service Oriented Comput.*, Amsterdam, The Netherlands, 2005, pp. 241–254.

[4] V. Paxson, "Strategies for sound Internet measurement," in *Proc. 4th ACM SIGCOMM Conf. Internet Meas.*, Taormina, Italy, 2004, pp. 263–271.

[5] Y. Zhang, S. Debroy, and P. Calyam, "Network-wide anomaly event detection and diagnosis with perfSONAR," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 666–680, Sep. 2016.

[6] M. A. Marvasti, A. V. Poghosyan, A. N. Harutyunyan, and N. M. Grigoryan, "An enterprise dynamic thresholding system," in *Proc. 11th Int. Conf. Auton. Comput. (ICAC)*, 2014, pp. 129–135.

[7] P. Calyam, J. Pu, W. Mandrawa, and A. Krishnamurthy, "OnTimeDetect: Dynamic network anomaly notification in perfSONAR deployments," in *Proc. IEEE Int. Symp. Model. Anal. Simulat. Comput. Telecommun. Syst. (MASCOTS)*, Miami Beach, FL, USA, 2010, pp. 328–337.

[8] T. Joshi *et al.*, "Soybean knowledge base (SoyKB): A Web resource for integration of soybean translational genomics and molecular breeding," *Nucleic Acids Res.*, vol. 42, no. D1, pp. D1245–D1252, Jan. 2014, doi: 10.1093/nar/gkt905.

[9] M. Bermana *et al.*, "GENI: A federated testbed for innovative network experiments," *Comput. Netw.*, vol. 61, pp. 5–23, Mar. 2014. [Online]. Available: http://www.geni.net/

[10] R. Govindan, I. Minei, M. Kallahalla, B. Koley, and A. Vahdat, "Evolve or die: High-availability design principles drawn from Google's network infrastructure," in *Proc. Conf. ACM SIGCOMM*, Florianópolis, Brazil, 2016, pp. 58–72.

[11] P. Kanuparthy and C. Dovrolis, "Pythia: Diagnosing performance problems in wide area providers," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*, Philadelphia, PA, USA, 2014, pp. 371–382.

[12] K.-H. Kim, H. Nam, V. Singh, D. Song, and H. Schulzrinne, "DYSWIS: Crowdsourcing a home network diagnosis," in *Proc. 23rd Int. Conf. Comput. Commun. Netw. (ICCCN)*, Shanghai, China, 2014, pp. 1–10.

[13] M. Grigoriev *et al.*, "E-center: A collaborative platform for wide area network users," in *Proc. Int. Conf. Comput. High Energy Nucl. Phys. (CHEP)*, 2012, Art. no. 042025.

[14] R. Soundararajan, E. Celebi, L. Spracklen, H. Muppalla, and V. Makhija, "A social media approach to virtualization management," *VMWare Tech. J.*, vol. 1, no. 2, pp. 59–68, Dec. 2012.

[15] Y. Tang, E. Al-Shaer, and K. Joshi, "Reasoning under uncertainty for overlay fault diagnosis," *IEEE Trans. Netw. Service Manag.*, vol. 9, no. 1, pp. 34–47, Mar. 2012.

[16] L. Xiong and L. Liu, "A reputation-based trust model for peer-to-peer e-commerce communities," in *Proc. IEEE Conf. E Commerce*, Newport Beach, CA, USA, 2003, pp. 275–284.

[17] L. Mui, M. Mohtashemi, and A. Halberstadt, "A computational model of trust and reputation," in *Proc. 35th Annu. Hawaii Int. Conf. Syst. Sci. (HICSS)*, vol. 7, 2002, pp. 2431–2439.

[18] A. Jøsang and R. Ismail, "The beta reputation system," in *Proc. 15th Bled Conf. Electron. Commerce*, vol. 5, 2002, pp. 2502–2511.

[19] I. Ivanov, P. Vajda, J.-S. Lee, and T. Ebrahimi, "In tags we trust: Trust modeling in social tagging of multimedia content," *IEEE Signal Process. Mag.*, vol. 29, no. 2, pp. 98–107, Mar. 2012.

[20] S. L. Lim and A. Finkelstein, "StakeRare: Using social networks and collaborative filtering for large-scale requirements elicitation," *IEEE Trans. Softw. Eng.*, vol. 38, no. 3, pp. 707–735, May/Jun. 2012.

[21] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, San Diego, CA, USA, 2012, pp. 448–456.

[22] Y. Zhang, S. Debroy, and P. Calyam, "Network measurement recommendations for performance bottleneck correlation analysis," in *Proc. IEEE Int. Symp. Local Metropolitan Area Netw. (LANMAN)*, Rome, Italy, 2016, pp. 1–7.

[23] P. Calyam, M. Dhanapalan, M. Sridharan, A. Krishnamurthy, and R. Ramnath, "Topology-aware correlated network anomaly event detection and diagnosis," *J. Netw. Syst. Manag.*, vol. 22, no. 2, pp. 208–234, Apr. 2014.

[24] P. Calyam *et al.*, "Enabling performance intelligence for application adaptation in the future Internet," *J. Commun. Netw.*, vol. 13, no. 6, pp. 591–601, Dec. 2011. [Online]. Available: http://ontime.rnet.missouri.edu/

[25] Y. Zhang, P. Calyam, S. Debroy, and M. Sridharan, "PCA-based network-wide correlated anomaly event detection and diagnosis," in *Proc. 11th Int. Conf. Design Rel. Commun. Netw. (DRCN)*, Kansas City, MO, USA, 2015, pp. 149–156.

[26] *NetEM Provides Network Emulation Functionality for Testing Protocols by Emulating the Properties of Wide Area Networks*. Accessed: Nov. 2017. [Online]. Available: https://wiki.linuxfoundation.org/networking/netem

[27] *HumHub: Open Source Social Network Kit*. Accessed: Nov. 2017. [Online]. Available: https://www.humhub.org/en

[28] *Publicly Accessible perfSONAR Data Sets Used in Evaluation Experiments*. Accessed: Nov. 2017. [Online]. Available: https://github.com/zhangyuanxun/OnTimeSocial.git

Yuanxun Zhang received the B.E. degree from Southwest Jiaotong University, China, in 2006. He is currently pursuing the Ph.D. degree from the University of Missouri–Columbia. His research interests include network performance monitoring, software-defined networking, and big data analytics.

Prasad Calyam (SM'14) received the M.S. and Ph.D. degrees from the Department of Electrical and Computer Engineering, Ohio State University, in 2002 and 2007, respectively. He is currently an Assistant Professor with the Department of Computer Science, University of Missouri–Columbia. His current research interests include distributed and cloud computing, computer networking, and cyber security.

Saptarshi Debroy (M'09) received the B.Tech. degree from the West Bengal University of Technology, India, in 2006, the M.Tech. degree from Jadavpur University, India, in 2008, and the Ph.D. degree in computer engineering from the University of Central Florida, in 2014. He is currently an Assistant Professor with the Department of Computer Science, City University of New York. His current research interests include cyber security, cloud computing, and wireless networks.

Sai Shreya Nuguri received the B.Tech. degree from the BNM Institute of Technology, Bengaluru, India, in 2015. She is currently pursuing the M.S. degree with the University of Missouri–Columbia. Her current research interests include cloud computing, big data analytics, computer networking, and security.