DOI: 10.3969/j. issn. 1673-5188. 2016. 02. 006
http://www.cnki.net/kcms/detail/34.1294.TN.20160405.1012.002.html, published online April 5, 2016

A Survey on Event Mining for ICT Network Infrastructure Management

LIU Zheng, LI Tao, and WANG Junchang
(Nanjing University of Posts and Telecommunications, Nanjing 210023, China)

Abstract

Managing large-scale complex network infrastructures is challenging due to the huge number of heterogeneous network elements. The goal of this survey is to provide an overview of event mining techniques applied in the network management domain. Event mining includes a series of techniques for automatically and effectively discovering valuable knowledge from historical event/log data. We present three research challenges (i.e., event generation, root cause analysis, and failure prediction) for event mining in network management and introduce the corresponding solutions. Event generation (i.e., converting messages in log files into structured events) is the first step in many event mining applications. Automatic root cause analysis can locate the faulty elements/components without the help of experienced domain experts. Failure prediction in proactive fault management improves network reliability. The representative studies to address the three aforementioned challenges are reviewed and their main ideas are summarized in the survey. In addition, our survey shows that using event mining techniques can improve the network management efficiency and reduce the management cost.

Keywords

event mining; failure prediction; log analysis; network infrastructure management; root cause analysis

1 Introduction

Nowadays in China, there are more than six hundred million netizens [1]. On April 11, 2015, the number of simultaneous online users of the Chinese instant message application QQ reached two hundred million [2]. The fast growth of the Internet pushes the rapid development of information technology (IT) and communication technology (CT). Many traditional IT service and CT equipment providers are facing the fusion of IT and CT in the age of digital transformation, and heading toward ICT enterprises. Large global ICT enterprises, such as Apple, Google, Microsoft, Amazon, Verizon, and AT&T, have been contributing to the performance improvement of IT service and CT equipment.

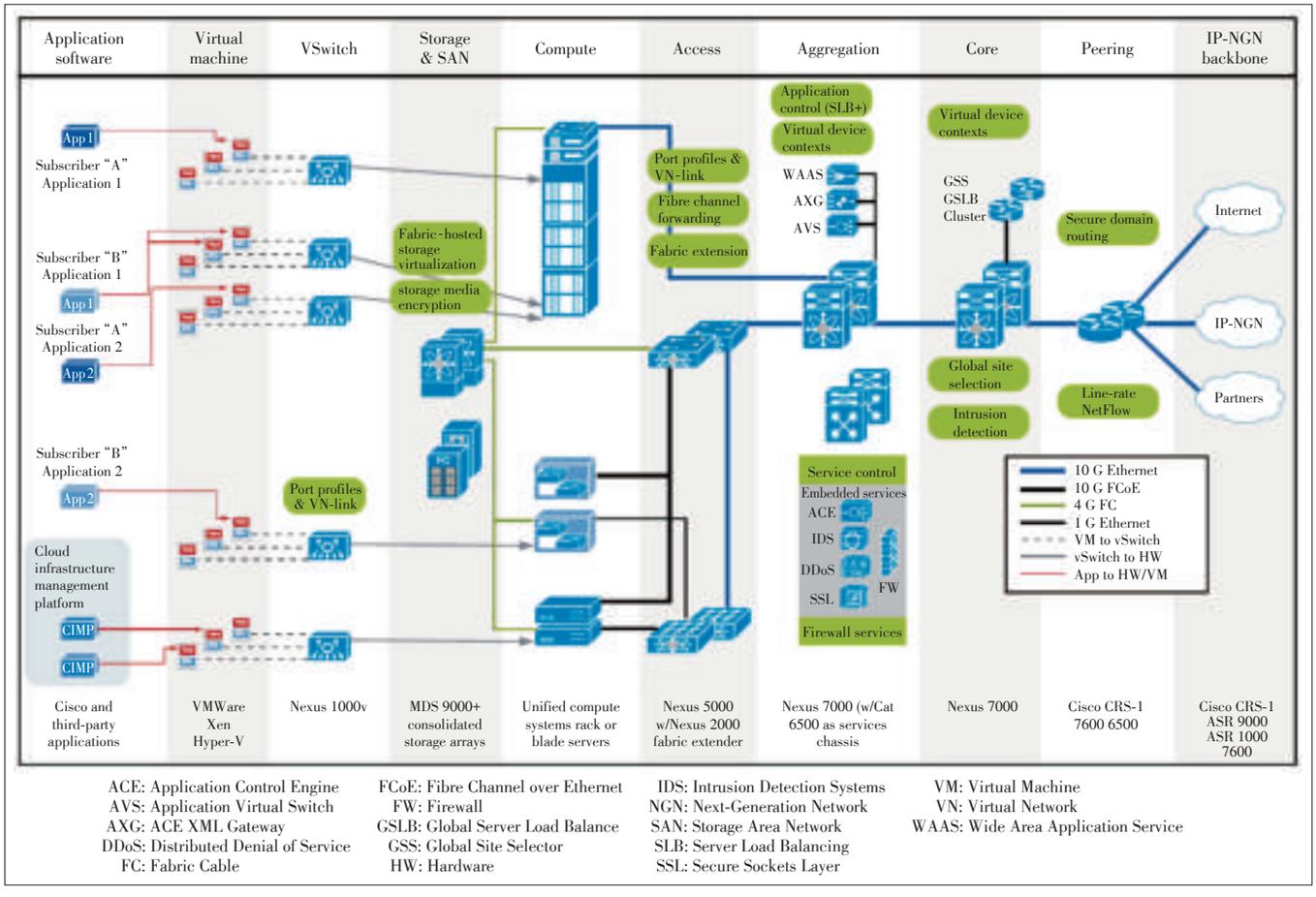
As a result, the performance of IT service and CT equipment has become increasing powerful. The speed of the world's top high-performance computing system, Chinese Tianhe-2 supercomputer, is 33.86 petaflop [3]. The data I/O of a modern Internet backbone router is more than tens of terabytes per seconds, while its routing table usually consists of millions of routes. The scale of modern networks becomes larger and larger. A global information grid [4] built by the US military is capable of collecting, processing, storing, disseminating and managing information from more than two million nodes.

These large-scale, high-performance ICT networks are supported by ICT network infrastructures. ICT network infrastructure refers to the combination of all computing and network hardware components, as well as software resources of an ICT network. Computing hardware components include computing servers, storage systems, etc. Network hardware components include routers, switches, LAN cards, etc. Software resources include virtual machine platforms, operating systems, security applications, network operation and management platforms, etc. These resources facilitate the communications and services between users and service providers.

The network infrastructure of a large ICT enterprise, e.g., a world-wide online shopping company like Amazon, usually has several world-wide data centers. Each data center has tens of thousands of servers, switches, routers, firewalls, as well as other affiliated systems like power supply systems or cooling systems. A typical architecture of data centers is shown in **Fig. 1** [5]. The ICT network infrastructure for Carriers is even more complex. For example, besides data centers, there are nation-wide communication networks in a 3G/4G network infrastructure (**Fig. 2**) [6]. Each communication network includes access network equipment, core network equipment, transport network equipment, and other application systems, containing tens of thousands of network elements that provide authentication, billing, data/voice communications, and multimedia services. These large-scale complex networks introduce many difficulties in designing, architecting, operating, and maintaining the corresponding network infrastructures, on which multiple

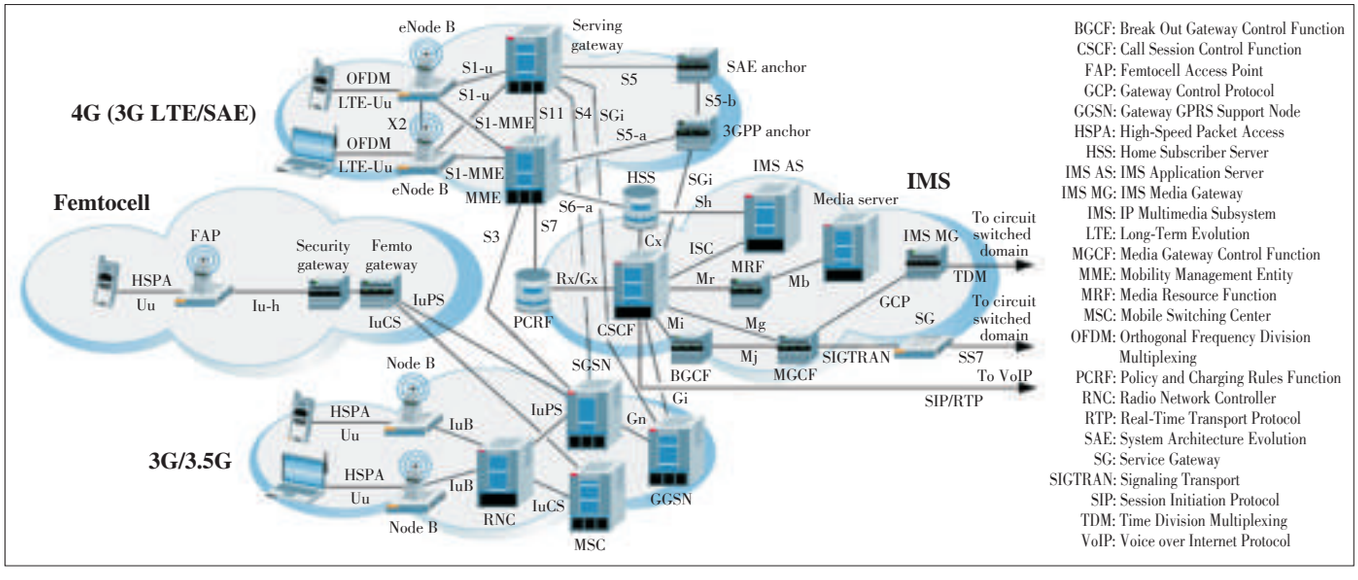
This work was supported in part by Ministry of Education / China Mobile joint research grant under Project No. 5-10, and Nanjing University of Posts and Telecommunications under Grants No. NY214135 and NY215045.

A Survey on Event Mining for ICT Network Infrastructure Management
LIU Zheng, LI Tao, and WANG Junchang



ACE: Application Control Engine
AVS: Application Virtual Switch
AXG: ACE XML Gateway
DDoS: Distributed Denial of Service
FC: Fabric Cable
FCoE: Fibre Channel over Ethernet
FW: Firewall
GSLB: Global Server Load Balance
GSS: Global Site Selector
HW: Hardware
IDS: Intrusion Detection Systems
NGN: Next-Generation Network
SAN: Storage Area Network
SLB: Server Load Balancing
SSL: Secure Sockets Layer
VM: Virtual Machine
VN: Virtual Network
WAAS: Wide Area Application Service

▲Figure 1. The typical architecture of a data center [5].



BGCF: Break Out Gateway Control Function
CSCF: Call Session Control Function
FAP: Femtocell Access Point
GCP: Gateway Control Protocol
GGSN: Gateway GPRS Support Node
HSPA: High-Speed Packet Access
HSS: Home Subscriber Server
IMS AS: IMS Application Server
IMS MG: IMS Media Gateway
IMS: IP Multimedia Subsystem
LTE: Long-Term Evolution
MGCF: Media Gateway Control Function
MME: Mobility Management Entity
MRF: Media Resource Function
MSC: Mobile Switching Center
OFDM: Orthogonal Frequency Division Multiplexing
PCRF: Policy and Charging Rules Function
RNC: Radio Network Controller
RTP: Real-Time Transport Protocol
SAE: System Architecture Evolution
SG: Service Gateway
SIGTRAN: Signaling Transport
SIP: Session Initiation Protocol
TDM: Time Division Multiplexing
VoIP: Voice over Internet Protocol

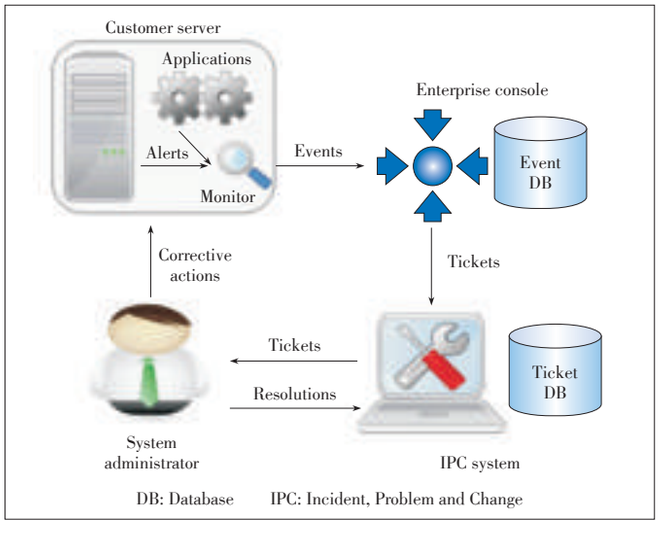
▲Figure 2. An example of 3G/4G network infrastructure [6].

complex systems are coordinated to ensure that the computation and communication functions work smoothly. Cloud technology is widely used in modern ICT network infrastructures due to the development of virtualization technology and its low

cost. But cloud technology also brings hierarchy and heterogeneity to network infrastructures. During the operation and maintenance of network infrastructures, equipment failure, communication error and system misconfiguration have high impact on the reliability of the whole network [7]–[9], as a result of unstable upper-level service and business. Traditionally, system administrators resolve the aforementioned incidents according to the workflow consisting of detection, localization and repair, by using network tools such as ping, traceroute, and tcpdump, or network monitor toolkits such as Nagios [10], Zabbix [11], and OpsView [12]. This process has been well-known and experienced as a labor-intensive and error-prone process and may not be effective when the systems/networks become large and complex.

Fortunately, several industry organizations have already paid attention to these issues and put lots of efforts on making specifications related to best practices in operating and maintaining large-scale complex systems/networks. In the IT service area, Information Technology Infrastructure Library (ITIL) [13] is a collection of specifications for service management, with which the best practices are organized according to the full life cycle of IT services including incident management, failure management, problem management, configuration management, and knowledge management. In the carrier service area, international organizations, such as ITU-T [14] and TM Forum [15], also make recommended specifications for managing telecommunication network infrastructures, partial ideas of which are borrowed from ITIL.

Fig. 3 shows a general workflow of problem detection, determination and resolution for IT service providers prescribed by the ITIL specifications [16]. The workflow aims at resolving incidents and quickly restoring the provision of services while relying on monitoring or human intervention to detect the malfunction of a component [16]. For problem detection, there is



▲**Figure 3. A general workflow of problem detection, determination and resolution [16].**

usually monitoring software running on servers or network elements, which continuously monitors the status of network elements and detects possible problems by computing metrics for the hardware and software performance at regular intervals. The monitoring software would issue an alert if those metrics are not acceptable according to predefined thresholds, known as monitoring situations, and emits an event if the alert does not disappear after a period. All events coming from the network infrastructure are consolidated in an enterprise console, where these events are analyzed and corresponding incident tickets are created, if necessary, in an Incident, Problem, and Change (IPC) system. System administrators are responsible for the problem determination and resolution based on the detailed information in these tickets. The efficiency of these resources is critical for the provisioning of the services [17].

However, the best practices in those specifications only provide the guidance on operating and maintaining network infrastructure, which is a standard workflow of consecutive procedures and definitions. Many key issues in these procedures are not answered in these specifications, especially in large-scale complex networks. The challenges in managing large-scale network infrastructures are listed as follows:

1) Large complex network infrastructures are heterogeneous and often consist of various network elements made by different equipment makers. There are different software components running on the various network elements and generating huge amount of messages and alerts in different types and formats. The heterogeneity complicates the management work [18], [19] and understanding these messages and alerts is not an easy task. In a small network, system administrators can analyze the messages and alerts one by one, and understand their corresponding event types. Apparently, it is not practical in large complex networks. Automatic event generation is important for reducing the maintenance cost with limited human resources.
2) The diagnosis and resolution depend on experienced system administrators who analyze performance metrics, alert logs, event information and other network characteristics. Unexpected behaviors are usually discovered in daily operation of large complex networks. Malfunction of certain network elements can cause alerts in both upper-level business applications and other connected network elements. The scale and complexity of root cause analysis [20] in such networks are often beyond the ability of human operators. Therefore, automatic root cause analysis is necessary in managing large complex network infrastructures.
3) Root cause analysis is to identify the actual network element that causes an alert, while failure prediction tries to avoid the situation where the expected services cannot be delivered [21], [22]. Proactive fault management can enhance the network reliability, which is usually done by system administrators based on predefined business rules. With failure prediction, proactive fault management can be more efficient.

Failure prediction based on historical incident tickets and server attributes plays an important role in managing large complex network infrastructures.

Mining valuable knowledge from events and tickets can efficiently improve the performance of system diagnosis. In this survey, we focus on recent research studies dealing with the above three challenges. The reminder of this survey is organized as follows. Section 2 reviews the event generation approaches. Root cause analysis and failure prediction are investigated in Section 3 and Section 4, respectively. Finally, Section 5 concludes the survey.

2 Event Generation

The monitoring software on network elements in large complex networks generates huge amount of alerts, alarms, and messages, indicating the equipment status at real time. These alerts, alarms, messages are usually collected in log files. Contents of the data in log files may include time, element name, the running states of software components (e.g., started, interrupted, connected, and stopped), and other performance parameter values. In this section, we mainly focus on the methodologies of event generation from log files.

The contents of log files in some systems are unstructured, that is, each event is stored as a short message in plain text files, such as server logs, Linux logs and Hadoop logs. In other systems, the logs may be semi-structured or structured, e.g., Window event logs, database management system logs. Such logs are often stored in a database. Each record in the database represents an event, often including time, server name, process name, error code and other related information. A lot of data mining algorithms are based on structured or semi-structured data, while unstructured textual logs cannot be handled by these algorithms. Event generation is to convert textual logs into structured events for later analysis.

A simple log example is shown in **Table 1** [23], in which messages from a Simple File Transfer Protocol (SFTP) log are collected from a FTP software called FileZilla. Each line in Table 1 is a short message describing a certain event. In order to analyze the behaviors of FTP visits, these raw log messages need to be translated into types of events. The generated events are usually organized by timeline so that people can understand the behaviors and discover event patterns [24]. In Table 1, Message 4 is the event of uploading a webpage to the FTP server, and Message 9 is an error alert that the operation of creating a new directory is not successful. By converting raw log messages into canonical events, these events are able to be correlated across logs from different elements.

It seems that obtaining events from the log files is not a difficult task. However, due to the heterogeneity of network infrastructures, each network element generates raw messages with its own format and contents. These messages may be disparate and inconsistent, which creates difficulty in deciphering events reported by multiple network elements [24]. For example, it is supposed that we need to perform the following task: if any element stops, the system administrator is notified by email. Given the variability among different network elements, one element might record "The server has stopped" in the log file, while another one might record "The server has changed the state from running to stop." The inconsistency in log files makes the above task difficult. All the messages indicating the stop status from all network elements must be collected, in order to write a program to automate this simple task. This is less possible in large complex networks with newly added network elements and many legacy network elements.

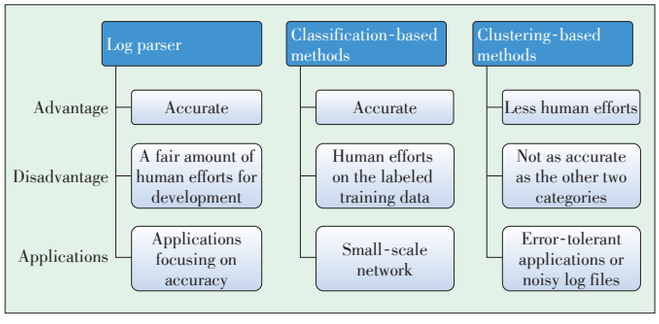
▼**Table 1. An example of messages in a simple file transfer protocol log [23]**

ID	Logs
1	2010-05-02 00:21:41 Command: cd "/disk/storage006/users/lt...
2	2010-05-02 00:21:42 Command: cd "/disk/storage006/users/lt...
3	2010-05-02 00:21:42 Command: put "E:/Tomcat/apps/record1.html" "/disk/...
4	2010-05-02 00:21:42 Status: Listing directory /disk/storage006/users/lt...
5	2010-05-02 00:21:42 Status: File transfer successful, transferred 1,232 bytes...
6	2010-05-02 00:21:42 Command: put "E:/Tomcat/apps/record2.html" "/disk/...
7	2010-05-02 00:21:42 Response: New directory is: "/disk/storage006/users/lt...
8	2010-05-02 00:21:42 Command: mkdir "libraries"
9	2010-05-02 00:21:42 Error: Directory /disk/storage006/users/lt...
10	2010-05-02 00:21:44 Status: Retrieving directory listing...
…	…

When one needs to analyze the historical event data across multiple elements, it is necessary to encode semantics in a system-independent manner. All raw log messages in log files should be consistent in semantics across similar fields, which allows the organization of common semantic events into categories. The converted canonical events provide the ability of describing the semantics of log data as well as the initial connection of syntax to semantics [24]. The research studies on event generation can be classified into three categories: log parser, classification, and clustering. The main characteristics of approaches in each category are summarized in **Fig 4**.

2.1 Log Parser

A straightforward solution is the log-parser-based approach, in which a log parser is built for log files in different formats. The system administrators must be familiar with the type and format of each raw message and understand its meaning, so that they can develop text parsers to extract the detailed semantic information from these messages accurately. Some messages might be easily parsed using simple regular expression. Clearly, the approach is not efficient for large complex network infrastructures, in which there are heterogeneous network elements having different log-generating mechanisms, and disparate formats and contents. In addition, the legacy systems with-



▲Figure 4. Main characteristics of different event generation approaches.

out reliable log generation libraries make this problem even harder.

The approaches of building log parsers based on the analysis of source code have been investigated by many researchers. Xu et al. [25], [26] proposed that the source code can be viewed as the schema of logs and the structure of raw log messages can be inferred from the schema. The event type and event format are determined based on the schema, and the variable values are extracted from source code as the attributes of events.

IBM autonomic computing toolkit allows general data collection from heterogeneous data sources and uses the Generic Log Adapter (GLA) [27] to convert raw log messages into the Common Base Event (CBE) format. Modern software packages are likely to be open sources, e.g., Hadoop and Apache Tomcat, to which the above approaches can be naturally applied. The advantage of log-parse-based approaches is that they are accurate, but on the other hand they require a fair amount of human efforts to fully understand the log formats and to develop log parser software.

2.2 Classification-Based Methods

Not all applications in network management require extracting all possible field variable values from log messages. Some of them only need to know event types of raw messages and focus on discovering the unknown relationship between different event types [24]. For example, a network firewall system only need to know its current state, that is, whether the current log message is related to a certain security issue, a particular performance status, or an unexpected program exception. Here, event generation is to determine the event types of raw messages, which is a text classification problem.

A simple classifier can be built using regular expression patterns. For each event type, there is a corresponding regular expression pattern [28]. But similar to the issue in log-parser-based approaches, using regular expression for classification requires experienced domain experts to write the expression in advance, which is inefficient in large complex network infrastructures with heterogeneous network elements.

When labeled log messages are available for training, popular classification algorithms like support vector machine (SVMs), can be applied to solve the text classification problem. A traditional approach for handling text information is the bag-of-words model [29], which splits the log messages into words or terms and uses binary vector representation. If a term exists in the message, the corresponding feature value is 1, otherwise, it is 0. Then, the classifier is built based on the joint distribution of the terms in log messages and corresponding event types. Pitakrat et al. [30] used supervised machine learning algorithms, e.g., decision trees and probabilistic representations, to classify log messages in order to conclude about computer system states, where log files are preprocessed and manually labeled for training the classifier.

Security log classification is an important research issue in log classification and has received a lot of research attention [31]–[34]. Network anomalies include DDoS attack, worm propagation, portscan activity, flash crowd, etc. One important task in security area is to categorize the anomalies into different types. Teufl et al. [32] built a classifier for combining different log messages into known event types based on relations between events or features. They proposed the concept of "Activation Patterns" which are generated from raw log messages for intrusion detection. Androulidakis et al. [31] focused on detection and classification of network anomalies in security application logs. They used an Entropy-base method to classify anomalies, where the entropy changes in each anomaly type. Kruegel et al. [33] analyzed the false alarm problem caused by incorrect classification of log messages, and proposed an event classification schema based on Bayesian network which improves the performance of model aggregation. Modi et al. [34] also used Bayesian classifier for network intrusion detection in Cloud platform by classifying alert messages from virtual machines.

The classification-based methods are accurate, but they need the labeled log messages for training. Obtaining the labeled data requires human efforts, which is often time consuming and costly. Classification-based methods are inappropriate for large complex networks due to the lack of experienced domain experts for labeling.

2.3 Clustering-Based Methods

Labeled training data is not required for clustering-based methods, because such the methods infer event types from raw log messages. Although clustering-based methods might not be very accurate, they are acceptable in certain event mining applications. Raw log messages are usually short but have a large vocabulary size, which leads to a vector space with very sparse high dimensional features. There are some recent studies [35], [36] on applying clustering techniques to partition log messages into separated groups, each of which represents an event type. The traditional clustering algorithms based on bag-of-words model cannot perform well due to the short message and large vocabulary, so these studies on clustering-based methods focus on the structured log messages.

Makanju et al. [36] proposed a log message clustering algorithm with the following four steps: 1) partitioning by the number of tokens; 2) partitioning by the word positions; 3) partitioning by the search for bijection; 4) discovering the descriptive words for each partition. Most frequent words are treated as template words. The log messages are partitioned based on these template words' positions. This method is quite efficient with linear time complexity. However, when the frequencies of these template words are flexible among different log files, the identification of template words is challenging.

Tang and Li [23] described an algorithm-independent framework for event generation from log messages named LogTree. LogTree utilizes the format and structural information in log messages and employs message segment table for effective and efficient event generation. LogTree builds tree patterns for log messages. The root of a tree pattern is the category of the log message, usually indicated by the message type, and the leaves are the field information in messages. The similarity between two tree patterns is defined based on the number of similar nodes and edges and nodes in higher levels are more discriminative.

Tang et al. [37] proposed to convert texture logs into events by clustering message signatures. Though log messages have various types, different log messages often have common subsequences. These common subsequences are treated as the signatures of event types. The most representative signatures are extracted from log messages, and based on these signatures, all messages are then partitioned into several segments by maximizing the total number of common pairs of signatures.

Makanju et al. [38] presented an approach for visualizing event log clusters, where log messages are partitioned into different groups for visualization using Simple Log File Clustering Tool (SLCT). SLCT can produce interpretable cluster results. Frequent attribute sets are found in an Apriori manner, and then the clusters based on their frequencies are built. Vaarandi and Pihelgas [39] presented the LogCluster algorithm for clustering textual log messages. LogCluster can discovery frequent occurring message patterns, as well as outliers. Sharma and Parvat [40] proposed to use k-mean clustering algorithm to partition network alert logs generated during network attacks. The generated clusters are used for further security alert analysis.

The advantage of clustering-based methods is that they do not require lots of human efforts, but they are not as accurate as log-parser-based or classification-based approaches. So clustering-based approaches should be applied when the applications are error-tolerant or the log files are noisy.

3 Root Cause Analysis

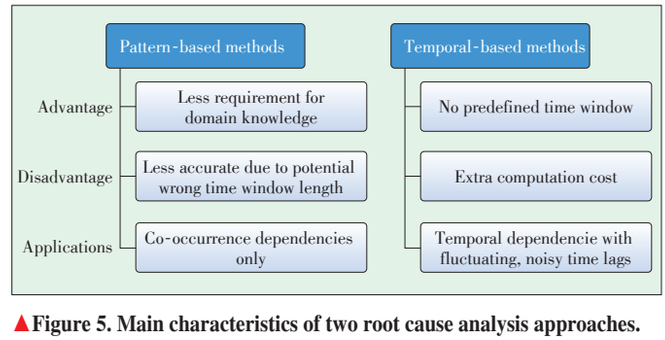
Network infrastructures usually include systems at multiple levels and these systems consist of computer servers, routers, and other network elements. For example, the system architecture of an enterprise portal website may contain Web servers, application servers, database servers and storage servers. When a system error occurs at a lower-level server, it might propagate to upper-level servers and cause system errors at different levels. If a storage server stops working, the database server may report I/O errors due to inaccessible files, resulting in malfunctioning application servers and Web servers. The alert messages from the Web server might be caused by any of these lower-level servers. To find the root cause of the fault, it is not possible to check the servers one by one to verify whether there is a hardware failure or a software exception. Therefore, automatic root cause analysis is needed. Event mining is a solution to highly efficient root cause analysis and cost reduction as well.

Most root cause analysis methods are based on the dependency graph of network elements [41], [42]. Dependency graphs could be built by experts if the network architecture is simple. For large complex networks, dependency graphs are built by finding the dependencies of network elements using event mining techniques. Root cause analysis can be done by locating the deepest element with alert messages on dependency graphs. Dependency might be bi-directional in practice, in which case we need to build a Bayesian network to calculate the probability of an element's status. Then the key step in root cause analysis is to discover the dependencies between events from log messages. Some of these approaches do not consider the time lag between events while others do. The research studies along this direction are divided into two categories: pattern-based methods and temporal-based methods. **Fig. 5** shows an overview of these two categories.

3.1 Pattern-Based Methods

Lou et al. [43] proposed an approach to find the hidden dependencies between components from unstructured logs. The raw log messages are parsed into keys and parameters first. Then, the dependent log pairs are found by co-occurrence analysis. Bayesian decision theory is adopted to estimate the dependency direction for each pair. The log pairs are further filtered by removing pairs with inconsistent time lags.

Nagaraj et al. [44] described an automated tool to help system administrators diagnose and correct performance issues in modern large-scale distributed systems. The tool leverages log



▲**Figure 5. Main characteristics of two root cause analysis approaches.**

data to reduce the required knowledge for administrators. Both the state and event information are extracted from log messages, and behavior patterns are discovered from the extracted states and events. During root cause analysis, the tool infers the most possible system components which might cause the performance issue using machine learning techniques.

Khan et al. [45] presented a tool for uncovering bugs in wireless sensor networks. Bugs in wireless sensor networks usually do not caused by a particular component but the unexpected interactions between multiple working components. The tool performs root cause analysis by discovering event sequences that are responsible for the faulty behavior. All log messages are divided into two categories, good and bad. Then all frequent event sequences up to a predefined length are generated. The good and bad frequent event sequences are used to perform discriminative analysis and these discriminative subsequences are used for bug analysis by matching.

3.2 Temporal-Based Methods

Zeng et al. [46] proposed to mine time lags of hidden temporal dependencies from sequential data for root cause analysis. Unlike traditional methods using a predefined time window, this method is used to find fluctuating, noisy, interleaved time lags. The randomness of time lags and the temporal dependencies between events are formalized as a parametric model. The parameters of the maximal likelihood model are inferred using an EM-based approach.

Tang et al. [47] presented a non-parametric method for finding the hidden temporal dependencies. By investigating the correlations between temporal dependency and other temporal patterns, both the pattern frequency and the time lag between events are considered in their proposed model. Two algorithms utilizing the sorted table in representing time lags are proposed to efficiently discover the appropriate lag intervals.

Yan et al. [48] described a generic root cause analysis platform for large IP networks. The Platform collects all kinds of network information including configurations, alarm logs, router logs, command logs, and performance measurements. With additional help from the spatial model of IP route, the platform supports tasks such as temporal/spatial correlation and Bayesian inference.

4 Failure Prediction

Failure prediction is a key step in proactive fault management of large complex networks. As mentioned, failure prediction tries to avoid service interrupt by applying resolution before fault happens. Most failure prediction approaches are similar in general. The main steps of failure prediction are summarized as follows:

1) extracting features from labeled training data based on historical failure log messages
2) building a prediction model using popular classifiers in machine learning techniques
3) monitoring continuously the current status of network elements to find whether a potential failure will happen in the near future based on classification results.

Salfner and Malek [49] presented an approach for online failure prediction in telecommunication systems using event-driven data sources. Hidden Semi-Markov Models (HSMMs) are used to model the failure event flow. The historical event sequence for failure and non-failure are collected for building two HSMMs. The failure likelihood of current event sequence is calculated using the two HSMMs.

Sipos et al. [50] presented a data-driven approach based on multiple-instance learning for failure prediction using equipment events. The log files contain both the daily operation records and the service details. Predictive features include event keywords, event codes, variations, sequence of event codes, etc. Keywords, event codes and variations are generated by parsers. Sequences of event codes are generated by applying sequential pattern mining techniques. A sparse linear classifier is trained with selected stable features for failure prediction.

Fronza et al. [51] introduced a method for predicting failures of a running system using log files. First, event sequences are extracted from log files. Supported Vector Machines (SVMs) are used to classify these event sequences into two categories: fail and non-fail. The process of extracting the event sequences is done in an incremental way. Each word in log files is assigned to a unique high dimensional index vector. When the log message is scanned, a context vector is calculated by summarizing index vectors in the sliding window.

Liang et al. [52] applied several classification methods on event logs collected from supercomputer IBM BlueGene/L and tried to predict the fatal event in the near future based on events in current window and historical observation period. There are six different types of events in the log files and for each event type. The following features are extracted from log files for training the classifiers: event number, accumulated event number, event distribution, interval between failures, and entry keywords in log messages.

Sahoo et al. [53] described a framework of a proactive prediction and control system for large clusters. Event logs and system activity reports are collected from a 350-node cluster for one year. A filtering technique is applied to remove the redundant and misaligned event data. They evaluated three different failure prediction approaches: linear time series models, rule-based classification algorithms, and Bayesian network models.

Fu and Xu [54] developed a spherical covariance model and a stochastic model to qualify the temporal correlation and the temporal correlation between events, respectively. The failure events are clustered into groups based on the correlations. Each group is represented by a failure signature which contains various attributes of computer nodes including type, I/O request, user information, system utilities, etc. Failure predic-

tion is done by predicting the future occurrences of each group.

Mohammed et al. [55] developed an approach for predicting failure and in categorical event sequences. Sequential data mining techniques are applied on the historical plan failure information for generating predictive rules. Normative, redundant, and dominated patterns are removed in order to select the most predictive rules for failure prediction.

5 Conclusions

In this paper, we present a comprehensive survey of event mining techniques applied in the domain of large-scale complex network management. Based on the general workflow of problem detection, determination and resolution, we present three challenges in modern network infrastructure management, which are related to event generation, root cause analysis, and failure prediction. For each challenge, we present the corresponding event mining techniques by reviewing the representative studies and summarizing their main ideas. In summary, mining valuable knowledge from events and logs greatly improves the reliability of large-scale complex network infrastructures.

References

[1] China Internet Information Center, "The 36th statistical report on the Internet development in China," China, 2015.
[2] C. Liang. (2014, Apr. 12). Tecnent techology news [Online]. Available: http://tech.qq.com/a/20140412/000129.htm
[3] TOP500.org. (2015, Jun.). TOP500 supercomputer sites [Online]. Available: http://www.top500.org/lists/2015/06
[4] National Security Agency. (2012, Apr. 23). Global information grid—NSA/CSS [Online]. Available: https://www.nsa.gov/ia/programs/global_information_grid/index.shtml
[5] K. Bakshi. (2009, Aug. 25). Cisco cloud computing—data center strategy, architecture [Online]. Available: http://www.cisco.com/c/dam/en_us/solutions/industries/docs/gov/CiscoCloudComputing_WP.pdf
[6] RCR Wireless News. (2014, May 13). Diagram, LTE network architecture [Online]. Available: http://www.rcrwireless.com/20140513/network-infrastructure/lte/lte-network-architecture-diagram
[7] D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage, "California fault lines: understanding the causes and impact of network failures," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 315–326, Oct. 2010. doi: 10.1145/1851275.1851220.
[8] D. Ford, F. Labelle, F. Popovici, et al., "Availability in globally distributed storage systems," in *the 9th USENIX Symposium on Operating Systems Design and Implementation*, Vancouver, Canada, Oct. 2010.
[9] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 350–361, Aug. 2011. doi: 10.1145/2043164.2018477
[10] Nagios Enterprises, LLC. (2015). Nagios: the industry standard in IT infrastructure monitoring [Online]. Available: http://www.nagios.org
[11] Zabbix LLC. (2015). Zabbix—an enterprise-class open source monitoring solution [Online]. Available: http://www.zabbix.com
[12] Opsview Ltd. (2015) Opsview: IT monitoring for networks, applications, virtual servers and the cloud [Online]. Available: http://www.opsview.com
[13] AXELOS. (2015). ITIL—information technology infrastructure librar [Online]. Available: https://www.axelos.com/best-practice-solutions/itil
[14] ITU. (2015). ITU-T Recommendations [Online]. Available: http://www.itu.int/en/ITU-T/publications/Pages/recs.aspx
[15] TM Forum. (2015). Forum, frameworx—TM [Online]. Available: https://www.tmforum.org/tm-forum-frameworx
[16] L. Tang, T. Li, L. Shwartz, F. Pinel, and G. Y. Grabarnik, "An integrated framework for optimizing automatic monitoring systems in large IT infrastructures," in *ACM International Conference on Knowledge Discovery and Data mining*, San Francisco, USA, Aug. 2013, pp. 1249–1257. doi: 10.1145/2487575.2488209.
[17] Y. Jiang, C.-S. Perng, T. Li, and R. Chang, "Intelligent cloud capacity management," in *IEEE Network Operations and Management Symposium*, Maui, USA, Apr. 2012, pp. 502–505. doi: 10.1109/NOMS.2012.6211941.
[18] A. Y. Halevy, N. Ashish, D. Bitton, et al., "Enterprise information integration: successes, challenges and controversies," in *ACM SIGMOD International Conference on Management of Data*, Baltimore, USA, Jun. 2005, pp. 778–787. doi: 10.1145/1066157.1066246.
[19] P. A. Bernstein and L. M. Haas, "Information integration in the enterprise," *ACM Communiation*, vol. 51, no. 9, pp. 72–79, Sep. 2008. doi: 10.1145/1378727.1378745.
[20] C. Zeng, L. Tang, T. Li, L. Shwartz, and G. Ya, "Mining temporal lag from fluctuating events for correlation and root cause analysis," in *IEEE International Conference on Network and Service Management*, Rio de Janeiro, Brazil, Nov. 2014.
[21] J. Bogojeska, I. Giurgiu, D. Lanyi, G. Stark, and D. Wiesmann, "Impact of HW and OS type and currency on server availability derived from problem ticket analysis," in *IEEE Network Operations and Management Symposium*, Krakow, Poland, May. 2014, pp. 1–9. doi: 10.1109/NOMS.2014.6838347.
[22] J. Bogojeska, D. Lanyi, I. Giurgiu, G. Stark, and D. Wiesmann, "Classifying server behavior and predicting impact of modernization actions," in *the 9th International Conference on Network and Service Management*, Zurich, Swizerland, Oct. 2013, pp. 59–66. doi: 10.1109/CNSM.2013.6727810.
[23] L. Tang and T. Li, "LogTree: a framework for generating system events from raw textual logs," in *IEEE International Conference on Data Mining*, Sydney, Australia, Dec. 2010, pp. 1550–4786. doi: 10.1109/ICDM.2010.76.
[24] T. Li, *Event Mining: Algorithms and Applications*. USA: Chapman and Hall/CRC, 2015.
[25] W. Xu, L. Huang, A. Fox, D. A. Patterson, and M. I. Jordan, "Mining console logs for large-scale system problem detection," in *the 3rd Workshop on Tackling System Problems with Machine Learning Techniques*, San Diego, USA, Dec. 2008, pp. 4–4.
[26] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *ACM 22nd symposium on Operating Systems Principles*, Big Sky, USA, Oct. 2010, pp. 117–132. doi: 10.1145/1629575.1629587.
[27] G. Grabarnik, A. Salahshour, B. Subramanian, and S. Ma, "Generic adapter logging toolkit," in *IEEE International Conference on Autonomic Computing*, May, 2004, pp. 308–309. doi: 10.1109/ICAC.2004.1301391.
[28] BalaBit IT Security. (2015). Pattern DB™—real-time syslog message classification [Online]. Available: http://www.balabit.com/network-security/syslog-ng/opensource-logging-system/features/pattern-db
[29] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge, UK: Cambridge University Press, 2008.
[30] T. Pitakra, J. Grunert, O. Kabierschke, F. Keller, and A. v. Hoorn, "A framework for system event classification and prediction by means of machine learning," in *the 8th International Conference on Performance Evaluation Methodologies and Tools*, Bratislava, Slovakia, Dec. 2014, pp. 173–180. doi: 10.4108/icst.valuetools.2014.258197.
[31] G. Androulidakis, N. Tech, V. Chatzigiannakis, and S. Papavassiliou, "Network anomaly detection and classification via opportunistic sampling," *IEEE Network*, vol. 23, no. 1, pp. 6–12, Jan./Feb. 2009. doi: 10.1109/MNET.2009.4804318.
[32] P. Teufl, U. Payer, and R. Fellner, "Event correlation on the basis of activation patterns," in *the 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, Pisa, Italy, Feb. 2010, pp. 631–640. doi: 10.1109/PDP.2010.80.
[33] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," in *the 19th Annual Computer Security Applications Conference*, Las Vegas, USA, Dec. 2003, pp. 14–23. doi: 10.1109/CSAC.2003.1254306.
[34] C. Modi, D. Patel, A. Patel, and R. Muttukrishnan, "Bayesian classifier and snort based network intrusion detection system in cloud computing," in *International Conference on Computing Communication & Networking Technologies*,

Coimbatore, India, Jul. 2012, pp. 1–7. doi: 10.1109/ICCCNT.2012.6396086.
[35] M. Aharon, G. Barash, I. Cohen, and E. Mordechai, "One graph is worth a thousand logs: uncovering hidden structures in massive system event logs," in *the European Conference on Machine Learning and Knowledge Discovery in Databases*, Bled, Solvenia, Sep. 2009, pp. 227–243. doi: 10.1007/978-3-642-04180-8_32.
[36] A. Makanju, A. N. Zincir-Heywood, and E. E. Milios, "Clustering event logs using iterative partitioning," in *ACM International Conference on Knowledge Discovery and Data Mining*, Paris, France, Jun. 2009, pp. 1255– 1264. doi: 10.1145/1557019.1557154.
[37] L. Tang, T. Li, and C.-S. Perng, "LogSig: generating system events from raw textual logs," in *ACM International Conference on Information and Knowledge Management*, Glasgow, UK, Oct. 2011, pp. 785– 794. doi: 10.1145/2063576.2063690.
[38] A. Makanju, S. Brooks, A. N. Zincir-Heywood, and E. E. Milios, "LogView: visualizing event log clusters," in *Annual Conference on Privacy, Security and Trust*, Fredericton, USA, Oct. 2008, pp. 99–108. doi: 10.1109/PST.2008.17.
[39] R. V. a. M. Pihelgas, "LogCluster—a data clustering and pattern mining algorithm for event logs," in *the 11th International Conference on Network and Service Management*, Barcelona, Spain, Nov. 2015, pp. 1– 7. doi: 10.1109/CNSM.2015.7367331.
[40] P. Sharma and T. J. Parvat, "Network log clustering using k-means algorithm," in *International Conference on Recent Trends in Information, Telecommunication and Computing*, Kochi, India, Aug. 2012, pp. 115–124. doi: 10.1007/978-1-4614-3363-7_14.
[41] M. Brodie, I. Rish, and S. Ma, "Intelligent probing: a cost-effective approach to fault diagnosis in computer networks," *IBM System Journal*, vol. 41, no. 3, pp. 372–385, Apr. 2002. doi: 10.1147/sj.413.0372.
[42] E. Kiciman and A. Fox, "Detecting application-level failures in component-basd inernet services," *IEEE Transactions on Neural Networks*, vol. 16, no. 5, pp. 1027–1041, Sep. 2005. doi: 10.1109/TNN.2005.853411.
[43] J.-G. Lou, Q. Fu, Y. Wang, and J. Li, "Mining dependency in distributed systems through unstructured logs analysis," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 1, pp. 91–96, Jan. 2010. doi: 10.1145/1740390.1740411.
[44] K. Nagaraj, C. Killian, and J. Neville, "Structured comparative analysis of systems logs to diagnose performance problems," in *the 9th USENIX conference on Networked Systems Design and Implementation*, San Jose, USA, Apr. 2012, pp. 26–26.
[45] M. M. H. Khan, H. K. Le, H. Ahmadi, T. F. Abdelzaher and J. Han, "Troubleshooting interactive complexity bugs in wireless sensor networks using data mining techniques," *ACM Transactions on Sensor Networks*, vol. 10, no. 2, Jan. 2014. doi: 10.1145/2530290.
[46] C. Z. Tang, T. Li, L. Shwartz, and G. Grabarnik, "Mining temporal lag from fluctuating events for correlation and root cause analysis," in *the 10th International Conference on Network and Service Management*, Rio de Janeiro, Brazil, Nov. 2014.
[47] L. Tang, T. Li, and L. Shwartz, "Discovering lag intervals for temporal dependencies," in *ACM International Conference on Knowledge Discovery and Data mining*, Beijing, China, Aug. 2012, pp. 633– 641. doi: 10.1145/2339530.2339633.
[48] H. Yan, L. Breslau, Z. Ge, et al., "G-RCA: a generic root cause analysis platform for service quality management in large IP networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 6, pp. 1734–1747, Mar. 2012. doi: 10.1109/TNET.2012.2188837.
[49] F. Salfner and M. Malek, "Using hidden semi-Markov models for effective online failure prediction," in *the 26th IEEE International Symposium on Reliable Distributed Systems*, Beijing, China, Oct. 2007, pp. 161– 174. doi: 10.1109/SRDS.2007.35.
[50] R. Sipos, D. Fradkin, F. Moerchen, and Z. Wang, "Log-based predictive maintenance," in *ACM International Conference on Knowledge Discovery and Data mining*, New York, USA, Aug. 2014, pp. 1867– 1876. doi: 10.1145/2623330.2623340.
[51] I. Fronzaa, A. Sillitti, G. Succia, M. Terhob, and J. Vlasenkoa, "Failure prediction based on log files using random indexing and support vector machines," *Journal of Systems and Software*, vol. 86, no. 1, pp. 2– 11, Jan. 2013. doi: 10.1016/j.jss.2012.06.025.
[52] Y. Zhang and A. Sivasubramaniam, "Failure prediction in IBM BlueGene/L event logs," in *IEEE International Symposium on Parallel and Distributed Processing*, Miami, USA, Apr. 2008, pp. 1–5. doi: 10.1109/IPDPS.2008.4536397.
[53] R. K. Sahoo, A. J. Oliner, I. Rish, et al., "Critical event prediction for proactive management in large-scale computer clusters," in *the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, Washington, USA, Aug. 2003, pp. 426–235. doi: 0.1145/956750.956799.
[54] S. Fu and C.-Z. Xu, "Exploring event correlation for failure prediction in coalitions of clusters," in *ACM/IEEE Conference on Supercomputing*, Reno, USA, Nov. 2007, pp. 1–12. doi: 10.1145/1362622.1362678.
[55] M. J. Zaki, N. Lesh, and M. Ogihara, "Predicting failures in event sequences," in *Data Mining for Scientific and Engineering Applications*. US: Springer, 2001. doi: 10.1007/978-1-4615-1733-7_27, pp. 515-539.

Manuscript received: 2016-01-15

Biographies

LIU Zheng (zliu@njupt.edu.cn) is an assistant professor at the school of computer science and technology, the Nanjing University of Posts and Telecommunications, China. He obtained his PhD from the Chinese University of Hong Kong, China in 2011. He was a research engineer with Huawei Technologies from 2011 to 2015. His research interests include mining and querying large graph data, mining multimedia data and mining event logs in network management. He has published research papers in major conferences including ICDE, ICDM, DASFAA, and PAKDD. He is an IEEE member.

LI Tao (taoli@cs.fiu.edu) is dean and professor at the school of computer science and technology, the Nanjing University of Posts and Telecommunications, China. He received his PhD in computer science from the University of Rochester, USA in 2004. His research interests are in data mining, information retrieval, and computing system management. He was a recipient of an NSF CAREER Award and multiple IBM Faculty Research Awards. He is on the editorial boards of *ACM Transactions on Knowledge Discovery from Data, IEEE Transactions on Knowledge and Data Engineering,* and *Knowledge and Information System Journal*.

WANG Junchang (wangjc@njupt.edu.cn) is a lecturer at Nanjing University of Posts and Telecommunications, China. He received a Ph.D. degree in computer science from University of Science and Technology of China. His research focuses on software defined networking (SDN), network management, and high-performance computing (HPC).