Improving Fault Device Identification Method using Alarm Clustering Approach

Yoichi Matsuo*, Yuichi Suto†, Yusuke Makino†, Kazumich Sato*
* *NTT Network Service Systems Laboratories, NTT Corporation, Tokyo, Japan*
† *NTT Network Innovation Center, NTT Corporation, Tokyo, Japan*

***Abstract*—Identifying the fault device is one of the important tasks for network operators so that operators can immediately take action for the fault device in a communication network. Many methods have been developed to automate the identification of the fault device, and these methods assume only a single cause, such as hardware failure, which generates alarms. However, alarms are generated simultaneously due to multiple events (i.e., failure, construction work, and notification for operators) in a production communication network. This paper addresses the challenge of identifying fault devices in communication networks, which is complicated by the simultaneous occurrence of alarms from multiple events. We introduce a method utilizing alarm clustering to improve the accuracy of fault device identification by effectively distinguishing between these mixed alarms. Our evaluation was conducted using dataset in a real-world production environment more than one month and demonstrated significant enhancements in identifying faulty devices.**

***Index Terms*—Identifying fault device, Alarm clustering, Network management, Bayesian Network**

I. Introduction

In recent decades, communication networks have been used in various fields, such as business, healthcare, and education. Therefore, to prevent prolonged failures in communication networks, which can significantly impact daily life activities, many studies and products have focused on identifying fault device [1]–[10] using generated alarms.

Those studies and products assume that multiple failures rarely occur simultaneously in a communication network, and they assume that only alarms related to a single failure are generated at the time of a failure. However, these assumptions may not hold in a production environment. Although multiple failures may not frequently occur, alarms are generated by not only failures but also construction work and notification for operators due to the customer's operation, such as rebooting a switch placed in the customer's building, which can be mixed for the following reasons. First, the alarms are not generated only when an event occurs but continue to be generated for a certain period, such as during construction work or until a hardware failure is recovered [11]. Therefore, alarms due to multiple events are mixed, although events did not occur simultaneously. Second, since construction work (e.g., adding or deleting devices) is performed frequently, alarms related to construction work and failure may be mixed due to failures during construction. Third, alarms due to operations by customers frequently are generated if a large number of customers use that communication network, which may overlap the alarms due to failure or construction work. As a result, the assumption of the existing methods and products does not hold, which deteriorates the estimation accuracy.

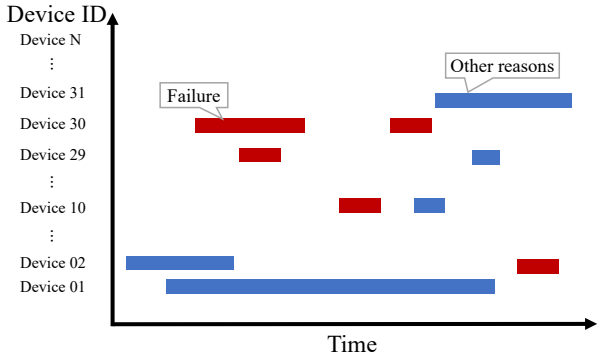

Fig. 1. Example of how alarms generated in a actual network

Fig. 1 shows an actual example of how alarms occurred in a certain communication network operated by NTT. The horizontal axis represents duration, the vertical axis represents each device in the communication network, and the red box indicates that the device has generated alarms due to a failure, while the blue box indicates that the device has generated alarms by other events. As explained in the previous paragraph, there are rarely simultaneous failures. However, the alarms generated by devices 29 and 30 due to failures are mixed up with alarms generated by construction work at device 1. As another case, the alarms generated by device 10 due to a failure are mixed up with alarms generated by notifications at device 30. Although previous studies have not considered mixed alarms due to multiple causes, these events do occur in real networks, making fault device estimation difficult.

In this paper, we propose a fault device identification method using an alarm clustering approach to adapt the mixed-up alarms and improve the accuracy of fault device estimation. Specifically, we combine a Bayesian Network-based fault device estimation method [2] with alarm clustering method [12], which also uses a Bayesian Network and allows us to split mixed alarms by clustering. Although various fault device estimation methods exist, rule-based methods should be used in practice. This is because it is hard to collect data for training the methods since actions are taken to prevent recurrence after a failure occurs, which means that the same

978-3-903176-66-9

failure rarely occurs. Therefore, we select one of the rule-based methods as a base method [2]. When mixed-up alarms due to multiple events are generated, the method proposed by Hata et.al [12] enables us to assign the ID to each alarm, indicating to which event the alarm is associated, by modeling the alarm generation process using the Bayesian Network. This method automatically identifies alarms with similar characteristics in the same cluster. Therefore, by combining these methods, the fault device can be identified in the production communication network environment.

In this paper, we evaluate the usefulness of the proposed method using alarms in a production environment over several weeks. The results show that our proposed method improves accuracy by 14% at most in the offline evaluation compared with the baseline method and achieves very high accuracy in the online evaluation.

The contributions of this paper are as follows.

- We pointed out that alarms due to multiple events are mixed up in the production environment using actual data, which is not considered in the previous papers. Since alarms caused by several events might overlap and deteriorate the estimation accuracy of existing methods, the method applicable to the alarms due to multiple events has to be developed.
- We proposed a fault device identification method that improves the estimation accuracy by splitting mixed alarms into alarms for each event.
- The proposed method was evaluated using a prepared dataset based on events collected from a production environment and a real-time online dataset in a production environment for more than a month total, while most of the existing papers use simulation datasets. The results show that the proposed method improves estimation accuracy compared with the baseline method.

II. Related Work

A. Identifying Fault Device

Methods to estimate the fault device have been extensively studied and can be categorized into rule-based and data-driven methods. However, rule-based methods should be used in practice since it is hard to collect data for training the methods since actions are taken to prevent recurrence after a failure occurs, which means that the same failure rarely occurs.

In the rule-based methods, Gestalt [1] is a fast estimation calculation algorithm for a network with more than a thousand routers. Matsuo et al. [2] proposed modifying the causal model so that the causal model can adapt to rare failures. G-RCA [3], Leila et al. [4], NetMedic [5], and Paramvir et al. [13] prepare templates that abstract causal dependencies between the fault device and observation data such as CPU usage, memory usage, traffic volumes, and syslog messages. The reason for using the various kinds of data is to reduce the overlooking of the fault device. Once the templates are prepared, the causal model can be constructed by inputting the communication network topology. Researchers [6]–[8] have proposed constructing a causal model using a certain assumption. Shrink [6], Score [7] construct a causal model between routers and interface alarms from the topology data. If an interface in a router fails, alarms regarding interface down are generated from a fault router and adjacent routers. Thus, these methods can localize the fault router using generated alarms. MicroRCA [8] localizes the fault container on Kubernetes (K8s) by constructing a causal model based on the requests between containers.

However, those existing methods do not explicitly consider the situation in mixed-up alarms due to multiple events being inputted to them, which deteriorates the estimation accuracy.

B. Alarm Correlation Method

Several alarm correlation methods have been developed, which analyze the correlation among occurred alarms and extract root-cause alarms to support the operator's investigation by presenting the root-cause alarm [14]–[18]. The paper proposed by Bouloutas et al. [14] developed a representation of the network devices connection using graphs and identifying fault locations based on the occurrence of alarms on these graphs. Authors [15] abstracted and modeled the occurrence of alarms in the microservice application using Bayesian networks. MAYOR proposed in [16] grouping the alarms so that a set of grouped alarms represents a unique event based on the calculated generation interval after all of the alarms occur due to a certain event. Kim et al. [17] proposed a correlation based on identifying the root-cause alarm and the fault device method. The method proposed in [18] extracts the subsequence of previous alarms related to the current alarm.

In contrast to the existing method, the proposed method with the method [12] clusters alarms in terms of events, taking into account the previous occurred alarms instead of extracting a root-cause alarm from generated alarms, since alarms have to be analyzed every a certain window size (e.g., every two minutes) in communication network operation. Also, although existing alarm correlation methods rely on topology information, this information is often updated manually, which can lead to errors. Hata et al. [12] proposes a method that does not rely on topology information by abstracting event mechanisms.

Furthermore, most of the existing studies are based on simulations or open datasets, while this paper evaluates the baseline method and the proposed method on data in production communication networks, both online and offline. It is valuable to check the practicality of these methods.

III. Proposed Method

The proposed method consists of the alarm clustering method, which splits the mixed-up alarms for each event (e.g., failure, construction work, notification for operators due to customer's operation) and the fault device identification method to be adapted to alarms due to multiple events simultaneously occurred in a production environment. Here, since information about construction work (e.g., when and in which device construction work will take place) is known in advance, alarms due to construction work can be filtered. However, if all alarms from devices under construction are turned off, it

will be impossible to confirm whether the devices are not in an unexpected state due to construction. In addition, partial filtering requires time and expert knowledge to determine in advance what alarms should be filtered out, making it impractical for each construction project.

This section first explains the alarm clustering and the fault device identification methods. Note that since both methods are already published, most of the parts in Sections III-A and III-B are coming from original papers [2], [12]. Then, we explain how both methods are used and the effect of combining both methods.

A. Alarm Clustering Method

Alarm clustering method [12] used in this paper correlates alarms on an event-by-event basis using a Bayesian Network that considers characteristics of alarms such as alarm type, time of occurrence, and location. In addition, by estimating the relationship between previous alarms and current alarms considering characteristics, the method calculates correlation among alarms even when only some alarms have yet to occur. This capability is important for network operators to take action immediately.

We describe how to abstract and model alarm generation in the communication network. A communication network consists of logical and physical connections.When a failure occurs at a transmission device in the physical layer, the following alarms occur.

- EQP alarm: indicating a hardware failure of the transmission device
- Section alarm: indicating an optical multiplex section error
- Optical alarm: indicating an error in the path between transmission devices
- Path alarm: indicating a path error between IP devices
- Linkdown alarm: indicating a connection error between IP devices
- Protocol alarm: indicating a routing protocol error

If an abnormality occurs in the transmission device, section alarms and other alarms listed above are considered to occur continuously. Thus, this method models the alarm generation mechanism using a Bayesian Network.

Figure 2 represents a constructed model of alarm generation mechanism using the Bayesian Network. Each node in the figure is a random variable that takes 0 or 1. Regarding the ALM nodes, if a certain alarm (e.g., section alarm) occurs, the corresponding node (e.g., Section ALM) takes 1. The remaining nodes, $T(\cdot,\cdot)$ and $L(\cdot,\cdot)$, are used for characterizing the time of occurrence and location, respectively. A node $T(\cdot,\cdot)$ indicates the difference in occurrence time between alarms, where it (e.g., $T(Sec, Opt)$) takes 1 if a current alarm (e.g., optical alarm) is generated after previous alarm (e.g., Section alarm) within a certain time interval (e.g., 5 seconds). Similarly, a node $L(\cdot,\cdot)$ indicates the difference of location, where it (e.g., $L(Sec, Opt)$) takes 1 if a current alarm is generated at the same location (e.g., same device, same prefecture). These settings are based on the assumption that alarms can be inferred to be caused by the same event if the occurrence times and location are close. This is because since devices generate alarms in real-time once anomalies occur in devices, the occurrence times of alarms for the same event tend to be close to each other. Similarly, because devices connected to each other detect anomalies and generate alarms, the locations of alarms for the same event tend to be close to each other. Then, conditional probability is defined between nodes using expert knowledge or hyperparameter tuning.

When the alarm occurs, the probability of the occurrence of the current alarm is calculated from the prior probability and conditional probability, using the information on previous alarms as evidence in the Bayesian Network. If the probability is greater than a threshold, the current alarm is assigned the same clustering ID as the previous alarms; otherwise, a new cluster ID is assigned. For example, an EQP ALM occurs at Tokyo-device01 four seconds after a Section ALM occurs at Tokyo-device02. The probability can be calculated as follows.

$$P(EQPALM = 1 | L(EQP, Sec) = 1, T(EQP, Sec) = 1) \tag{1}$$

This method can assign cluster IDs to alarms in real-time by sequentially calculating the probability of alarms using previous alarms when an alarm appears, which allows alarms to be split by event.

B. Localizing the Fault Device Method

We explain how to construct the causal model using the Bayesian Network to identify the fault device in the communication network.

The causal model represents which devices are affected (i.e., whether alarms occur) when a certain device fails. When alarms occur, the fault device is identified using the causal model. The causal model consists of device nodes and observation-data nodes, where both nodes are random variables, edges, prior probabilities, and conditional probabilities. Device nodes represent the status of routers and servers in the communication network. Observation-data nodes represent the status of data collected from the communication network, such as CPU and memory usage, traffic volumes, and syslog messages. Note that this paper uses only alarms as data inputted to observation-data nodes. Thus, j-th observation-data node represents whether j-th device generates alarm or not during a certain time window, and the number of device nodes and observation-data nodes is the same.

Let x_i and y_j be random variables that represent the status of the i-th device and whether the j-th device generates alarms. Then, we define $X = (x_1, x_2, \ldots, x_N), x_i \in \{0, 1\}$ and $Y = (y_1, y_2, \ldots, y_N), y_j \in \{0, 1\}$, where N is the number of devices, 0 means normal, and 1 means abnormal. For instance, if $x_i = 1$, i-th device is fault. Also, if j-th device generates alarms, the status of y_j takes 1; otherwise, 0.

The prior probabilities represent how much each device tends to be normal or abnormal status. We define the prior probabilities to equipment nodes as follows.

$$P(X|\alpha) = \Pi_{i=1}^{N} P(x_i|\alpha) = \Pi_{i=1}^{N} (1-\alpha)^{1-x_i} \alpha^{x_i}, \tag{2}$$

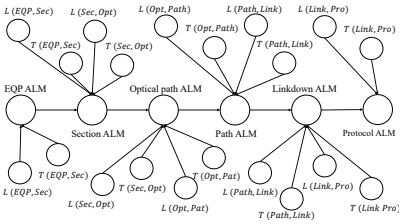

Fig. 2. Constructed model of alarm generation mechanism

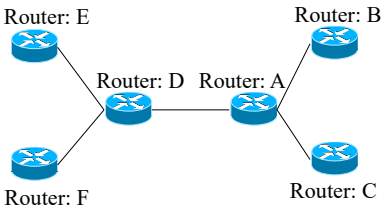

Fig. 3. Example of the topology

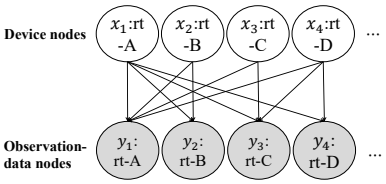

Fig. 4. Example of the causal model

where α is constant value that is a hyperparameter of prior probabilities.

The edges and conditional probabilities represent causal dependencies and the degree of the causal dependencies, respectively. If device i has a causal dependency with the j-th device, we add an edge $e_{i,j}$ between x_i and y_j.

How to add the edges depends on methods. In this paper, we set an assumption that alarms are generated by the fault device and the devices connected to the device when a failure. Thus, the edge $e_{i,j}$ is set if the i-th and j-th devices are connected. Figures 3 and 4 show examples of how the causal model is constructed, where we omit several device nodes and observation-data nodes due to space limitations. For instance, since router A connects to router B, C, and D, the there are edges from x_1, which represents the status of router A, to y_1, y_2, y_3, and y_4, which represent the status of router A, B, C, and D. This means that if the router A becomes abnormal status, alarms will be generated from router A, B, C, and D.

Let E be a set of all edges between X and Y, and $\phi_{i,j}$ be a parameter that represents the index of an edge $e_{i,j}$ as follows.

$$\phi_{i,j} = \begin{cases} 1 & (e_{i,j} \in E) \\ 0 & (\text{otherwise}) \end{cases}. \tag{3}$$

Then, we define the conditional probabilities between X and Y as follows.

$$P(Y|X, \beta, \Phi) = \Pi_{j=1}^{M} \Pi_{i=1}^{N} (1-\beta)^{\delta(x_i=y_j)\phi_{i,j}} \beta^{\delta(x_i \neq y_j)\phi_{i,j}}, \tag{4}$$

where β is constant value that is a hyperparameter of prior probabilities, δ is a delta function, and Φ is a set of parameter $\phi_{i,j}$.

Finally, when the observation-data nodes Y are given, the statuses of each device node $\tilde{X}$ are formulated as below using Equations (2), and (4),

$$\tilde{X} = \arg\max_X P(X|Y, \beta, \Phi, \alpha) = \arg\max_X \frac{1}{C} \Pi_{j=1}^{M} \Pi_{i=1}^{N} \psi_{i,j}, \tag{5}$$

where $\psi_{i,j} = (1-\beta)^{\delta(x_i=y_j)\phi_{i,j}} \beta^{\delta(x_i \neq y_j)\phi_{i,j}} (1-\alpha)^{1-x_i} \alpha^{x_i}$ and C is a constant value for normalization, and under the condition that each x_i flips independently. Equation (5) is solved by the belief propagation inference algorithm.

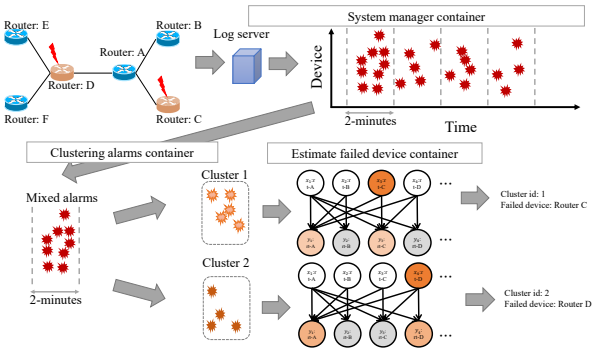

Fig. 5. Overview of implemented system and effect of the proposed method

C. Overview and Effect of Proposed Method

To evaluate our proposed method, we implemented a system that collects alarms from a certain production communication network, inputs the collected alarms into the alarm clustering method, and inputs alarms in each cluster into the identifying fault device method.

Figure 5 illustrates the implemented system's overview and how the proposed method treats alarms and estimates the fault devices. This system consists of a system manager container, an alarm clustering container, and an identifying fault device container using Docker and Fluentd. When multiple events (e.g., events at router C and D) occur, mixed-up alarms are generated. The system manager container forwards these alarms to the alarm clustering container and the identifying fault device container every two minutes. Here, the window size was set to 2 minutes based on the advice from network operators. In this case, if all the alarms are input into the identifying fault device container, it is unclear whether routers D and C are estimated as fault devices. However, using our proposed method, the alarms cluster into the event, which means that in this case, the alarms are split into alarms due to failure at router C (cluster id 1) and router D (cluster id 2). Then, the fault device identification method is executed twice using alarms with cluster IDs 1 and 2, respectively.

IV. Experimented Communication Network and Dataset Information

In this paper, we implemented our proposed method in a certain production communication network operated by NTT as described in Section III-C. This communication network

consists of more than 1,000 devices and is subject to construction work and minor failures at any time. In our evaluation, we selected less than 200 devices that play a crucial role among all devices as the first step of our project.

While collecting the dataset, failures such as interface down and package restart occurred during construction work such. Here, failures and construction work rarely occur on the same device more than once. Thus, it is difficult to determine the fault devices and alarms using a data-driven method.

A. Dataset for Offline Evaluation

We prepared data for the offline evaluation in approximately six weeks, from mid-January to March 2023. During the six weeks, construction work and failures occurred 52 times. It should be noted, however, that the actual number of alarms is much higher than 52 because each event generated many alarms for several minutes to several hours. During this period, there were alarms caused by the customer's operation and mixed-up alarms caused by construction work and failures, as shown in Figure 1. To evaluate the proposed method, the network operator labeled each alarm as to which event it belongs to and the fault device for each event.

In the experiments, data was prepared as follows, as illustrated in Figure 6.

1) 52 events were divided into 17 events for hyperparameter tuning and 35 events for evaluation. Here, the 17 events for parameter tuning were not taken from the front in chronological order but were extracted so that there would be no bias by considering the contents of events (e.g., failures and construction work).
2) Then, the time information in each alarm included in each event for evaluation was edited so that the time between the last alarm by a certain event and the first alarm by the next event was set to 20 minutes. This was done to fill in the gaps in evaluation events since several events were extracted for hyperparameter tuning or to fill in the gaps during the six weeks when no events occurred so that events occur at 20-minute intervals. However, the types of alarms generated by the event and the order in which alarms are generated within each event have not changed. Therefore, alarms in this dataset also precisely reproduce the alarms in a production environment.
3) Then, for each event, these alarms were segmented into 2-minute intervals, and these segmented alarms were used as input for the proposed and baseline methods.

B. Dataset for Online Evaluation

For the online evaluation, we constructed an evaluation environment in which alarms generated on the production environment are forwarded to the system implementing the proposed method in real-time, and the analysis results are output to a network management system. The evaluation was conducted using online datasets 1 and 2, which are 22 cases during four weeks from mid-November to mid-December in 2023 and 10 cases of construction work and failures that occurred during February 2024, respectively.

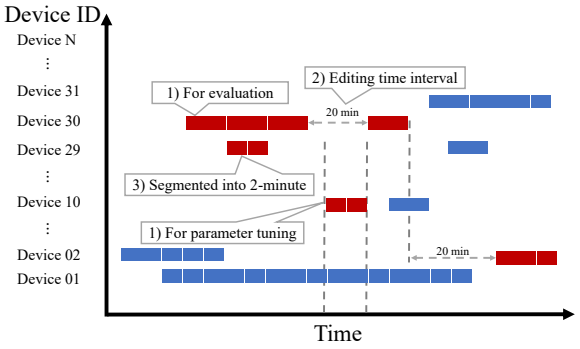

Fig. 6. How to prepare the offline dataset

V. Evaluation

To demonstrate the usefulness of the proposed method, offline and online evaluations were conducted using data from a communication network provided by NTT.

Regarding evaluation metrics, the homogeneity score and Adjusted Rand Index (ARI), are used for alarm clustering method. The homogeneity score represents the degree to which each data belongs to the correct cluster. If labels of all alarms in a certain cluster indicate the same cluster, the homogeneity score is 1 (complete homogeneity). ARI is one of the evaluation metrics for clustering, which calculates the ratio of how many pairs sampled from each cluster match per all of the pairs.

For the fault device identification method, we used two original matrices PMA and MA, by modifying accuracy. This is to incorporate the fact that the number of fault devices is very small compared to the total number of devices and that, in network operations, missed fault devices can lead to prolonged failures. PMA metric is defined as follows.

$$EFD(k) = \begin{cases} 1 & \left(x_i = 1, {}^{\forall} i \in FD_k, x_i = 0, {}^{\forall} i \in ND_k\right) \\ 0 & (\text{otherwise}) \end{cases} \tag{6}$$

$$PMA = \frac{\sum_{k=1}^{K} EFD(k)}{K}, \tag{7}$$

where FD_k and ND_k represent the set of index of fault devices and non-fault devices in k-th execution, respectively, and K is the total number of execution of identifying fault device method. Thus, PMA represents ratio of perfect match of estimation results per the number of execution. Similarly, MA metric is defined as follows.

$$EFD'(k) = \begin{cases} 1 & \left(x_i = 1^{\forall} i \in FD_k \cup A_k, \right. \\ & \left. \quad x_i = 0^{\forall} i \in ND_k \backslash A_k\right) \\ 0 & (\text{otherwise}) \end{cases} \tag{8}$$

$$MA = \frac{\sum_{k=1}^{K} EFD'(k)}{K}, \tag{9}$$

where A_k is any two indices not in FD_k. Thus, MA allows up to two normal devices to be mis-estimated as fault devices. This is because although missing fault devices is unacceptable,

a few misestimated fault devices have little impact on network operations.

In this evaluation, we set a fault device identification method without alarm clustering as a baseline, i.e, identifying fault device using all alarms for two minutes at once.

A. Alarm Investigation

Since we pointed out that alarms due to multiple events can be mixed in a production environment in Section I, we checked whether such alarms occur using an online dataset. First, we calculated the duration of alarm occurrence due to each event in the online dataset by measuring the time from the first alarm occurrence due to a certain event until the alarm due to that event stopped occurring. Due to a small number of long-duration construction works, the median duration was approximately one hour, and the average duration was about one day. This means that an event does not necessarily generate an alarm only at the moment of event occurrence. On the other hand, alarms cease to occur after about one hour, and multiple events are unlikely to occur frequently within an hour, indicating that mixed alarm cases do not occur frequently. However, as the average duration is one day, it is quite possible that alarms due to multiple events (e.g., one day of construction work and failure) may occur, resulting in a mix-up.

In the online dataset, 110 cases of mixed alarms due to multiple events within 2 minutes over approximately two months were recorded.

B. Offline Evaluation

We evaluate our proposed method using a dataset from mid-January to March 2022. In this dataset, the alarm clustering method was executed 589 times during 35 events.

The homogeneity score and ARI for the alarm clustering method were 1.0 and 0.995, respectively. Since the homogeneity score is 1.0, alarms caused by multiple events were not mixed in a cluster. This result is useful because it is very important to eliminate the mixing of alarms in actual network operations and the fault device identification method. By checking the clustered alarms, we found that the alarm clustering method cleanly split alarms generated by the customer's operation and alarms caused by failures and construction work. On the other hand, the ARI of 0.995 indicates that alarms due to a certain event were not overly split. These results show that the alarm clustering method can appropriately divide alarms into events in the production communication network.

Table I shows the proposed and baseline method results for the offline dataset. In the table, numbers in parentheses indicate the number of EFD for PMA or EFD' for MA and total cases. Here, the number of executions of the fault device identification method was higher than that of the alarm clustering method. This is because if the alarms are divided into multi clusters (e.g., 2 clusters in the case Figure 5), the fault device identification method will be executed for each cluster (e.g., two times in the case Figure 5). If only a single event has occurred, the proposed method is the same as the baseline method since a single cluster ID is assigned to all alarms. Therefore, to check the effect of the alarm clustering method more precisely, we extracted only the cases where alarms are divided into multiple clusters and summarized the results as only m-clusters row.

TABLE I
THE RESULTS OF OFFLINE EVALUATION

		Baseline	Proposed
Offline dataset	PMA	0.89 (525/589)	0.96 (921/951)
	MA	0.97 (576/589)	0.98 (938/951)
Offline dataset only m-clusters	PMA	0.82 (335/352)	0.96 (685/714)
	MA	0.96 (386/352)	0.98 (701/714)

TABLE II
EFFECT OF ALARM CLUSTERING METHOD

		Proposed	
		Correct	Incorrect
Baseline	Correct	289	0
	Incorrect	34	29

In Table I, the proposed method outperforms the baseline method by 7% in PMA for all offline datasets. When the proposed method is compared with the baseline method, the improvement in accuracy is smaller for MA than for PMA. This is because MA allows two false positives. The improvement of PMA and MA between the proposed and baseline methods for multi-clustered events increases compared with the result in the table. Especially, PMA increases 14%. This is because the baseline method cannot accurately estimate the fault device (i.e., PMA) by inputting all alarms at once when multiple alarms occur, while the proposed method can estimate each fault device by input alarms split by each event using the alarm clustering method.

For a more detailed analysis, Table II summarizes how the proposed methods improved the correct and incorrect cases in the baseline method. This table aims to see if the alarms clustering method causes cases where the correct cases in the baseline method become incorrect in the proposed method. In the table, correct means $\sum EDF$. As shown in the table, 289 cases were correct for the proposed and baseline methods. By using the alarm clustering method, 34 cases that were incorrect in the baseline method became correct in the proposed method, while no cases became incorrect in the proposed method from correct in the baseline method.

C. Online Evaluation

We evaluated the proposed method using a system implemented in the production communication network. In the online experiment, the proposed method was executed, and the estimated fault devices were outputted in real-time. In total, the fault device identification method was executed 182 times for online dataset 2 and 363 times for online dataset 2. Here, in this evaluation, since the system implements only the proposed method, we cannot compare the proposed method with the baseline method.

TABLE III
THE RESULTS OF ONLINE EVALUATION

		PMA	MA
online 1	Proposed (all)	0.84 (153/182)	0.91 (165/182)
	Proposed (m-cluster)	0.10 (3/31)	0.48 (15/31)
online 2	Proposed (all)	0.97 (355/363)	1.0 (363/363)
	Proposed (m-cluster)	0.94 (75/79)	1.0 (79/79)

Table III shows the result of the proposed method. All and m-cluster in the parentheses represent all cases and cases when alarms split into multi-clusters, respectively. The proposed method for the online dataset 1 is highly accurate, especially with MA exceeding 90%. However, when the alarms are divided into multiple clusters, MA is about 0.5, and PMA is 0.1. This is because, for the online dataset 1, hyperparameters of the alarm clustering method and fault device identifying method were tuned using the offline dataset. However, trends of alarm occurrence in online evaluation had changed significantly from the time when the offline dataset was obtained. Specifically, configuration changes were made to partially filter out alarms and other notifications to operators due to customer operations. Since hyperparameters of both methods were tuned under the assumption that these alarms would occur, PMA and MA deteriorated.

Therefore, we re-tuned the hyperparameters using the online dataset 1 and conducted the online evaluation again in February. The result is shown in dataset 2. As shown in Table III, the proposed method achieved high accuracy and 1 for MA metric.

These results show that even in a production environment, there is a limitation that the proposed method requires appropriate hyperparameter tuning when the way of alarm generation changes. However, by hyperparameter tuning, the proposed method can appropriately split alarms caused by multiple events into separate events and estimate the fault device in real-time with high accuracy.

VI. CONCLUSION

This paper highlights the challenges and complexities in accurately identifying fault devices within communication networks, particularly when faced with mixed alarms from multiple events such as hardware failures, construction activities, and notification of customer operations. The proposed method leverages an alarm clustering approach that integrates a rule-based model using Bayesian networks to effectively distinguish between these mixed alarms, thereby enhancing the accuracy of fault localization. The online and offline evaluations of this approach in a real-world production environment confirm a significant improvement in fault localization accuracy, enabling a shorter network operation when the failure occurs.

The feature work is to expand the target devices in the communication network since we selected key role devices from all devices. Another feature work involves evaluating the proposed method with other existing methods using a larger dataset from the production environment.

REFERENCES

[1] R. N. Mysore, R. Mahajan, A. Vahdat, and G. Varghese, "Gestalt: Fast, Unified Fault Localization for Networked Systems," *USENIX ATC*, pp. 255–267, 2014.

[2] Y. Matsuo, Y. Nakano, A. Watanabe, K. Watanabe, K. Ishibashi, and R. Kawahara, "Root-cause diagnosis for rare failures using bayesian network with dynamic modification," *Proc. ICC*, 2018.

[3] H. Yan, L. Breslau, Z. Ge, D. Massey, D. Pei, and J. Yates, "G-RCA: A Generic Root Cause Analysis Platform for Service Quality Management in Large IP Networks," *IEEE Trans. Net.*, vol. 20, no. 6, pp. 1734–1747, 2012.

[4] L. Bennacer, Y. Amirat, A. Chibani, A. Mellouk, and L. Ciavaglia, "Self-diagnosis technique for virtual private networks combining bayesian networks and case-based reasoning," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 354–366, 2015.

[5] S. Kandula, R. Mahajan, P. Verkaik, S. Agarwal, J. Padhye, and P. Bahl, "Detailed diagnosis in enterprise networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, p. 243, 2009.

[6] S. Kandula, D. Katabi, and J.-p. Vasseur, "Shrink: A tool for failure diagnosis in IP networks," *Proc. ACM SIGCOMM workshop on Mining network data*, pp. 173–178, 2005.

[7] R. Kompella, J. Yates, a. Greenberg, and a.C. Snoeren, "Fault Localization via Risk Modeling," *IEEE Trans. Dependable Secure Comput.*, vol. 7, no. 4, pp. 1–14, 2010.

[8] L. Wu, J. Tordsson, E. Elmroth, and O. Kao, "Microrca: Root cause localization of performance issues in microservices," in *NOMS*, 2020, pp. 1–9.

[9] COMARCH, 2024, accessed on 6th, June, 2024. [Online]. Available: https://www.comarch.com/

[10] I. C. P. for AIOps, 2024, accessed on 6th, June, 2024. [Online]. Available: https://www.ibm.com/products/cloud-pak-for-aiops

[11] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," in *Proceedings of the ACM SIGCOMM 2011 Conference*, ser. SIGCOMM '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 350–361. [Online]. Available: https://doi.org/10.1145/2018436.2018477

[12] Y. Hata, N. Hayashi, Y. Makino, A. Takada, and K. Yamagoe, "Alarm correlation method using bayesian network in telecommunications networks," in *Proc. APNOMS*, 2022, pp. 1–4.

[13] P. Bahl, R. Chandra, A. Greenberg, S. Kandula, D. A. Maltz, and M. Zhang, "Towards highly reliable enterprise network services via inference of multi-level dependencies," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, p. 13, 2007.

[14] A. Bouloutas, S. Calo, and A. Finkel, "Alarm correlation and fault identification in communication networks," *IEEE Transactions on Communications*, vol. 42, no. 234, pp. 523–533, 1994.

[15] L. Wu, J. Tordsson, E. Elmroth, and O. Kao, "Causal inference techniques for microservice performance diagnosis: Evaluation and guiding recommendations," in *Proc. ACSOS*, 2021, pp. 21–30.

[16] R. Mijumbi, A. Asthana, C. Bernal, and M. Castejon, "Mayor: Machine learning and analytics for automated operations and recovery," in *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, 2019, pp. 1–9.

[17] D. S. Kim, H. Shinbo, and H. Yokota, "An alarm correlation algorithm for network management based on root cause analysis," in *13th International Conference on Advanced Communication Technology (ICACT2011)*, 2011, pp. 1233–1238.

[18] G. Manca and A. Fay, "Detection of historical alarm subsequences using alarm events and a coactivation constraint," *IEEE Access*, vol. 9, pp. 46 851–46 873, 2021.