SmtRPTG: Highly-Efficient Monitoring Scheme to Capture Network Status Changes Accurately

Ziye Lu, Zichen Xu, Zhihuang Ma and Zuqing Zhu†
School of Information Science and Technology, University of Science and Technology of China, Hefei, China
†Email: {zqzhu}@ieee.org

***Abstract*—As network monitoring is crucial for ensuring the performance of network operations, one key challenge is how to optimize the tradeoff between its overheads and accuracy. This paper proposes a smart reporting mechanism, namely SmtRPTG, which continuously optimizes the scheme of network status collecting and reporting to properly balance the tradeoff. SmtRPTG estimates the distributions of status data of various types based on sampled results, and lets network elements make local decisions on whether and what type of status data should be reported based on the estimations. We formulate a probabilistic model with hidden variables, based on which an algorithm is designed to estimate data distributions for SmtRPTG. Extensive simulations verify the effectiveness of our proposal on balancing the tradeoff between overheads and accuracy of network monitoring.**

***Index Terms*—Network monitoring, Network status reporting.**

I. Introduction

Since the inception of Internet, network monitoring has been essential to facilitate effective network control and management (NC&M). Nowadays, we have witnessed dramatic changes in the Internet landscape, due to the rise of datacenter networks [1], the emerging of new physical layer technologies [2–4], and the fast development of software-defined networking (SDN) [5, 6] and network function virtualization (NFV) [7, 8]. These changes have made network environments more complex and traffic patterns more unpredictable, bringing new challenges to network monitoring techniques as if they still want to visualize network status changes timely and accurately.

Here, one key challenge is how to properly balance the tradeoff between the overheads and accuracy of network monitoring [9]. This is because highly-dynamic network environments necessitate frequent network status collecting and reporting, while collecting network status data and reporting it consume not only the computing and memory resources on network elements (NEs) (*e.g.*, switches and routers) but also the bandwidth resources on network connections [10], no matter whether the network monitoring uses a traditional out-of-band method (such as SNMP [11] and Netflow [12]) or the recently-developed in-band network telemetry (INT) [13].

Traditional out-of-band network monitoring methods like SNMP and Netflow place an agent on each NE to collect its status data, which is then collected by the NC&M system with a polling-based scheme. The advance on programmable data plane (PDP) [14] has led to the proposal of INT, which can monitor networks in a programmable, fine-grained and real-time manner. Specifically, with INT enabled, each NE along a flow's routing path collects its status during forwarding each packet of the flow, and encodes the status data as specific INT field(s) in the packet. Therefore, for both the traditional out-of-band methods and INT, the key of balancing the tradeoff between the overheads and accuracy of network monitoring is to optimize the mechanism for network status collecting and reporting. In other words, the mechanism should be designed such that it only collects and reports the status data that is truly necessary for the execution of NC&M tasks.

Previously, people have developed event-driven mechanisms for traditional out-of-band methods, *i.e.*, the agent on an NE only collects and reports status data when an NC&M-interested event has happened [10]. However, as network environments are becoming more and more complex, it will be increasingly difficult to predefine NC&M-interested events, especially for soft failures [15]. On the other hand, a few selective INT schemes [16–18] proposed to reduce the overheads of INT by sampling packets to encode INT fields. Nevertheless, all these studies have not answered the question of how to adjust the sampling scheme adaptively to balance the tradeoff between the overheads and accuracy of network monitoring on-the-fly in a highly-dynamic network environment. In [19, 20], we proposed to let NEs make local decisions on whether and what type of INT fields should be encoded in a packet, based on the amount of information that can be conveyed by the INT fields. However, when and how to update the information content of status data on each NE to capture network status changes timely were not addressed.

Note that, in a highly-dynamic network environment, an NE needs to adaptively update the importance of status data (*i.e.*, the information content that the data can convey to NC&M tasks) to capture real-time changes in network status, thereby optimizing the tradeoff between the overheads and accuracy of its network monitoring. Therefore, in this work, we propose a smart reporting mechanism, namely SmtRPTG, which continuously optimizes the scheme of network status collecting and reporting to only report the status data that is truly necessary for the execution of NC&M tasks. Specifically, SmtRPTG estimates the real distributions of status data of various types based on sampled results, and lets NEs make local decisions on whether and what type of status data should be reported based on the estimations (*i.e.*, calculating status data's importance accordingly). We formulate a probabilistic model with hidden variables, based on which the problem of distribution estimation is solved with the maximum likelihood

estimation (MLE). We also propose an algorithm to determine when the data distributions maintained by NEs should be updated. Extensive simulations confirm the effectiveness of our proposal on improving the efficiency of network monitoring.

The rest of the paper is organized as follows. Section II describes the system architecture of SmtRPTG and formulates the probabilistic model for the samples of network status data. In Section III, we design the algorithms for SmtRPTG to estimate the distributions of status data and to determine whether data reporting strategy should be updated, respectively. The simulations for performance evaluation are discussed in Section IV. Finally, we summarize the paper in Section V.

II. Operation Principle

This section presents the system architecture of SmtRPTG, explains its key problem of estimating the distributions of status data, and derives a probabilistic model for the problem.

A. System Architecture and Problem Definition

We assume that SmtRPTG takes the generic architecture of a network monitoring system, which consists of monitoring agents on NEs and an NC&M system. Each monitoring agent is responsible for collecting and reporting network status data about its NE, while the NC&M system monitors NEs' status in the global manner and updates the NEs' operation states and network monitoring strategies accordingly. Depending on the way that the agents report network status data to the NC&M system, the architecture can fit in both the traditional out-of-band methods and INT. Specifically, if the agents report status data in the out-of-band way and count on the NC&M system to analyze the data, SmtRPTG is utilized with traditional out-of-band network monitoring, and if it is used with INT, status data is encoded as INT fields in packets by NEs and then received and analyzed by data analyzers at network edges.

Fig. 1 shows the architecture of SmtRPTG. As the network status on an NE normally changes when it processes packets, we assume that for a type of network status, if the NE samples it on a per-packet basis, the network status is monitored with the highest accuracy, which means that the sampled results can be treated as the "ground truth" of the status data for this type. As shown in Fig. 1, on an NE at each packet processing, there is a corresponding **ground truth data** for each network status type, denoting as $\{d_1, d_2, \cdots, d_N\}$, where N is the number of types. The **collected data** is a subset of the ground truth data (*e.g.*, $\{d_1, d_3, d_7, \cdots\}$ in Fig. 1), which is selected by looking up the data reporting table (DRT), to reduce the overheads of network monitoring. The DRT contains the probability distribution of each type of network status data, and the importance of a data sample to NC&M tasks generally becomes larger if its value is observed with a smaller probability [19]. As the probability distribution of a type of network status data normally cannot be known in advance, it should be estimated by the NC&M system during network operation based on the collected data from the NE.

We design SmtRPTG to offload the distribution estimation to the NC&M system because an NE (*i.e.*, a switch/router) usually has limited memory and should focus its computing power on packet processing. The NC&M system stores received collected data in the traffic engineering database (TED), estimates the distributions of network status data accordingly, and updates the DRTs on NEs when necessary. As shown in Fig. 1, each column of the DRT corresponds to a data type, while each entry in the column denotes the probability of a range/value. Specifically, for the i-th type, if its data is continuous, we divide the possible values into M_i ranges: $\{v_{i,1}, v_{i,2}, \cdots, v_{i,M_i}\}$, and if the data is discrete, $v_{i,j}$ is just the j-th possible value. Meanwhile, the DRT records the probability that a data sample falls in or equals $v_{i,j}$ as $p_{i,j}$.

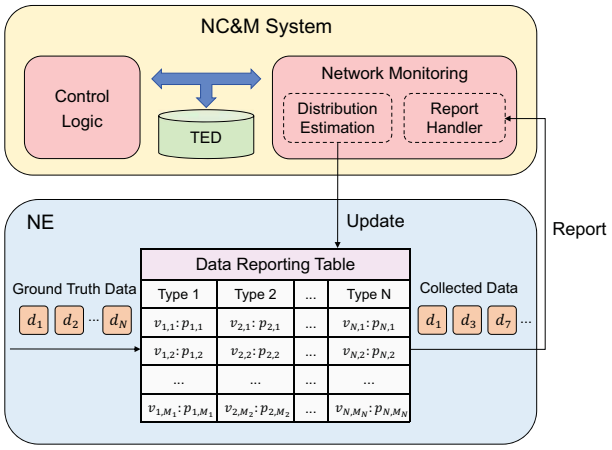


Fig. 1. System architecture of SmtRPTG.

B. Probabilistic Model for Distribution Estimation

We denote the collected data that the NE reports to the NC&M system at time t as $\mathcal{D}_t$ (*e.g.*, $\mathcal{D}_t = \{d_1, d_3, d_7, \cdots\}$ in Fig. 1). Hence, over a period of processing T packets, the set of collected data is $\mathbf{D} = \{\mathcal{D}_t, t \in [1, T]\}$. Then, for the i-th type, the probability distribution of its data in period T is estimated as $\mathcal{P}_i = \{p_{i,1}, p_{i,2}, \cdots, p_{i,M_i}\}$. The DRT summarizes all the estimated distributions as $\mathbf{P} = \{\mathcal{P}_i,\ i \in [1, N]\}$.

Note that, at each sampling time t, the data of the i-th type can be either collected or ignored. However, $\mathcal{P}_i$ can only be accurately obtained when the values of all the data samples are known/estimated. Therefore, we define a set of hidden variables $\mathcal{H}_t = \{h_{1,t}, h_{2,t}, \cdots, h_{N,t}\}$ to denote the ranges/values that each type of data falls in or equals at sampling time t. Specifically, if the data of the i-th type is collected at time t ($d_i \in \mathcal{D}_t$) and its value belongs to $v_{i,j}$, we get the probability distribution of $h_{i,t}$ as

$$P(h_{i,t} = n) = \begin{cases} 1, & n = j, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Otherwise, if the data is not collected at time t ($d_i \notin \mathcal{D}_t$), the probability distribution of $h_{i,t}$ becomes

$$P(h_{i,t} = n) = \xi_{i,t}(n), \tag{2}$$

where $\xi_{i,t}(n)$ is the probability that needs to be estimated and $\sum_{n=1}^{M_i} \xi_{i,t}(n) = 1$. We can solve the distribution estimation with

maximum likelihood estimation. For the set of collected data $\mathcal{D}_t$, the probability of its occurrence under distribution $\mathbf{P}$ is

$$\begin{aligned} P\left(\mathcal{D}_t \mid \mathbf{P}\right) &= \sum_{h_{1,t}=1}^{M_1} \cdots \sum_{h_{N,t}=1}^{M_N} P\left(\mathcal{D}_t, h_{1,t}, \cdots, h_{N,t} \mid \mathbf{P}\right) \\ &= \sum_{\mathcal{H}_t} P(\mathcal{D}_t, \mathcal{H}_t \mid \mathbf{P}). \end{aligned} \tag{3}$$

If we assume that the data samples of a type are independent of each other, the likelihood function of the set of collected data $\mathbf{D}$ over the entire period T can be obtained as

$$\begin{aligned} P(\mathbf{D} \mid \mathbf{P}) &= \prod_{t=1}^{T} P\left(\mathcal{D}_t \mid \mathbf{P}\right) \\ &= \prod_{t=1}^{T} \left[\sum_{\mathcal{H}_t} P(\mathcal{D}_t, \mathcal{H}_t \mid \mathbf{P}) \right]. \end{aligned} \tag{4}$$

Hence, by leveraging MLE, we transform the distribution estimation to finding the $\mathbf{P}$ that maximize the likelihood function $P(\mathbf{D} \mid \mathbf{P})$, *i.e.*, calculating the $\overline{\mathbf{P}}$ that satisfies:

$$\overline{\mathbf{P}} = \underset{\mathbf{P}}{\operatorname{argmax}} \left[P(\mathbf{D} \mid \mathbf{P}) \right]. \tag{5}$$

III. Algorithm Design

In this section, we propose an algorithm to solve Eq. (5).

A. Expectation-Maximization Process

We solve Eq. (5) by leveraging the procedure of expectation-maximization (EM) [21], which is an iterative approach to estimate unknown variables when the observed data is incomplete. For a probabilistic model with hidden variables, EM estimates its parameters by iterating the expectation step (E-step) and maximization step (M-step) continuously until convergence. Hence, to solve Eq. (5), we leverage EM to maximize $P(\mathbf{D} \mid \mathbf{P})$ in Eq. (4) by gradually increasing its lower bound. Specifically, by applying $\log(\cdot)$ to $P(\mathbf{D} \mid \mathbf{P})$ and utilizing the Jensen's inequality [21], we can get the iteration equation for the k-th iteration as

$$\mathbf{P}^k = \underset{\mathbf{P}}{\operatorname{argmax}} \left\{ \sum_{t=1}^{T} \sum_{\mathcal{H}_t} P(\mathcal{H}_t \mid \mathcal{D}_t, \mathbf{P}^{k-1}) \log \left[P(\mathcal{D}_t, \mathcal{H}_t \mid \mathbf{P}) \right] \right\}, \tag{6}$$

where $\mathbf{P}^k$ denotes the parameters obtained at the k-th iteration.

We design the EM procedure to solve Eq. (5) as follows:

1) Parameters Initialization: At the beginning of a period T, we need to initialize the to-be-estimated variables in $\mathbf{P}$. If the period is the first one, we initialize $\mathbf{P}$ with preset values, which are set empirically on previous experience, and use the $\mathbf{P}$ obtained in the last period, otherwise.

2) E-Step: In this step, we calculate the distribution of hidden variables $P(\mathcal{H}_t \mid \mathcal{D}_t, \mathbf{P}^{k-1})$ based on the current estimations (*i.e.*, $\{\xi_{i,t}^k(n),\ n \in [1, M_i]\}$ for each i). As the strategy is known, we assume that for any $\mathcal{H}_t$, we can tell whether there is a contradiction between $\mathcal{H}_t$ and $\mathcal{D}_t$ with

$$P(\mathcal{D}_t, \mathcal{H}_t \mid \mathbf{P}) = \begin{cases} 0, & P(\mathcal{H}_t \mid \mathcal{D}_t, \mathbf{P}^{k-1}) = 0, \\ P(\mathcal{H}_t \mid \mathbf{P}), & \text{otherwise.} \end{cases} \tag{7}$$

Then, we can get the expectation function $Q(\mathbf{P}, \mathbf{P}^{k-1})$ as

$$\begin{aligned} Q(\mathbf{P}, \mathbf{P}^{k-1}) &= \sum_{t=1}^{T} \sum_{\mathcal{H}_t} \left\{ P(\mathcal{H}_t \mid \mathcal{D}_t, \mathbf{P}^{k-1}) \log \left[P(\mathcal{D}_t, \mathcal{H}_t \mid \mathbf{P}) \right] \right\} \\ &= \sum_{t=1}^{T} \sum_{\mathcal{H}_t} \left\{ P(\mathcal{H}_t \mid \mathcal{D}_t, \mathbf{P}^{k-1}) \log [P(\mathcal{H}_t \mid \mathbf{P})] \right\} \\ &= \sum_{t=1}^{T} \left\{ \sum_{n_1=1}^{M_1} \cdots \sum_{n_N=1}^{M_N} \left[\prod_{i=1}^{N} \xi_{i,t}^k(n_i) \cdot \log \left(\prod_{i=1}^{N} p_{i,n_i} \right) \right] \right\}. \end{aligned} \tag{8}$$

3) M-Step: In this step, the updated $\mathbf{P}$ is determined by maximizing the expectation function $Q(\mathbf{P}, \mathbf{P}^{k-1})$, which is equivalent to solving the following optimization:

$$\mathbf{P}^k = \underset{\mathbf{P}}{\operatorname{argmax}} \left[Q(\mathbf{P}, \mathbf{P}^{k-1}) \right] \tag{9a}$$

$$s.t. \begin{cases} \sum_{n_1=1}^{M_1} \cdots \sum_{n_N=1}^{M_N} \left(\prod_{i=1}^{N} p_{i,n_i} \right) = 1, \\ 0 \le p_{i,n_i} \le 1,\ \forall i. \end{cases} \tag{9b}$$

4) Iteration Termination Condition: The E-Step and M-Step are executed alternatively, until one of the two termination conditions is satisfied: 1) the number of iterations reaches a threshold, and 2) the difference between the results obtained in two adjacent iterations is less than a threshold.

In E-Step, to obtain the distribution of $h_{i,t}$, we introduce a set of boolean matrices $\mathbf{G} = \{G_1, G_2, \cdots, G_N\}$ to denote the possible values/ranges of status data of different types at each time t, where the size of G_i is $T \times M_i$. The operation principle of SmtRPTG ensures that the importance of uncollected data is smaller than that of collected data, thereby for the i-th type, if its ground truth data falling in $v_{i,j}$ at t is not contradictory to the collected data in $\mathcal{D}_t$, we call $v_{i,j}$ a *possible* value/range. Hence, the value of element $G_i(t,j)$ is

$$G_i(t,j) = \begin{cases} 1, & v_{i,j} \text{ is a possible value/range at } t, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Therefore, at the k-th iteration, we can obtain the posterior probability $\xi_{i,t}^k(j)$ as

$$\xi_{i,t}^k(j) = \frac{G_i(t,j) \cdot p_{i,j}^{k-1}}{\sum_{j'=1}^{M_i} \left[G_i(t,j') \cdot p_{i,j'}^{k-1} \right]}. \tag{11}$$

Since the problem in Eq. (9) is a convex optimization (*i.e.*, $Q(\mathbf{P}, \mathbf{P}^{k-1})$ is a concave function), M-Step can solve it by leveraging the Lagrange multiplier method [22]. Hence, we can get the joint probability as

$$\begin{aligned} \prod_{i=1}^{N} p_{i,n_i} &= \frac{\sum_{t=1}^{T} \left[\prod_{i=1}^{N} \xi_{i,t}^k(n_i) \right]}{\sum_{t=1}^{T} \sum_{n_1=1}^{M_1} \cdots \sum_{n_N=1}^{M_N} \left[\prod_{i=1}^{N} \xi_{i,t}^k(n_i) \right]} \\ &= \frac{\sum_{t=1}^{T} \left[\prod_{i=1}^{N} \xi_{i,t}^k(n_i) \right]}{T}. \end{aligned} \tag{12}$$

Then, we can calculate the marginal probability as

$$p_{i,j}^k = \frac{\sum_{t=1}^{T} \xi_{i,t}^k(j)}{T}. \tag{13}$$

Algorithm 1 shows our proposed procedure for estimating distributions of status date, which will be executed in the NC&M system in SmtRPTG. We first initialize T, $\{G_i\}$ and $\{\tilde{\mathcal{P}}_i\}$ in *Lines* 1-2, where $\{\tilde{\mathcal{P}}_i\}$ compose the initial DRT. Then, the while-loop of *Lines* 3-17 describe how SmtRPTG operates to estimate the distributions of status date upon receiving each set of collected data $\mathcal{D}_t$. Specifically, we calculate the T-th row of each matrix G_i in *Line* 5, and then check whether the procedure of distribution estimation should be invoked in *Line* 6. Here, we introduce ΔT as the shortest interval of estimating the distributions (*i.e.*, the estimation is performed every time when the NC&M system has received ΔT sets of new collected data). *Lines* 7-14 show the procedure of estimating $\mathcal{P}_i$ with the EM approach. We execute E-Step to get the posterior probability of hidden variables $\{\xi_{i,t}^k(j)\}$ (*Line* 11) and M-Step to calculate $\{p_{i,j}^k\}$ (*Line* 12) in each iteration until the termination condition is satisfied. *Line* 15 leverages *Algorithm* 2 to determine whether updating the DRT on an NE is necessary, and we will explain it in the next subsection.

Algorithm 1: Estimation of Probability Distributions

```
1  T = 0, G_1 = G_2 = ··· G_N = 0;
2  initialize P̃_1, P̃_2, ··· P̃_N randomly;
3  while received a set of collected data D_t do
4      T = T + 1;
5      calculate the T-th row of each G_i with Eq. (10);
6      if T % ΔT = 0 then
7          for each i ∈ [1, N] do
8              k = 0, P_i^k = P̃_i;
9              while termination condition is not met do
10                 k = k + 1;
11                 execute E-Step with Eq. (11);
12                 execute M-Step with Eqs. (12) and (13);
13             end
14         end
15         apply Algorithm 2 to tell whether updating DRT
           is necessary;
16     end
17 end
```

B. Algorithm for Determining DRT Update Time

Note that, although *Algorithm* 1 re-estimates the distributions in the DRT every ΔT, the NC&M system might not need to communicate with the NEs to update the DRTs there every ΔT. Therefore, to save the bandwidth overheads caused by the communications for updating DRTs, we design *Algorithm* 2 to adjust the interval of DRT updating adaptively according to the latest network status.

Definition 1: We define the **gap** between two distributions $\mathcal{A} = \{\alpha_1, \alpha_2, \cdots \alpha_M\}$ and $\mathcal{B} = \{\beta_1, \beta_2, \cdots \beta_M\}$ as

$$S(\mathcal{A}, \mathcal{B}) = \frac{1}{M} \sum_{j=1}^{M} \left| \frac{\alpha_j - \beta_j}{\alpha_j} \right|, \tag{14}$$

where α_j and β_j denote the probabilities of a same value/range in the two distributions, respectively, *i.e.*, $\sum_{j=1}^{M} \alpha_j = \sum_{j=1}^{M} \beta_j = 1$.

Algorithm 2: Decision Making for DRT Updating

```
Input: Minimum and maximum DRT update intervals
       T_min and T_max, respectively, probability sets
       {P_i^k} and {P̃_i} for each i, and gap threshold η.
1  if T ≤ T_min then
2      continue;
3  else
4      if T ≥ T_max then
5          for each i ∈ [1, N] do
6              update DRT on NE with current {P_i^k};
7              P̃_i = P_i^k;
8          end
9          T = 0;
10     else
11         for each i ∈ [1, N] do
12             calculate S(P_i^k, P̃_i) with Eq. (14);
13             if S(P_i^k, P̃_i) ≥ η then
14                 update DRT on NE with current {P_i^k};
15                 P̃_i = P_i^k, T = 0;
16                 break;
17             end
18         end
19     end
20 end
```

As for the inputs of *Algorithm* 2, $T_{\min}$ and $T_{\max}$ denote the minimum and maximum DRT update intervals, respectively, $\{\mathcal{P}_i^k\}$ are the current estimation results of $\{\mathcal{P}_i\}$, and $\{\tilde{\mathcal{P}}_i\}$ denote the estimations in the current DRT on the NE, and η is the preset threshold for the gap $S(\mathcal{P}_i^k, \tilde{\mathcal{P}}_i)$. If the interval since the last DRT update is less than $T_{\min}$, we will just skip the DRT update directly (*Lines* 1-2). Otherwise, if the interval has exceeded $T_{\max}$, we will update the DRT regardless of the gap $S(\mathcal{P}_i^k, \tilde{\mathcal{P}}_i)$ (*Lines* 4-9). Finally, if the interval falls in $(T_{\min}, T_{\max})$, we only update the DRT when any gap $S(\mathcal{P}_i^k, \tilde{\mathcal{P}}_i)$ exceeds the threshold η (*Lines* 11-18), to ensure that the difference between the distributions estimated by the NC&M system and those in the DRT will not be significantly large. With *Algorithm* 2, we can adjust the DRT update interval adaptively, enabling more efficient network monitoring.

IV. Performance Evaluation

In this section, we discuss the simulations to evaluate the performance of SmtRPTG for efficient network monitoring.

A. Convergence Performance

We first perform simulations to verify the convergence of the EM approach used in SmtRPTG. Specifically, we leverage

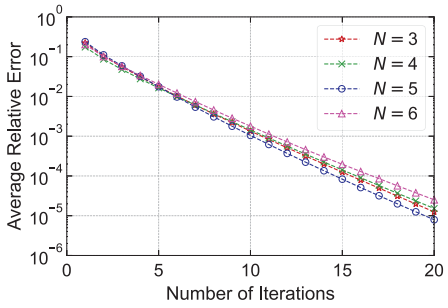


Fig. 2. Convergence performance of EM approach.

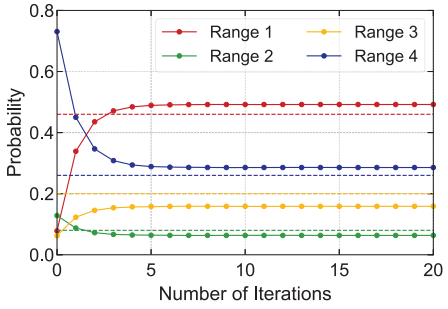


Fig. 3. Example on distribution estimation with EM approach.

the gap defined in *Definition* 1 to compare the distributions obtained in two adjacent iterations of the EM approach.

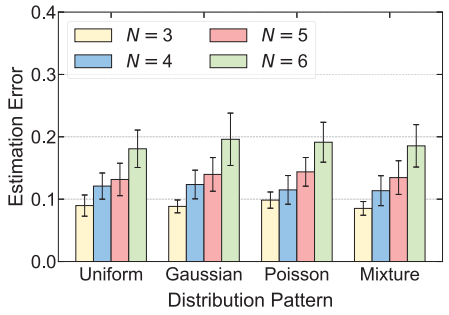
Definition 2: We define the **average relative error (ARE)** between the results in the k-th and $(k-1)$-th iterations as

$$\delta^k_{\text{ARE}} = \frac{1}{N}\sum_{i=1}^{N} S(\mathcal{P}_i^{k-1}, \mathcal{P}_i^k). \tag{15}$$

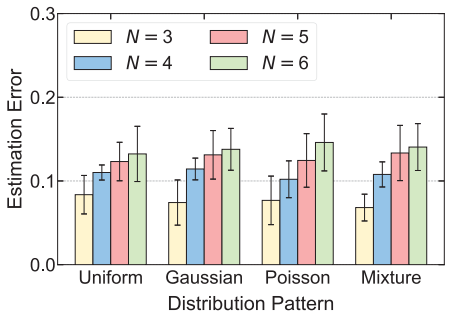
We first verify that the EM approach in *Algorithm* 1 can converge, by setting the status data types as $N \in [3, 6]$, and for each case, we generate the boolean matrices $\mathbf{G}$ randomly and fix $T = 200$. The results on δ^k_{ARE} are shown in Fig. 2, which indicates that the ARE of each case monotonically decreases towards 0 as the iterations progress, verifying the convergence of the EM approach. Meanwhile, as the results suggest that δ^k_{ARE} is normally reduced to $\sim 10^{-3}$ after 10 iterations, we set the iteration termination condition in *Algorithm* 1 as: 1) the number of iterations reaches 10, or 2) when we have $\delta^k_{\text{ARE}} < 10^{-3}$, in the following simulations. To further illustrate the effectiveness of our proposed EM approach on distribution estimation, Fig. 3 shows an example on its iterations to estimate the probabilities of the ranges of a Gaussian-distributed status data type, where we divide the possible values of the data into 4 ranges and the estimated probabilities and ground truths are plotted with solid and dished lines, respectively. We can see that the estimated probabilities converges to the values that are close to the ground truths after only ~ 5 iterations.

B. Accuracy of Distribution Estimation

Next, we evaluate the accuracy of data distribution estimation of SmtRPTG, by generating the ground truth data of each type with different distributions and comparing the distributions estimated by SmtRPTG with their ground truths in terms of the ARE in Eq. (15). The simulations set $T_{min} = 100$, $T_{max} = 200$, $\Delta T = 10$, $N \in [3, 6]$ and the update threshold



(a) $\eta = 0.2$



(b) $\eta = 0.4$

Fig. 4. Estimation error of SmtRPTG.

as $\eta \in \{0.2, 0.4\}$, and consider four scenarios, each of which generates N types of status data with a different distribution pattern, *i.e.*, the uniform distribution, Gaussian distribution, Poisson distribution, and the mixture of the three distributions. Each simulation collects 1000 sets of collected data samples.

Fig. 4 shows the average results on the estimation error (*i.e.*, the ARE between the estimated distributions and their ground truths). We can see that the estimation error of SmtRPTG does not change significantly with the distributions of status data types, proving that the performance of our proposal is almost independent of status data distributions. The estimation error increases with N, which is well expected because it is more difficult to estimate the distributions of more types of status data. However, in the worst case with $N = 6$ and $\eta = 0.2$, the average relative error between the estimated distributions and their ground truths is still less than 0.2, confirming the accuracy of SmtRPTG in distribution estimation. Meanwhile, it is interesting to observe that the estimation error becomes smaller when we increase the update threshold η from 0.2 to 0.4. We suppose that it is because when η is larger, SmtRPTG each time waits for a longer interval to update DRT, leading to use more collected data for each distribution estimation.

C. Tradeoff Analysis

Finally, we check the performance of SmtRPTG on balancing the tradeoff between the overheads and accuracy of network monitoring by applying it with INT[1] and comparing the scheme with two representative benchmarks, *i.e.* the Sel-INT [17] and EntropyINT [19]. Here, Sel-INT samples each type of status data with a preset sampling rate, while EntropyINT

[1]Note that, in addition to INT, SmatRPTG can also be used together with a traditional out-of-band network monitoring method (*e.g.*, SNMP).

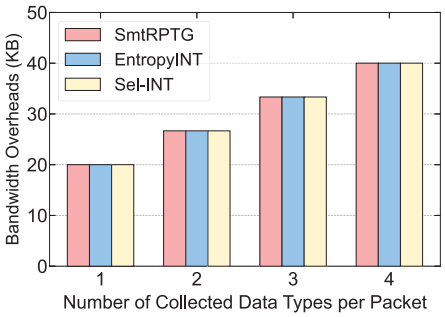


Fig. 5. Bandwidth overheads of three network monitoring schemes.

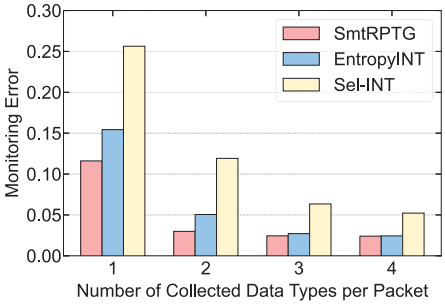


Fig. 6. Monitoring errors of network monitoring schemes.

generally follows the same operation principle of SmtRPTG except for that it uses a pre-estimated DRT without updating it during operation. We assume that the three schemes use the same packet format, where the length of an INT field that contains a sample of one status data type is 12 Bytes and its length increases by 4 Bytes for each additional data type.

This time, we use the traces collected in real-world networks [23], each of which contains 5000 traffic samples, as one status data type (*i.e.*, *Bandwidth*), and assume that the other status data types are static or slow-varying ones (*e.g.*, *Port Number* and *Processing Latency* [19]). We fix the number of data types as $N = 6$ and each INT packet contains $[1, 4]$ types of status data. For fair comparisons, we set the parameters of the three monitoring schemes to ensure that their bandwidth overheads in the data plane are the same (as shown in Fig. 5). Then, based on the sampled results from each scheme, we use linear interpolation to reconstruct the trace of each data type, and compare its distribution with that of the original trace to obtain the relative error between them. The average relative error of each scheme is treated as its monitoring error. As shown in Fig. 6, we can see that SmtRPTG achieves the smallest monitoring error, verifying that it can balance the tradeoff between the overheads and accuracy of network monitoring the best.

V. Conclusion

In this paper, we proposed SmtRPTG as a smart reporting mechanism that can continuously optimize the scheme of network status collecting and reporting to balance the tradeoff between the overheads and accuracy of network monitoring. Specifically, SmtRPTG estimates the distributions of status data of various types based on sampled results, and lets NEs make local decisions on whether and what type of status data should be reported based on the estimations. We designed an algorithm based on the EM approach to solve the problem of distribution estimation for SmtRPTG. Extensive simulations confirmed the effectiveness of our proposal and indicated that it can balance the tradeoff between the overheads and accuracy of network monitoring better than two existing schemes.

Acknowledgments

This work was supported by the NSFC project 62371432.

References

[1] P. Lu *et al.*, "Highly-efficient data migration and backup for Big Data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, pp. 36–42, Sept./Oct. 2015.

[2] Z. Zhu, W. Lu, L. Zhang, and N. Ansari, "Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing," *J. Lightw. Technol.*, vol. 31, pp. 15–22, Jan. 2013.

[3] L. Gong *et al.*, "Efficient resource allocation for all-optical multicasting over spectrum-sliced elastic optical networks," *J. Opt. Commun. Netw.*, vol. 5, pp. 836–847, Aug. 2013.

[4] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 450–460, Feb. 2014.

[5] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: An intellectual history of programmable networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–98, Apr. 2014.

[6] S. Li *et al.*, "Protocol oblivious forwarding (POF): Software-defined networking with enhanced programmability," *IEEE Netw.*, vol. 31, pp. 58–66, Mar./Apr. 2017.

[7] M. Zeng, W. Fang, and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *J. Lightw. Technol.*, vol. 34, pp. 3330–3341, May 2016.

[8] J. Liu *et al.*, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Netw. Serv. Manag.*, vol. 14, pp. 543–553, Sept. 2017.

[9] S. Tang, S. Zhao, X. Pan, and Z. Zhu, "How to use in-band network telemetry wisely: Network-wise orchestration of Sel-INT," *IEEE/ACM Trans. Netw.*, vol. 31, pp. 421–435, Feb. 2023.

[10] S. Lee, K. Levanti, and H. Kim, "Network monitoring: Present and future," *Comput. Netw.*, vol. 65, pp. 84–98, Jun. 2014.

[11] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "A simple network management protocol (SNMP)," *RFC 1157*, May 1990. [Online]. Available: https://tools.ietf.org/html/rfc1157.

[12] B. Claise, "Cisco systems NetFlow services export version 9," *RFC 3954*, Oct. 2004. [Online]. Available: https://tools.ietf.org/html/rfc3954.

[13] INT dataplane specification. [Online]. Available: https://github.com/p4lang/p4-applications/blob/master/docs/INT_v2_1.pdf.

[14] P. Bosshart *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–95, Jul. 2014.

[15] R. Govindan *et al.*, "Evolve or die: High-availability design principles drawn from Google's network infrastructure," in *Proc. of ACM SIGCOMM 2016*, pp. 58–72, Aug. 2016.

[16] Y. Kim, D. Suh, and S. Pack, "Selective in-band network telemetry for overhead reduction," in *Proc. of CloudNet 2018*, pp. 1–3, Oct. 2018.

[17] S. Tang *et al.*, "Sel-INT: A runtime-programmable selective in-band network telemetry system," *IEEE Trans. Netw. Serv. Manag.*, vol. 17, pp. 708–721, Jun. 2020.

[18] B. Basat *et al.*, "PINT: Probabilistic in-band network telemetry," in *Proc. of ACM SIGCOMM 2020*, pp. 662–680, Aug. 2020.

[19] Z. Xu, S. Tang, and Z. Zhu, "Entropy-driven adaptive INT and its applications in network automation of IP-over-EONs," *J. Sel. Topics Quantum Electron.*, vol. 28, p. 3700313, Jul./Aug. 2022.

[20] Z. Xu, Z. Lu, and Z. Zhu, "Information-sensitive in-band network telemetry in P4-based programmable data plane," *Submitted to IEEE/ACM Trans. Netw.*, 2023.

[21] D. Mitrinovic, E. Barnes, D. Marsh, and J. Radok, *Elementary Inequalities*. Noordhoff Groningen, 1964.

[22] D. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 2014.

[23] S. Liu and Z. Zhu, "Generating data sets to emulate dynamic traffic in a backbone IP over optical network," *Tech. Rep.*, 2019. [Online]. Available: https://github.com/lsq93325/Traffic-creation/blob/master/README.md?tdsourcetag=s_pctim_aiomsg