Computers and Electrical Engineering 000 (2017) 1–19

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

A learning-based measurement framework for traffic matrix inference in software defined networks☆

Mehdi Malboubi[a,*], Shu-Ming Peng[a], Puneet Sharma[b], Chen-Nee Chuah[a]

[a] Department of Electrical and Computer Engineering, University of California, Davis, CA, USA
[b] Networked Systems Group at Hewlett Packard Labs, Palo Alto, CA, USA

ARTICLE INFO

Article history:
Received 24 November 2016
Revised 16 November 2017
Accepted 16 November 2017
Available online xxx

Keywords:
Network measurement and inference
Traffic matrix estimation
Software defined networking
Compressed sensing
Multi-armed bandit algorithms

ABSTRACT

In this paper, we propose an intelligent framework for Traffic Matrix (TM) inference in Software Defined Networks (SDN) where the Ternary Content Addressable Memory (TCAM) entries of switches are partitioned into two parts to: 1) effectively aggregate part of incoming flows for aggregate measurements, and 2) de-aggregate and directly measure the most informative flows for per-flow measurements. These measurements are then processed to effectively estimate the size of network flows. Under hard resource constraints of limited TCAM sizes, we show how to design the optimal and efficient-compressed flow aggregation matrices. We propose an optimal Multi-Armed Bandit (MAB) based algorithm to adaptively measure the most rewarding flows. We evaluate the performance of our framework using real traffic traces from different network environments and by considering two main applications: TM estimation and Heavy Hitter (HH) detection. Moreover, we have implemented a prototype of our framework in Mininet to demonstrate its effectiveness.

1. Introduction

In a network, a flow is a sequence of packets that shares common fields that can be readily extracted from packet headers, such as source and destination IP addresses, port numbers and protocol. The size of an Origin-Destination Flow (ODF) quantifies the volume of traffic between the source node and the destination node (in packets or bytes), and the traffic matrix represents the volume of network flows between all nodes. Fine-grained estimates of traffic matrices provide essential information that is central in many networking applications including network design, network traffic monitoring and management [1,2], and network anomaly detection and security [3]. There are two main approaches for TM measurement and estimation. Direct flow-based measurements such as NetFlow and Sampled Flow (sFlow) offer fine-grained measurements that can support different measurement tasks. However, such an approach not only requires dedicated hardware and specialized algorithms, but it is also often challenging, inefficient or even infeasible to monitor each and every flow due to exploding traffic volume and limited monitoring resources (e.g., the number of TCAM entries, storage capacity and processing power). Accordingly and due to non-scalability of direct flow measurement techniques in large-scale networks, intelligent sampling and streaming algorithms have been proposed to estimate statistics or answer specific queries of, e.g.,

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. Zhihan Lu.

* Corresponding author.
E-mail addresses: mmalboubi@ucdavis.edu (M. Malboubi), shmpeng@ucdavis.edu (S.-M. Peng), puneet.sharma@hpe.com (P. Sharma), chuah@ucdavis.edu (C.-N. Chuah).

https://doi.org/10.1016/j.compeleceng.2017.11.020
0045-7906/

approximate size of elephant flows [4]. These solutions are task-specific and lack the full flexibility to dynamically choose which traffic sub-population to measure, and how, depending on application requirements.

An alternate approach is estimating the traffic matrix based on a limited set of measurements using Network Inference (NI) and network tomography methods. However, most network inference problems are naturally formulated as ill-posed Under-Determined Linear Inverse (UDLI) problems where the number of measurements are not sufficient to uniquely and accurately determine the solution. Hence, side information from different sources must be incorporated into the problem formulation to improve estimation precision [5,6].

In today's large-scale networks, the flows of interest can be sparse, or highly fluctuating over time and/or space, due to the presence of a variety of applications and services with different behaviors and Quality of Services (QoS) requirements. The management and operation of today's networks, and providing high QoS for a variety of applications in these complex and highly dynamic networks, depend on the accurate measurement and estimation of the size of network flows. In fact, many recent network operation, monitoring and security applications require timely estimates of both large and small traffic flows with high accuracy. Therefore, the network measurement infrastructure must be agile enough to cope with the dynamic network and traffic conditions. Such a flexible architecture can be achieved due to the recent advent of software defined networking. In fact, an SDN enabler, such as OpenFlow, nicely separates the measurement data plane and control plane functions, and provides the capability to control/re-program the internal configurations of switches in dynamic environments. Consequently, SDN allows applying more complex network monitoring and management applications which results in providing more effective and efficient network services [7,8].

An important resource in SDN switches and routers is TCAM entries which are essential for effective flow classification and forwarding in today's high-speed networks. TCAMs are special computer memory with the capability of real-time and exact-match search which can be used for noncontiguous and arbitrary bitwise packet matching and classifications in network switches and routers [9]. However, TCAMs are expensive hardware and consume a lot of power; hence, the number of TCAM entries in network switches and routers are limited. Therefore, to efficiently allocate such an important resource in dynamic environments, where the behaviors of operating network change, effective optimization and adaptive learning algorithms must be utilized. In fact, adaptive learning algorithms and optimization techniques give us the capability of learning the behavior of network flows and optimally allocating TCAM resources among different network operation and monitoring applications. For this purpose, the run-time programmability of SDN plays an important role to achieve such an intelligent framework with the ability of reconfiguring measurement modules and constructing effective TCAM rules, and to adaptively and easily collect a set of timely-effective flow statistics using counters associate with TCAM entries [10–13].

Accordingly, we propose a framework called intelligent SDN based Traffic (de)Aggregation and Measurement Paradigm (iSTAMP). In iSTAMP, measurement modules can be configured on-the-fly to collect fine-grained measurements of specific traffic sub-populations of interest that directly reflect the monitoring application requirements. Based on the philosophy that not all attributes of interest are equally important, iSTAMP utilizes an intelligent sampling algorithm to select the most informative traffic flows, using information gathered throughout the measurement process. It also exploits Compressed Sensing (CS) inference methods that are effective for estimating highly fluctuated sparse unknown quantities from a set of well-defined compressed linear measurements [14,15].

In order to do this, iSTAMP leverages OpenFlow to dynamically partition the TCAM entries of a switch/router into two parts. In the first part, a set of incoming flows are optimally aggregated to provide well-compressed aggregated flow measurements that can lead to the best possible estimation accuracy via network inference process. The second portion of TCAM entries are dedicated to track/measure the most *rewarding* flows (defined as flows with the highest impact on the ultimate monitoring application performance) to provide accurate per-flow measurements. These flows are selected and "stamped" as important (or *rewarding* from monitoring's perspective) using an intelligent multi-armed bandit based algorithm. These two sets of measurements (aggregated and sampled flows) are then jointly processed to estimate the size of all network flows using different optimization techniques.

Therefore, iSTAMP is scalable framework because it uses a limited number of aggregated flow measurements to estimate the size of a large number of flows. In addition, iSTAMP produces accurate flow estimates since: 1) it uses compressed sensing NI techniques which are effective for estimating highly fluctuated network flows, and 2) part of the flows are perfectly and directly measured by intelligently tracking them, depending on the application.

1.1. Related work

There is a rich literature on improving the accuracy of traffic flow measurements and estimation. We will briefly discuss the most relevant work here. In [16], ProgMe proposes a re-programmable architecture allowing statistics collection based on the notion of flowsets (arbitrary traffic sub-populations) using a flexible flowset composition language. From SDN perspective, most current research has focused on leveraging the run-time programmability/reconfigurability of SDN to passively measure the size of the sub-set of network traffic flows, or identify heavy hitters and/or Hierarchical Heavy Hitters (HHH), which is of a particular importance in different traffic monitoring and network security applications. In [7,8,17], SDN reconfigurable measurement architectures are proposed where a variety of sketches for different direct measurement tasks can be defined and installed by the operator. In [18], an adaptive flow counting method is proposed that controls the temporal and spatial aggregation using a linear prediction method. In addition, in [19], OpenTM directly measures the network's TM by keeping track of statistics for each flow. However, these methods suffer from two challenges: first, the pure SDN

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

based passive flow measurement systems [19] can not provide per-flow measurements of all ODFs in large-scale networks, due to the hard constraint of measurement resources, and second, SDN-based network inference methods, such as [7,8,17], are mainly focused on sub-population flow size estimation and they can not provide the complete visibility of all flows over long time-horizon. Furthermore, these approaches do not directly optimize the usage of available flow-table entries.

iSTAMP, in contrast, leverages off-the-shelf SDN devices and existing OpenFlow API to construct a global view of the available measurement resources which can be used for optimal resource allocation, and consequently, provides the most informative flow measurements at the right time and place. In fact, iSTAMP addresses the problem of SDN-based fine-grained traffic matrix estimation under monitoring resource constraints that has not been addressed previously. In particular, it leverages *statistics* field in OpenFlow to collect traffic measurements. Hence, network TM estimation can be accomplished without affecting packet forwarding behavior or performance. In this paper we concentrate on the centralized implementation of iSTAMP. However, as it has been shown in [10,11,13], iSTAMP paves the way toward providing a unified-distributed framework for traffic matrix estimation in SDN enabled networks which can be used for a variety of networking applications.

1.2. Our contributions

iSTAMP is simple, scalable, and efficient with the ability to intelligently track and measure the most rewarding flows, and effectively aggregate others. iSTAMP adaptively allocates TCAM entries amongst the constantly evolving flows to achieve the highest flow estimation accuracy. Therefore, it is robust in dynamic environments (with highly fluctuating flows over time/space) and under hard resource constraints where, TCAM entries, storage capacity, and network bandwidth are severely limited. In fact, iSTAMP can be easily deployed on commodity OpenFlow-enabled routers/switches to enhance the performance of various network monitoring applications, including TM estimation, Traffic Engineering (TE), Heavy Hitter Identification (HHI), hierarchical heavy hitter detection and different security applications. It also offers timely estimates of network flows with low computation and communication overhead between control and data planes. Furthermore, by providing a well compressed form of aggregated measurements, iSTAMP significantly reduces the required storage for monitoring tasks.

The iSTAMP framework has been originally published in our paper in the proceeding of the 33rd IEEE International Conference on Computer Communication (INFOCOM) in 2014 [10]. This journal version significantly strengthens the previous conference version by improving the presentation of the main concepts, adding new technical contents, developing the mathematical foundation of our main ideas, and providing additional results. In particular, we have added four new sections where: 1) we provide the mathematical foundation for our TCAM partitioning technique (Section 3); 2) we add our new results to indicate the performance of iSTAMP in hierarchical heavy hitter identification (Section 6.3); 3) we demonstrate the prototype of our centralized implementation of iSTAMP framework in mininet (Section 7), and 4) we present the mathematical proofs in Appendix A.

Our main contributions in this paper are summarized as follows:

- iSTAMP is a SDN based framework for network *TM estimation* in which both aggregated flow and per-flow measurements are incorporated into its inference engine. In the context of compressed sensing, we theoretically explain why such a hybrid framework is effective under hard resource constraint of TCAM entries.
- We formulate and solve the problem of designing an optimal binary aggregation (or observation) matrix to maximize the estimation accuracy while providing a compressed form of measurements that are compatible with flow aggregation constraints. Also, we propose a method for designing an efficient-compressed flow aggregation matrix which is effective for estimating highly fluctuated TMs under hard resource constraints of limited TCAM sizes.
- We propose a simple, efficient and an optimal MAB based algorithm to adaptively track and measure the most *rewarding* flows while achieving logarithmic regret over time. The optimality of this multi-armed bandit flow sampling algorithm is proved in Theorem 1.
- We evaluate the performance of iSTAMP using real traffic traces from a variety of network environments and by considering different networking applications, including TM estimation, HH detection, and HHH identification.
- We implement a prototype of iSTAMP in Mininet environment. The feasibility and effectiveness of this prototype has been successfully demonstrated in the 21th GENI Engineering Conference (GEC 21).

The rest of this paper is organized as follows. Section 2 provides an overview of iSTAMP and different network inference techniques that we have used, and Section 3 introduces our main compressed sensing network inference technique. In Section 4 we describe our optimal aggregation matrix design procedure, and in Section 5 we present our intelligent MAB based flow sampling algorithm. In Section 6, we evaluate the performance of iSTAMP considering three main applications: ODF estimation, heavy hitter detection, and hierarchical heavy hitter detection. Section 7 demonstrates the results of our centralized implementation of iSTAMP in mininet environment. Finally, Section 8 summarizes our most important results.

2. Network model and system description

Consider a network with N nodes where an IP address is assigned to each node. In the context of SDN, a flow can be identified by multiple fields in a flow-table entry. For developing the main ideas in this paper, without loss of generality, a flow is defined by a tuple consisting of a source IP address and a destination IP address as $<\text{source IP}, \text{destination IP}>$. In addition, to address the problem of flow measurement and estimation under hard resource constraints of TCAM sizes in

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

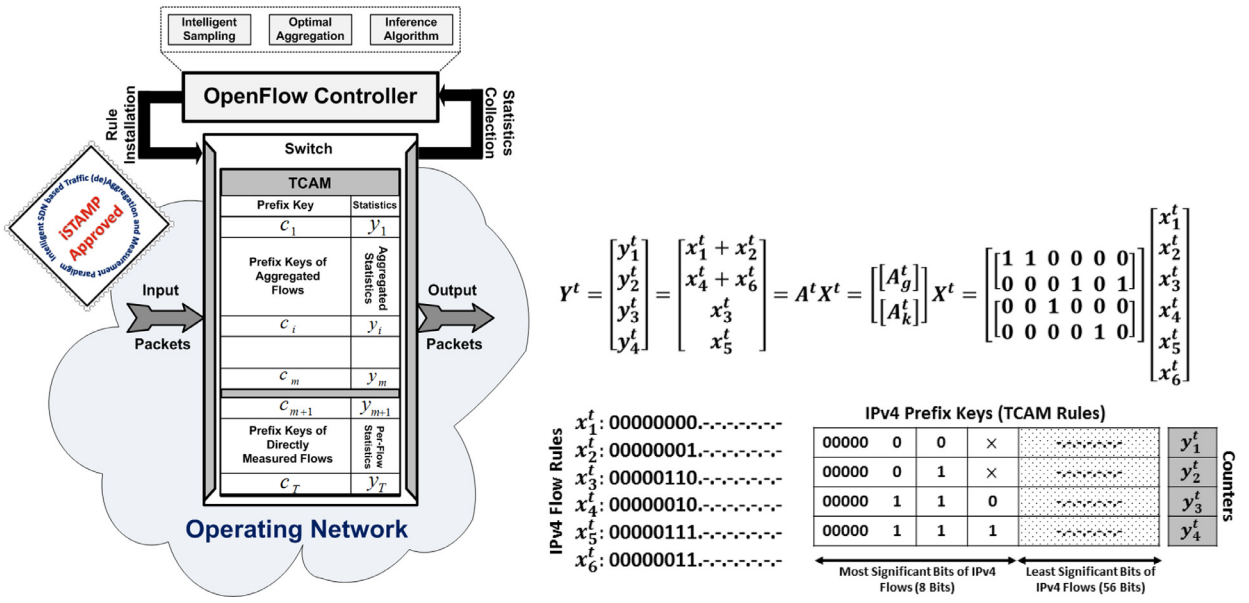

Fig. 1. The architecture of centralized iSTAMP framework, and an illustrative example for configuring TCAM rules.

SDN switches, it is assumed that the size of a flow is measured by counters associated with TCAM entries. However, the same idea can be applied to the cases where other technologies such as virtual/software-based switches are used for flow classification and forwarding.

Fig. 1 shows the general block diagram of the centralized iSTAMP framework where the TCAM at the switch or router is partitioned into two parts for (i) aggregate measurements, and (ii) per-flow monitoring of selected flows, respectively. The effective aggregate statistics from the first part provide a set of well-formed compressed linear measurements as input to a network inference process to perform flow size estimation. The second part of TCAM is allocated for tracking individual flows that are deemed *important* (or rewarding) for the monitoring application in question. This process is called the intelligent sampling of flows. For illustrative purpose, we consider ODF size estimation, and more generally, traffic matrix (as a measure of origin-destination traffic size between nodes) estimation, as the driving application in this section. In this context, ODFs with the largest volume will be the 'important' candidates for per-flow monitoring. Note that a single router may only observe a subset of ODFs (or partial TM) in the network. We will first discuss how iSTAMP can be deployed on a single router/switch to estimate the ODFs it observes. Due to the flexibility and scalability of iSTAMP, such a hybrid framework (where both aggregated and per-flow measurements are used for TM estimation) can be easily extended to a distributed monitoring case [11,13], where multiple routers running iSTAMP can coordinate to estimate the complete TM for the network.

Assume there are n ODFs in the network and T entries of TCAM at the switch which can be used for network measurement where $n >> T$. At each measurement interval τ, K out of T entries are used to track and measure the most rewarding ODFs and m entries are used to optimally aggregate other ODFs (i.e. $T = m + K$). For this purpose, the flexibility of OpenFlow is used to install TCAM wildcard matching rules (prefix keys) c_i and collect associated statistical counts y_i in each measurement interval τ_t. At the controller, which can be co-located on the switch or reside on a separate machine/server, the measurement statistics/counts are processed. In the next epoch τ_{t+1}, the best ODFs (determined based on the application requirement) for direct measurement are selected and their corresponding prefix keys are installed in K TCAM entries. This process is called de-aggregation and the TCAM lookup mechanism, which is based on highest wildcard matching rule, facilitates the implementation of this process. In addition, m entries of TCAM is used for optimal aggregation of $n - K$ flows. The controller can poll the statistic counts periodically or in different measurement intervals, the frequency of which is limited by practical constraints.

At each epoch t, the set of measurement statistics Y^t is represented by the under-determined linear system of equations shown in Eq. (1) where Y_g^t denotes the aggregated measurements and Y_k^t denotes the direct per-flow measurements. In this equation, A^t is an $(T \times n)$ measurement matrix consisting of an $(m \times n)$ binary aggregation matrix A_g^t, where non-zero entries in each row of A_g^t represents the set of flows aggregated (or mapped) to that entry, and an $(k \times n)$ binary matrix A_k^t, where the only one non-zero entry in each row demonstrates the selected flow for direct measurement in tth measurement interval τ_t. Also, X^t denotes the tth vector of flows where X_g^t is the set of ODFs that are being mapped to m groups for aggregate measurements (that must be estimated) and X_k^t is the set of directly measured ODFs (i.e. $X_k = Y_k$). Having these measurements, the set of unknown flows X_g^t can be estimated using the following general optimization formulations Eqs. (2) and (3) where $A_g^t(:, I_g^t)$ indicates a sub-matrix of A_g^t consisting of all its rows (indicated by the colon operator) and its columns

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

Table 1
The most commonly used notations where $I_g^t \bigcap I_k^t = \emptyset$.

Notation	Description
N	Number of nodes in the network.
n	Number of flows in the network ($n \leq N(N-1)$).
T	Number of TCAM entries at the switch ($T = m + K$).
m	Number of TCAM entries used for flow aggregation.
K	Number of TCAM entries used for direct per-flow measurement.
t	Notation used to denote time.
I_g^t	Set of flow indices that must be aggregated at tth epoch.
I_k^t	Set of flow indices that must be directly measured at tth epoch.
X^t	Vector representation of traffic matrix at tth epoch.
$X_g^t = X^t(I_g^t)$	Vector of aggregated ODF sizes at tth epoch.
$X_k^t = X^t(I_k^t)$	Vector of directly measured ODF sizes at tth epoch.
A^t	Flow measurement matrix of size $T \times n$ at tth epoch.
A_g^t	Flow aggregation matrix of size $m \times n$ at tth epoch.
A_k^t	Direct per-flow measurement matrix of size $K \times n$ at tth epoch.
Y^t	Flow measurement vector at tth epoch.
Y_g^t	Aggregated flow measurement vector at tth epoch.
Y_k^t	Direct per-flow measurement vector at tth epoch.
Y_S	Vector of SNMP link-load measurements.
H	Routing Matrix.
τ_t	Flow measurement interval at tth epoch.
λ	Regularization parameter.
(p, q, F)	Notations used to denote norm operators.
$\|X\|_1$	$\sum_{j=1}^{n} \left\|x_j\right\|$.
$\|X\|_2^2$	$\sum_{j=1}^{n} \left\|x_j\right\|^2$.
$\|A\|_F^2$	$\sum_{i=1}^{m} \sum_{j=1}^{n} \left\|a_{ij}\right\|^2$.

indicated by set I_g^t. Having both sets of estimated flow-sizes $\hat{X}_g^t$ and directly measured flow-sizes X_k^t, $\hat{X}^t$ is constructed as $\hat{X}^t = \hat{X}^t(I_g^t) \bigcup \hat{X}^t(I_k^t)$ where $\hat{X}^t(I_g^t) = \hat{X}_g^t$ and $\hat{X}^t(I_k^t) = X_k^t$. In Eqs. (2) and (3), parameters (p, q, λ) are defined based on different optimization techniques. The particular choices of $p = 2$, $q = 1$ and $\lambda = \lambda_0 (\in \mathbb{R})$ defines a realization of Compressed Sensing (CS) inference technique, called Least Absolute Shrinkage and Selection Operator (LASSO) [14]. The compressed sensing inference techniques (defined in Eq. (5) in general form) are effective for estimating highly fluctuating unknown quantities (e.g. network flows) from a set of limited number of linear combinations of unknowns of interest. Furthermore, Eq. (3) incorporates side information Y_S into the optimization problem to improve the performance of the overall TM estimation process (in both centralized and extended multi-point/distributed monitoring case). Here, side information Y_S^t is the SNMP link load measurements $Y_S^t = HX_g^t$ where H is the routing matrix. Table 1 lists our commonly used notations.

$$Y^t = \begin{bmatrix} Y_g^t \\ Y_k^t \end{bmatrix} = A^t X^t \text{ where } A^t = \begin{bmatrix} A_g^t \\ A_k^t \end{bmatrix} \tag{1}$$

$$\begin{aligned} &\hat{X}^t = \underset{X_g^t}{\text{minimize}} \left\| Y_g^t - A_g^t(:, I_g^t) X_g^t \right\|_p^2 + \lambda \left\| X_g^t \right\|_q \\ &\text{s.t.} \quad X_g^t \geq 0 \end{aligned} \tag{2}$$

$$\begin{aligned} &\hat{X}^t = \underset{X^t}{\text{minimize}} \left\| Y_S^t - HX^t \right\|_p^2 + \lambda \left\| X_g^t \right\|_q \\ &\text{s.t.} \quad Y_g^t = A_g^t X^t, \quad X_k^t = A_k^t X^t, \quad X_g^t \geq 0 \end{aligned} \tag{3}$$

2.1. TCAM configuration in iSTAMP: an illustrative example

Given the set of IPv4 flows $\left\{x_1^t, \ldots, x_6^t\right\}$, the flow aggregation matrix A_g^t and per-flow measurement matrix A_k^t at tth epoch, Fig. 1 illustrates how TCAM rules can be configured to provide the set of desirable flow statistics $\left\{y_1^t, \ldots, y_4^t\right\}$. In this example, it is assumed that network flows are distinguishable based on the Most Significant Bits (MSB), and accordingly, TCAM rules can be constructed by appropriately configuring of the MSBs of network flow tuples where $\times$ indicates the don't-care operator. TCAM rules can be (re)configured using OpenFlow protocol, as a standard protocol to configure SDN switches. Note that, arbitrary bitwise matching (for IP addresses) is supported by OpenFlow 1.2 and higher. Accordingly, the iSTAMP controller uses flow-table modification and flow statistics request messages to configure flow-table entries and collect statistics.

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

2.2. iSTAMP: centralized and distributed implementations

Considering the above optimization formulations in Eqs. (2) and (3), iSTAMP framework can be implemented in both centralized (i.e. a single-point) and distributed (i.e. multi-points) measurement scenarios. In this paper, the focus is on developing the main ideas and establishing the foundations of an adaptive and effective network flow inference framework in the centralized scenario [10] which can be easily extended to distributed scenarios. As a matter of fact, in [11,13], we have addressed the implementation of iSTAMP framework in a distributed scenario with considering more practical flow-aggregation and routing constraints. In iSTAMP framework, the vector of ODF sizes X^t represents different notion of flows at different levels. For example, ODFs can be traffic between routers in an ISP network, or they can be traffic between servers in different racks in data centers.

Under single point measurement, we consider two sub-cases: (a) when OpenFlow Switch (OFS) is used for both routing and measurement, and hence the aggregation matrix A_g cannot be arbitrary in order to preserve routing of the aggregate flows, and (b) the OpenFlow switch is used primarily for flow measurement, and routing does not have to be preserved, as in the case of random or hash-based routing. These two cases are compatible with both formulations in Eqs. (2) and (3). In the former case A_g^t is the local routing matrix and part of the TCAM entries is used to directly measure a set of the most informative flows. Note that reserved-empty TCAM entries are always available at switches. In the latter case, all TCAM entries of the OFS are used for flow measurement; accordingly, along with directly measured flows, A_g^t can be optimally designed to enhance the accuracy of estimation. Likewise, there are two possible cases in the multi-point measurement scenario, where installation rules are transmitted from Central Controller (CC) to local OFSs and their measurements are transmitted to the CC. If all TCAM entries of OFSs are used for measurement, then both Eqs. (2) and (3) are applicable and A_g^t can be optimally designed. However, if local OFSs are used for routing and measurement, then only Eq. (2) can be applied because, in this case, $A_g^t \cong H$. Please refer to [20] for more details.

In iSTAMP, the communication overhead between the controller and switch is low because only a few flows are directly measured and other entries are used to aggregate a large number of flows. Hence, T instead of n (where $T << n$) aggregated and per-flow measurements are transmitted to the controller which is important under hard constraints of limited TCAM sizes and in multi-point measurement scenario. iSTAMP is also computationally efficient because it exploits existing low-complexity NI techniques [14,21].

3. Insight into compressive network inference in iSTAMP

Today's network traffic are highly fluctuating over time and/or space. These high variations can be modeled as sparse signals (modeling large flows known as heavy hitters or elephants) contaminated with noise (modeling small flows known as mice). Therefore, efficient compressed sensing estimation algorithms with low computational complexity, such as Orthogonal Matching Pursuit (OMP), can be used for recovery of sparse signals [14]. For this purpose, it has been shown that if the coherence of the observation matrix A_g (defined as μ in Eq. (4)) is sufficiently small, then the convex optimization program Eq. (5), can exactly recover a sparse signal with l non-zero quantities (in some basis) from $m = O(l \log(n/l))$ properly designed linear observations. In fact, the recovery succeeds if $l \leq 0.5\left(1 + \frac{1}{\mu}\right)$. Thus, the smaller μ, the higher the bound on the sparsity of X_g [14]. It should be noted that for the sake of simplicity, in the following discussions, we drop the notion t and assume $A_g = [a_{ij}] \in \mathbb{R}^{m \times n}$ $(i = 1:m, j = 1:n)$ is written as $A_{g_{(m \times n)}} := [a_1, \ldots, a_n]$ where $\{a_j\}_{j=1}^{n} \in \mathbb{R}^m$.

$$\mu = \max_{i \neq j} \frac{\left|a_i^T a_j\right|}{\|a_i\|_2 \left\|a_j\right\|_2} \tag{4}$$

$$\hat{X_g} = \min_{X_g} \|X_g\|_1 \qquad \text{s.t.} \qquad Y_g = A_g(:, I_g) X_g \tag{5}$$

In iSTAMP we partition the T entries of TCAM at the switch or router into two parts for (i) aggregate measurements (with m TCAM entries where $m = T - K$), and (ii) per-flow monitoring of the K heaviest flows (using K entries of TCAM), respectively. The main intuitions behind the proposed hybrid technique are as follows. First, this technique allows us to provide a set of possible aggregated/compressed measurements of the larger subset of flows such that a good estimation accuracy is achieved via compressive network inference process. Second, measuring the K heaviest flows and reducing the large variations in the input traffic enhance the performance of network inference techniques. This is consistent with the previous studies where it has been shown that in real networks a small number of flows may account for a large fraction of traffic volume [16,22,23]. Therefore, this approach improves the performance of HH identification, Load Balancing or Denial of Service (DoS) attack detection.

To explain and quantify the effectiveness of this method we consider the simple scenario of CS recovery in the presence of White Gaussian Noise (WGN). For this purpose, we compute the number of required measurements for the exact CS recovery of an l-sparse signal (i.e. a signal with l non-zero elements). To be consistent with our framework, the number of required measurements is denoted by T. Here for the exact CS recovery $T \geq C_0 l \log\left(\frac{n}{l}\right)$ where depending on the original

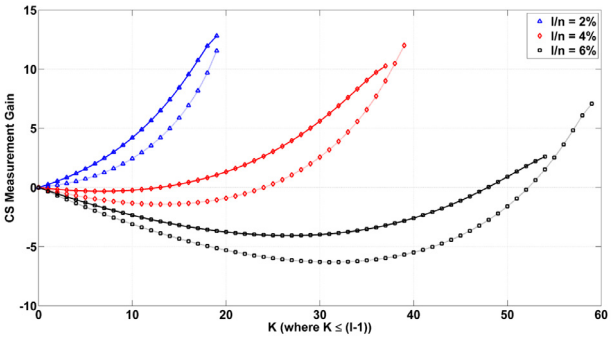

Fig. 2. CSMG for different values of $\frac{l}{n}$ where $n = 1000$ and $\tilde{a}_m = 1$ (solid line: SNR = 10 dB, dotted line: SNR = 15 dB).

Table 2
Real Datasets under study.

Network	Date	Duration	Resolution	TM size ($n \times T_c$)
GEANT [30]	2005-01-08	1 week	15 min	529 × 672
Abilene [29]	2004-05-01	1 week	5 min	144 × 2016
Data Center (Univ1) [23]	2010	≈ 1 h	1 s	48(/192) × 3500

Signal-to-Noise Ratio[1] (denoted by SNR_o) C_0 is computed as $C_0 = \frac{2}{\log(1+SNR_o)}$ [24]. Accordingly, in our hybrid architecture where the K heaviest flows are directly measured, the number of required measurements for the CS recovery of a sparse signal of dimension $(n-K)$ with sparsity $(l-K)$ must be greater than $\acute{C}_0(l-K)\log\left(\frac{n-K}{l-K}\right)$, where $\acute{C}_0 = \frac{2}{\log(1+SNR_n)}$ and SNR_n is the new SNR. Therefore, the improvement achieved by our hybrid framework can be quantified by the Compressed Sensing Measurement Gain (CSMG), as defined in Eq. (6). The CSMG indicates the number of measurements that we gain to accurately infer an l-sparse signal via the CS recovery process in our hybrid fine-grained flow measurement framework comparing against the CS inference where all n flows are aggregated using all T entries of TCAM without direct flow measurement. The positivity of CSMG shows that, by measuring the K heaviest flows in our hybrid technique, the number of required measurements for the exact CS recovery of an l-sparse signal is reduced.

$$CSMG = C_0 l\log\left(\frac{n}{l}\right) - \left(\acute{C}_0(l-K)\log\left(\frac{n-K}{l-K}\right) + K\right) \tag{6}$$

To evaluate the performance of this technique, first, we quantify CSMG for a general case where the original signal X consists of l non-zero elements with amplitude $\tilde{a}_m$ and $(n-l)$ zeros, contaminated with WGN[2] with variance σ^2. Accordingly, the SNR of original signal is $SNR_o = \frac{l\tilde{a}_m^2}{n\sigma^2}$. By measuring K non-zero elements, the new SNR is computed as $SNR_n = \frac{(l-K)\tilde{a}_m^2}{(n-K)\sigma^2}$. Fig. 2 shows the CSMG which indicates that there is a positive K for the range of values of $\frac{l}{n}$ (equivalently the percentage of the number of large flows in network traffic) and SNRs that we are facing in network monitoring applications [6,25]. Note that, the CSMG is higher at low or moderate SNRs, that is, at low SNRs it is more constructive to directly measure part of the attributes of interest. Therefore, it is beneficial to partition the TCAM entries (or measurement resources, in general) into two parts and directly measure the K heaviest flows. For example, in GEANT network (Table 2), by setting the threshold for identifying large flows to 10% of the link capacity, traffic flows can be approximately modeled as sparse signal contaminated with additive noise with SNR = 22.85 dB, $\frac{l}{n} = 0.0335$ and $C_0 = 0.6305$; here, the power of the signal and noise are computed as the power of flows higher and less than threshold, respectively. Accordingly, the CSMG is positive for all values of K where $\acute{C}_0$ changes from 0.6305 to 0.6766.

In addition, in [14] and references therein, it has been shown that for measurements contaminated with WGN, the upper bound of estimation error $\left\|X - \hat{X}\right\|_2$ is reduced for a smaller μ, and also, it is proportional to $\sqrt{l\log(n)}$. Hence, the measurement accuracy may be improved by measuring K largest flows (i.e. lowering l), supporting the effectiveness of our hybrid technique for measuring the K heaviest flows.

Measuring the K largest quantities can also enhance the estimation accuracy of other inference methods. For example, since Least Square Estimation (LSE) methods are sensitive to unusual inputs, reducing the number of large variations could enhance the estimation accuracy. Therefore, it is of particular importance to track and measure the K heaviest flows in highly

[1] The Signal-to-Noise Ratio (SNR) is defined as the ratio of the power of the signal to the power (or variance) of the noise.
[2] Note that flow measurement is a noisy process due to sampling and disalignment of polling intervals which are modeled as WGN [6].

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

dynamic network environments where flows are rapidly fluctuating over time and space. In the following, we introduce effective methods for the efficient-compressed aggregation and optimal tracking of heaviest flows.

4. Optimal flow aggregation: methodology and challenges

In this section, we explain how to design the optimal compressed sensing aggregation matrix A_g^t in our iSTAMP framework. We also discuss the various challenges involved and how we address them.

In the literature of compressed sensing, it has been shown that random observation matrices with i.i.d. Gaussian or random ± 1 entries, and sufficient number of rows can achieve small coherence with overwhelmingly high probability. However, it is a valid question to ask what the optimal observation (i.e. aggregation) matrix is to maximize the performance of CS recovery. Such a direct optimization approach for designing the optimal observation matrix A_g^{Opt}, to maximize the performance of CS recovery technique, is prohibitive due to the complexity of the process [26]. Therefore, to simplify the process of designing the optimal observation matrix other objective functions are considered in the optimization process, while accepting the unavoidable sacrifice in performance [26].

In [27], it has been demonstrated that by minimizing all off-diagonal elements of the corresponding Gram matrix $G = A_g^T A_g$, an observation matrix with small coherence can be designed. For this purpose, $\left\|A_g^T A_g - I\right\|_F^2$ is minimized using continuous optimization approaches. The main motivation behind this technique is that minimizing the sum of the squared inner products of all columns(/atoms) of A_g results in an observation matrix with more orthogonal columns which improve the performance of the CS recovery process.

Accordingly, here the same strategy is used for designing the flow aggregation matrix. However, to design the optimal flow aggregation (i.e. observation) matrix, we must note that Openflow switches typically classify the incoming packets (e.g. forward/measure or drop) based on the longest prefix match against pre-configured TCAM/flow-table entries. As a result, our aggregation matrix is typically a binary matrix (i.e. $a_{ij} \in \{0, 1\}$). Also, since in [27] the column of the observation matrix is normalized, that is, $\left\{\left\|a_j\right\| = 1\right\}_{j=1}^{n}$; therefore, in our binary formulation, we consider the diagonal matrix ϕ, instead, to force the optimization engine to focus on minimizing the sum of off diagonal elements of the objective function $\left\|A_g^T A_g - \phi\right\|_F^2$, and to control the number of ones's in each column. Proposition 1 represents lower and upper bounds for our objective function in Eq. (8). This Proposition shows that the performance of the objective function Eq. (8) is a function of the size of matrix and ϕ.

Proposition 1. *Assume A_g is a real-positive $m \times n$ matrix where $m < n$ and ϕ is a positive diagonal matrix. Then objective function $\left\|A_g^T A_g - \phi\right\|_F^2$ is bounded as the following (for complete proof please refer to [20]).*

$$\left\|A_g^T A_g - \phi\right\|_F^2 \geq Tr\left[\phi^2\right] - m\phi_{ii_{max}}^2$$

$$\left\|A_g^T A_g - \phi\right\|_F^2 \leq \left[Tr\left(A_g^T A_g\right) - \phi_{ii_{min}}^2\right]^2 + Tr\left[\phi^2\right] - \phi_{ii_{min}}^2$$

In this paper, we introduce the following *generic* integer optimization program Eq. (8) to design the optimal aggregation matrix where the first constraint emphasizes that A_g is a binary matrix. The second constraint controls the redundancy between aggregated measurements where, in general, redundancy can be achieved by observing a flow at multiple TCAM entries.[3] The non-zero entries of diagonal matrix ϕ is chosen properly by the designer to control the redundancy among measurements. Note that, C_l and C_u are important parameters that are constrained by routing rules, flow table configurations, and TCAM lookup limitations.

The third constraint enforces the optimization engine to design an optimal aggregation matrix that is compatible with any other set of desirable flow aggregation rules denoted by set $\mathcal{F}$ where $|\mathcal{F}|$ indicates the number of flow aggregation rules. Realization of $\mathcal{F}$ can be the set of rules representing flow aggregation policies or network routing constraints. In practical implementations, these constraints can be eventually mapped into internal forwarding rules in SDN switches [11,13]. In the third constraint, $a_{i:}$ denotes the ith row of the aggregation matrix A_g, and $\odot$ denotes the matching operator where the compatibility of $a_{i:}$ with jth rows of $\mathcal{F}$ (denoted by $\mathcal{F}_j$) is examined by bit-by-bit comparison. If $a_{i:}$ is compatible with jth row of $\mathcal{F}$ then $\left|a_{i:} \odot \mathcal{F}_j\right| = 1$. Accordingly, in OpenFlow 1.2 and higher that matching against arbitrary bitmasks for IP addresses is allowed (where bitmasks can be in a given field and need not be adjacent or at the beginning of the field), the generated optimal flow aggregation rules can be feasibly configured on TCAM entries.

Considering the fact that A_g is a binary matrix and a_{ij}'s are binary variables, we have shown that such a complex, non-convex and non-linear objective function in Eq. (8) can be reformulated as Eq. (9) that is still a non-linear integer program. To effectively convert Eq. (9) into a linear integer optimization program we use the trick that each multiplication xy of binary variables x and y can be linearized by rewriting the multiplication term xy as a new variable z, where $z = xy$, and considering three constraints: $z \leq x$, $z \leq y$ and $z \geq x + y - 1$. Accordingly, Eq. (9) is converted to the linear integer optimization program in Eq. (10) that can be solved using standard integer optimization tools, such as CPLEX, to design the optimal CS

[3] In fact, this constraint is effective for the distributed implementation of iSTAMP where a flow can be measured by different aggregation rules at different SDN switches, distributed across the network [13].

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

aggregation matrix. Here, to reduce the computational complexity, we have ignored the third constraint in Eq. (8). In Eq. (10), C_j denotes the set of z_j and z_i where the corresponding a_{ij} satisfies the second constraint in Eq. (8), respectively. Note that, if $C_l = C_u = 1$, then the objective function in Eq. (10) is simplified by reducing K_u to $mn(n-1)$ and removing $\sum_{i=1}^{K_v} v_i$; in this case $\phi = I$. In addition, parameter ϕ is given, based on desirable redundancy amongst measurements. To justify the performance of binary optimization framework Eq. (10), we used CPLEX to implement our optimization engine. An example of optimal A_g with size $(m = 4, n = 8)$ and $\mu = 0.7066$ has been shown in Eq. (7).

$$A_{g_{4\times 8}} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$\underset{A_g}{\text{minimize}} \left\| A_g^T A_g - \phi \right\|_F^2 = \underset{A_g}{\text{minimize}}\ Tr[(A_g^T A_g - \phi)^2] \tag{8}$$

$s.t.$

$$① : \left\{a_{ij}\right\}_{(i,j)=(1,1)}^{(m,n)} \in \{0,1\}$$

$$② : C_l \le \sum_{i=1}^{m} a_{ij} \le C_u \quad \forall j = 1, \ldots, n$$

$$③ : \sum_{j=1}^{|\mathcal{F}|} \left| a_{i:} \odot \mathcal{F}_j \right| \ge 1 \quad \forall i = 1, \ldots, m$$

$$\begin{aligned} \underset{A_g}{\text{minimize}} \left\| A_g^T A_g - \phi \right\|_F^2 = \underset{A_g}{\text{minimize}} \Bigg\{ & \sum_{i=1}^{m}\sum_{j=1}^{n}(1-2\phi_{jj})a_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1,k\neq j}^{n} a_{ij}a_{ik} \\ & + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1,k\neq i}^{m} a_{ij}a_{kj} + \sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}\sum_{k=1}^{n}\sum_{l=1,l\neq k}^{n} a_{ik}a_{il}a_{jk}a_{jl} + \sum_{j=1}^{n}\phi_{jj}^2 \Bigg\} \end{aligned} \tag{9}$$

$$\underset{Z,U,V}{\text{minimize}} \left\{ \sum_{i=1}^{m}\sum_{j=1}^{n}(1-2\phi_{jj})z_{(i-1)n+j} + \sum_{j=1}^{n}\phi_{jj}^2 + \sum_{i=1}^{K_u} u_i + \sum_{i=1}^{K_v} v_i \right\} \tag{10}$$

where $A^T(:) = Z = [z_1, \ldots, z_{mn}], U = [u_1, \ldots, u_{K_u}], V = [v_1, \ldots, v_{K_v}]$,
$K_u = mn(n-1) + nm(m-1)$ and $K_v = m(m-1)n(n-1)$
$s.t.$
(a) for each multiplication $z_j z_k : u_i \le z_j, u_i \le z_k, u_i \ge 0$,
$u_i \ge z_j + z_k - 1$
(b) for each multiplication $z_j z_k z_l z_m : v_i \le z_j, v_i \le z_k, v_i \le z_l, v_i \le z_m$,
$v_i \ge 0, v_i \ge z_j + z_k + z_l + z_m - 3$
(c) $C_l \le \sum_{j\in C_j} z_j \le C_u$
(d) $z_i \in \{0,1\}$

4.1. Challenges in optimal aggregation matrix design and solutions

In the previous sections, we explained that it is very hard to directly design the optimal CS observation matrix to maximize the ultimate performance of NI problems and other metrics must be optimized, instead. Accordingly, we proposed a method for designing optimal binary observation matrix for network monitoring applications. However, in reality for a switch with a single TCAM set, the wildcard matching rule in current TCAM lookup technology does not allow a particular flow to be mapped into more than one TCAM entry in each switch, that is, $C_l = C_u = 1$. This imposes hard constraints on our optimization program Eq. (10), which otherwise could have benefited from redundant measurements when there can be multiple ones in each column and the coherence of the aggregation matrix can be decreased which can potentially leads to a higher estimation accuracy via CS estimation techniques. On the other hand, the optimal aggregation matrix framework Eq. (8), belongs to the set of binary combinatorial optimization problems which are NP-Complete. Hence, for large-scale networks the computational complexity of inputting this problem into a solver and solving it is intractable. Considering these

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

challenges, in Section 5 we introduce a new heuristic aggregation matrix which can provide accurate estimates for highly fluctuated TMs. Such a matrix can be designed using information from other sources.

Among these, making use of auxiliary SNMP link counts, which are readily and reliably available, can improve the performance of our TM estimation framework. SNMP link-loads are provided using $Y_S^t = HX^t$ [21] where H is the routing matrix with low average coherence as there are multiple ones in each column, and with the capability of providing redundant flow measurements. Accordingly, Eq. (3) can be used for TM estimation where additional aggregated statistics are used as auxiliary measurements which can significantly improve the estimation accuracy. This approach is compatible with both of our single-point and multi-point measurement scenarios.

5. Online flow identification and sampling

The notion of flow sampling here denotes the process of sequential allocation of K out of T entries of TCAM at epoch t to target flows, which are selected based on the information collected up to that time. The main goal is to adaptively track and measure the most rewarding(/informative) traffic flows that, if measured accurately, can yield the best improvement of overall measurement utility (the exact performance metric is dependent on the monitoring application). For this purpose, multi-armed bandit sequential resource allocation algorithms are used. A classic MAB problem involves a number of independent arms, each of which, when played, offers random reward drawn from a distribution with unknown mean. At each time, a player chooses a subset of arms to play, aiming to maximize reward or minimize regret over some time horizon T_c. The Restless Multi-Armed Bandit (RMAB) optimal policies [28] can be applied effectively to our framework for intelligent flow sampling in dynamic networks where: a) flows are independent; b) the dynamics of flows are not-known, and c) flow sizes can vary stochastically with time. In general, the optimal policies consists of two phases: exploration or learning phase, in which system dynamics is learned, and exploitation phase, where the most rewarding flow(s) (i.e. arm(s)) are measured (i.e. played).

In our problem, we would like to identify, track, and measure the K largest flows amongst n flows using K out of T entries of TCAM to: 1) improve the ability of our optimization techniques in Eqs. (2) and (3) for fine-grained flow estimation and 2) enhance the capability of the monitoring system in heavy hitter identification where heavy hitters are flows with a flow size larger than a threshold θ. Note that HHI is of particular importance in both network management and security applications. For a flow sampling strategy, to function effectively and efficiently in dynamic environments and under hard resource monitoring constraints (where $T << n$), the duration of the learning phase must be short. This is important because in many applications measurement intervals τ_ts are long, for example 5–15 min (see Table 2) which leads to very long learning durations. Therefore, after a comprehensive survey, we adopted the Upper Confidence Bound (UCB) algorithm in [28] and modified it based on our application requirements to propose our Modified Upper Confidence Bound algorithm (Algorithm 1).

Algorithm 1 Modified Upper Confidence Bound (MUCB).

Input: Time horizon T_c and parameter α.
Output: At each epoch t, the set of sorted indices of incoming flows (I^t) in descending order where $I^t = I_k^t \bigcup I_g^t$.
while True **do**
- Set $t = 1$, measure all n flows $\{x_j\}_{j=1}^n$ using all T entries of TCAM over $\lceil \frac{n}{T} \rceil$ epochs, and set $t_c = n$.
while $t < T_c$ **do**
- Compute flow indices $I_j^t = \alpha \bar{x}_j + \sqrt{\frac{2\log(t_c)}{t_j(t_c)}}$ for all flows $j = 1, \ldots, n$ where $\bar{x}_j$ is the average flow size for jth flow, $t_j(t_c)$ is the number of times flow j has been measured upto time t_c and t_c is the overall number of measurements done so far.
- Sort the set I^t and report indices in descending order as $I^t = I_k^t \bigcup I_g^t$.
- Allocate k measurement entries to the k flows with indices in I_k^t and measure them.
- $t = t + 1$, $t_c = t_c + K$.
end while
end while

In MUCB algorithm, we modify the original UCB algorithm [28] to choose the K most rewarding flows. For this purpose, first, the time horizon T_c is determined. Then all T entries of TCAM are used to measure all n incoming flows over $\lceil \frac{n}{T} \rceil$ measurement interval and the indices of incoming flows, sorted in descending order, are reported including the indices of K heaviest flows (I_k^t) and the indices of flows that must be aggregated (I_g^t). Accordingly, the direct flow measurement matrix A_k^t is defined as $A_k^t(:, I_k^t) = 1$ where : denotes all rows of matrix. In computing flow indices the first term $\bar{x}_j$ favors measurement resources toward flows that are historically larger. In fact, larger flows that are measured more frequently are dominated by the first term and smaller flows that are observed less frequently are dominated by the second term. To improve the agility of the algorithm in dynamic environment, for observing a variety of flows, we also modify the original UCB algorithm [28] by multiplying the first term $\bar{x}_j$ by a coefficient α where $0 < \alpha \leq 1$. In this algorithm, the parameters T_c

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

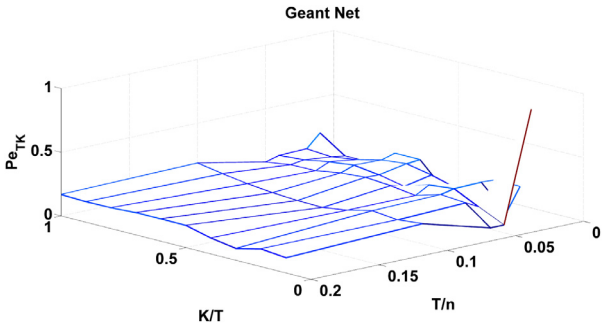

Fig. 3. Pe_{TK} for Geant network with different configurations where $\alpha = 1$ and the average of Pe_{TK} is taken over duration Tc, defined in Table 2.

and α can be adjusted by the user. We also propose the following method in Eq. (11) to use the temporal auto-correlation of flows to compute T_c as:

$$T_c = \frac{1}{n}\sum_{j=1}^{n} t_{R_x^j}(\beta) \quad \text{and} \quad \beta \leq 1 \tag{11}$$

where $t_{R_x^j}(\beta)$ is the delay (t_d) that the auto-correlation of jth flow x_j (i.e. $R_x^j := E[x_j(t)x_j(t-t_d)]$) drops to $\beta \max(R_x^j)$. Typically, β is set to 0.5, but smaller β leads to larger T_c. In Theorem 1, we have proved that the total regret using this algorithm can be bounded, logarithmically if α satisfies a specific convergent condition. In practice, α can be in the interval (0, 1] without losing the performance, as we have shown in Section 6 through our direct experiment.

Theorem 1. *For all $n > 1$, if policy MUCB is run on n machines having arbitrary reward distributions $\{P_i\}_{i=1}^n \in [0, U]$ with expected values $\{v_i\}_{i=1}^n$, then, the total expected regret after any number t_c of plays is at most:*

$$\tilde{R}_K(t_c) \leq \sum_{j=1}^{K}\left[8\alpha^2U^2\sum_{i:\tilde{v}_i<\tilde{v}_j^*}\left(\frac{\log(t_c)}{\tilde{\Delta}_i^j}\right) + (1+\psi(U,\alpha))\Sigma_j\right]$$

where $\tilde{\Delta}_i^j := \tilde{v}_j^* - \tilde{v}_i$, $\tilde{v}_j^*$ denotes the jth maximum of the set $\{\tilde{v}_i\}_{i=1}^n$, A_j^S is the set of arms in jth play, $\Sigma_j = \sum_{i\in A_j^S}\tilde{\Delta}_i^j$, and $\psi(U, \alpha)$ is a bounded function of U and α (for complete proof please refer to Appendix A).

The MUCB algorithm is simple to implement and the exploration(/learning) phase is very short which is an important factor in network monitoring applications in dynamic environments and under hard resource constraints. Furthermore, it is effective for heavy hitter detection in dynamic environments as highly possible large flows are tracked and directly measured; this set of large flows is indicated by I_k^t. The performance of this algorithm in selecting the most K heaviest flows is measured by $Pe_{TK} := \frac{1}{T_c}\sum_{t=1}^{T_c}\frac{e_t}{k}$ where e_t is the number of errors in selecting K largest flows. Hence, Pe_{TK} denotes the probability of selecting K largest flows using Algorithm 1. Fig. 3 shows Pe_{TK} for the different values of K and T (for a given n) indicating that Algorithm 1 has low probability of error in selecting the heaviest flows.

In MUCB algorithm, the set I_g^t in Algorithm 1 arranges other not-directly measured flows in descending order based on their flow sizes. Consequently, an effective flow aggregation matrix is designed in the following section which is compatible different optimization techniques.

5.1. Efficient-compressed flow aggregation in practice

Now lets suppose the set I_g^t is given where its elements declare the indices of network flow sizes in descending order. This is a great source of information that can be used for designing aggregation matrices which are not only effective for estimating highly fluctuated TMs under hard constraint of limited TCAM sizes but also they are compatible with different under-determined NI techniques. Accordingly, considering the fact that grouping attributes with similar values can improve the estimation accuracy of NI techniques, then an efficient aggregation matrix in a compressed form (i.e. with small number of rows m) can be designed to aggregate $n - K$ flows using m TCAM entries. For this purpose, the following exponential aggregation algorithm is proposed (Algorithm 2) where more measurement resources (TCAM entries) are allocated to the earlier indices in the set I_g^t, indicating larger flows. Then, a large number of smaller flows with more stable behaviors are aggregated using smaller number of TCAM entries. Parameters ρ and δ control the number of flows aggregated per entries which are the inputs of the algorithm defined by the user, properly. As we show in Section 6, this aggregation technique is effective not only for estimating highly fluctuated TMs and heavy hitter detection but also for the estimation of the sub-population of small size flows which is of particular importance in network security applications such as Distributed

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

Algorithm 2 Exponential Aggregation Technique (EAT).

Input: Aggregation parameters ρ and δ.
Output: Aggregation Matrix A_g^t.
Initialization: Set $i_c = 0$ and $A_g^t = 0_{m\times n}$.
for i = m **to** 1 **do**
- $r = \left\lceil \rho(n-K)\frac{1}{\delta^i} \right\rceil + 1$
for j = 1 **to** r **do**
- $A_g^t(i, I_g^t(i_c + j)) = 1$
end for
- $i_c = i_c + r$
end for

Denial of Service (DDOS) attack detection. The estimation accuracy of iSTAMP framework, in general, and this exponential aggregation technique, in specific, can be significantly improved if we incorporate SNMP link counts into our formulation and use Eq. (3).

6. *i*STAMP performance evaluation

In this section, the effectiveness of iSTAMP network measurement and inference framework is justified. For this purpose, three real networks (Table 2) with different configurations are considered. The Abilene [29] and GEANT [30] networks are well known networks with publicly available traffic traces. The routing matrices $H_{Abilene}$ and H_{Geant} are (30×144) and (74×529) binary matrices with full row-ranks. To consider a more dynamic environment with highly varying flows, we processed the publicly available data center packet traces in [23] and randomly chose a subset of flows, which are non-zero over the period of experiment and they rapidly fluctuate over time/space. We also assumed a Fat-Tree topology where the number of servers communicating among different racks (n_{Ext}) vary and ECMP routing is used between aggregation and core levels. In this case, flows are defined as traffic between servers in different racks.

Each configuration is defined by choosing an appropriate aggregation matrix and by selecting different values for T and K and α. These parameters can be defined by the designer of the system based on the prior knowledge/experience and through a trial and error process. Also, Algorithm 1 is used to directly measure the highest possible large flows. In addition, we evaluate the performance of three different aggregation techniques: 1) Block Aggregation Technique (BAT); 2) Random Aggregation Technique (RAT), and 3) Exponential Aggregation Technique (EAT).

In BAT, m entries of TCAM are equally shared among $(n-K)$ flows where each block contains a set of $\left\lceil \frac{n-K}{m} \right\rceil$ consecutive aggregable flows which models the mapping of flows into TCAM entries in many practical cases. RAT is an abstract model that we used, through a Monte-Carlo process, to show the performance of our measurement framework where the routing/aggregation structure is not under our control. In EAT, our aggregation method in Algorithm 2 is used to efficiently allocate TCAM entries to flows, and to produce a well compressed form of linear aggregated measurements. We also use different network inference methods in Eqs. (2) and (3) to verify the compatibility of our framework with a variety of optimization techniques which can be efficiently solved by existing low-complexity algorithms [14]. In the following, using Eq. (2) for flow estimation based on aggregated statistics and without Side Information (SI), is denoted by "w/o SI". We use "w/ SI" to denote the case where SNMP link loads are incorporated into our formulation using Eq. (3).

Eq. (12) defines the metrics used in our performance evaluation. *NMSE* is a metric that we used to measure the accuracy of ODF estimation for each configuration. To justify the effectiveness of our framework for HH detection, we first set the threshold θ as a fraction of the link capacity C_L, and then, the average probability of detection (P_{HH}^d) and the average probability of false alarm (P_{HH}^{fa}) are computed as in Eq. (12). Given n, these metrics are measured based on two quantities by varying T and K. First, T/n represents the ratio of the number of TCAM entries, used for both aggregation and direct measurement in our framework, and the number of flows. Note that low T/n asserts the hard resource constraint regime where a large number of flows (n) has been represented by T measurements including both per-flow and aggregated flow counts. Thus, T/n also indicates the Flow Compression Ratio (FCR). Second, K/T represents the ratio of the number of TCAM entries used for direct measurement and the total number of available TCAM entries.

Furthermore, we measure the effectiveness of our measurement framework in Sub-Population (SP) flow size estimation, which is of particular importance in some security applications such as DDoS detection. In Eq. (12), SP^t denotes the subset of I_g^t where $X(I_g^t) < \theta_l$ and SP_l^t denotes the lth disjoint subset of SP^t where the sum of flows in SP_l^t can be large (note that $\theta_l < \theta$). Also, π_0^t and π_1^t are prior probabilities which are computed experimentally. In our Monte-Carlo evaluation process, $\left|SP_l^t\right|$ is randomly chosen from interval $[\alpha_l(n-K) \quad \alpha_u(n-K)]$ where parameters α_l and α_u are selected properly to cover a wide range of SP sizes. Due to space limitation, we also summarize our results by averaging our performance criteria in Eq. (12) over different choices of T and K (e.g. $NMSE_{Avg}$).

$$NMSE = \frac{1}{T_c}\sum_{t=1}^{T_c}\frac{\left\|X^t - \hat{X}^t\right\|_2}{\|X^t\|_2}$$

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

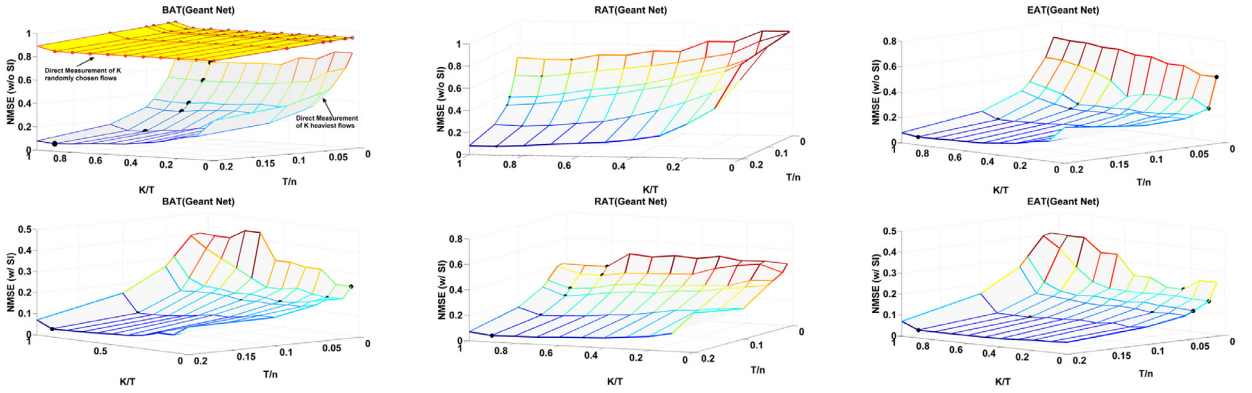

Fig. 4. *NMSE* for Geant Net. with different configurations using Eqs. (2) and (3) with (p, q, λ)=(2,1,0.01), $\alpha = 1$, $\rho = 1$ and $\delta = 1.975$. Note that, using only SNMP link-loads $NMSE = 0.6153$ is provided.

$$P_{HH}^{d} = \frac{1}{T_c}\sum_{t=1}^{T_c} Pr\left(\hat{X}^t \geq \theta \,|\, X^t \geq \theta\right)$$

$$P_{HH}^{fa} = \frac{1}{T_c}\sum_{t=1}^{T_c} Pr\left(\hat{X}^t \geq \theta \,|\, X^t < \theta\right)$$

$$SPE = \frac{1}{T_c}\sum_{t=1}^{T_c}\frac{1}{|SP^t|}\sum_{l\in SP^t}\frac{\left|\left\|\hat{X}_{SP_l^t}^t\right\|_2 - \left\|X_{SP_l^t}^t\right\|_2\right|}{\left\|X_{SP_l^t}^t\right\|_2}$$

$$P_{SP}^{d} = \frac{1}{T_c}\sum_{t=1}^{T_c}\pi_1^t Pr\left(\left\|\hat{X}_{SP^t}^t\right\|_2 \geq \theta \,|\, \left\|X_{SP^t}^t\right\|_2 \geq \theta\right) + \frac{1}{T_c}\sum_{t=1}^{T_c}\pi_0^t Pr\left(\left\|\hat{X}_{SP^t}^t\right\|_2 \leq \theta \,|\, \left\|X_{SP^t}^t\right\|_2 \leq \theta\right)$$

$$P_{SP}^{fa} = \frac{1}{T_c}\sum_{t=1}^{T_c} Pr\left(\left\|\hat{X}_{SP^t}^t\right\|_2 \geq \theta \,|\, \left\|X_{SP^t}^t\right\|_2 < \theta\right) \tag{12}$$

6.1. iSTAMP performance on ISP networks

Fig. 4 shows the *NMSE* for different configurations on Geant network where CS inference techniques Eqs. (2) and (3) are used for flow size estimation. The first plot in Fig. 4 justifies our intuition that direct measurement of K heaviest flows in iSTAMP can significantly improve the estimation accuracy compared to the case where K flows are randomly selected for direct per-flow measurement and its performance is evaluated through the Monte-Carlo simulation. In the absence of side information in Fig. 4, a better accuracy is achieved at higher K/T's. However, in general, the optimal choice for the number of direct flow measurements K which yields to the minimum *NMSE* (indicated by solid black points) is obtained at $K/T < 1$. In other words, the best accuracy is often achieved by allocating at least part of the TCAM entries for direct measurement and the remaining for flow aggregation. Among these, EAT can enhance the estimation accuracy, for example, $NMSE_{Avg}^{BAT} = 0.38$ and $NMSE_{Avg}^{EAT} = 0.29$; this improvement is significantly higher at low T/n's.

As it is shown in Fig. 4, incorporating side information (i.e. SNMP link counts) and using Eq. (3) can remarkably improve the precision of the estimation using BAT, RAT and EAT methods. In this case, EAT still can improve the accuracy ($NMSE_{Avg}^{BAT} = 0.19$ and $NMSE_{Avg}^{EAT} = 0.16$); however, since aggregated measurements are interpreted as side information in Eq. (3), the difference between BAT and EAT is low.

In addition, although increasing T/n improves the accuracy, the slope of the improvement is slow after some near-optimal FCR (e.g. $T/n \approx 0.1$). On the other hand, Fig. 5 shows the normalized error in sub-population size estimation, indicated by *SPE* in Eq. (12), in the presence of SNMP side informations. This figure represents that EAT is also effective in SP size estimation and, in general, a good performance is achieved when part of TCAM entries are used for measuring K flows, directly. Also, note that, the slope of the improvement is slow after some optimal FCR ($= T/n$). Furthermore, Fig. 6 represents the effectiveness of iSTAMP for reliably detecting both heavy hitters and heavy sub-populations. Among these, a high probability of detection and low probability of false alarms are achieved for HH detection where threshold θ is set to 10% of link capacity

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

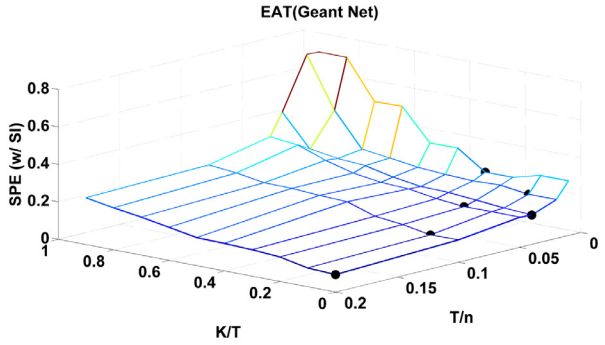

Fig. 5. *SPE* for Geant Network with different configurations where $\alpha = 1$, $(\alpha_l, \alpha_u) = (0.01,0.1)$, $\rho = 1$ and $\delta = 1.975$.

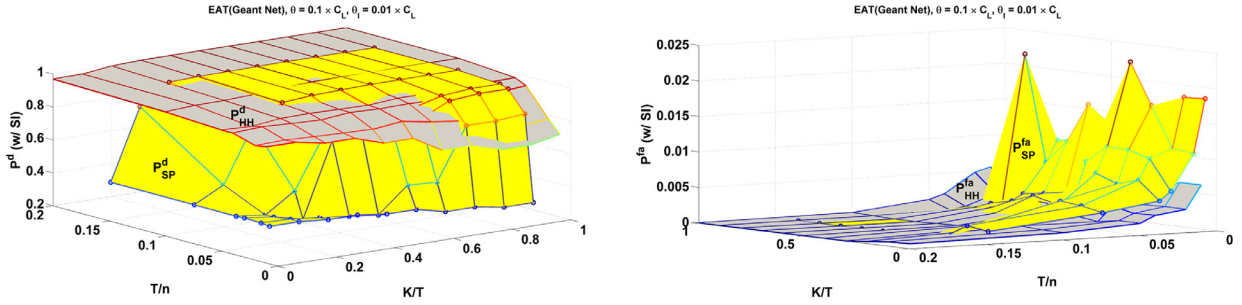

Fig. 6. P^{pd} and P^{fa} in Geant Network with different configurations for both HH and SP detection where $(\alpha_l, \alpha_u) = (0.01,0.1)$, $\rho = 1$ and $\delta = 1.975$.

Table 3
The average performance for Abilene network where $(p, q, \lambda) = (\infty, 0,0)$ in Eqs. (2) and Eq. (3), $\alpha = 1$, $\theta = 0.01C_L$, $\theta_l = 0.0025C_L$, $(\alpha_l, \alpha_u) = (0.01,0.2)$, $C_L =$ 1Mbps, $\rho = 1$ and $\delta = 5$. Note that, using only SNMP link-loads $NMSE = 0.5953$ is provided.

Aggregation technique	$NMSE_{Avg}$ (w/o SI)	$\bar{P}^{d}_{HH}$ (w/o SI)	$\bar{P}^{fa}_{HH}$ (w/o SI)
BAT	0.3857	0.7762	0.0696
RAT	0.5239	0.4508	0.0900
EAT	0.3602	0.7282	0.0496
Aggregation technique	$NMSE_{Avg}$ (w/ SI)	$\bar{P}^{d}_{HH}$ (w/ SI)	$\bar{P}^{fa}_{HH}$ (w/ SI)
BAT	0.1663	0.8473	0.0396
RAT	0.2578	0.7361	0.0571
EAT	0.1762	0.8173	0.0341
Aggregation technique	SPE_{Avg} (w/o SI)	$\bar{P}^{d}_{SP}$ (w/o SI)	$\bar{P}^{fa}_{SP}$ (w/o SI)
BAT	0.4943	0.7960	0.1360
Aggregation technique	SPE_{Avg} (w/ SI)	$\bar{P}^{d}_{SP}$ (w/ SI)	$\bar{P}^{fa}_{SP}$ (w/ SI)
BAT	0.1765	0.9019	0.0086

$C_L = 10$Mpbs. This figure also shows that the detection capability for heavy sub-populations (with $\theta_l = 0.01C_L$) is still high for many choices of T and K. Under very hard resource constraints, although the sub-population detection performance is lower, it is still acceptable and it improves, rapidly.

Therefore, iSTAMP achieves high estimation accuracy using a small fraction of available TCAM entries at low FCRs and using different aggregation techniques. Hence, it can produce well-compressed aggregated flow measurements and remarkably decrease the required storage capacity and reduce the communication overhead between control and data planes. iSTAMP is also effective for highly reliable heavy hitter and heavy SP detection which is of particular importance in different networking applications. Table 3 summarizes the average performances for Abilene network. These results re-emphasize the flexibility and capability of iSTAMP for different network monitoring applications under hard resource constraints. Note that, for both Geant and Abilene networks, T_c is set to the duration of the data in Table 2 (i.e. small β in Eq. (11)) which is the worst case scenario. Better accuracy can be achieved with smaller T_c's.

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

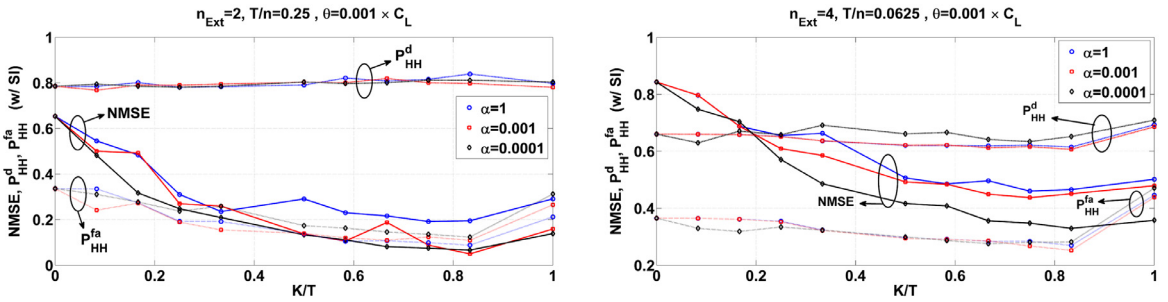

Fig. 7. *NMSE*, P^{d}_{HH} and P^{fa}_{HH} for Data Center with different configurations where α varies and $\beta = 0$ and (p, q, λ)=(2,0,0) in Eq. (3).

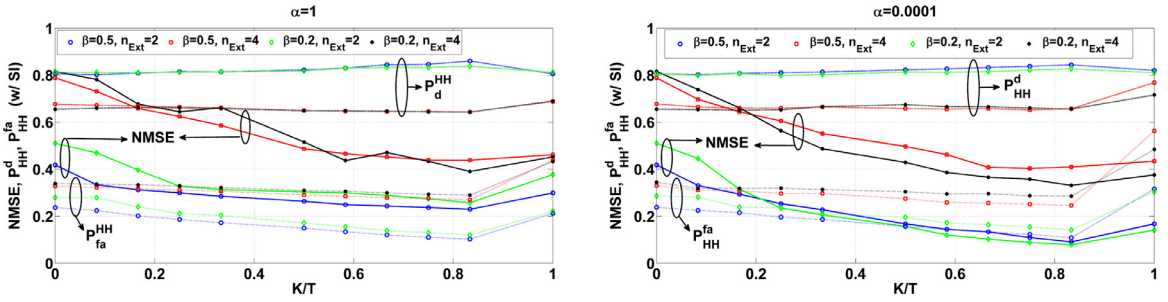

Fig. 8. *NMSE*, P^{d}_{HH} and P^{fa}_{HH} for Data Center with different configurations where (p, q, λ)=(2,0,0) in Eq. (3) and α, β (i.e. T_c) and n_{Ext} vary.

6.2. iSTAMP performance in Data Center (DC) environment

In this section, the performance of our measurement framework under very hard resource constraint of TCAM sizes (e.g. example $T/n = 0.0625$) in a data center environment with highly fluctuating flows is evaluated. We consider both the absence/presence of SNMP side information, that is, using Eqs. (2) and (3) where (p, q, λ)=(2,0,0). Here, the aggregation matrix is a given block aggregation matrix and routing matrix H is computed for the fat-tree topology assuming the ECMP routing between aggregation and core level [23]. As it was explained in Section 5, in dynamic environments we need to explore different flows more frequently and, also, repeatedly learn the behavior of flows. For this purpose, proper values for α and β in our Algorithm 1 are chosen; note that smaller β leads to larger T_c. Fig. 7 shows the performance of iSTAMP framework for our two main applications in a highly dynamic Data Center environment. In fact, it shows that decreasing α can improve the accuracy of flow size estimation for different values of T/n's. These results re-emphasize that there is an optimal K for direct measurements; therefore, our hybrid technique for providing aggregated and direct flow measurements is effective. Fig. 8 shows the effect of different choices of T_c using different values of β in Eq. (11) when n_{Ext} vary. Here, $(\beta = 0.5, n_{Ext} = 2)$, $(\beta = 0.5, n_{Ext} = 4)$, $(\beta = 0.2, n_{Ext} = 2)$ and $(\beta = 0.2, n_{Ext} = 4)$ correspond to $T_c = 186(\mathrm{s})$, $T_c = 260(\mathrm{s})$, $T_c = 1051(\mathrm{s})$ and $T_c = 1085(\mathrm{s})$, respectively. Also, $n_{Ext} = 2$ and $n_{Ext} = 4$ yields the number of flows $n = 48$ and $n = 192$. It is clear that, even in a very hard resource constraint ($T/n << 1$), our measurement framework can obtain a good estimation accuracy and detecting capability by choosing optimum K/T. In practice, the designer of the system must appropriately set the controlling parameters T, K, α and β based on the history of the system and the required performances through a trial and error process.

6.3. Hierarchical heavy hitter identification using iSTAMP

Hierarchical heavy hitter are the longest prefixes that includes a high volume of traffic without considering any HHH descendants in each interval [7]. A novel SDN based approach for HHH detection in [7], proposes an Online Tree Search (OTS) algorithm for identifying HHHs. To compare the performance of this algorithm with iSTAMP, we run an offline algorithm over the real traffic trace of Geant network for all leaf nodes in the trie. Similarly, we identify all possible HHHs by running the same offline algorithm in each interval over TM estimates computed by iSTAMP framework. In addition, we run OTS algorithm in [7] over the whole duration of the traffic. Table 4 shows the precision and recall (as defined in [7]) for these two algorithms indicating that, compared to OTS, iSTAMP can also provide a good performance for detecting HHHs (i.e. high precision and recall) with less or very close number of TCAM entries. Note that iSTAMP is a general framework that can provide the complete visibility of all flows over long time-horizon and it is useful in different applications.

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

Table 4
HHH detection using OTS and iSTAMP algorithms.

Algorithm	Precision (%)	Recall (%)	# of TCAM used
OTS Alg. $\theta = 0.05C_L$	87.17	80.22	137
iSTAMP $\theta = 0.05C_L$	60.56	64.43	112
OTS Alg. $\theta = 0.10C_L$	89.50	68.19	105
iSTAMP $\theta = 0.10C_L$	74.82	75.74	112

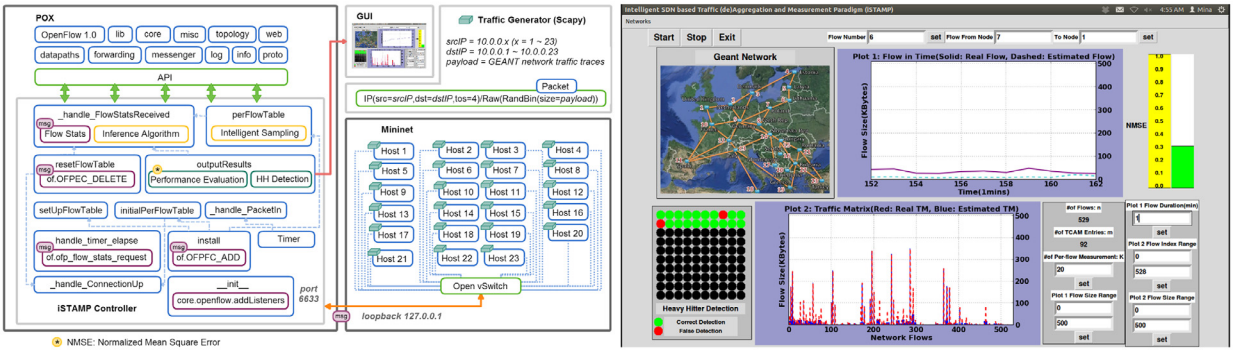

Fig. 9. System architecture diagram of iSTAMP in mininet and a snapshot of centralized iSTAMP GUI ($m = 92$, $K = 20$).

Table 5
iSTAMP performance where $m = 92$, and K varies ($T = m + K$).

Perf. metric	$K = 0$	$K = 20$	$K = 30$	$K = 50$	$K = 75$	$K = 100$
NMSE	0.8785	0.3003	0.1889	0.1226	0.0763	0.0575
P_{HH}^{d}	0.1179	0.5973	0.8790	0.9814	0.9852	0.9936
P_{HH}^{fa}	0.8821	0.4027	0.1210	0.0186	0.0148	0.0064

7. *i*STAMP: a mininet Implementation

To show the feasibility and effectiveness of iSTAMP, we have implemented a prototype of iSTAMP in mininet which is a network testbed for developing OpenFlow and SDN experiments. Fig. 9 shows the system architecture diagram of iSTAMP where we create a centralized configuration of Geant Network in mininet including 23 nodes (i.e. hosts), with distinct IP addresses, connected to a single SDN enabled switch. These nodes send IP packets into the network (using scapy package in python) to reproduce the real Geant traffic, according to Table 2. The controller, which has been implemented using POX, runs the iSTAMP algorithm. The main modules of iSTAMP controller are 1) the modules for interacting with the switch to reset, configure and install the routing and measurement rules in flow-tables; 2) the modules for periodically collecting flow statistics at predetermined time-intervals, and 3) the modules for applying network inference technique, and evaluating and demonstrating the results.

In this implementation, without loss of generality and for demonstrating purposes, a flow is defined as 3-tuple ⟨src IP, dst IP, protocol⟩ and each TCAM (i.e. flow-table) entry is constructed to provide desired per-flow and/or aggregated flow measurements. Incoming packets are matched against TCAM rules and forwarded to a specific output port, determined by the action field for each rule. At the time of this implementation, mininet did not support OpenFlow 1.2 (or higher); consequently, packet classification performed based on the longest IP-prefix matching. Therefore, 92 destination-based routing rules (i.e. $m = 92$) are configured on the Open vSwitch. That is, $A_g^t = H$ where H is the routing matrix. Accordingly, the iSTAMP framework collected both aggregated and per-flow statistics (at one minute intervals) and solved the network inference Eq. (2) using CVXPY package where $A^t = \begin{bmatrix} A_g^t \\ A_k^t \end{bmatrix}$, and A_g^t denotes routing matrix H excluding A_k^t, which indicates flows that must be directly measured. Accordingly, the performance of iSTAMP is computed by evaluating *NMSE* and Pe_{TK}.

Table 5 shows the performance of centralized implementation of iSTAMP framework in mininet indicating that estimation accuracy can be improved by increasing K. This centralized iSTAMP has been implemented using more than 2000 lines of codes in Python; further results and details of the implementation are available in [20]. These results, along with an implementation of centralized iSTAMP on the SDN switch HP2920, have been successfully demonstrated in the 21th GENI Engineering Conference (GEC 21). Fig. 9 shows a snapshot of the GUI designed for this demonstration.

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

8. Conclusion

We introduced iSTAMP, an intelligent framework for measuring and inferring network flow sizes in software defined networks. Our philosophy in developing iSTAMP was that under hard resource constraint of measurement resources, efficient and scalable network inference techniques must be effectively utilized to provide the fine-grained estimate of network flows. Using the SDN flexibility, we showed that: 1) a global view of available measurement resources in a network can be achieved which can be used to effectively allocate TCAM/flow-table entries for both aggregated and per-flow measurements that are required for network inference, and 2) the behavior of network flows in dynamic environments can be learned to measure the most informative flows at the right time and place. Therefore, based on the application, it is essential to use appropriate network inference and learning algorithms. We showed the effectiveness of iSTAMP for different applications, through extensive evaluations using real traffic traces and network topologies. By implementing a prototype of iSTAMP, we also demonstrated the feasibility and effectiveness of iSTAMP for fine-grained TM estimation.

Acknowledgments

This work is supported by National Science Foundation CNS-1321115 grant and HP Labs Innovation Research Award. Also, we would like to thank Shiliang Tang for implementing our integer optimization problem in CPLEX.

Appendix A

Considering the original UCB1 algorithm [28], it can be shown that:

Lemma 1. *For all $n > 1$, if policy UCB1 is run on n machines having reward distributions $\{P_i\}_{i=1}^{n} \in [0, U]$ with mean $\{\nu_i\}_{i=1}^{n}$ where the average reward of jth arm is computed as $\tilde{\bar{x}}_j = \alpha \frac{1}{t_j(t_c)} \sum_{t=1}^{t_c} x_j(t)$ with $0 < \alpha < \frac{2}{U\sqrt{3}}$. Then, the regret after t_c number of plays is bounded as:*

$$R(t_c) \leq 8\alpha^2 U^2 \sum_{i:\tilde{\nu}_i < \tilde{\nu}^*} \left(\frac{\ln(t_c)}{\tilde{\Delta}_i} \right) + (1 + \psi(U, \alpha)) \left(\sum_{j=1}^{n} \tilde{\Delta}_j \right) \tag{13}$$

where $\tilde{\Delta}_i := \tilde{\nu}^* - \tilde{\nu}_i$, $\tilde{\nu}^* = \max_{1 \leq j \leq n} \tilde{\nu}_j$ and $\psi(U, \alpha)$ is a bounded function of U and α (please refer to [20] for complete proof).

Proof of Theorem 1. Let Q_K^* denotes the set of indices of the K arms with highest mean and $Q_{n-K} = \{1, \ldots, n\} \setminus Q_K^*$. Then, arm $k \in Q_{n-K}$ is selected if the index of kth arm in MUCB policy is greater than all best arms Q_K^*; that is, $\{I_k > I_i\}_{i \in Q_K^*}$. Therefore, the total regret of playing MUCB strategy and sampling K arms with highest indices is not worse than playing K simultaneous MAB plays where, in the jth play, the arm with the highest index in the $(j-1)$th play has been removed from the available set of arms. The set of arms for jth play A_j^S $(j = 1, \ldots, K)$ are defined as:

$$\text{1th MAB play (j=1)} : A_1^S = \{1, \ldots, n\}$$

$$\text{2th MAB play (j=2)} : A_2^S = \left\{ \{1, \ldots, n\} \setminus \bigcup_{l=1}^{j-1} S_l \right\} \quad \text{where}$$

$$S_1 = \{\text{the index of the arm with the highest index in} (j-1)\text{st play}\}$$

$$\vdots$$

$$K\text{th MAB play (j=K)} : A_K^S = \left\{ \{1, \ldots, n\} \setminus \bigcup_{l=1}^{j-1} S_l \right\} \quad \text{where}$$

$$S_{K-1} = \{\text{the index of the arm with the highest index in} (K-1)\text{st play}\}$$

According to the Lemma 1, the expected regret for the jth play is bounded as:

$$8\alpha^2 U^2 \sum_{i:\tilde{\nu}_i < \tilde{\nu}_j^*} \left(\frac{\ln(t_c^j)}{\tilde{\Delta}_i^j} \right) + (1 + \psi(U, \alpha)) \left(\sum_{i \in A_j^S} \tilde{\Delta}_i^j \right)$$

where t_c^j is the overall number of plays for the jth play. Considering the facts that $t_c^j \leq t_c$ and $\tilde{\Delta}_i^{j+1} \leq \tilde{\Delta}_i^j$; therefore, the total expected regret can be bounded as:

$$\tilde{R}_K(t_c) \leq \sum_{j=1}^{K} \left[8\alpha^2 U^2 \sum_{i:\tilde{\nu}_i < \tilde{\nu}_j^*} \left(\frac{\ln(t_c)}{\tilde{\Delta}_i^j} \right) + (1 + \psi(U, \alpha)) \Sigma_j \right]$$

□

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

References

[1] Medhi D. Network routing: algorithms, protocols, and architectures. Morgan Kaufmann; 2007.
[2] Mardani M, Giannakis G. Estimating traffic and anomaly maps via network tomography. IEEE/ACM Trans Networking 2016;24.
[3] Soule A, Salamatian K, Taft N. Combining filtering and statistical methods for anomaly detection. ACM SIGCOMM; 2005.
[4] Kumar A, Xu J. Sketch guided sampling - using on-line estimates of flow size for adaptive data collection. IEEE INFOCOM; 2006.
[5] Medina A, Taft N, Salamatian K, Bhattacharyya S, Diot C. Traffic matrix estimation:existing techniques and new directions. ACM SIGCOMM; 2002.
[6] Zhao Q, Ge Z, Wang J, Xu J. Robust traffic matrix estimation with imperfect information: making use of multiple data sources. ACM SIGMETRICS; 2006.
[7] Jose L, Yu M, Rexford J. Online measurement of large traffic aggregates on commodity switches. ACM Hot-ICE; 2011.
[8] Yu M, Jose L, Miao R. Software defined traffic measurement with opensketch. ACM NSDI; 2013.
[9] Ligatti J, Kuhn J, Gage C. A packet-classification algorithm for arbitrary bitmask rules, with automatic time-space tradeoffs. IEEE ICCCN; 2010.
[10] Malboubi M, Wang L, Chuah C-N, Sharma P. Intelligent sdn based traffic (de)aggregation and measurement paradigm (*i*stamp). IEEE INFOCOM; 2014.
[11] Gong Y, Wang X, Malboubi M, Wang S, Xu S, Chuah C-N. Towards accurate online traffic matrix estimation in software-defined networks. ACM SOSR; 2015.
[12] Malboubi M, Gong Y, Xiong W, Chuah C-N, Sharma P. Software defined network inference with passive/active evolutionary-optimal probing (sniper). IEEE ICCCN; 2015.
[13] Liu C, Malboubi M, Chuah C-N. Openmeasure: adaptive flow measurement and inference with online learning in sdn. In: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 9th IEEE Global Internet Symposium; 2016.
[14] Eldar YC, Kutyniok G. Compressed sensing: theory and applications. Cambridge University Press; 2012.
[15] M R, Zhang Y, Willinger W, Qiu L. Spatio-temporal compressive sensing and internet traffic matrices. IEEE/ACM Trans Networking 2012;20:662–76.
[16] Yuan L, Chuah C-N, Mohapatra P. Progme: towards programmable network measurement. IEEE/ACM Trans Networking 2011;19(1).
[17] Moshref M, Yu M, Govindan R, Vahdat A. Dream: dynamic resource allocation for software-defined measurement. ACM SIGCOMM; 2014.
[18] Zhang Y. An adaptive flow counting method for anomaly detection in sdn. ACM CoNEXT; 2013.
[19] Tootoonchian A., Ghobadi M., Ganjali Y. Opentm: traffic matrix estimator for openflow networks2010;.
[20] Malboubi, M., Peng, S.-M., Sharma, P., Chuah, C.-N. A Learning-based Measurement Framework for Traffic Matrix Inference in Software Defined Networks, Technical report, at: https://ucdavis.box.com/s/tx0hk5u5z7h10x7ar43drgin.
[21] Malboubi M, Vu C, Chuah C-N, Sharma P. Decentralizing network inference problems with multiple-description fusion estimation (mdfe). IEEE INFOCOM, April; 2013.
[22] Benson T, Anand A, Akella A, Zhang M. Microte: fine grained traffic engineering for data centers. ACM CoNEXT; 2011.
[23] Benson DMT, Akella A. Network traffic characteristics of data centers in the wild. ACM IMC; 2010. Data Center Data is Available At: http://pages.cs.wisc.edu/~tbenson/IMC10_Data.html.
[24] Sarvotham S, Baron D, Baraniuk R. Measurements vs. bits: compressed sensing meets information theory. Allerton conference on communication, control, and computing, September; 2006.
[25] Papagiannaki K, Taft N, Bhattacharyya S, Thiran P, Salamatian K, Diot C. A pragmatic definition of elephants in internet backbone traffic. ACM SIGCOMM; 2002.
[26] Elad M. Optimized projections for compressed sensing. IEEE Trans Signal Process 2007;55(12):5695–702.
[27] Abolghasemi V, Ferdowsi S, Makkiabadi B, Sanei S. On optimization of the measurement matrix for compressive sensing. IEEE EURASIP; 2010.
[28] Auer P, Cesa-Bianchi N, Fischer P. Finite-time analysis of the multiarmed bandit problem. Mach Learn 2002;47(2).
[29] Abilene traffic. www.cs.utexas.edu/~yzhang/research/AbileneTM.
[30] Geant network. http://totem.info.ucl.ac.be/dataset.html.

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020

Mehdi Malboubi received his PhD in Electrical Engineering from the University of California, Davis, in 2015. His main research interests include networking, communications, signal/image processing, machine learning/intelligence, and data analysis/mining. His current researches include cloud computing, software defined networking, internet of things, and 5th generation of mobile/wireless networks.

Shu Ming Peng is currently a Senior Product Development Engineer at Verizon. Her passion lies in computer networking, microservices and distributed cloud computing. She completed her MS degree at the University of California at Davis in March 2015. Her major is Electrical and Computer Engineering.

Puneet Sharma is Head of Networked Systems Group at Hewlett Packard Labs. He received his PhD in Computer Science from the University of Southern California and his B.Tech. from Indian Institute of Technology (IIT) Delhi. He has published 60+ research and has been granted 30+ US patents. He is an IEEE Fellow and a Distinguished Scientist of ACM.

Chen-Nee Chuah is a Professor in Electrical and Computer Engineering at the University of California, Davis. She received her Ph.D. in Electrical Engineering and Computer Sciences from the University of California, Berkeley. She is a Fellow of the IEEE and an ACM Distinguished Scientist. Her research interests include Internet measurements, network management, security detection, digital healthcare, and intelligent transportation systems.

Please cite this article as: M. Malboubi et al., A learning-based measurement framework for traffic matrix inference in software defined networks, Computers and Electrical Engineering (2017), https://doi.org/10.1016/j.compeleceng.2017.11.020