Reformulating the Monitor Placement Problem: Optimal Network-Wide Sampling

Gion Reto Cantieni, Gianluca Iannaccone, Chadi Barakat, Christophe Diot, Patrick Thiran *

ABSTRACT

Confronted with the generalization of monitoring in operational networks, researchers have proposed placement algorithms that can help ISPs deploy their monitoring infrastructure in a cost effective way, while maximizing the benefits of their infrastructure. However, a static placement of monitors cannot be optimal given the short-term and long-term variations in traffic due to re-routing events, anomalies and the normal network evolution. In addition, most ISPs already deploy router embedded monitoring functionalities. Despite some limitations (inherent to being part of a router), these monitoring tools give greater visibility on the network traffic but raise the question on how to configure a network-wide monitoring infrastructure that may contain hundreds of monitoring points.

We reformulate the placement problem as follows. Given a network where all links can be monitored, which monitors should be activated and which sampling rate should be set on these monitors in order to achieve a given measurement task with high accuracy and low resource consumption? We provide a formulation of the problem, an optimal algorithm to solve it, and we study its performance on a real backbone network.

1. INTRODUCTION

Network operators perform traffic measurements as part of their day by day network management activities that include traffic engineering, anomaly detection, accounting and capacity planning. There exist several ways to perform traffic measurements. Some involve router support (e.g., SNMP counters, Netflow [4]), while others require additional equipment to be installed in the network to perform passive or active measurements.

The various solutions present a trade-off between the accuracy of the measurement and the amount of computing resources they require. SNMP counters, for example, represent a very low cost solution (in terms of router processing resources) but give the operator only a rough idea on the traffic that is traversing the network. The aggregate counters are of little use to operators interested in users' perceived performance [16] or in estimating network traffic demands [20]. At the other extreme, passive monitoring equipment, that captures every packet on a link, allows extremely accurate measurements, but scales very poorly for large networks, given the high unit cost for deployment and maintenance. In addition, space and power constraints may prevent operators from deploying this equipment where it would be the most useful, while, once deployed, routing or network changes may render it useless.

The solution that many network operators have adopted is then to use Netflow [4] or similar solutions [23, 19]. NetFlow allows a ubiquitous deployment of traffic monitors and provides a detailed enough view of the traffic streams. Netflow is today widely used by Internet Service Providers (ISPs) and several tools exist to process, analyze and visualize NetFlow data [9, 24]. However, enabling NetFlow can have an impact on packet forwarding performance. To address this problem, router vendors have introduced versions of Netflow that sample the incoming packets and update the flow information only with sampled packets.

Network operators then face two options: (i) enable Netflow on all routers but using very low sampling rates to minimize potential network impact, or, (ii), enable Netflow on a chosen set of routers where the sampling rates are set depending on the measurement task and the target accuracy.

Currently the first option is the one followed by ISPs because no automated method exists for the second. This work aims at filling this gap.

The contributions of the paper are threefold. First, we define a general framework to approach the problem of sampling traffic data in large IP networks. Our framework allows to combine and solve in one step the selection of traffic monitors and the setting of the sampling rates for each monitor. We show how this framework can be applied to a general class of measurement tasks. Second, we provide an optimal algorithm to solve the sampling and placement problem. We validate the algorithm using network data collected from the GEANT's backbone network [10]. Finally, we discuss how to deploy our solution in a real backbone network and introduce additional methods that allow to adapt the sampling rates to changes in the traffic due to time-of-day effects, failures or anomalies.

*G.Cantieni and P.Thiran are with EPFL, Switzerland. G.Iannaccone is with Intel Research, Cambridge UK. C.Barakat is with INRIA Sophia Antipolis, France. C. Diot is with Thomson Research, Paris, France.

The paper is organized as follows. In Section 2, we describe the objectives and challenges of this work. Section 3 presents related work. Sections 4 and 5 formally define the problem and our approach to solve it. Section 6 evaluates the performance of our algorithm in the GEANT backbone network. Section 7 describes how to deploy our solution in a real backbone and how to deal with traffic changes when the measurement task runs in a continuous fashion. Section 8 concludes the paper.

2. OBJECTIVES AND CHALLENGES

The nature of the measurement task has a clear impact on the choice of traffic monitors to be used. For example, in order to estimate the traffic demands, a network operator would ideally monitor all ingress links in the network or at least all peering links [8]. However, when the network operators want to focus on an individual network prefix or Autonomous System (AS), they may require a very different layout of monitors in the network.

Very often, network operators do not have prior knowledge of the measurement tasks the monitoring infrastructure will have to perform. This is particularly true with security applications. For example, a specific network prefix that is "below the radars" for traffic engineering purposes may play an important role in the early detection of anomalies.

Furthermore, network traffic demands are subject to short term variations due to failures and other anomalous events as well as longer term variations due to the addition of new customers, peering links, Points of Presence, etc. These changes quickly make a static placement of traffic monitors perform sub-optimally.

For these reasons, Internet Service Providers (ISP) prefer widespread monitoring infrastructures that provide visibility over the entire network. Netflow [4] is a perfect example of such monitoring system. Embedded into the routers, it maintains a list of flow records that describe the traffic forwarded by the router. The flow records are then exported to a collector for analysis and storage. Netflow has become today the de-facto standard of flow monitoring solutions: it is the most widely deployed and various router manufactures support compatible monitoring applications [19]. A standardization effort is also in progress at the IETF [15].

ISPs configure Netflow on all routers to the same "safe" sampling rate — router vendors usually recommend a value of 1/1000 packets — while little attention is paid to the accuracy of the results. Low sampling rates reduce the stress on the routers but introduce large errors in the measurement.

Clearly, this approach leads to an inefficient use of the resources both in the routers and in the collector, and to a less than desirable measurement accuracy (we will quantify this in Section 6.3). Our objective is to find a method that, given a measurement task and a target measurement accuracy, has the following properties:

- It selects the monitors that need to participate in the measurement. This reduces the processing overhead on the collector side where it is desirable that a large proportion of the records received, processed and stored are actually relevant to the measurement task.
- It provides a set of sampling rates for the active monitors to guarantee an optimal use of the resources while making sure that the sampling rates are low enough to not be a concern for the operator.
- The resource consumption, as measured by the total number of sampled and processed packets, is minimal and adapts to the changes in the traffic. The objective is to maintain the overall resource consumption stable while keeping the accuracy close to target.
- The method requires a minimal amount of configuration from the network operator and is robust to incorrect input parameters.

In addition, our method should support a general class of monitoring applications and be easily adapted to new measurement tasks. We choose one example of such tasks to illustrate the contributions of this paper. We find the sampling rates to estimate the amount of traffic flowing among a set of origin-destination pairs selected by the network operator. In our terminology, origin and destination could refer to any combinations of end-hosts, network prefixes, Autonomous Systems, etc. We have chosen this task because it helps in illustrating our contribution and it is, at the same time, a canonical measurement task for classical traffic engineering, security and accounting applications.

3. RELATED WORK

Identifying the strategic locations for traffic monitors is a hard problem that has attracted significant interest in the literature. Several solutions have been proposed for different contexts. For example, in [17], the authors focus on the placement of measurement devices for active monitoring (more specifically for the construction of distance maps). Others have addressed the placement problem in an active monitoring infrastructure to measure delays and detect link failures [1, 13, 21].

In the passive monitoring domain, Suh et al. [28] address the problem of placing monitors and set their sampling rates in order to maximize the fraction of IP flows being sampled. They propose a two phase approach where they first find the links that should be monitored and then run a second optimization algorithm to set the sampling rates. Their approach bears some similarities with our work, but the analysis is limited to a generic monitoring goal (maximize the overall sampled traffic) and only considers a static placement of monitors. Their formulation leads to a set of heuristics that find near-optimal solutions. Our approach, instead, allows to indicate whether a solution corresponds to the global optimum.

Many researchers have also explored ways to improve NetFlow to automate some of the features and help network operators configure the routers. Estan et al. [7] propose a set of techniques to let a router running NetFlow adapt the sampling rate in order to keep a fixed resource consumption. The adaptation is a local and independent decision of the router, and is not tied to any measurement objective. Our work is complementary to [7] in the sense that it provides a global sampling strategy for a specific monitoring goal. The individual routers could then apply local decisions in order to minimize their memory usage.

There is a large body of literature that addresses the problem of inversion of traffic properties from sampled traffic [6, 5, 12]. Duffield et al. [6] show that periodic and random sampling provide roughly the same result on high speed links. Random sampling can thus be used in the mathematical analysis for its appealing features.

Traffic matrix estimation techniques (e.g., [20, 29, 26]) address a measurement task similar to the one we are considering as example. However, the focus in those works is the inference of the traffic matrix from partial information (e.g., link loads). Indeed, they often use sampled Netflow data to validate their methods.

4. PROBLEM FORMULATION

In this section we formalize the placement problem. Our approach to solve it will be described in the next section.

We represent the network by a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ corresponds to the set of nodes and $\mathcal{E}$ is the set of edges. The traffic load on edge $e \in \mathcal{E}$ is denoted by U_e. The routing of each origin-destination pair (OD pair) is specified by the routing matrix R, whose entries $r_{i,j} = 1$ if the OD pair i traverses edge j and 0 otherwise.

The measurement tasks are defined over a set $\mathcal{F} = \{1, 2, \ldots, F\}$ of OD pairs. We indicate the subset of links traversed by the OD pairs in $\mathcal{F}$ by $\mathcal{L} \subseteq \mathcal{E}$. The optimization framework we propose in this paper is general and can be applied to any definition of node (e.g., end-host, network prefix, Autonomous System, etc.).

The quality of a measurement for an OD pair $k \in \mathcal{F}$ is computed via a utility function $M : \mathbb{R}_+ \to \mathbb{R}_+$, whose argument ρ_k is the *effective* sampling rate of OD pair $k \in \mathcal{F}$, defined as the probability that a packet of the kth OD pair is sampled at least once by at least one monitor deployed in the network. Note that this definition assumes that we have means to discern whether the same packet is sampled at multiple locations in the network. In Section 6, we will address this aspect of the monitoring infrastructure in more detail. Assuming that the packets are sampled in an i.i.d. (independent and identically distributed) fashion at each monitor, with p_i denoting the packet sampling probability of the monitor deployed on link i, and that the sampling processes of different monitors are statistically independent, we have that

$$\rho_k = 1 - \prod_{i \in \mathcal{L}} (1 - p_i)^{r_{k,i}}. \tag{1}$$

Clearly, the larger the effective sampling rate, the more information it brings. However, the marginal rate of information is usually smaller for large values of ρ_k than for small values. These two observations lead us to reasonably assume that $M(\rho_k)$ is an increasing and strictly concave function of ρ_k.

Our objective is to choose the vector of sampling rates $\boldsymbol{p} = (p_i)_{i \in \mathcal{L}}$ that maximizes

$$\sum_{k \in \mathcal{F}} M\left(\rho_k\left(\boldsymbol{p}\right)\right), \tag{2}$$

under the constraints

$$p_i \geq 0 \quad \text{for all } i \in \mathcal{L} \tag{3}$$

$$p_i \leq \alpha_i \quad \text{for all } i \in \mathcal{L} \tag{4}$$

$$\sum_{i \in \mathcal{L}} p_i U_i \leq \theta, \tag{5}$$

where θ is the *capacity* of the system, defined as the maximum total number of packets that can be sampled in the entire network, and α_i represents the maximum sampling rate that can be applied to the individual link i.

All constraints are linear and therefore define a convex solution space Ω defined by $\{\boldsymbol{p} \mid \sum_{i \in \mathcal{L}} p_i U_i \leq \theta, \quad 0 \leq p_i \leq \alpha_i \quad \forall i \in \mathcal{L}\}$. As the utility function M is strictly concave, the optimization problem, given by (2), (4) and (5), has a unique maximizer, that we denote $\boldsymbol{p}^*$ (see e.g., [3, Chapter 2]).

Note that an alternative objective could be to maximize the minimum of the utilities, i.e., $min_{k \in \mathcal{F}} M(\rho_k(\boldsymbol{p}))$. The two formulations have their advantages and limitations. Maximizing the sum of utilities gives us more flexibility in setting the sampling rates. Indeed, we can compensate the poor accuracy of one OD pair increasing the accuracy of another. We will discuss the implications of this in greater detail in Section 6. On the other hand, maximizing the minimum utility may lead to increase significantly the sampling rate on those links that carry small OD pairs. Furthermore, the minimum of the utilities is not a differentiable function over the whole parameter space and this may impact the convergence of the algorithm (see Section 5.4). We leave the study of alternative objective functions for future work.

5. METHOD

In this section we first reformulate the optimization problem using Lagrange multipliers. Second, we introduce and comment on the assumptions that simplify the computation of the optimal solution. Third, we discuss the choice of a suitable utility function $M(\cdot)$. Last, we briefly present the algorithm used to solve the optimization problem and discuss its advantages and pitfalls.

5.1 Our approach

The most common approach to solve a constrained optimization problem, such as the one in (2), (3), (4) and (5), is to define the corresponding Lagrangian:

$$\begin{aligned} L(\boldsymbol{p}, \lambda, \boldsymbol{\mu}, \boldsymbol{\nu}) &= \sum_{k \in \mathcal{F}} M\left(\rho_k(\boldsymbol{p})\right) - \lambda \left(\sum_{i \in \mathcal{L}} p_i U_i - \theta \right) \\ &\quad - \sum_{i \in \mathcal{L}} \mu_i (p_i - \alpha_i) + \sum_{i \in \mathcal{L}} \nu_i p_i, \end{aligned} \tag{6}$$

where $(\lambda, \boldsymbol{\mu}, \boldsymbol{\nu}) = (\lambda, \mu_i, \nu_i)_{i \in \mathcal{L}}$ is the set of Lagrange multipliers. Each Lagrange multiplier enforces the satisfaction of one of the constraints (3), (4) and (5) with an equality sign. A constraint met with an equality sign is called an *active constraint*. For example, if $p_i = \alpha_i$, then the ith constraint (4) is active, whereas if $p_i < \alpha_i$, it is inactive.

In order to find the unique maximizer $\boldsymbol{p}^*$, we can solve the system of equations provided by the Karush-Kuhn-Tucker (KKT) conditions [11, Chapter 5]. As the solution space is a convex hull and the objective function is concave, the KKT conditions are sufficient for optimality (see e.g., [3, Chapter 5.5]). The difficulty in using the KKT conditions to solve the problem given by (2), (3), (4) and (5) is that it requires to know the set of active and inactive constraints in advance, which is not possible. Therefore, we need to rely on an iterative procedure, such as a gradient projection method to explore the solution space. Our approach to find the optimal solution $\boldsymbol{p}^*$ is given in more details further below.

5.2 Assumptions

Let us recall the assumptions made so far: the utility function M is strictly increasing and concave; each monitor sampling process is i.i.d. and is statistically independent from the sampling process of the other monitors.

From a practical perspective, we expect to obtain sampling rates that are in the order of 0.01 and lower. Moreover, we rarely expect to have more than one or two monitors observing the traffic of the same OD pair. This allows us to approximate the effective sampling rate (1) by

$$\rho_k = \sum_{i \in \mathcal{L}} r_{k,i} p_i. \tag{7}$$

This set of assumptions allows to make the optimization problem more tractable and, as we show in Section 6.2, have a minor impact on the performance of our method.

Next, we force the constraint (5) to be met with an equality sign. This assumption is pretty straightforward to make, as there is no practical interest not to use all the provided resources. Hence (5) becomes

$$\sum_{i \in \mathcal{L}} p_i U_i = \theta. \tag{8}$$

5.3 Choosing the utility function

The choice of the utility function $M(\rho_k)$ is dictated by the following conditions. First, the utility function has to quantify the information provided by the measurement for each OD-flow in $\mathcal{F}$. Second, the function has to comply with the requirements set by the optimization framework: as mentioned earlier, it is strictly increasing and concave. Without loss of generality, we assume that $M(0) = 0$, i.e. that the utility is zero if no packet is sampled at all. Third, the function M must be easy to use in our optimization algorithm. This requires the function to be twice continuously differentiable. We next derive a possible function which combines the above specified properties.

A first straightforward choice for the utility function M is to use the relative error between the actual metric of interest that we want to measure (in the present case, the flow size) and its value estimated from the sampled packets. Let S_k be the actual size of the kth OD pair (number of packets of the kth OD pair) in a given time interval, and X_k be the number of sampled packets from this OD pair in the same time interval. Because of the assumptions that the sampling processes at different monitors are independent and that packets are sampled at most once (Section 5.2), the distribution of X_k conditionally to S_k is binomial with parameters (S_k, ρ_k). In this paper, we consider the squared relative error (SRE) between the estimated size X_k/ρ_k and the actual size of the flow S_k, i.e.

$$\text{SRE} = \left(\frac{X_k/\rho_k - S_k}{S_k} \right)^2, \tag{9}$$

whose expected value is

$$\mathbb{E}[\text{SRE}](\rho_k) = \int_0^\infty \frac{1-\rho_k}{\rho_k s} d\mathbb{P}(S_k \leq s) = \mathbb{E}\left[\frac{1}{S_k}\right]\left(\frac{1}{\rho_k} - 1\right).$$

Define $A(\rho_k) = 1 - \mathbb{E}[\text{SRE}](\rho_k)$, that we call the mean squared relative accuracy. A possible candidate for $M(\rho_k)$ would be to take it equal to $A(\rho_k)$, since it is a strictly increasing, concave function of ρ_k. However, this function is not yet adequate to be used as utility function M. The problem is that the function $A(\rho_k)$ is not defined at the origin. This is required, as we expect to have zero utility for zero sampling. To fix this problem, we divide the interval $[0, 1]$ in two intervals $[0, x_0]$ and $[x_0, 1]$. We take $M(\rho_k) = 1 - \mathbb{E}[\text{SRE}](\rho_k)$ if $x_0 \leq \rho_k \leq 1$ and $M(\rho_k) = A'(\rho_k)$ for $0 \leq \rho_k < x_0$, where A' is defined in such a way that it is strictly concave and increasing and twice differentiable over $[0, x_0]$, with $A'(0) = 0$, $A'(x_0) = A(x_0)$, $\partial A'(x_0)/\partial \rho_k = \partial A(x_0)/\partial \rho_k$ and $\partial^2 A'(x_0)/\partial \rho_k^2 = \partial^2 A(x_0)/\partial \rho_k^2$. A function that complies with these requirements is the quadratic expansion of $A(\rho_k)$ at x_0, that reads

$$A'(\rho_k) = A(x_0) + (\rho_k - x_0)\frac{\partial A}{\partial x}(x_0) + \frac{(\rho_k - x_0)^2}{2}\frac{\partial^2 A}{\partial x^2}(x_0),$$

where x_0 is given by the relation $A'(0) = 0$, that is,

$$A(x_0) - x_0 \frac{\partial A}{\partial x}(x_0) + \frac{x_0^2}{2}\frac{\partial^2 A}{\partial x^2}(x_0) = 0.$$

To summarize, we define the utility function M as

$$M(\rho_k) = \begin{cases} A'(\rho_k) & \text{if } 0 \leq \rho_k \leq x_0 \\ A(\rho_k) & \text{if } x_0 \leq \rho_k \leq 1. \end{cases} \tag{10}$$

We plot the function $M(\rho_k)$ for $\mathbb{E}[1/S_k] = 0.002$ and for $\mathbb{E}[1/S_k] = 0.0002$ in Figure 1. We emphasize that this manipulation on the function has the sole purpose of making it suitable for our optimization framework. The interval $0 \leq \rho_k < x_0$, on which we use the quadratic expansion $A'(\rho_k)$, is of little practical interest because it corresponds to low values of utility. Hence, we expect that this necessary manipulation does not affect much the optimization results. We will validate this observation in Section 6.

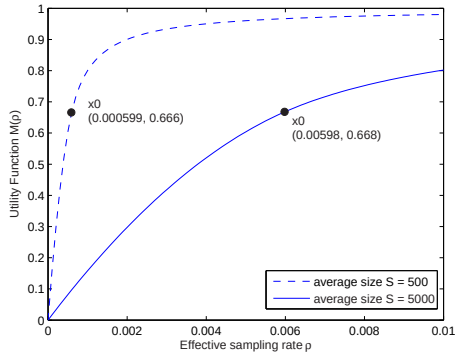

Figure 1: The utility function M with respect to the effective sampling rate ρ_k. The utility function is defined over the interval [0,1], such as given in (10).

5.4 The algorithm

We solve our optimization problem by use of the gradient projection method for constrained optimization (Refer to [11, Chapter 5]). At each iteration step n, this method consists in projecting first the gradient of the objective function onto the subspace spanned by the active constraints (that is, the set of points $\boldsymbol{p}$ that satisfy the active constraints). This projected gradient gives the search direction $\boldsymbol{s}(n)$, along which the current feasible solution $\boldsymbol{p}(n)$ is moved until either the objective function is maximized along this line or an inactive constraint is hit. In the latter case, this inactive constraint has to be activated and incorporated into the next computation of the gradient projection. In the former case, the maximization of the objective function along $\boldsymbol{s}(n)$

reduces to a one-dimensional search. We choose Newton's method (refer to [3], chapter 9.5) to perform this search. Newton's method shows fast convergence but requires the objective function to be twice continuously differentiable. Once the new solution $\boldsymbol{p}(n+1)$ that maximizes the objective along $\boldsymbol{s}(n)$ is found, a new search direction $\boldsymbol{s}(n+1)$ has to be computed. The new search direction will be orthogonal to the previous one. The successive search directions $\boldsymbol{s}(n), \boldsymbol{s}(n+1), \ldots$ form therefore a zigzag path in the subspace spanned by the active constraints, which may result in a poor convergence depending on the shape of this subspace. A better approach is to add, with some weighting factor, the previous search direction to the new one. This weighting factor is usually chosen according the Polak-Ribiere rule. More details can be found e.g. in [18].

We start our search with a feasible solution $\boldsymbol{p}(0)$ arbitrarily chosen on the plane defined by the active constraints (5). We then follow the search direction $\boldsymbol{s}(0)$, that coincides with the projection of the gradient of the objective function (2) on the subspace spanned by the active constraints, until we hit the constraint $p_1 \geq 0$. This forces us to activate this constraint, i.e. to set $p_1 = 0$, and to recompute next the search direction $\boldsymbol{s}(1)$. With Newton's method we finally find $\boldsymbol{p}(2)$.

We mentioned earlier that in order to find the optimal solution, we eventually have to know the set of active and inactive constraints of our optimization problem. That is, we have to know which monitors i are going to be used (whose corresponding constraint $p_i > 0$ is inactive) and which ones not (whose corresponding constraint $p_i = 0$ is active). This combinatorial explosion is unavoidable as the monitor placement problem is NP-hard. Probing each possible combination is not feasible in practice, hence we use the above introduced gradient method to find iteratively the optimum. However, we do not have any guarantee that the gradient projection method always converges to the optimum. It might well happen that the projection of the gradient $\boldsymbol{s}$ converges to a zero vector at some point, say $\boldsymbol{p}'$, which forces us to stop the search, even though $\boldsymbol{p}'$ is not optimal. This means that the algorithm has not generated the optimal set of active and inactive constraints, and thus that some monitors are turned on/off on suboptimal links. Given the feasible solution $\boldsymbol{p}'$ we have obtained, we apply the Karush-Kuhn-Tucker (KKT) conditions [11, Chapter 5] that determine whether this solution is optimal or not. To do so, we have to compute the Lagrange multipliers defined in (6). Having some of them negative indicates that the algorithm cannot converge to the optimum with the current set of active constraints. It may be possible to continue the search by removing (making inactive) some of the currently active constraints.

A strategy is to remove (make inactive) the subset of active constraints associated with negative Lagrange multipliers (a similar strategy is given in [11, Chapter 5]). After this operation, we recompute the gradient projection on the subspace spanned by the *new* set of active constraints and proceed with the search. We continue until we either reach a point that satisfies the KKT conditions and therefore is the optimal solution by KKT's theorem, or we abort the search because the number of iterations (a new iteration starts each time a new search direction has to be computed) exceeds a threshold we set on the maximum number of iterations. In our experiments described in Section 6, this threshold is set to 2000 to keep the execution time of the algorithm in the order of few seconds. We observe that in 98.6 percent of cases the optimum is found in less than 2000 iterations.

The performance of the gradient projection method can also be measured by the number of times we end up in the situation where we have to remove (make inactive) the active constraints associated with negative Lagrange multipliers. To provide a number, we conducted 200 independent executions of the algorithm, each time with a different set of input parameters (different OD pair sizes, different link loads, different capacity θ). The input parameters are given from the data set discussed in Section 6. Even though we cannot have a formal upper bound on the number of times we have to remove some active constraints, we observe an average of 1.64 situations per execution of the algorithm, where we have to remove (make inactive) the constraints associated with negative Lagrange multipliers, with a standard deviation of 1.17. These low numbers, that are specific to the monitoring problem we solve, justify our choice of the gradient projection method to solve efficiently our optimization problem.

6. EVALUATION

We study the performance of our method by defining and simulating a measurement task on GEANT, the European Research network [10]. Our goal is to verify the following properties of the solution we obtain with the method described in Section 5: (*i*) the sampling rates are low and validate the approximation in (7); (*ii*) the solution results in a fair allocation of resources to each OD pair; (*iii*) the solution is clearly superior to other solutions that could be derived without running any optimization algorithm.

6.1 Data

We use sampled Netflow data collected on all the 23 routers of the GEANT network. The links have varying speeds from OC-3 (155 Mbps) to OC-48 (2.5 Gbps). We also collect in a continuous fashion BGP and ISIS updates and use them to build GEANT routing matrix.

Every GEANT router has NetFlow-compatible monitoring capabilities [19] enabled with a sampling rate of 1/1000. The packets are classified by the 5-tuple (source and destination IP address, source and destination port number and protocol number) and the flow records are exported every minute by the routers. Each record contains the following information (in addition to the 5-tuple) that is relevant to this study: (*i*) *Flow start and end time*. The start time is the timestamp of the first sampled packet of the flow. The end time is the timestamp of the last packet of the flow. Flow termination is triggered either by a FIN packet or by an idle timeout (set to 30 seconds). (*ii*) *Sampled packets and bytes*. The total number of sampled packets in the flow and their cumulative size in bytes. (*iii*) *Source and Destination Autonomous System (AS)*. The AS numbers to which the source and destination IP addresses belong. (*iv*) *Input and output interface*. The index of the router interface on which the flow was received and sent.

Before using NetFlow data, we need to perform some additional post-processing of the records. First, we aggregate all flow records (exported by the routers every minute for all active flows) in 5 minutes bins according to their start time. We choose 5 minutes as our measurement interval to reduce the impact of synchronization issues that could have arisen when collecting flow data from different routers. Then, we

adjust the sampled packet and byte count by multiplying them by the inverse of the sampling rate (1000 in our case).

Henceforth, we consider the post-processed NetFlow data to represent the *actual traffic* traversing the GEANT network. Our experiments then consist in simulating a sampling process with the rates set by the optimal algorithm and comparing the results with the post-processed data. Although our validation does not depend on a perfect reconstruction of the traffic dynamics, the sampled Netflow data present a potential bias against small flows that can affect the relative contribution of each OD pair of interest. Unfortunately, this is the only type of data available today for research and we are not aware of any other public dataset that contains full unsampled information for a network of the size of GEANT.

6.2 Results

For the evaluation we choose to estimate the traffic sent by JANET (UK Research Network, AS number 786) to each individual GEANT PoP through the UK PoP.

This task gives us a set $\mathcal{F}$ of 20 OD pairs. The OD pairs traverse 22 of the 72 unidirectional links of GEANT. We use flow data from November 22nd, 2004 where we associate to each flow record the egress PoP, computed from the destination IP address using the technique presented in [8]. The first two columns of Table 1 present a summary of the OD pairs and their sizes in packets/sec. The results in this section are computed over a single measurement interval.

This task shows that the method operates on OD pairs where origins (i.e., the JANET AS) and destinations (i.e, all the GEANT PoPs) are of a different nature. In addition, the OD pair sizes cover the entire spectrum: JANET to Netherlands (NL) consist of more than 30,000 packets/sec while JANET to Luxembourg is made of a mere 20 packets/sec. As it will become clear later, this is one of the main strengths of the method, i.e., the ability to track indifferently small and large OD pairs.

The remaining columns in Table 1 represent the sampling rates that the optimal solution provides. For each link i, we indicate the sampling rate p_i and the OD pairs that traverse (and are sampled on) that link. All links that are not present in the table have $p_i = 0$, i.e., those monitors do not need to participate in the measurement task. Finally, the last two rows of the table show the load in packets/sec on the links and the relative contribution to the total capacity. In this experiment we have chosen a value of $\theta = 100,000$, that is, at most $100,000$ packets can be sampled in each 5 minutes measurement interval in the entire network. We also set $\alpha_i = 1$ for all links, i.e., we do not define an upper limit for the sampling rates. Thus, we assume the operator has no prior knowledge of the network traffic.

We can immediately observe that the sampling rates are extremely low on most links even if no upper limit was set. Only two links (FR-LU and CZ-SK) need a sampling rates somewhat higher (around 0.9%), but they are lightly loaded links needed to accurately estimate the two smallest OD pairs (JANET-SK and JANET-LU). Furthermore, the low sampling rates we obtain and the fact that each OD pair is sampled in at most two links validate the assumptions on the effective sampling rate made in Section 5.2.

The last two columns in Table 1 show the value of the utility function (10) and the accuracy of the measurement. We define the accuracy of an OD pair size as 1 minus the absolute relative error: $1- \mid x/\rho - s \mid /s$, where s is the actual size of the OD pair, x is the sampled size and ρ is the effective sampling rate as in (7).

We use the accuracy instead of the utility M (10) to validate the impact of two assumptions we made in the definition of M: (i) the quadratic expansion to force M to zero when the sampling rate is zero (see Section 5.3); (ii) the approximation (7) for the effective sampling rate ρ, that may result in overestimating the size of certain OD pairs.

We run 20 sampling experiments on the flow records and compute the average accuracy over the 20 runs. Each sampling experiment consists in simulating a random sampling process on the flow records observed on link i using the sampling rate p_i in Table 1. The values in those two columns demonstrate that the method achieves good fairness among OD pairs. Although the algorithm maximizes the sum of the utilities, the results indicate that the individual utilities are well balanced. Moreover, the accuracy of the measurement is extremely good being on average above 0.89 for any OD pair.

6.3 Comparison with other solutions

The last aspect we want to address is how the optimal solution compares to naïve solutions. Clearly, if any solution performs well enough, then there is no reason to add the complexity of the optimization algorithm we propose.

The first naïve solution would consist in monitoring only the JANET access link to GEANT. This solution has the advantage that every sampled packet would belong to one of the OD pairs of interest. However, in order to track a small OD pair (e.g., JANET to Luxembourg) with a similar accuracy to the one we obtain in Table 1, the network operator would be forced to sample the link at a rate of about 1%, i.e., the effective sampling rate for JANET-LU. Given the high load on that link, this would require the capacity θ to be 70% higher than the one needed by our method to give the same measurement accuracy[1].

Furthermore, apart from being a suboptimal solution, monitoring the access link may not be feasible in all scenarios given that, in corporate networks for example, the edge routers (i.e., the router that is connected to the ISP) is often directly owned and managed by the ISP that provides network connectivity. These routers, usually called CPE (Customer Premise Equipment), can only be accessed by the ISP network operators. Therefore, in order to keep our method general and applicable to a wide range of network scenarios, we do not include the access links in the set of possible links to monitor. Even with this additional constraint, Table 1 shows that our method performs extremely well.

An alternative to the monitoring of the access link is to monitor all links that connect the UK PoP to the other PoPs in GEANT. This solution allows to "balance" the sampling rates over six links, instead of just one, and gives more freedom to reduce the resource consumption.

In order to compare this solution with the optimum we run our algorithm and restrict the choice of available monitors to just the six UK links. Figure 2 shows the comparison in terms of the accuracy over a wide range of values of the capacity θ. With respect to the optimum, this simple solu-

[1]Adding up the values in the second column of Table 1 we obtain 57,933 packets per second. At a sampling rate of 1%, this results in 173,798 sampled packets on average over a 5 minutes interval.

OD pair	pkt/s	p_5 UK-FR	p_7 UK-SE	p_8 UK-NL	p_9 UK-NY	p_{17} SE-PL	p_{30} UK-PT	p_{31} IT-IL	p_{33} FR-BE	p_2 FR-LU	p_{28} CZ-SK	Utility	Avg Accuracy
JANET-NL	30123	-	-	0.0016	-	-	-	-	-	-	-	0.9999	0.993
JANET-NY	9387	-	-	-	0.0002	-	-	-	-	-	-	0.9982	0.965
JANET-DE	4300	-	-	0.0016	-	-	-	-	-	-	-	0.9995	0.982
JANET-SE	4080	-	0.0003	-	-	-	-	-	-	-	-	0.9973	0.960
JANET-CH	4033	0.0013	-	-	-	-	-	-	-	-	-	0.9994	0.979
JANET-FR	1723	0.0013	-	-	-	-	-	-	-	-	-	0.9985	0.969
JANET-PL	1400	-	0.0003	-	-	0.0003	-	-	-	-	-	0.9960	0.950
JANET-GR	1080	-	-	0.0016	-	-	-	-	-	-	-	0.9981	0.964
JANET-ES	1003	0.0013	-	-	-	-	-	-	-	-	-	0.9974	0.959
JANET-SI	913	-	-	0.0016	-	-	-	-	-	-	-	0.9977	0.961
JANET-IT	873	0.0013	-	-	-	-	-	-	-	-	-	0.9971	0.956
JANET-AT	790	0.0013	-	-	-	-	-	-	-	-	-	0.9968	0.954
JANET-CZ	590	-	-	0.0016	-	-	-	-	-	-	-	0.9965	0.952
JANET-BE	490	0.0013	-	-	-	-	-	-	0.0002	-	-	0.9955	0.946
JANET-PT	463	-	-	-	-	-	0.0011	-	-	-	-	0.9935	0.937
JANET-HU	377	-	-	0.0016	-	-	-	-	-	-	-	0.9945	0.940
JANET-HR	237	-	-	0.0016	-	-	-	-	-	-	-	0.9912	0.924
JANET-IL	87	0.0013	-	-	-	-	-	0.0018	-	-	-	0.9877	0.910
JANET-SK	43	-	-	0.0016	-	-	-	-	-	-	0.0092	0.9929	0.932
JANET-LU	20	0.0013	-	-	-	-	-	-	-	0.0090	-	0.9840	0.897
Link Loads (pkt/s)		63603	51833	57756	37286	23680	19950	15213	11173	6133	2600		
Contribution to θ		24.5%	5.1%	26.9%	2.1%	2.1%	6.8%	8.3%	0.7%	16.5%	7.1%		

Table 1: Optimal sampling rates p_i for each link i. For each OD pair, it indicates the size (pkts/sec), the link(s) where it is monitored, their respective sampling rates and the average accuracy. All other links have zero sampling rate.

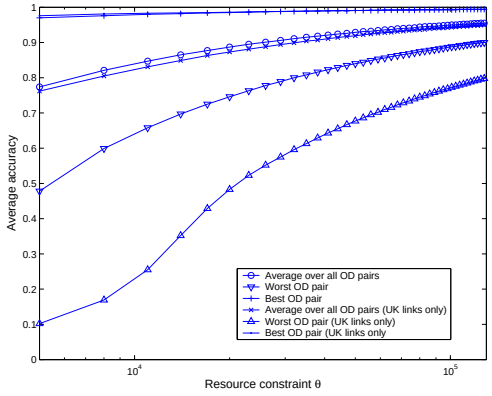

Figure 2: Accuracy of the measurement for two different solutions as a function of θ.

tion has poor performance with respect to small OD pairs. This is expected given that the UK links are heavily loaded and the high sampling rate required to accurately estimate a small OD pair results in a high resource consumption.

In summary, we have seen that both in terms of resource usage and measurement accuracy there is a clear advantage in using our method for setting the sampling rates when compared to other naïve solutions. This advantage comes from the network-wide approach to packet sampling. Indeed, the optimization method finds those links across the entire network, where the small OD pairs "manifest" themselves with a small amount of cross traffic from other large OD pairs. Several studies [2, 8] have shown that this is a general property of current network design.

7. DEPLOYMENT ON A BACKBONE NETWORK

Before deploying our method in a real backbone network, we need to address two additional issues: (i) how to set the input parameters and (ii) how to deal with traffic variations over time.

As described in Section 5, the input parameters consist of the quantities θ, $E[1/S_k]$ and U_i, for all $k \in \mathcal{F}$ and $i \in \mathcal{L}$. In order to obtain these quantities, the network operator needs to have some prior knowledge about the network. In particular, the operator needs to have a rough idea of the sizes, S_k, of the OD pairs that are under study. This is a classical bootstrapping problem and we will present a solution in Section 7.1.

The second deployment issue has to do with the temporal fluctuations of the OD pair sizes and link loads. In the flow data, we observe that these fluctuations may be very large and abrupt. This phenomenon is also more pronounced for small OD pairs and confirms previous measurement studies [27].

It is out of the scope of this paper to investigate the reasons behind these fluctuations. They may be due to normal time-of-day effects, failures in the network, flash crowd events or other anomalies. What is important to observe is that these fluctuations may lead our method to "overshoot" the capacity θ when the link loads increase abruptly, or to miss the target accuracy when an OD pair size decreases. Section 7.2 shows the impact of the fluctuations on the performance of our method, while Section 7.3 introduces a simple heuristic to effectively cope with them.

In the remainder of this section we will refer to flow data collected over a 12 hours period from 8AM to 8PM on November 22nd, 2004.

7.1 Bootstrapping phase

The bootstrapping method we propose is very simple. Given the presence of traffic monitors on every link in the network, an operator can get a rough estimate of the OD pair sizes and link loads by activating all monitors at once. The monitors should be activated at a very low sampling rate, that we call *background sampling rate* ($\tilde{p}$), to guarantee a minimal impact on the performance of the routers and a limited overall resource usage.

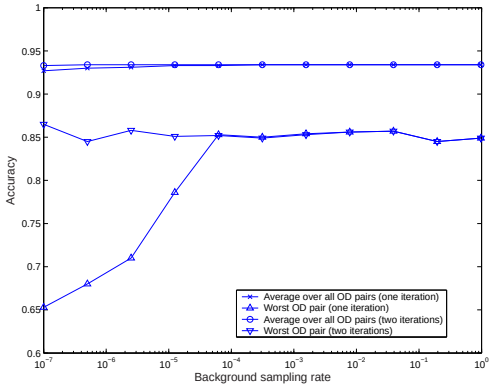

Figure 3: Performance of the bootstrapping algorithm as a function of the background sampling rate.

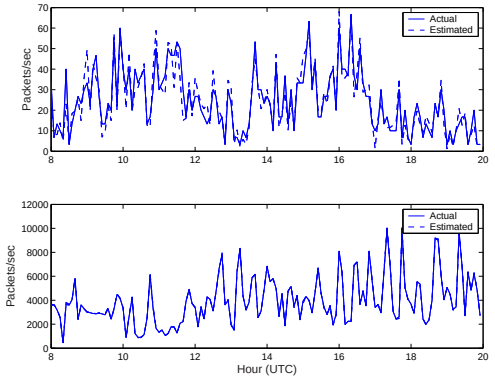

Figure 4: Estimated size of JANET to Luxembourg (top) and JANET to Switzerland (bottom) OD pairs. Sampling rates are fixed over the entire duration of the experiment

We can then use these estimates as input parameters to our method, derive a new set of sampling rates for all the routers (many of which will be set to zero), re-configure all routers, wait one measurement interval and get a new estimate of the input parameters. We iterate this procedure until the set of sampling rates converges and no router needs to change its settings.

This approach, however, raises three questions: (i) how do we estimate the size of the OD pairs? (ii) what is an appropriate background sampling rate? (iii) how many iterations are needed to perform the bootstrapping?

The optimization algorithm needs the values of $\mathbb{E}[1/S_k]$ for all OD pairs of interest. After the first measurement interval, we obtain X_k, the sampled size of each OD pair k. If $X_k > 0$, we replace $\mathbb{E}[1/S_k]$ in (10) with ρ_k/X_k – note that $p_i = \tilde{p}$, for all $i \in \mathcal{L}$. In the case $X_k = 0$, the OD pair has not been sampled at all. This means that the original OD pair size is likely smaller than $1/\tilde{p}$. For our purposes, here, any value below $1/\tilde{p}$ is a good enough estimate of its size. Hence, we replace $\mathbb{E}[1/S_k]$ with $c\tilde{p}$, where c is a constant arbitrarily set to 10 in all our experiments.

Note that finding an accurate estimator for $\mathbb{E}[1/S_k]$ is outside the scope of this paper. In fact, we deliberately choose to use one single sample of the size as the estimator because it is simple and *clearly imperfect*. Our aim is to show that our bootstrapping method is robust to incorrect input parameters and requires no configuration effort.

To answer the last two questions, we run an experiment over the flow data where we vary the value of $\tilde{p}$ from 10^{-7} to 1. We estimate the size of the OD pairs from JANET to all PoPs and the link loads across the entire network. Then, we run the optimization algorithm using this set of input parameters and a value of $\theta = 100,000$ (as in Section 6) to derive a set of sampling rates.

Figure 3 shows the average and worst accuracy over all OD pairs as a function of the background sampling rates. The figure compares the performance after one or two iterations. It is clear that two iterations are sufficient to complete the bootstrapping. Indeed, after two iterations the accuracy of the measurement does not depend anymore on the background sampling rate and thus on the quality of the input parameters. We also observe that a background sampling rates of 10^{-7} is enough if the algorithm is allowed at least two iterations. This demonstrates that the bootstrapping algorithm is very robust against incorrect input parameters.

7.2 Performance over time

We evaluate the performance of our method by estimating the size of all OD pairs originating from JANET over the entire period 8AM to 8PM. As in Section 6, we set a capacity $\theta = 100,000$. We perform the bootstrapping algorithm as described in the previous section at 7.55AM with a sampling rate of 10^{-7}. The sampling rates obtained from the optimization algorithm are then set on all the routers and unchanged for the entire 12 hour period.

Figure 4 compares the estimated and actual size for two OD pairs (JANET to Luxembourg and JANET to Switzerland). The figure shows the result for one run of the sampling process over the 12 hour period. The estimate is extremely accurate and closely follows all fluctuations over time for the JANET to Switzerland OD pair (the two lines overlap completely in the figure). The estimate is somewhat less accurate for JANET to Luxembourg (our worst case) but this is well expected given the small size of this OD pair (it peaks at 70 packets/sec).

In Figure 5 we plot the accuracy of the measurement over 20 experiments. Each experiment consists in one run of the sampling process across the entire 12 hour period. We show the average accuracy over all OD pairs and the accuracy for the worst OD pair over time. The average accuracy is always above 0.95 indicating that we are able to closely track the fluctuations in all OD pairs. Even, the worst OD pair is tracked with an accuracy around 0.90, but in certain measurement intervals, the accuracy drops below 0.85.

A second metric that plays a crucial role in our method is the evolution over time of the measured resource consumption defined as the total number of sampled (and thus processed) packets, $\hat{\theta}(t) = \sum_i p_i U_i(t)$. We are interested in comparing this value with θ^*, the overall capacity. Figure 6 shows the relative difference computed as $(\hat{\theta}(t) - \theta^*)/\theta^*$.

As we can see, the actual resource consumption is always above the constraint and it may exceed the constraint by as much as 120%. This is a consequence of having set the sampling rate early in the morning without taking into account time-of-day effects that lead to an increase in traffic in the later hours of the day. Therefore, the sampling rates are too high for the resource constraint, but result in good

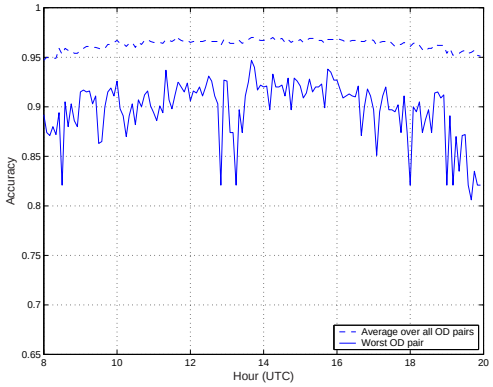

Figure 5: Accuracy over all OD pairs averaged over 20 experiments. Sampling rates are fixed over the entire duration of the experiments

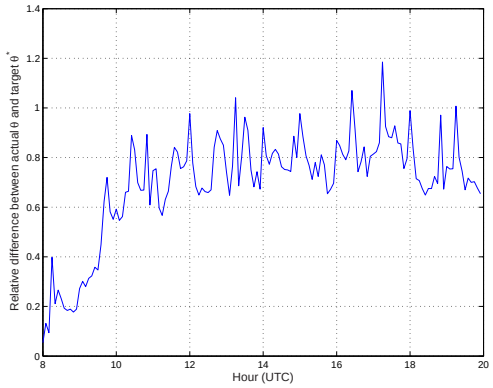

Figure 6: Relative difference between the actual resource consumption $\theta(t)$ and the target constraint θ^*. Sampling rates are fixed over the entire duration of the experiment

accuracy.

7.3 Adapting to traffic fluctuations

Ideally, the network operator would like to define a measurement task and a target accuracy (or a limit on the resource consumption) and then have the monitoring infrastructure that provides the best possible results independently of when the measurement is started or which network events occur during the measurement.

There are three specific events that make a given set of sampling rates suboptimal: (1) The monitored links start to carry a larger amount traffic that is not of interest for the measurement task. This leads to an increase in the resource consumption that does not correspond to an increase in the accuracy (as described in Section 7.2). (2) An OD pair of interest decrease in size requiring a higher sampling rate in order to keep the target accuracy. (3) The OD pairs of interest are routed differently in the network and some monitors are only capturing traffic that is not of interest.

In the following, we present solutions that allow us to cope with each one of these events.

Fluctuations in the link loads. The link loads vary over time for a variety of reasons, including time-of-day effects, anomalies, flash crowds, etc. At peak times the traffic on a link may be orders of magnitude larger than during quiet times. If the sampling rates are fixed and set during quiet times, this may lead to exceed the resource constraint as shown in Figure 6.

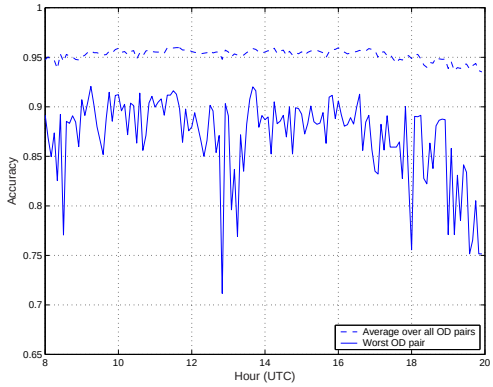

Figure 7: Average accuracy over all OD pairs and worst accuracy. Monitors are reconfigured as soon as the resource usage $\hat{\theta}(t)$ is 10% above or below the capacity θ.

A possible solution is to define two thresholds θ_L and θ_H around the capacity θ and then predict the number of sampled packets in the next interval, $\bar{\theta}(t+1)$. When the prediction exceeds those thresholds, the collector can trigger a recomputation of the optimal sampling rates and (potentially) reconfigure the monitors.

At the end of each measurement interval, the collector can estimate $\bar{\theta}(t+1)$ from the number of sampled packets $\hat{\theta}(t)$ derived from the received flow records. Identifying an accurate predictor of the resource usage is outside the scope of this paper. Our goal is to provide an analysis of the performance of this method with the simplest predictor: we use $\hat{\theta}(t)$ as the predictor of the resource usage in the next interval, $\bar{\theta}(t+1)$.

Figures 7 and 8 show the performance of this method when $\theta_L = 0.9\theta$ and $\theta_H = 1.1\theta$. The results are averaged over 20 experiments. Each experiment is started at 8AM with the bootstrapping taking place at 7.55AM (UTC). As we can see this method succeeds in keeping the resource consumption within the desired bounds even if the bootstrapping is performed during quiet times (early morning). With this adaptive method, the network operator can start the measurement at any time and for any duration without incurring the risk of overloading the monitoring infrastructure.

OD pair fluctuations. The case of fluctuations in an OD pair traffic volume is the most difficult to address. Indeed, if the size of an OD pair decreases, in order to preserve the same measurement accuracy, there exists no alternative but to increase the sampling rates. If the OD pair accounts only for a small portion of the traffic on the monitored link, increasing the sampling rate causes a large amount of traffic not relevant for the measurement task to be collected.

We can see this phenomenon in Figure 7, where the accuracy of the worst OD pair (JANET to Luxembourg in our network scenario) drops significantly around 1PM and 7PM.

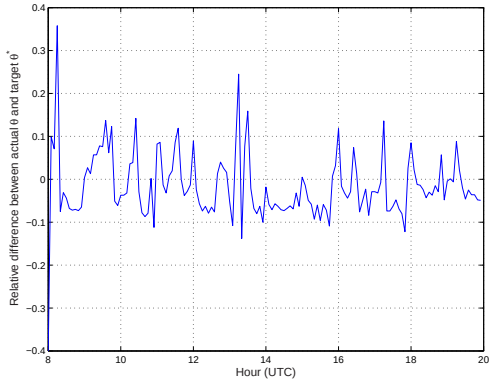

Figure 8: Relative difference between the resource usage $\hat{\theta}(t)$ and the capacity θ. Monitors are reconfigured as soon as $\hat{\theta}(t)$ is 10% above or below θ.

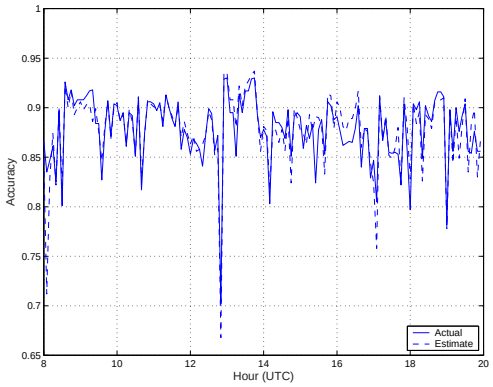

Figure 9: Comparison between estimated and actual accuracy for JANET to Luxembourg OD flow

These drops are consistent with the decrease in size of the JANET to Luxembourg OD pair (see Figure 4), but they do not correspond to equivalent decreases in the link loads.

Therefore, in the cases where the network operator is interested in preserving a target measurement accuracy, the capacity θ (i.e., the total number of sampled packets) needs to adapt to network conditions as well.

We now face a second challenge. Given a target accuracy set by the network operator, how can we estimate if, given the sampled size $X_k(t)$ of each OD pair k, the sampling rate is "good enough" to meet our target in the next intervals?

To this end, we approximate the accuracy as 1 minus the square root of (10). As we did in Section 7.1, we replace $\mathbb{E}[1/S_k]$ by $\rho_k/X_k(t)$, where ρ_k is the effective sampling rate for OD pair k.

Figure 9 compares the estimated accuracy with the actual accuracy from the data for the JANET to Luxembourg OD pair (the smallest OD pair). As we can see, our estimate closely follows the actual accuracy.

We can then use this approximation of the measurement accuracy to trigger a new reconfiguration of the traffic monitors as soon as the worst accuracy drops below the defined target.

We first run the optimization algorithm with the same value of θ used in the previous interval. We then compute

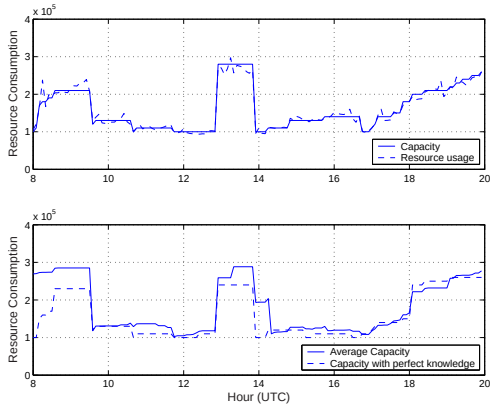

Figure 10: Adapting θ to preserve the target accuracy. Top: actual resource consumption over one experiment. Bottom: average θ over 20 experiments compared with the case of complete knowledge of the traffic.

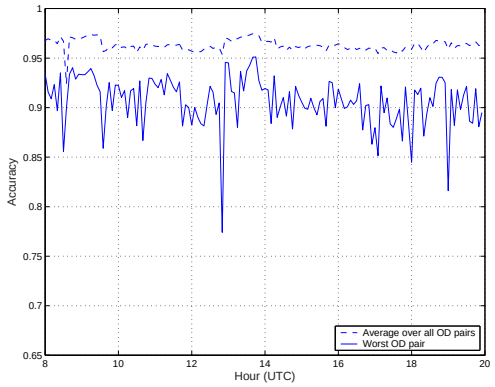

Figure 11: Average accuracy over all OD pairs when θ is adapted to meet target accuracy (0.85)

again the estimate accuracy using the same size estimate $X_k(t)$ but with the new sampling rates. If the estimated accuracy is still below the target, we increase θ by 10% until we reach the target accuracy for all OD pairs.

Furthermore, in order to avoid keeping a θ unnecessary large for long periods, we trigger a new reconfiguration if the estimated worst accuracy is above our target for one consecutive hour.

Figure 10 shows how the capacity θ varies over time when the target accuracy is set to 0.85. The top graph shows, for one experiment, the evolution of the actual resource usage and the capacity. The bottom graph shows the average over 20 experiments and compares it with the capacity that could be set knowing the exact size of each OD pair. Finally, Figure 11 illustrates the performance in terms of accuracy over all OD pairs and worst OD pair.

Around 1PM we observe a significant increase of the capacity θ. This corresponds to a significant drop in accuracy for the worst OD pair. Note that in the previous non-adaptive schemes, the poor accuracy would persist for more than one hour (see Figure 7). Increasing θ allows instead to quickly bring the accuracy above the target. Moreover, θ is decreased as soon as there is no more need of the additional

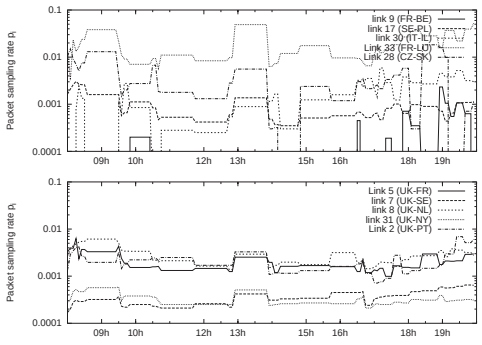

Figure 12: Evolution of the sampling rates over time. Top: lightly loaded links. Bottom: heavily loaded links.

resources.

A last aspect we want to investigate is the evolution of the sampling rates over time. We are interested in having sampling rates that are always low. Moreover, we want to verify that to track effectively OD pairs, a static placement of monitors is not sufficient.

Figure 12 shows the sampling rates for the 10 monitors that are activated during the day. The top graph shows the sampling rates for the lightly loaded links, while the bottom graph shows the sampling rates on the heavily load links.

The number of monitors used to track the 20 OD pairs of interest varies between eight and ten. Often the algorithm includes one or more of the lightly loaded links in order to improve the accuracy or reduce the resource consumption. The sampling rates exceed 1/100 only for the lightly loaded France to Luxembourg link that is instrumental to accurately track the fluctuations of the JANET to Luxembourg OD pair. The heavily loaded links (all links from the UK, a major PoP in GEANT's network) are all set to low sampling rates and experience much less variability over time.

Routing matrix changes. Network failures or BGP updates may modify the path taken by an OD pair under study across the network. When this occurs, some monitors may be recording traffic that is not of interest for the operator. Detecting routing changes is very simple given that routing messages need to propagate to all routers. ISPs often deploy systems to record and process all intra-domain and inter-domain routing messages [14, 25]. The monitoring infrastructure can therefore be aware of routing changes that affect the OD pairs of interest and trigger a new bootstrapping phase. The bootstrap would allow to find where the OD pairs are now entering the network (e.g., for multihomed customers) and get a new estimate of their size. Therefore, after two measurement intervals the monitoring infrastructure will have reconfigured itself with a new set of optimal monitors and sampling rates (see Section 7.1).

7.4 Implementation feasibility

We have shown that it is possible to design a self-configuring, self-adapting monitoring infrastructure in today's backbone networks. Our proposed method can be deployed today in any network running NetFlow (or any other widespread monitoring infrastructure). The network operator has just to select a measurement task (e.g., track any set of OD pairs) and a measurement strategy (i.e., focus on the resource usage or on the accuracy). The proposed mechanisms will then choose the monitors that allow to track those OD pairs with an optimal use of the resources and adapt to changes in the network conditions.

However, we did not look at the time it takes to reconfigure NetFlow on today's routers. All our results assume zero time to run the optimization algorithm (that is instead in the order of a few seconds), and to set the new sampling rates on the routers. Unfortunately, we do not have any figures about the time needed to remotely configure NetFlow, as we did not have any access to the routers in the network but only to the flow records stored at the collector. We leave this task as future work.

Our approach uses only the information stored (or that can be derived) from NetFlow records. This is one of the strengths of our method given that it is generic and can be applied to new NetFlow versions without modifications. Newer versions of NetFlow are constantly increasing the amount of information that is exported in each record. For example, NetFlow v9 records contain BGP next-hop information that would significantly simplify the task of finding the egress PoP for a given network prefix [22]. Moreover, router vendors are reportedly working on ways to support variable sampling rates as in [7]. In this case, our method can be used to define lower bounds on the sampling rates across the entire network, while the individual routers adapt to the local traffic conditions given their available resources.

The method gives the best performance when compared to alternative solutions (Section 6.3) if different network links have different load and carry a diverse set of OD pairs. Indeed, lightly utilized links are candidate for higher sampling rates that allow to compensate for the low sampling rates that we set on heavily loaded links. This is true for the GEANT network and other measurement studies have indicated that this is a common case for other ISP networks as well [2, 8].

Finally, there are a number of ways in which the adaptive methods described in this section can be extended. The choice to increase θ by steps of 10% and to decrease it after an hour is in fact somewhat arbitrary (see Section 7.3). It showed to perform well in practice but one can find smoother ways to adapt θ to the traffic conditions. It is also possible to use different predictors for the resource usage, as well as estimators for the OD pair size. All the results show the baseline performance when the prediction is done using just the previous sample. Even with this simple techniques, our method has demonstrated to work well and achieve the target accuracy. However, other predictors may perform better for different datasets.

8. CONCLUSION

We reformulated the monitor placement problem to adapt it to the reality of network operations and management in the Internet. We have proposed an optimization method to select and configure passive monitors in a backbone network. The method receives as input the network topology, the routing matrix and the set of OD pairs of interest. It returns a set of monitors (and their sampling rates) that is optimal with respect to the measurement task to perform. We have described the performance of our method considering a canonical measurement task, i.e., the estimation of the size of a set of OD pairs.

Although the evaluation in this paper is in terms of estimating OD pair sizes, the optimization method is not specific to them. The method can be applied to a wide range of measurement tasks for which a utility function can be sought. Our ongoing work is centered on defining new expressions for the utility function for applications such as anomaly detection and network performance analysis. We are also studying alternative formulations for the objective function as well as trying to evaluate the performance of our method in larger networks and different datasets.

ACKNOWLEDGMENTS

We wish to thank DANTE for granting us access to the NetFlow, BGP and ISIS dataset of the GEANT network to perform this study.

9. REFERENCES

[1] Y. Bejerano and R. Rastogi. Robust monitoring of link delays and faults in IP networks. In *Proceedings of IEEE Infocom*, Apr. 2003.

[2] S. Bhattacharyya, C. Diot, J. Jetcheva, and N. Taft. Geographical and temporal characteristics of inter-PoP flows: View from a single PoP. *European Transactions on Telecommunications*, 13(1):5–22, Feb. 2002.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] Cisco Systems. NetFlow services and applications. White Paper, 2000.

[5] N. G. Duffield, C. Lund, and M. Thorup. Properties and prediction of flow statistics from sampled packet streams. In *Proceedings of ACM Internet Measurement Workshop*, Nov. 2002.

[6] N. G. Duffield, C. Lund, and M. Thorup. Estimating flow distributions from sampled flow statistics. In *Proceedings of ACM Sigcomm*, Aug. 2003.

[7] C. Estan, K. Keys, D. Moore, and G. Varghese. Building a better NetFlow. In *Proceedings of ACM Sigcomm*, Aug. 2004.

[8] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True. Deriving traffic demands for operational ip networks: Methodology and experience. In *Proceedings of ACM Sigcomm*, Sept. 2000.

[9] M. Fullmer and S. Romig. The OSU flow-tools package and Cisco NetFlow logs. In *Proceedings of Usenix LISA*, Dec. 2000.

[10] GEANT. http://www.dante.net.

[11] R. T. Haftka and Z. Guerdal. *Elements of Structural Optimization*. Kluwer Academic Publishers, third edition.

[12] N. Hohn and D. Veitch. Inverting sampled traffic. In *Proceedings of ACM Internet Measurement Conference*, Oct. 2003.

[13] J. Horton and A. Lopez-Ortiz. On the number of distributed measurement points for network tomography. In *Proceedings of ACM Internet Measurement Conference*, Nov. 2003.

[14] G. Iannaccone, C.-N. Chuah, S. Bhattacharyya, and C. Diot. Feasibility of IP restoration in a tier-1 backbone. *IEEE Network*, 18(2):13–19, Mar. 2004.

[15] IP Flow Information eXport Working Group. Internet Engineering Task Force. http://www.ietf.org/html.charters/ipfix-charter.html.

[16] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. Inferring TCP connection characteristics through passive measurements. In *Proceedings of IEEE Infocom*, Mar. 2004.

[17] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. On the placement of internet instrumentation. In *Proceedings of IEEE Infocom*, Apr. 2000.

[18] A. K. John Hertz and R. G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley.

[19] Juniper Traffic Sampling. http://www.juniper.net. Netflow-compatible implementation.

[20] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic matrix estimation: Existing techniques and new directions. In *Proceedings of ACM Sigcomm*, Pittsburgh, Aug. 2002.

[21] H. Nguyen and P. Thiran. Active measurement for multiple link failures diagnosis in IP networks. In *Proceedings of PAM Workshop*, Apr. 2004.

[22] K. Papagiannaki, N. Taft, and A. Lakhina. A distributed approach to measure IP traffic matrices. In *Proceedings of ACM Internet Measurement Conference*, Oct. 2004.

[23] P. Phaal, S. Panchen, and N. McKee. InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks. RFC 3176, Sept. 2001.

[24] D. Plonka. Flowscan: A network traffic flow reporting and visualization tool. In *Proceedings of Usenix LISA*, Dec. 2000.

[25] A. Shaikh, M. Goyal, A. Greenberg, R. Rajan, and K. Ramakrishnan. An OSPF topology server: Design and evaluation. *IEEE Journal on Selected Areas in Communications*, 20(4), May 2002.

[26] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot. Traffic matrices: Balancing measurements, inference and modeling. In *Proceedings of ACM Sigmetrics*, June 2005.

[27] A. Soule, A. Nucci, E. Leonardi, R. Cruz, and N. Taft. How to identify and estimate top largest OD flows in a dynamic environment. In *Proceedings of ACM Sigmetrics*, June 2004.

[28] K. Suh, Y. Guo, J. Kurose, and D. Towsley. Locating network monitors: Complexity, heuristics and coverage. In *Proceedings of IEEE Infocom*, Mar. 2005.

[29] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg. Fast accurate computation of large-scale IP traffic matrices from link loads. In *Proceedings of ACM Sigmetrics*, June 2003.